

# Journal of the ACM

## Complexity Theory

- Article 31**    **J. Fearnley**  
(48 pages)    **P. Goldberg**  
                 **A. Hollender**  
                 **R. Savani**

The Complexity of Computing KKT Solutions of  
Quadratic Programs

## Computational Complexity

- Article 32**    **V. Guruswami**  
(40 pages)    **B. Lin**  
                 **X. Ren**  
                 **Y. Sun**  
                 **K. Wu**

Parameterized Inapproximability Hypothesis under ETH

## Theory of computation, Models of computation, Formal methods

- Article 33**    **M. Künnemann**  
(27 pages)    **F. Mazowiecki**      Coverability in VASS Revisited: Improving Rackoff's  
                 **L. Schütze**          Bounds to Obtain Conditional Optimality  
                 **H. Sinclair-Banks**  
                 **K. Węgrzycki**

**continued on back cover**



Association for  
Computing Machinery

*Advancing Computing as a Science & Profession*

# Journal of the ACM

ACM  
1601 Broadway, 10th Floor  
New York, NY 10019-7434  
Tel.: (212)-869-7440  
Fax: (212)-869-0481  
<https://www.acm.org>

**Home Page:** <https://jacm.acm.org/>

## Editor-in-Chief

Venkatesan Guruswami	<i>University of California, Berkeley, United States</i>
----------------------	--

## Area Editors

Pankaj Agarwal	<b>Computational Geometry</b> Duke University, United States
Amal Ahmed	<b>Programming Languages and Verification</b> Northeastern University, United States
Albert Atserias	<b>Logic and Complexity Theory</b> Technical University of Catalonia, Spain
Peter Bartlett	<b>Machine Learning</b> University of California, Berkley, United States
Avrim Blum	<b>Machine Learning and Computational Learning Theory</b> Toyota Technological Institute at Chicago, United States
David Forsyth	<b>Computer Graphics and Computer Vision</b> University of Illinois, Urbana-Champaign, United States
Seth Gilbert	<b>Distributed Computing</b> National University of Singapore, Singapore
Paul Goldberg	<b>Economics and Computation; Algorithms and Complexity</b> Oxford University, United Kingdom
Monika Henzinger	<b>Algorithms and Data Structures</b> Institute of Science and Technology Austria, Austria
Sanjeev Khanna	<b>Algorithms</b> University of Pennsylvania, United States
Ravi Kumar	<b>Web Algorithms and Data Mining</b> Google, United States
Orna Kupferman	<b>Formal Methods</b> Hebrew University, Israel
Kasper Green Larsen	<b>Data Structures and Algorithms; Learning Theory</b> Aarhus University, Denmark
Shachar Lovett	<b>Complexity</b> University of California, San Diego, United States

Rafail Ostrovsky

## Cryptography and Security

*University of California, Los Angeles,  
United States*

## Logic, Verification, and Computation

*Max Planck Institute for Software  
Systems, Germany*

## Scientific Computing and High Performance Computing

*Purdue University, United States*

## Complexity and Information Theory

*Tata Institute of Fundamental Research,  
India*

## Algorithms and Combinatorial Optimization

*University of Washington, United States*

## Parallel Computing and Architecture

*Universität Paderborn, Germany*

## Database Systems and Theory

*University of Washington, United States*

## Quantum Computing

*California Institute of Technology,  
United States*

## Artificial Intelligence

*NICTA and University of New South Wales,  
Australia*

## Online Algorithms and Distributed Systems

*California Institute of Technology,  
United States*

Joël Ouaknine

Alex Pothen

Jaikumar Radhakrishnan

Thomas Rothvoß

Christian Scheideler

Dan Suciu

Thomas Vidick

Toby Walsh

Adam Wierman

## Journal Administrator

Tracey Glazener

*Technica Editorial Services, United States*

## Headquarters Staff

Scott Delman

*Director of Publications*

Sara Kate Heukerott

*Associate Director of Publications,  
Journals*

Yubing Zhai

*Editor*

Stacey Schick

*Senior Associate Editor*

Craig Rodkin

*Manager, Publishing Operations*

Barbara Ryan

*Intellectual Property Rights Manager*

Bernadette Shade

*Publishing Production Manager*

Anna Lacson

*Content QA Specialist*

Darshanie Jattan

*Administrative Assistant*

The Journal of the ACM (ISSN: 0004-5411; e-ISSN: 1557-735X) is published bimonthly by the Association for Computing Machinery (ACM), 1601 Broadway, 10th Floor, New York, NY 10019-7434.

For manuscript submissions, subscription, and change of address information, see inside backcover.

Copyright © 2025 by the Association for Computing Machinery (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Request permission to republish from: [permissions@acm.org](mailto:permissions@acm.org) or fax Publications Department, ACM, Inc. Fax +1 212-869-0481.

For other copying of articles that carry a code at the bottom of the first or last page or screen display, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923.



Association for  
Computing Machinery

Advancing Computing as a Science & Profession

# The Complexity of Computing KKT Solutions of Quadratic Programs

JOHN FEARNLEY, University of Liverpool, Liverpool, United Kingdom

PAUL GOLDBERG, University of Oxford, Oxford, United Kingdom

ALEXANDROS HOLLENDER, University of Oxford, Oxford, United Kingdom

RAHUL SAVANI, The Alan Turing Institute, London, United Kingdom and University of Liverpool, Liverpool, United Kingdom

---

It is well known that solving a (non-convex) quadratic program is NP-hard. We show that the problem remains hard even if we are only looking for a Karush–Kuhn–Tucker (KKT) point, instead of a global optimum. Namely, we prove that computing a KKT point of a quadratic polynomial over the domain  $[0, 1]^n$  is complete for the class  $\text{CLS} = \text{PPAD} \cap \text{PLS}$ .

CCS Concepts: • Theory of computation → Problems, reductions and completeness; Quadratic programming;

Additional Key Words and Phrases: Quadratic programming, KKT, continuous local search

**ACM Reference Format:**

John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. 2025. The Complexity of Computing KKT Solutions of Quadratic Programs. *J. ACM* 72, 5, Article 31 (October 2025), 48 pages. <https://doi.org/10.1145/3745017>

---

## 1 Introduction

**Quadratic programming (QP)** is the problem of optimizing a quadratic function of a collection of real variables, subject to linear constraints on those variables. It has widespread applications, numerous software implementations, and an extensive literature on its theoretical analysis, dating back more than 50 years. A fairly standard formulation is the following:

$$\min_{x \in \mathbb{R}^n} f(x) := x^\top Qx + c^\top x \text{ subject to } a_i^\top x \leq b_i, \forall i \in \{1, \dots, m\}. \quad (1)$$

In (1), the matrix  $Q$  is usually (and without loss of generality) taken to be symmetric, and there has been much work on restrictions of the problem based on assumed properties of  $Q$ , some

---

A preliminary version of this article appeared at STOC 2024.

J.F. and R.S. were supported by EPSRC Grant EP/W014750/1. P.W.G. was supported by EPSRC Grant EP/X040461/1. A.H. was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number MB22.00026. Authors' Contact Information: John Fearnley, University of Liverpool, Liverpool, United Kingdom; e-mail: john.fearnley@liverpool.ac.uk; Paul Goldberg, University of Oxford, Oxford, United Kingdom; e-mail: Paul.Goldberg@cs.ox.ac.uk; Alexandros Hollender, University of Oxford, Oxford, United Kingdom; e-mail: alexandros.hollender@cs.ox.ac.uk; Rahul Savani, The Alan Turing Institute, London, United Kingdom and University of Liverpool, Liverpool, United Kingdom; e-mail: rahul.savani@liverpool.ac.uk.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART31

<https://doi.org/10.1145/3745017>

of which we touch on below. The main result of the present article is that for QP it is computationally hard to compute a *Karush–Kuhn–Tucker* (KKT) point, an important kind of solution consisting of one that is locally optimal with respect to gradient descent. Moreover, our hardness result applies to a special case of interest known as “box constraints” (e.g., [3, 8]), in which the feasible region (i.e., the region  $a_i^T x \leq b_i, \forall i \in \{1, \dots, m\}$ ) consists of an axis-aligned hypercuboid; here, we use  $[0, 1]^n$ . (Indeed, the general version of the problem sometimes assumes that the linear constraints include  $x \in [0, 1]^n$  (e.g., [5]), which guarantees that the feasible region is compact).

**KKT solutions and other types of solution.** Informally, a KKT point is one that constitutes a local optimum of gradient descent. It may be a point at which the gradient is zero (a stationary point), or one where the gradient is non-zero, but further downhill progress is obstructed by one or more of the boundary constraints. A key feature of KKT solutions of (1) is that they have concise certificates: roughly, the gradient together with the binding constraints at a point of interest. Moreover, if the domain is compact, there is guaranteed to be at least one KKT point, since the global optimum is a KKT point. These two observations indicate that the problem of searching for a KKT point belongs to the complexity class TFNP, search problems in which easily checkable solutions must exist. In particular, this means that the problem is not expected to be NP-hard, unless NP = co-NP [23]. Apart from KKT points, the other main solution concepts for continuous optimization problems are the following:

- Global optimum,  $x$  for which  $f(x) \leq f(x')$  for all  $x'$  in the feasible region;
- Stationary point (a.k.a. critical point), where  $\nabla f(x) = 0$ ;
- Local minimum,  $x$  where for some  $\varepsilon > 0$ , we have  $f(x) \leq f(x')$  for all  $x'$  within  $\varepsilon$  of  $x$ ; at a *strict* local minimum, we have  $f(x) < f(x')$  for all  $x'$  in the feasible region within  $\varepsilon$  of  $x$ .

For QPs of the form (1), stationary points are not guaranteed to exist, and global optima and local minima are NP-hard to compute. KKT points relate to other solution concepts as follows. Global or local minima are KKT points; thus, KKT points are at least as easy to compute as global/local minima. Any stationary point is a KKT point, but stationary points are not guaranteed to exist, even for the box-constrained feasible regions that we consider here. To see that stationary points can be searched for in polynomial time, note that they are given by  $\nabla f = 0$ ; hence, define a linear subspace, and checking for points in the subspace that satisfy the boundary constraints  $a_i^T x \leq b_i$  amounts to solving an LP. In the unconstrained case (in which there are no boundary constraints), stationary points are the same as KKT points, so then the problem of searching for either kind of solution is tractable. A stationary point need not be a local minimum: For the problem of minimizing  $f(x) = -x^2$  over the interval  $[0, 1]$ ,  $x = 0$  is a stationary point but not a local minimum.

**Hardness results for global/local optima.** There has been much work studying the circumstances in which one can efficiently compute a solution of one of the above kinds, also on determining whether a given point  $x$  is one of the above solutions.

The global optimization problem for quadratic programming is amongst the earliest problems to be shown NP-hard [32, 33],<sup>1</sup> although containment in NP had to wait until substantially later [35].<sup>2</sup> NP-hardness has also been established for various restrictions of the problem, for example, [28] obtain NP-hardness when  $Q$  has rank 1 with one negative eigenvector (in a sense the simplest kind

<sup>1</sup>Indeed, even before NP-completeness, [24] reduce MAX CLIQUE to a version of QP in which the feasible region is a simplex. In Sahni’s reduction (from SUBSET SUM), the feasible region is a box.

<sup>2</sup>Vavasis [35] showed, in particular, that there exist optimal solutions having polynomial bit complexity.

of non-convex program that can be expressed as a QP). A simple reduction from MAX-CLIQUE to QP [2, 24, 30] yields NP-hardness for QPs that are square-free quadratic forms (diagonal entries of the matrix  $Q$  in (1) are zero, and there are no linear terms  $c^\top x$ ); the feasible region is a simplex as opposed to a box.

Regarding local optima, there are hardness results known for computing them, as well as for checking whether a given point is locally optimal. It is shown in [2] that it is hard to find an approximation to a local minimum. The problems of checking whether a given point  $x$  is a local optimum, or a strict local optimum, are NP-hard [25, 27] (in particular, when the feasible region is the unit box  $[0, 1]^n$  and  $x$  is the origin). In the unconstrained case ( $m = 0$ ) [1] show that it is possible in polynomial time to determine whether any version of local optimal solution exists.

There are strong hardness results even for computing approximations to the global optimum of QP. Bellare and Rogaway [5] obtained hardness of approximation for QP by reducing a two-prover one-round proof (with polylogarithmic communication) to QP in quasi-polynomial time. From this it follows that assuming NP problems are not solvable in quasi-polynomial time, there is no non-trivial constant factor approximation algorithm for QP. Bellare and Rogaway [5, Theorem 1.3] also show that for some constant  $\mu \in (0, 1)$ , QP is NP-hard to approximate within a factor  $\mu$ ; these hardness results even apply assuming numbers are given in unary. Also in the context of two-prover one-round interactive proof systems, [17] show how the search for an optimal strategy for the provers can be expressed as a QP. They study a relaxed version of the QP that corresponds with an upper bound on the value of the game played by the provers (and is poly-time computable); this leads to a general-purpose heuristic for problems in NP, and, in turn, a general kind of algorithm for diverse problems in P. Feige and Kilian [16] (Corollary 4) use this to conclude that (unless  $P = NP$ ) there is no constant-factor approximation algorithm for QP.

## 1.1 NP Total Search Problems and the Class CLS

As a solution concept, KKT points have two appealing properties: guaranteed existence (provided the feasible region is bounded) and polynomial-time checkability (we can efficiently verify that a point is KKT). These properties mean that the problem of *computing* one belongs to the complexity class TFNP: Total (as opposed to partial) functions that belong to NP. Problems that belong to TFNP are classified by various syntactic subclasses associated with the proof principle underlying the existence guarantee. Here, the relevant classes are PLS [21], PPAD [26], and CLS [12], the latter having been shown to be equal to PPAD  $\cap$  PLS [15].

The complexity class CLS (for “continuous local search”) was introduced by [12] in an effort to understand the complexity of certain seemingly hard search problems that belong to both PPAD and PLS. The problems they list include the search for a KKT point of a given polynomial over a domain given by linear constraints. Such problems are unlikely to be complete for either PPAD or PLS, since such a result would indicate that one of PPAD or PLS contains the other. Recently [15] showed that CLS is equal to the intersection of PPAD and PLS, in the process showing that it is CLS-complete to find KKT solutions of piecewise polynomial functions defined by a certain class of arithmetic circuits. Building on these results, [4] showed that computing a (possibly mixed) Nash equilibrium of a congestion game is CLS-complete and furthermore (of more relevance to the present article) that local optimisation (in the KKT sense) of degree-5 polynomials is also CLS-complete. Since the CLS-completeness results of [4, 15], other problems in game theory have been shown CLS-complete [13, 34] via comparatively direct reductions. The main result of [15] indicates that CLS-complete problems are unlikely to have polynomial-time algorithms. Moreover, the hardness of CLS can also be based on various cryptographic assumptions such as particular versions of indistinguishability obfuscation [19], soundness of Fiat–Shamir [10], or Learning With Errors [20].

Regarding our main result and its significance, we have noted that quadratic programming is a fundamental problem of general interest. Our main result answers an open question raised in [29] (see problem 3) and reiterated in [2]. The problem POLYNOMIAL KKT [12] is a generalization of (1), in which  $f$  is allowed to be any polynomial, written down as a sum of monomials. Babichenko and Rubinstein [4] showed that the POLYNOMIAL KKT problem is CLS-complete for degree-5 polynomials, which naturally raises the question, pointed out in [15], of whether such a result holds for lower degree. Here, we identify the lowest degree for which a hardness result holds, since for degree 1 the problem is linear programming. On the other hand, our result does not hold for some versions of interest, such as taking a standard simplex as the feasible region,<sup>3</sup> for example, [6]. Another important class of QPs that differs from the one studied here involves optimising quadratic functions over a unit sphere, or an intersection of spheres, for example, [36, 37, 39].

Our main result is for the computation of *exact* (as opposed to approximate) solutions. Fortunately, any problem instance has rational-valued KKT solutions whose bit complexity is polynomial in the bit complexity of numbers appearing in the problem instance.<sup>4</sup> If we consider natural notions of approximation, computation of exact solutions is polynomial-time equivalent to the computation of  $\varepsilon$ -approximate solutions for inverse-exponential  $\varepsilon$ . Ye [38] gives an algorithm that computes  $\varepsilon$ -KKT points, whose runtime dependence on  $\varepsilon$  is  $O((1/\varepsilon)\log(1/\varepsilon)\log(\log(1/\varepsilon)))$  (there is also polynomial dependence on  $n$ , and a factor representing the difference between the maximal and minimal objective values). So we give a negative answer to the question of whether a logarithmic dependence on  $\varepsilon$  is possible. Finally, our hardness result also highlights a contrast with convex optimisation, in which KKT points and global optima coincide, and many algorithms are known that find  $\varepsilon$ -approximate solution in time  $O(\log(1/\varepsilon))$  [7].

**THEOREM (MAIN RESULT).** *It is CLS-complete to compute KKT solutions of (1), even when the feasible region consists of the unit box  $[0, 1]^n$ .*

## 1.2 Technical Overview

Our result is proved in two steps. First, we present a reduction from the problem of computing (some sort of) a KKT point of a type of arithmetic circuit to the problem of computing a KKT point of a QP with box constraints. Namely, we consider *linear arithmetic circuits* that compute piecewise linear functions using a single kind of (fairly general, multi-purpose) gate. In the second step, we show that computing a KKT solution of such a circuit is a CLS-hard problem by reducing from a version of the problem for more general circuits, that is known to be CLS-hard [15]. Together, these two reductions establish the CLS-hardness of computing a KKT point of a QP.

While the first step is certainly the most interesting part of this combined reduction, the second step is surprisingly technical and requires a certain number of new ideas, which are likely to be useful in future works. We now present the main challenges in both parts, as well as the new ideas that were needed to overcome them.

**Step 1: Reducing from a circuit to a QP.** The first challenge in this part is the following main obstacle.

**Challenge 1: We can only use terms of degree at most two.** Unsurprisingly, the techniques used in [15] to show CLS-hardness of finding a KKT point of a general arithmetic circuit are of no

<sup>3</sup>In subsequent work, our result was extended to also hold for such simplex constraints, by presenting a direct reduction from box constraints to simplex constraints [18].

<sup>4</sup>This does not hold for objective functions of degree 3 or more. The distinction is analogous to the distinction between Nash equilibria of 2-player games versus 3-player games.

use here, since we are reducing to an explicit polynomial. Rather, just as in [4], we will just use the result of [15] as a starting point for reductions.

The techniques used in [4] to reduce to a degree-5 polynomial are much more relevant here. Their reduction is highly non-trivial and also relies on some older ideas used in the context of proving PLS-hardness of a version of local-max-SAT [22]. However, the restriction here to degree-2 polynomials makes a big difference and we mostly cannot re-use their ideas.<sup>5</sup> We encounter a fundamental obstacle to the use of *guide variables* (called guide players in [4], since their reduction is presented in terms of a game). Very briefly, the role of these guide variables, which were already used in [22] in a somewhat simpler form, is to be able to “deactivate” some interactions between two (or more) other variables. This deactivation is absolutely crucial for the approach of [4], as it already was in [22]. Given that in any reasonable construction of a quadratic polynomial, the interaction between two variables would yield a quadratic term (e.g.,  $x_i x_j$  or perhaps  $(x_i - x_j)^2$ ), the addition of a guide variable on top of that immediately takes us up to degree 3.

The inability to use guide variables, or any of the other involved machinery from [4], forces us to start from the ground up. As a toy example to illustrate our approach, consider a circuit  $C$  that takes two inputs  $x_1, x_2 \in [0, 1]$  and consists of only two gates. The first gate computes  $x_3 := x_1 + x_2$ , and then the second gate, which is the output of the circuit, computes  $x_4 := -2x_3$ . Thus, the circuit  $C$  simply computes the function  $f : [0, 1]^2 \rightarrow \mathbb{R}$ ,  $(x_1, x_2) \mapsto -2(x_1 + x_2)$ . This is a linear function and so finding a KKT point over the simple domain  $[0, 1]^2$  is very easy: The only KKT point (for the minimization problem) is at  $(1, 1)$ . Now, let us attempt to simulate this circuit by a quadratic polynomial that implements each gate separately.<sup>6</sup> Consider the polynomial

$$p(x_1, x_2, x_3, x_4) := (x_3 - x_1 - x_2)^2 + (x_4 + 2x_3)^2,$$

which consists of one squared term for each gate.<sup>7</sup> Intuitively, minimizing  $p$  will force  $x_3 = x_1 + x_2$  and  $x_4 = -2x_3$ . To simplify the exposition in this part of the overview, we think of  $x_3$  and  $x_4$  as being unconstrained, while  $x_1, x_2 \in [0, 1]$ . This makes the KKT conditions for  $x_3$  and  $x_4$  easier to work with,<sup>8</sup> as they just correspond to  $\frac{\partial p}{\partial x_3} = 0$  and  $\frac{\partial p}{\partial x_4} = 0$ . Now, the partial derivative of  $p$  with respect to  $x_4$  is

$$\frac{\partial p}{\partial x_4} = 2(x_4 + 2x_3).$$

At a KKT point, this must be zero, so we obtain  $x_4 = -2x_3$ . Next, we have

$$\frac{\partial p}{\partial x_3} = 2(x_3 - x_1 - x_2) + 4(x_4 + 2x_3).$$

Setting this to zero, and using  $x_4 = -2x_3$ , we obtain  $x_3 = x_1 + x_2$  as desired. Thus, any KKT point  $(x_1, x_2, x_3, x_4)$  of  $p$  satisfies  $x_4 = f(x_1, x_2)$ . In other words, we have correctly simulated the

<sup>5</sup>One notable exception to this is the idea of simulating the evaluation of a circuit by constructing an objective function that consists of a sum of terms, one for each gate of the circuit, with gates deeper down in the circuit having smaller weights. This idea of exponentially decreasing weights, already used in [22] in the context of discrete local optimisation, ensures that gates are correctly simulated and that their output is not biased by other gates that use it as an input.

<sup>6</sup>Obviously, there is a trivial reduction here that just lets the quadratic polynomial be the linear function  $f$  itself. However, we are interested in a construction that implements each gate separately, because we will ultimately need to implement (slightly) more general gates. Indeed, a circuit that only consists of linear gates represents a linear function, and it is easy to find KKT points of such functions.

<sup>7</sup>Because we use such squared terms, our polynomial will not be multi-linear. In particular, our result has no implications for games, unlike [4].

<sup>8</sup>The arguments still work if  $x_3$  and  $x_4$  are constrained to lie in some sufficiently large interval. For example, we could impose the constraints  $x_3 \in [-10, 10]$  and  $x_4 \in [-30, 30]$ , since this ensures that  $x_3$  and  $x_4$  never lie on the boundary at a solution. In any case, we will later revert to  $[0, 1]$  constraints for all variables and these will indeed be used in a very crucial way in our construction.

evaluation of the circuit. However, this is not enough. We want any KKT point  $(x_1, x_2, x_3, x_4)$  of  $p$  to yield a KKT point  $(x_1, x_2)$  of  $f$ , and this is currently not the case. What is missing is that the QP is not “aware” of the fact that it should attempt to minimize the output of the circuit, namely,  $x_4$ . An initial attempt to fix this by redefining

$$p(x_1, x_2, x_3, x_4) := (x_3 - x_1 - x_2)^2 + (x_4 + 2x_3)^2 + x_4$$

fails because it introduces big errors in the evaluation of the gates. This can be mitigated by using the idea of exponentially decreasing weights from [22] (which was also heavily used in [4]). Indeed, we can instead define

$$p(x_1, x_2, x_3, x_4) := (x_3 - x_1 - x_2)^2 + \delta(x_4 + 2x_3)^2 + \delta^2 x_4,$$

where  $\delta > 0$  is small. Performing the same analysis as above, yields  $x_4 = -2x_3 - \delta/2$ , and then  $x_3 = x_1 + x_2 + \delta^2$ . So, the gates are no longer evaluated exactly, but have some additive error. Fortunately, the error can be made arbitrarily small by making  $\delta$  sufficiently small.

The interesting observation, however, is that

$$\frac{\partial p}{\partial x_1} = -2(x_3 - x_1 - x_2) = -2\delta^2$$

and similarly  $\frac{\partial p}{\partial x_2} = -2\delta^2$ . This forces any KKT point  $(x_1, x_2, x_3, x_4)$  of  $p$  to satisfy  $(x_1, x_2) = (1, 1)$ , which is indeed the correct KKT point of the original function  $f$ ! Moreover, notice that  $-2\delta^2$  is equal to  $\delta^2 \frac{\partial f}{\partial x_1}$ , that is, it is proportional to the partial derivative of the original function  $f$ . In other words, this seemingly arbitrary error term actually carries useful information. This is not a coincidence. The errors, starting from the error in the output gate due to the new term  $\delta^2 x_4$ , propagate backwards in the circuit evaluation, until they reach the input variables  $x_1$  and  $x_2$ . In doing so, every traversed gate modifies the error in a very particular way, until, finally, the signal seen by the input variables corresponds to the gradient of the original function  $f$ . Indeed, at every gate, the error is modified in a way that corresponds to applying the rules for computing the gradient of a circuit using the *backpropagation* technique. This technique, widely used in machine learning, computes the gradient of a function by starting from the output and repeatedly applying the chain rule for differentiation until the inputs are reached. Indeed, the following can be proved by induction over the depth of the circuit:

**LEMMA (LINEAR BACKPROPAGATION LEMMA).** *When  $p$  is constructed from a depth- $m$  circuit  $C$  with linear gates computing a function  $f$ , any KKT point of  $p$  satisfies*

$$\frac{\partial p}{\partial x_i} = \delta^m \cdot \frac{\partial f}{\partial x_i}$$

for all input variables  $x_i$ .

**Challenge 2: Circuits with linear gates are easy.** The Linear Backpropagation Lemma implies that any KKT point of  $p$  must yield a KKT point of the original function  $f$ . Unfortunately, this is not enough to prove that our problem is intractable, because computing a KKT point of a linear function is an easy problem. In order to reduce from existing hard functions [4, 15], we would at least need the circuit  $C$  to also consist of multiplication gates  $x_k := x_i x_j$ . But to implement such a gate, we would need terms of the form  $(x_k - x_i x_j)^2$ , which have degree 4.

The crucial observation here is that we have not yet used the boundary of the domain in any way to implement gates. The boundary constraints suggest a natural generalization of linear gates.

Indeed, if we consider a term such as  $(x_3 - x_1 - x_2)^2$  and now—unlike we did before—also constrain  $x_3 \in [0, 1]$ , then we see that any KKT point of this term must satisfy

$$x_3 = T(x_1 + x_2),$$

where  $T : \mathbb{R} \rightarrow [0, 1]$  denotes truncation to the  $[0, 1]$  interval, i.e.,  $T(z) = \min\{1, \max\{0, z\}\}$ . More generally, we can simulate any gate of the form  $x_k := T(ax_i + bx_j + c)$  by the term  $(x_k - ax_i - bx_j - c)^2$ . We call such a gate a *truncated linear gate*. No efficient algorithm for computing KKT points of such circuits seems to be known, so there is hope that we might be able to prove intractability.

Unfortunately, before we can start considering proving such an intractability result, there is a more pressing issue: This reduction only works for a single gate. Although we still have correct (approximate) evaluation of the circuit by picking a sufficiently small  $\delta$ , the errors no longer correctly simulate backpropagation when truncation occurs. In order to restore the behavior that we observed in the setting without truncation, we need to find a way to simulate truncated linear gates that also works with backpropagation.

We modify the simulation as follows. A truncated linear gate  $x_k := T(ax_i + bx_j + c)$  is now simulated by the term

$$(x_k + z^+ - z^- - ax_i - bx_j - c)^2 + 2z^+z^- + 2z^+(1 - x_k) + 2z^-x_k,$$

where  $z^+$  and  $z^-$  are new auxiliary variables. Intuitively,  $z^+$  is here to “pick up the slack” between  $x_k$  and  $ax_i + bx_j + c$ , when the latter is strictly larger than 1. The variable  $z^-$  has a similar function when  $ax_i + bx_j + c < 0$ . Note that the derivative of the new term with respect to  $x_k$  is the same as the derivative of  $(x_k - ax_i - bx_j - c)^2$ . So, from the point of view of  $x_k$ , this new term is the same as the old one; in particular,  $x_k$  will again (approximately) take the value  $T(ax_i + bx_j + c)$ . The difference is in what the variables  $x_i$  and  $x_j$  see. The derivative of the old term with respect to  $x_i$  was just  $-2a(x_k - ax_i - bx_j - c)$ ; so when truncation occurred, this derivative would essentially correspond to the truncation gap, whereas we would want it to be 0, which is the correct backpropagation signal (because a small change in  $x_i$  would not change  $x_k$  if truncation occurs). On the other hand, the derivative of the new term with respect to  $x_i$  is  $-2a(x_k + z^+ - z^- - ax_i - bx_j - c)$ . In this derivative, the variables  $z^+$  and  $z^-$  fill the gap between  $x_k$  and  $ax_i + bx_j + c$  when truncation occurs, and thus we obtain the correct backpropagation signal 0. If truncation does not occur, then  $z^+ = z^- = 0$  and the new term behaves just like the old term.

It is thus tempting to try to establish the following lemma.

**DESIRED LEMMA (IDEAL BACKPROPAGATION LEMMA).** *When  $p$  is constructed from a depth- $m$  circuit  $C$  with truncated linear gates computing a function  $f$ , any KKT point of  $p$  satisfies*

$$\left( \frac{\partial p}{\partial x_1}(x_1, x_2), \frac{\partial p}{\partial x_2}(x_1, x_2) \right) \in \delta^m \cdot \partial f(x_1, x_2),$$

where  $x_1$  and  $x_2$  are the input variables, and  $\partial f$  is the generalized gradient<sup>9</sup> of the (almost everywhere differentiable) function  $f$ .

**Challenge 3: The Ideal Backpropagation Lemma does not hold.** Unfortunately, this lemma fails to hold. The reason for this is quite fundamental: backpropagation does not really work for such circuits. Consider the following simple example: The circuit  $C$  has a single input  $x_1$ , and outputs the value  $T[2x_1]/2 + T[x_1 - 1/2]$ . This can easily be implemented by using three truncated linear gates. Note that the circuit computes the (linear!) function  $[0, 1] \rightarrow [0, 1]$ ,  $x_1 \mapsto x_1$ , which

<sup>9</sup>At a point where  $f$  is differentiable, the generalized gradient is the singleton set consisting of the gradient; where  $f$  is not differentiable, it is the set of convex combinations of well-defined gradients close to that point.

has derivative 1 everywhere. Now, consider performing backpropagation when the input is  $x_1 = 1/2$ . Since this is the threshold for both truncations, the value 0 is a valid derivative for both of those gates. As a result, the gradient computed by backpropagation could theoretically output 0. In Section 3, we provide a slightly more involved example (Example 1), where this indeed happens in our QP construction: The correct gradient value is  $1/2$  everywhere, but at  $x_1 \approx 1/2$ , we have  $\frac{\partial p}{\partial x_1}(x_1) = 0$ . In particular, our construction would incorrectly output  $x_1 \approx 1/2$  as a KKT point. This shows that the Ideal Backpropagation Lemma cannot hold, even in some approximate version where we would also consider points in the vicinity of  $(x_1, x_2)$ .

However, the example suggests that a weaker statement might hold. Indeed, the issue occurs because both truncations have a threshold at  $1/2$ . If we were to slightly perturb these thresholds, then we would indeed see a derivative of 0 appear. In other words, it seems reasonable to think that the backpropagation that occurs computes some convex combination of gradients of various perturbed versions of the circuit  $C$ . To be more precise, in a perturbed version of  $C$ , every gate  $x_k := T(ax_i + bx_j + c)$  is replaced by a gate  $x_k := T(ax_i + bx_j + c + \pi_k)$  for some  $\pi_k \in \mathbb{R}$ . By picking  $\delta$  sufficiently small, we can ensure that all  $\pi_k$  are as small as required. Indeed, we can prove the following result.

**LEMMA (BACKPROPAGATION LEMMA (INFORMAL)).** *When  $p$  is constructed from a depth- $m$  circuit  $C$  with truncated linear gates, any KKT point of  $p$  satisfies*

$$\left( \frac{\partial p}{\partial x_1}(x_1, x_2), \frac{\partial p}{\partial x_2}(x_1, x_2) \right) \in \delta^m \cdot \text{conv} \left\{ \nabla \tilde{f}(x_1, x_2) : \tilde{f} \text{ computed by small perturbation of } C \right\},$$

where  $x_1$  and  $x_2$  are the input variables.

This leads us to define the new notion of a generalized gradient of a linear arithmetic circuit to capture this behavior. See the subsequent preliminaries section for more details on this.

With the Backpropagation Lemma in hand, we can now leave quadratic polynomials behind us and focus on circuits with truncated linear gates. We will also refer to these by the simpler name *linear arithmetic circuits* (which is usually used to refer to circuits with  $+, -, c, \times c, \min, \max$  gates [15]), since our circuits can easily simulate such circuits and vice-versa. In the next step, we construct a class of such circuits that is robust to perturbations, and for which, it is CLS-hard to find a KKT point (with respect to the new definition of generalized gradient).

**Step 2(a): Designing a robust function: the mesa construction.** In order to show CLS-hardness of this problem, we have to reduce from an existing CLS-hard problem. We reduce from the problem of computing an approximate KKT point of a smooth function defined on the two-dimensional grid  $[0, 1]^2$ , which is known to be CLS-complete when the function is represented by an arithmetic circuit with more general gates [15]. Indeed, being able to work on a two-dimensional domain (as opposed to a high-dimensional one if we used [4] instead) allows us to avoid having to unnecessarily complicate the construction.

At a high level, given such a smooth function defined on  $[0, 1]^2$ , we would like to construct a piecewise linear function that has (approximately) the same KKT solutions as the original function. Importantly, our piecewise linear function must be represented by a linear arithmetic circuit. This is already challenging, even if we ignore the perturbations.

**Challenge 1: Existing linear circuit constructions give no guarantees about the gradient.** Interpolations of continuous functions by linear arithmetic circuits have been given in prior work [9, 15, 31] and indeed these have been pivotal in proving important results in this field. However, these constructions only aim to obtain a piecewise linear function that closely approximates the original function in terms of *function value*. The usage of the *averaging trick* [11], which is

common to all of these, means that the (generalized) gradient of the piecewise linear function can be wildly different from the original gradient and introduce spurious KKT solutions.

We are thus forced to move away from this type of construction. Putting circuits aside for a bit, there is a standard interpolation by a piecewise linear function that does (approximately) maintain the gradient. Simply pick a sufficiently fine standard triangulation of  $[0, 1]^2$ , define the value of the function at the vertices of the triangulation to agree with the original function, and interpolate linearly within each triangle. This interpolation would be sufficient for our purposes, because it is not hard to show that any KKT point of the new function must correspond to an approximate KKT point of the original function.

The “catch” is that we would have to construct a linear arithmetic circuit that represents this piecewise linear interpolation. Existing techniques, which all use the averaging trick, are unable to achieve this. However, it turns out that using some new ideas (most of which we end up using in our final construction and which are highlighted below), it is in fact possible to construct a linear arithmetic circuit that *exactly* computes this piecewise linear interpolation. At this point, if the Ideal Backpropagation Lemma stated earlier was true, we would be done. Unfortunately, this is not the case, and we also have to argue about what happens to the circuit when gates are slightly perturbed.

**Challenge 2: The standard piecewise linear interpolation is not robust to perturbations.** Unfortunately, this circuit is not robust to perturbations, that is, perturbed versions of the circuit would introduce new solutions that did not appear in the original function. In order to illustrate the issues that can occur, as well as explain how they can be overcome, it is useful to take a step back and think about what would happen if the domain was **one-dimensional (1D)**, instead of **two-dimensional (2D)**.

In 1D case, with domain  $[0, 1]$ , the standard interpolation is indeed very simple. Given some sufficiently fine discretization of  $[0, 1]$ , we simply interpolate linearly between adjacent points. Implementing this using a linear arithmetic circuit is still non-trivial (because the discretization is exponentially fine), but it is possible using the new ideas hinted at above. Without going into too much detail, this piecewise linear function is constructed by taking the maximum of an exponential<sup>10</sup> number of simple functions. Every simple function implements a single linear segment of the interpolation and then very quickly decreases in value as soon as we leave the segment. Each of the simple functions can be constructed by taking the minimum of the three linear functions that they each consist of; see Figure 1. Taking the maximum of all these simple functions then indeed correctly yields the desired interpolation.

Now, let us see what happens when we allow small perturbations at gates of the circuit. Of course, it is hard to think about all the ways in which perturbations can interfere with a construction (especially since we have not given many details here), but a good starting point, and in a certain sense, the absolute minimum requirement is: If we slightly perturb each of the linear functions that are used to construct each of the simple functions by some small additive term, no new solutions should appear.

Unfortunately, the construction already fails this simple test, as can be seen in the example in Figure 2. Furthermore, note that the issue appears as soon as we allow non-zero perturbations; requiring the perturbations to be very small does not help. However, this example suggests a somewhat different approach: Instead of always trying to faithfully approximate the gradient of the original function, we can relax this requirement as long as we do not introduce any new KKT

---

<sup>10</sup>Of course, this would not be efficient, so the actual construction has to do something more clever. Nevertheless, this (incomplete) description is sufficient for this part of the technical overview.

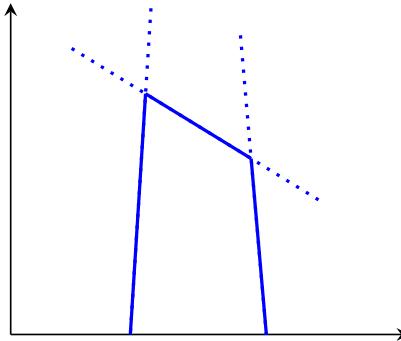


Fig. 1. Creating a 1D mesa as the minimum of three lines.

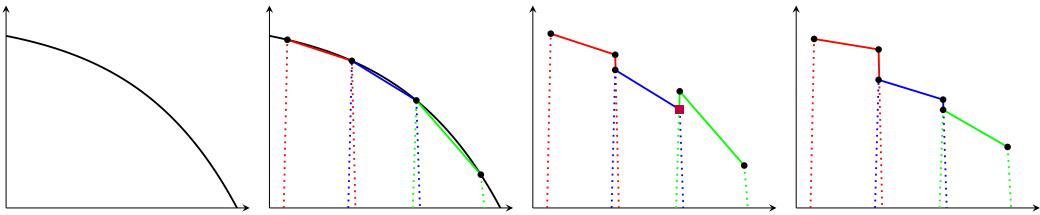


Fig. 2. Left to right: smooth function we want to interpolate (with no stationary points); standard interpolation as the maximum of adjacent mesas without perturbations (still no stationary points); additive perturbations introduce an unwanted stationary point (indicated as a brown square); this unwanted stationary point does not arise when the halved-gradient trick is applied.

points. Indeed, we can avoid the issue by using the following “halved-gradient” trick: For each linear segment, halve its slope while keeping the same value at the middle of the segment. As long as the perturbations are kept sufficiently small, this simple trick ensures that linear pieces with “bad” gradients are no longer visible, that is, they disappear when we take the maximum over all simple functions. See Figure 2 for an example. The only case where such a “bad” piece might appear is if the slope of the linear segment is very flat. But in that case, the original function must have an approximate KKT point there.

This approach indeed works for 1D setting. It turns out that the easiest way to generalize this to two dimensions is not to try to apply this to a triangulation of the unit square, but rather to a grid over the unit square. For any point on the grid, we construct a corresponding square segment with a gradient that is half the gradient of the original function at that point. When we leave that square, the function value decreases very quickly. We call this simple function a mesa due to its shape, which is reminiscent of a flat-topped hill with steep sides; see Figure 3. The final function is then obtained by taking the maximum of (an exponential number of such) mesa functions, one at each grid point.

We show that this mesa construction does not introduce any new KKT points, except in the vicinity of approximate KKT points of the original function. Importantly, this continues to hold even if we add an arbitrary, but small, perturbation to each linear piece of each mesa.

In the last part of this technical overview, we give some details about how the mesa construction can be implemented by using only the gates available in a linear arithmetic circuit (namely, truncated linear gates, as well as other gates that can be simulated by them, such as max and min). Furthermore, we need to make sure that the perturbations, which appear in any gate, can only im-

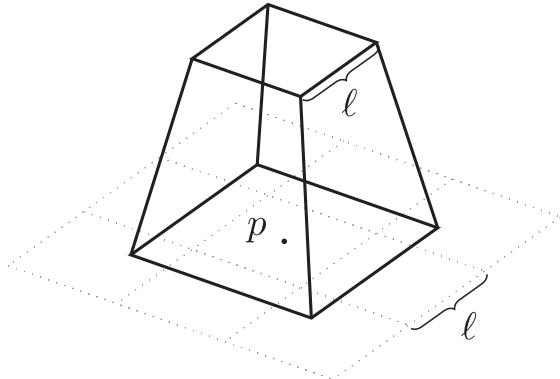


Fig. 3. An illustration of a 2D mesa function centered at point  $p$ .

pact the construction in the way described above (i.e., perturbing each linear piece of each mesa), and not in any other way.

**Step 2(b): Robustly implementing the mesa construction with a linear circuit.** From Step 2(a), we get a grid of points  $G$  covering  $[0, 1]^2$ , and Boolean circuits that, for each point,  $y \in G$  define a mesa centered at  $y$ . If  $m(x, y)$  is the height of the mesa centered at  $y$  at the point  $x$ , then we must implement a linear circuit that computes

$$f(x) = \max_{y \in G} m(x, y).$$

Since  $G$  contains exponentially many points, it is clearly infeasible to compute  $f$  directly. Instead, we first perform a bit extraction on  $x$  to obtain binary encodings of a small set  $S \subseteq G$  of nearby grid points, and we then evaluate  $f(x) = \max_{y \in S} m(x, y)$  instead. This works because each mesa can only achieve the maximum used in  $f$  in a small radius around its center, so we can disregard the mesas that are far from  $x$  when computing  $f$ .

While the technique of extracting bits from  $x$  to succinctly compute some function  $f(x)$  has been used before, we must overcome several challenges to make this work for our setting.

**Challenge 1: Dealing with bit extraction failures.** The bit extraction process requires us to implement an inherently discontinuous function, and since linear circuits can compute only continuous functions, we are forced to rely on bit extractors that can fail for a small subset of the inputs. This is a well-known problem, and prior work has addressed it through the use of the “averaging trick”, in which one extracts bits for  $x$  and also a large number of points that are close to  $x$ . By arranging the process such that only a small number of the bit extractions can fail, then the results over all of the points can be averaged, giving us a function  $\tilde{f}$  with  $|\tilde{f}(x) - f(x)| \leq \varepsilon$  for some small  $\varepsilon$ .

For prior work, which was usually interested only in ensuring that  $\tilde{f}(x)$  was far from zero whenever  $f(x)$  was also far from zero, this approach was good enough. However, for our purposes, any deviation in the desired output, no matter how small, may fatally undermine the construction by introducing new gradients, which could create new solutions.

To overcome this, we apply averaging in a new way that ensures that any errors arising from bit extraction failures do not make their way into the output. Specifically, we divide the points in the grid into four sets  $S_1, S_2, S_3, S_4 \subseteq G$ , where each set contains points from a grid of double the width of the original. The four sets are shown in four different colors on the left side of Figure 4.

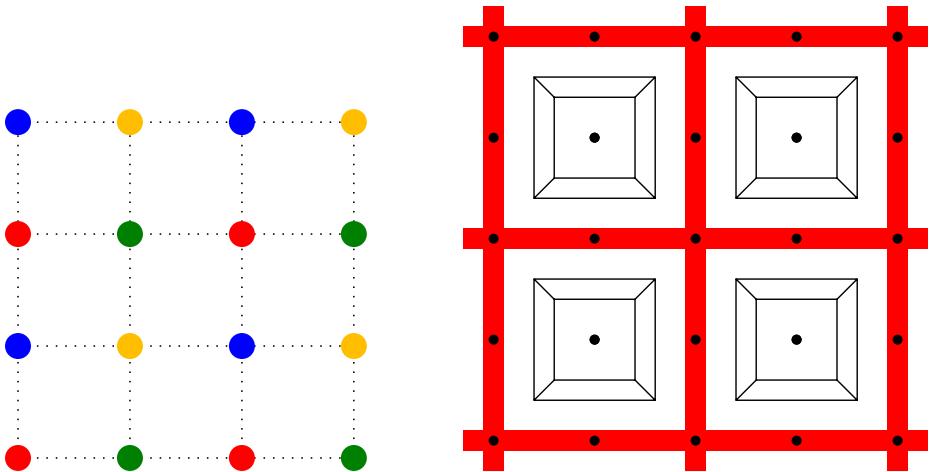


Fig. 4. Left: the division of the grid into four sets. Right: one of the four  $g_i$  functions.

For each set  $S_i$ , we build a function  $g_i(x) = \max_{y \in S_i} m(x, y)$ , which takes the maximum only over the mesas whose centers are in  $S_i$ . We then set  $f(x) = \max(g_1(x), g_2(x), g_3(x), g_4(x))$ .

The function  $g_i$  is shown on the right of Figure 4. The red lines in the figure show the locations at which a bit extraction might fail. Importantly, these regions occur only in places at which  $m(x, y) \leq 0$  for all mesa centers  $y \in S_i$ . We ensure that at most two bit extractions can fail, and that when they do fail they result in an output that is at most 1. So, by averaging over 12 points near  $x$ , we can ensure that  $g_i(x) \leq 1/6$  whenever a bit extraction failure occurs.

However, we also ensure that  $f(y) \geq 1/3$  for all points  $y$ . Therefore, the function  $g_j$  that is responsible for the mesa that defines  $f(x)$  will satisfy  $g_j(x) \geq 1/3$ . Since  $f$  is defined to be the maximum of the  $g$  functions, this means that any errors arising from failed bit decodings in  $g_i$  will be masked completely by another function  $g_j$  that has correctly decoded its input. Ultimately, this means that any spurious gradient arising from a failed bit extraction is well below the value of  $f(x)$ , and so these gradients can never make their way into the output.

**Challenge 2: Multiplying two variables.** We are given Boolean circuits that define the mesas that we must output. This means that a Boolean circuit will tell us to output an affine function at  $x = (x_1, x_2)$  with gradient  $g = (g_1, g_2)$  and additive value  $a$ , in which case, we will need to output

$$x_1 \cdot g_1 + x_2 \cdot g_2 + a.$$

This is problematic, however, because  $x$  and  $g$  are both variables, and a linear circuit does not allow us to multiply two variables together.

We circumvent this by showing that it is possible to compute  $x \cdot y$  when  $x$  is a continuous variable and  $y$  is a variable encoded in binary. So, we represent gradients in binary in our circuit, and this allows us to precisely control the gradients that we output.

**Challenge 3: Perturbations.** It is not enough to create a linear circuit that implements  $f$ , because our linear circuit must also be robust to perturbations. These perturbations will slightly alter the values that are outputted by min, max, and truncation operations, and we need to ensure that they do not alter the gradients of the mesas that appear in the output of the function.

We are able to show that evaluating a Boolean circuit and decoding a binary value can be carried out exactly with no errors even in the presence of perturbations, while introducing perturbations

in the bit extraction process only slightly increases the region in which the bit extraction will fail. Then, we show that each mesa can be computed with gradients that are exactly correct, but where each of the five pieces might be additively perturbed by a small amount. Finally, we show that the step of splitting the points into the  $g_i$  functions and then maximizing over them to produce  $f$  only adds an additional small perturbation, while not altering the gradients of any of the mesas.

## 2 Preliminaries

All numbers appearing as inputs in our problem are assumed to be rational. A rational number  $x$  is represented by its numerator and denominator (in binary) of the irreducible fraction for  $x$ . We let  $\text{size}(x)$  denote the number of bits needed to represent  $x$  in this way. We also extend this notation in the natural way to the case where  $x$  is a vector with rational entries.

*Definition 1.* The QUADRATIC-KKT problem is defined as follows. We are given a degree-2 polynomial  $p$  over  $n$  variables, and the goal is to compute a KKT point of the following optimization problem:

$$\begin{aligned} \min \quad & p(x) \\ \text{s.t.} \quad & 0 \leq x_i \leq 1 \quad \forall i \in [n]. \end{aligned} \tag{2}$$

A point  $x \in [0, 1]^n$  is a KKT point of (2) if, for all  $i \in [n]$ ,

- if  $x_i > 0$ , then  $\frac{\partial p}{\partial x_i}(x) \leq 0$ , and
- if  $x_i < 1$ , then  $\frac{\partial p}{\partial x_i}(x) \geq 0$ .

The QUADRATIC-KKT problem lies in the class **CLS**, even for more general domains,<sup>11</sup> namely, any non-empty compact domain given by linear inequalities [15]. This is because the problem can be solved (inefficiently) by gradient descent. The problem is guaranteed to always admit at least one rational solution with polynomially bounded bit complexity; see Appendix A for a proof sketch. Our main result is the following theorem.

**THEOREM 2.1.** *The QUADRATIC-KKT problem is **CLS**-complete.*

The problem remains **CLS**-complete even if we only ask for an  $\varepsilon$ -KKT point, where  $\varepsilon > 0$  is allowed to be exponentially small (i.e., is given in binary). Indeed, by standard arguments, finding an exact solution reduces to finding an approximate solution; see Appendix A. For the definition of  $\varepsilon$ -KKT points, we just replace “ $\leq 0$ ” and “ $\geq 0$ ” by “ $\leq \varepsilon$ ” and “ $\geq -\varepsilon$ ” (respectively) in the definition of QUADRATIC-KKT.

We prove Theorem 2.1 by reducing from the 2D-LINEAR-KKT problem, which we introduce below. In Section 3, we reduce from 2D-LINEAR-KKT to QUADRATIC-KKT, and in Section ??, we show that 2D-LINEAR-KKT is **CLS**-hard.

### 2.1 KKT Points of Linear Arithmetic Circuits

A linear arithmetic circuit is a circuit that consists of gates implementing piecewise linear operations. As a result, the function represented by a linear circuit is a piecewise linear function. Such circuits can be evaluated in polynomial time [15].

In this article, we consider linear arithmetic circuits that consist of a single<sup>12</sup> type of gate: truncated linear gates. A truncated linear gate is defined by rational parameters  $a, b, c \in \mathbb{Q}$ . The

<sup>11</sup>We omit the definition of a KKT point for more general domains; see, for example, [15].

<sup>12</sup>Of course, various other types of gates can be simulated using truncated linear gates, and we will use this later in the article.

gate takes as input two variables  $x_i, x_j$  of the circuit and outputs  $x_k := \text{T}(ax_i + bx_j + c)$ , where  $\text{T} : \mathbb{R} \rightarrow [0, 1]$  denotes truncation (i.e., projection) to the  $[0, 1]$  interval.

**Generalized gradients.** Let  $C$  be a linear arithmetic circuit with  $n$  inputs and one output. We let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  denote the function computed by the circuit  $C$ . This is a piecewise linear function that is almost everywhere differentiable. The generalized gradient of  $f$  at point  $y$  can be defined as

$$\partial f(y) := \text{conv} \left\{ \lim_{t \rightarrow \infty} \nabla f(y_t) : (y_t)_t \text{ converging to } y \text{ such that } f \text{ is differentiable at } y_t \right. \\ \left. \text{and } \nabla f(y_t) \text{ also converges} \right\}.$$

For our purposes, we have to introduce a new more general notion of generalized gradient of a circuit. Let  $C$  be a linear arithmetic circuit consisting of  $m$  truncated linear gates. For any  $\pi = (\pi_i)_{i \in [m]} \in \mathbb{R}^m$ , we let  $C^\pi$  denote the circuit  $C$  perturbed by  $\pi$ , namely, for each  $i \in [m]$ , the  $i$ th gate  $x_i := \text{T}(ax_j + bx_k + c)$  is replaced by  $x_i := \text{T}(ax_j + bx_k + c + \pi_i)$ . We let  $f^\pi : \mathbb{R}^n \rightarrow \mathbb{R}$  denote the function represented by the perturbed circuit  $C^\pi$ .

For any  $\delta > 0$  and any such circuit  $C$ , the  $\delta$ -generalized circuit gradient of  $C$  at point  $y \in \mathbb{R}^n$  is defined as

$$\tilde{\partial}_\delta C(y) := \text{conv} \{ \nabla f^\pi(y) : \pi \in [-\delta, \delta]^m \text{ such that } f^\pi \text{ is differentiable at } y \}.$$

It can be shown that  $\partial f(y) \subseteq \tilde{\partial}_\delta C(y)$  for all  $\delta > 0$ . Although it is tempting to think that  $\tilde{\partial}_\delta C(y) \rightarrow \partial f(y)$  as  $\delta \rightarrow 0$ , this is not the case. Indeed, Example 1 together with the Backpropagation Lemma (Lemma 3.2) provide a counter-example.

We can now define the intermediate computational problem, which will act as a bridge between existing **CLS**-hard problems and **QUADRATIC-KKT**.

**Definition 2.** The **2D-LINEAR-KKT** problem is defined as follows. We are given  $\varepsilon, \delta > 0$ , and a linear arithmetic circuit  $C$  with two inputs and one output, and consisting only of truncated linear gates. The goal is to find a point  $y \in [0, 1]^2$  that satisfies the  $\varepsilon$ -KKT conditions with respect to the  $\delta$ -generalized circuit gradient of  $C$ , that is, such that there exists  $u \in \tilde{\partial}_\delta C(y)$  satisfying

- if  $y_i > 0$ , then  $u_i \leq \varepsilon$
- if  $y_i < 1$ , then  $u_i \geq -\varepsilon$

for  $i = 1, 2$ .

We note that it is not clear whether **2D-LINEAR-KKT** lies in **TFNP**, because it is not clear whether we can efficiently check if some given  $y$  is a solution. Nevertheless, we establish in Section 4 that this problem is **CLS**-hard, which is all we require from this intermediate problem.

### 3 Reduction from **2D-LINEAR-KKT** to **QUADRATIC-KKT**

The main result of this section is the following.

**PROPOSITION 3.1.** *There is a polynomial-time reduction from **2D-LINEAR-KKT** to **QUADRATIC-KKT**.*

The remainder of this section proves this result. We begin with the detailed construction of the quadratic polynomial and a statement of the Backpropagation Lemma (Lemma 3.2), and explain why it implies Proposition 3.1. Then, we prove some simple properties of the construction, before the technical culmination of this section, namely the proof of the Backpropagation Lemma.

### 3.1 Construction and Backpropagation Lemma

Let  $C$  be a linear arithmetic circuit that has two inputs and one output, and that consists only of truncated linear gates.

Let  $n$  denote the number of variables in the circuit  $C$ . We use  $x_i$  to denote the  $i$ th variable in the circuit, and assume that  $x_1, \dots, x_n$  are ordered such that the gate computing  $x_i$  uses inputs  $x_{\ell(i)}, x_{r(i)}$  with  $\ell(i) < i$  and  $r(i) < i$ . In particular,  $x_1$  and  $x_2$  are the input variables, and  $x_n$  is the output variable. For every  $i \in [n] \setminus \{1, 2\}$ , the  $i$ th gate of  $C$  is the gate computing  $x_i$ . It will be more convenient to write the  $i$ th gate's function  $x_i = T(a_i x_{\ell(i)} + b_i x_{r(i)} + c_i)$  as  $x_i = T(\sum_{j=1}^{i-1} a_{ij} x_j + c_i)$ , where

$$a_{ij} = \begin{cases} a_i & \text{if } j = \ell(i) \\ b_i & \text{if } j = r(i) \\ 0 & \text{otherwise.} \end{cases}$$

Let also  $K \geq 1$  be such that  $K \geq \max_{i \in [n] \setminus \{1, 2\}} (\sum_{j=1}^{i-1} |a_{ij}| + |c_i|)$ .

We now construct a polynomial  $p$  on  $n+2(n-2) = 3n-4$  variables. In more detail, the polynomial will have the following variables:

- For each  $i \in [n]$ , a variable  $y_i$ , corresponding to each variable  $x_i$  of  $C$ .
- For each  $i \in [n] \setminus \{1, 2\}$ , two auxiliary variables  $z_i^+$  and  $z_i^-$  to help with the implementation of the  $i$ th gate, which computes  $x_i$ .

For each gate  $i \in [n] \setminus \{1, 2\}$ , we construct a polynomial  $q_i$  on variables  $y = (y_1, \dots, y_n), z = (z_3^+, z_3^-, \dots, z_n^+, z_n^-)$

$$q_i(y, z) := \left( y_i + Kz_i^+ - Kz_i^- - \sum_{j=1}^{i-1} a_{ij} y_j - c_i \right)^2 + 2K^2 z_i^+ z_i^- + 2Kz_i^+(1 - y_i) + 2Kz_i^-(y_i).$$

For a given  $\delta \in (0, 1)$ , the final polynomial  $p$  is then constructed as follows:

$$p(y, z) := \delta^{n+1} y_n + \sum_{i=3}^n \delta^i q_i(y, z).$$

**QUADRATIC-KKT instance.** The instance of QUADRATIC-KKT we consider is thus

$$\begin{aligned} \min \quad & p(y, z) \\ \text{s.t.} \quad & (y, z) \in [0, 1]^{3n-4}. \end{aligned} \tag{3}$$

We are now ready to state the main technical lemma of this section.

**LEMMA 3.2 (BACKPROPAGATION LEMMA).** *Let  $(y, z)$  be a KKT point of the constructed QP (3), for some  $\delta \in (0, 1/16K^2)$ . Then, we have*

$$\frac{1}{\delta^{n+1}} \cdot \left( \frac{\partial p}{\partial y_1}(y, z), \frac{\partial p}{\partial y_2}(y, z) \right) \in \tilde{\partial}_{\delta'} C(y_1, y_2),$$

where  $\delta' = 8K^2\delta$ .

Let us see how Proposition 3.1 follows from this lemma. Let  $\varepsilon'$ ,  $\delta'$ , and  $C$  be the inputs to a 2D-LINEAR-KKT instance. We construct the polynomial  $p$  described above with  $\delta := \min\{\delta'/8K^2, 1/32K^2\}$ . Clearly, this can be done in polynomial time. Now, consider any KKT point  $(y, z)$  of the resulting QP (3). We claim that  $(y_1, y_2)$  must be a solution to the original 2D-LINEAR-KKT instance. Indeed, let

$$u := \frac{1}{\delta^{n+1}} \cdot \left( \frac{\partial p}{\partial y_1}(y, z), \frac{\partial p}{\partial y_2}(y, z) \right).$$

By the Backpropagation Lemma, we have that  $u \in \tilde{\partial}_{\delta'} C(y_1, y_2)$ . Furthermore, since  $(y, z)$  is a KKT point of (3), we, in particular, have for  $i = 1, 2$

- if  $y_i > 0$ , then  $\frac{\partial p}{\partial y_i}(y, z) \leq 0$ , and thus  $u_i \leq 0$ ,
- if  $y_i < 1$ , then  $\frac{\partial p}{\partial y_i}(y, z) \geq 0$ , and thus  $u_i \geq 0$ .

In other words,  $(y_1, y_2)$  satisfies the KKT conditions (and thus, in particular, the  $\epsilon'$ -KKT conditions) with respect to the  $\delta'$ -generalized circuit gradient of  $C$ .

Before proceeding with the proof of the Backpropagation Lemma, we present an example showing that a stronger version of the lemma—where we ask for the generalized gradient of  $f$  at  $(y_1, y_2)$  (or even of some point in the vicinity) to be zero—fails.

*Example 1.* Consider the circuit  $C$  that has one single input  $x_1$  and computes  $x_2 := T(2x_1)$ ,  $x_3 := T(x_1 - 1/2)$ , and outputs  $x_4 := T(x_2/2 + x_3 - x_1/2)$ . It is easy to see that this circuit computes the function  $f : [0, 1] \rightarrow [0, 1], x_1 \mapsto x_1/2$ . Thus, the only KKT point of  $f$  is at  $x_1 = 0$ . However, it can be checked that if we construct the polynomial  $p$  as described above from  $C$ , then, for any sufficiently small  $\delta > 0$ , the QP (3) will have a KKT point at  $y_1 = 1/2 + \delta^2/4$  (and where we have  $y_2 = 1, y_3 = 0$ , and  $y_4 = 1/4 - \delta^2/8 - \delta/2$ ). In particular, this means that the backpropagation computes gradient 0 at that point, even though the actual gradient of  $f$  is always  $1/2$ . As a result, no general backpropagation result can be proved for this kind of circuit without taking into account perturbed versions of the circuit.

### 3.2 Properties of KKT Points

In this section, we prove some simple properties that are satisfied by any KKT point of the QUADRATIC-KKT instance (3). Recall that a point  $(y, z) \in [0, 1]^{3n-4}$  is a KKT point of (3) if, for all  $i \in [n]$ ,

- if  $y_i > 0$ , then  $\frac{\partial p}{\partial y_i}(y, z) \leq 0$ , and
- if  $y_i < 1$ , then  $\frac{\partial p}{\partial y_i}(y, z) \geq 0$ ,

and similarly for the other variables  $z_i^+$  and  $z_i^-$  for all  $i \in [n] \setminus \{1, 2\}$ .

**Truncation.** The following Lemma 3.3 states that, at any KKT point, the auxiliary variables  $z$  enforce truncation, in a certain sense.

LEMMA 3.3. *Let  $(y, z)$  be a KKT point of (3). Then, for all  $i \in [n] \setminus \{1, 2\}$*

$$T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) = \sum_{j=1}^{i-1} a_{ij} y_j + c_i - K z_i^+ + K z_i^-.$$

PROOF. We show the following stronger fact, namely that

$$K z_i^+ = \max \left\{ 0, \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) - T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) \right\} \quad (4)$$

and

$$K z_i^- = \max \left\{ 0, T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) - \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) \right\}. \quad (5)$$

In order to prove (4), note that the variable  $z_i^+$  only appears in  $q_i$ , and thus  $\frac{\partial p}{\partial z_i^+} = \delta^i \frac{\partial q_i}{\partial z_i^+}$  and

$$\begin{aligned}\frac{\partial q_i}{\partial z_i^+}(y, z) &= 2K \left( y_i + Kz_i^+ - Kz_i^- - \sum_{j=1}^{i-1} a_{ij}y_j - c_i \right) + 2K^2 z_i^- + 2K(1 - y_i) \\ &= 2K \left( 1 + Kz_i^+ - \sum_{j=1}^{i-1} a_{ij}y_j - c_i \right).\end{aligned}$$

We now consider two cases. If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \leq T(\sum_{j=1}^{i-1} a_{ij}y_j + c_i)$ , then it must be that  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \leq 1$ . As a result,  $\frac{\partial p}{\partial z_i^+}(y, z) = \delta^i \frac{\partial q_i}{\partial z_i^+}(y, z) \geq \delta^i \cdot 2K^2 z_i^+$ . By the KKT conditions, it follows that  $z_i^+ = 0$ . Indeed, if  $z_i^+ > 0$ , then we would have  $\frac{\partial p}{\partial z_i^+}(y, z) > 0$ , which contradicts the KKT conditions.

If, on the other hand,  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i > T(\sum_{j=1}^{i-1} a_{ij}y_j + c_i)$ , then it must be that  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i > 1$ . As a result,  $\frac{\partial p}{\partial z_i^+}(y, z) = \delta^i \frac{\partial q_i}{\partial z_i^+}(y, z) < \delta^i \cdot 2K^2 z_i^+$ . In particular, we cannot have  $z_i^+ = 0$ , since that would imply  $\frac{\partial p}{\partial z_i^+}(y, z) < 0$ , which is not allowed by the KKT conditions at  $z_i^+ = 0$ . We also cannot have  $z_i^+ = 1$ . Indeed, by the KKT conditions, that would imply that  $\frac{\partial p}{\partial z_i^+}(y, z) \leq 0$ , which translates to

$$\delta^i \cdot 2K \left( 1 + K - \sum_{j=1}^{i-1} a_{ij}y_j - c_i \right) \leq 0,$$

which is impossible, since  $K \geq 1$  was chosen such that  $K \geq \sum_{j=1}^{i-1} |a_{ij}| + |c_i| \geq \sum_{j=1}^{i-1} a_{ij}y_j + c_i$ . As a result, we must have  $z_i^+ \in (0, 1)$ , which implies that the KKT condition is  $\frac{\partial p}{\partial z_i^+}(y, z) = 0$ . This yields

$$Kz_i^+ = \sum_{j=1}^{i-1} a_{ij}y_j + c_i - 1 = \sum_{j=1}^{i-1} a_{ij}y_j + c_i - T \left( \sum_{j=1}^{i-1} a_{ij}y_j + c_i \right)$$

as desired. We have thus shown that (4) always holds at a KKT point.

In order to prove (5), we again note that  $\frac{\partial p}{\partial z_i^-} = \delta^i \frac{\partial q_i}{\partial z_i^-}$  and

$$\begin{aligned}\frac{\partial q_i}{\partial z_i^-}(y, z) &= -2K \left( y_i + Kz_i^+ - Kz_i^- - \sum_{j=1}^{i-1} a_{ij}y_j - c_i \right) + 2K^2 z_i^+ + 2Ky_i \\ &= 2K \left( Kz_i^- + \sum_{j=1}^{i-1} a_{ij}y_j + c_i \right),\end{aligned}$$

and then perform a similar case analysis. □

**Approximate evaluation.** The next Lemma 3.4 states that the gates of the circuit are correctly simulated at a KKT point of (3), up to some small additive error depending on the parameter  $\delta$ . The lemma also gives a precise expression for the value of each variable  $y_i$  at a KKT point, which will be useful for the next section. In order to state this precise expression, we first have to introduce some additional notation. We define, for any  $i \in [n] \setminus \{1\}$ ,

$$p_i(y, z) := \delta^{n+1} y_n + \sum_{\ell=i+1}^n \delta^\ell q_\ell(y, z).$$

In particular,  $p_2 = p$ , and  $p_n(y, z) = \delta^{n+1} y_n$ . We are now ready to state the lemma.

LEMMA 3.4. Let  $(y, z)$  be a KKT point of (3). Then, for any  $i \in [n] \setminus \{1, 2\}$

$$y_i = T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i - \frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \right) = T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) \pm (2K\delta)^{n+1-i}.$$

PROOF. Let us first prove the first equality. Note that

$$\begin{aligned} \frac{\partial p}{\partial y_i}(y, z) &= \delta^i \frac{\partial q_i}{\partial y_i}(y, z) + \frac{\partial p_i}{\partial y_i}(y, z) \\ &= \delta^i \left( 2 \left( y_i + Kz_i^+ - Kz_i^- - \sum_{j=1}^{i-1} a_{ij} y_j - c_i \right) - 2Kz_i^+ + 2Kz_i^- \right) + \frac{\partial p_i}{\partial y_i}(y, z) \\ &= 2\delta^i \left( y_i - \sum_{j=1}^{i-1} a_{ij} y_j - c_i \right) + \frac{\partial p_i}{\partial y_i}(y, z). \end{aligned}$$

Hence, if  $y_i > \sum_{j=1}^{i-1} a_{ij} y_j - c_i - \frac{1}{2\delta^i} \frac{\partial p_i}{\partial y_i}(y, z)$ , then  $\frac{\partial p}{\partial y_i}(y, z) > 0$ , and by the KKT conditions, we must have  $y_i = 0$ . If, on the other hand,  $y_i < \sum_{j=1}^{i-1} a_{ij} y_j - c_i - \frac{1}{2\delta^i} \frac{\partial p_i}{\partial y_i}(y, z)$ , then,  $\frac{\partial p}{\partial y_i}(y, z) < 0$ , and by the KKT conditions, we must have  $y_i = 1$ . Thus, in all cases, we have  $y_i = T(\sum_{j=1}^{i-1} a_{ij} y_j - c_i - \frac{1}{2\delta^i} \frac{\partial p_i}{\partial y_i}(y, z))$ .

In order to prove the second equality, we show that  $\frac{1}{2\delta^i} |\frac{\partial p_i}{\partial y_i}(y, z)| \leq (2K\delta)^{n+1-i}$  by induction. For  $i = n$ , we have  $\frac{\partial p_n}{\partial y_n}(y, z) = \delta^{n+1}$ , and thus  $\frac{1}{2\delta^n} |\frac{\partial p_n}{\partial y_n}(y, z)| = \delta/2 \leq 2K\delta$ . Now, assume that the statement holds for  $i+1, \dots, n$ . We show that it also holds for  $i$ . We can bound

$$\begin{aligned} \frac{1}{2\delta^i} \left| \frac{\partial p_i}{\partial y_i}(y, z) \right| &\leq \frac{1}{2\delta^i} \sum_{\ell=i+1}^n \delta^\ell \left| \frac{\partial q_\ell}{\partial y_i}(y, z) \right| \leq \frac{1}{2\delta^i} \sum_{\ell=i+1}^n \delta^\ell \cdot 2K \cdot (2K\delta)^{n+1-\ell} \\ &= \delta^{n+1-i} \cdot K \cdot \sum_{\ell=i+1}^n (2K\delta)^{n+1-\ell} \\ &\leq \delta^{n+1-i} \cdot K \cdot 2 \cdot (2K)^{n-i} = (2K\delta)^{n+1-i}, \end{aligned}$$

where we bound  $|\frac{\partial q_\ell}{\partial y_i}(y, z)|$  by

$$\begin{aligned} \left| \frac{\partial q_\ell}{\partial y_i}(y, z) \right| &= \left| -2a_{\ell i} \left( y_\ell + Kz_\ell^+ - Kz_\ell^- - \sum_{j=1}^{\ell-1} a_{\ell j} y_j - c_\ell \right) \right| \\ &= \left| -2a_{\ell i} \left( y_\ell - T \left( \sum_{j=1}^{\ell-1} a_{\ell j} y_j - c_\ell \right) \right) \right| \\ &\leq 2K \left| y_\ell - T \left( \sum_{j=1}^{\ell-1} a_{\ell j} y_j - c_\ell \right) \right| \\ &\leq 2K \cdot (2K\delta)^{n+1-\ell} \end{aligned}$$

using Lemma 3.3,  $|a_{\ell i}| \leq K$ , and the induction hypothesis for  $\ell \geq i+1$ .  $\square$

### 3.3 Proof of the Backpropagation Lemma

In this section, we prove the Backpropagation Lemma. We begin by recalling some notation, as well as introducing some new notation. We let  $f : \mathbb{R}^2 \rightarrow \mathbb{R}$  denote the function represented by the

circuit  $C$ . For any  $\pi = (\pi_i)_{i \in [n] \setminus \{1, 2\}} \in \mathbb{R}^{n-2}$ , we let  $C^\pi$  denote the circuit  $C$  perturbed by  $\pi$ , namely, for each  $i \in [n] \setminus \{1, 2\}$ , the  $i$ th gate  $x_i := T(\sum_{j=1}^{i-1} a_{ij}x_j + c_i)$  is replaced by  $x_i := T(\sum_{j=1}^{i-1} a_{ij}x_j + c_i + \pi_i)$ . We let  $f^\pi : \mathbb{R}^2 \rightarrow \mathbb{R}$  denote the function represented by the perturbed circuit  $C^\pi$ . For any sign vector  $s = (s_i)_{i \in [n] \setminus \{1, 2\}} \in \{+1, -1\}^{n-2}$ , we let  $s \cdot \pi \in \mathbb{R}^{n-2}$  denote the coordinate-wise product of vector  $s$  with vector  $\pi$ , that is,  $[s \cdot \pi]_i = s_i \pi_i$  for all  $i \in [n] \setminus \{1, 2\}$ . Below we also use  $\lambda_{-i}$  to denote  $1 - \lambda_i$ .

The Backpropagation Lemma is a consequence of the following technical lemma.

**LEMMA 3.5.** *Let  $(y, z)$  be a KKT point of QP (3), for some  $\delta \in (0, 1/16K^2)$ . Then, there exists a perturbation vector  $\pi = (\pi_i)_{i \in [n] \setminus \{1, 2\}} \in \mathbb{R}^{n-2}$  satisfying*

$$- |\pi_i| \leq 8K^2\delta \text{ for all } i \in [n] \setminus \{1, 2\},$$

– for all  $s \in \{+1, -1\}^{n-2}$ ,  $f^{s \cdot \pi}$  is differentiable in a small neighborhood around  $(x_1, x_2) = (y_1, y_2)$ .

In addition, there exists  $\lambda = (\lambda_i)_{i \in [n] \setminus \{1, 2\}} \in [0, 1]^{n-2}$  such that for  $k = 1, 2$

$$\frac{\partial p}{\partial y_k}(y, z) = \delta^{n+1} \sum_{s \in \{+1, -1\}^{n-2}} \left( \prod_{j=3}^n \lambda_{s_j \cdot j} \right) \frac{\partial f^{s \cdot \pi}}{\partial x_k}(y_1, y_2).$$

Before moving to the proof of the technical lemma, let us see why it implies the Backpropagation Lemma. From the two bullets, we obtain by definition of the generalized circuit gradient that

$$\nabla f^{s \cdot \pi}(y_1, y_2) = \left( \frac{\partial f^{s \cdot \pi}}{\partial x_1}(y_1, y_2), \frac{\partial f^{s \cdot \pi}}{\partial x_2}(y_1, y_2) \right) \in \tilde{\partial}_{\delta'} C(y_1, y_2)$$

for all  $s \in \{+1, -1\}^{n-2}$ , and where we let  $\delta' := 8K^2\delta$ . As a result of the last part of the technical lemma, we can write

$$\frac{1}{\delta^{n+1}} \cdot \left( \frac{\partial p}{\partial y_1}(y, z), \frac{\partial p}{\partial y_2}(y, z) \right) = \sum_{s \in \{+1, -1\}^{n-2}} \left( \prod_{j=3}^n \lambda_{s_j \cdot j} \right) \nabla f^{s \cdot \pi}(y_1, y_2) \in \tilde{\partial}_{\delta'} C(y_1, y_2),$$

since this is a convex combination of elements in  $\tilde{\partial}_{\delta'} C(y_1, y_2)$ , and this set is convex by definition. This is exactly the statement of the Backpropagation Lemma.

**3.3.1 Proof of the Technical Lemma.** Let  $(y, z)$  be a KKT point of (3). For  $i \in [n] \setminus \{1\}$ , let  $\varepsilon_i := (2K\delta)^{n-i}$ .

**Construction of  $\pi$ .** Let  $i \in [n] \setminus \{1, 2\}$ . We construct  $\pi_i$  as follows:

- If  $|\sum_{j=1}^{i-1} a_{ij}y_j + c_i - 1| \leq 2K\varepsilon_{i-1}$ , then, we set  $\pi_i := -4K\varepsilon_{i-1}$ .
- If  $|\sum_{j=1}^{i-1} a_{ij}y_j + c_i - 0| \leq 2K\varepsilon_{i-1}$ , then, we set  $\pi_i := 4K\varepsilon_{i-1}$ .
- In all other cases, we set  $\pi_i := 0$ .

Note that the two first cases cannot both occur, since  $2K\varepsilon_{i-1} \leq 2K\varepsilon_{n-1} = 4K^2\delta < 1/4$ , since  $\delta < 1/16K^2$ . Furthermore, since  $2K\varepsilon_{i-1} < 1/4$ , we also have that  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i + \pi_i \notin (-2K\varepsilon_{i-1}, 2K\varepsilon_{i-1}) \cup (1-2K\varepsilon_{i-1}, 1+2K\varepsilon_{i-1})$ , and similarly  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i - \pi_i \notin (-2K\varepsilon_{i-1}, 2K\varepsilon_{i-1}) \cup (1-2K\varepsilon_{i-1}, 1+2K\varepsilon_{i-1})$ .

**Construction of  $\lambda$ .** Let  $i \in [n] \setminus \{1, 2\}$ . We construct  $\lambda_i \in [0, 1]$  as follows:

- If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i < -2K\varepsilon_{i-1}$ , then set  $\lambda_i := 0$ .
- If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i > 1 + 2K\varepsilon_{i-1}$ , then set  $\lambda_i := 0$ .
- If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \in (2K\varepsilon_{i-1}, 1 - 2K\varepsilon_{i-1})$ , then set  $\lambda_i := 1$ .

– Otherwise, pick  $\lambda_i \in [0, 1]$  as a solution of the equation

$$\frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \cdot \lambda_i = T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) - T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i - \frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \right). \quad (6)$$

Note that such  $\lambda_i \in [0, 1]$  always exists.

In fact, it is not hard to see that  $\lambda_i$  satisfies (6) in all four cases. This follows from the fact that by the proof of Lemma 3.4

$$\left| \frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \right| \leq (2K\delta)^{n+1-i} = \varepsilon_{i-1} \leq K\varepsilon_{i-1}.$$

Furthermore, note that by Lemma 3.4, we can rewrite the equation satisfied by  $\lambda_i$  as

$$\frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \cdot \lambda_i = T \left( \sum_{j=1}^{i-1} a_{ij} y_j + c_i \right) - y_i. \quad (7)$$

Before stating and proving an important claim satisfied by  $\pi$  and  $\lambda$ , we introduce some additional notation. For  $i \in [n] \setminus \{1, 2\}$  and any  $s_i \in \{+1, -1\}$ , we define the function  $\phi_i^{s_i \cdot \pi_i} : \mathbb{R}^{i-1} \rightarrow \mathbb{R}$  by  $\phi_i^{s_i \cdot \pi_i}(x_1, \dots, x_{i-1}) = T(\sum_{j=1}^{i-1} a_{ij} x_j + c_i + s_i \cdot \pi_i)$ . Recall that we use  $\lambda_{-i}$  to denote  $1 - \lambda_i$ .

**CLAIM 1.** For  $i \in [n] \setminus \{1, 2\}$  and for any  $s_i \in \{+1, -1\}$ , the function  $\phi_i^{s_i \cdot \pi_i}$  is differentiable (with respect to its inputs  $x_1, \dots, x_{i-1}$ ) over  $\prod_{j \in [i-1]} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$  and we have

$$\frac{\partial \phi_i^{s_i \cdot \pi_i}}{\partial x_k}(v_1, \dots, v_{i-1}) = \frac{\partial \phi_i^{s_i \cdot \pi_i}}{\partial x_k}(y_1, \dots, y_{i-1})$$

for all  $k \in [i-1]$  and all  $v \in \prod_{j \in [i-1]} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ . Furthermore, we also have

$$\lambda_i \cdot \frac{\partial \phi_i^{\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) + \lambda_{-i} \cdot \frac{\partial \phi_i^{-\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = \lambda_i \cdot a_{ik}$$

for all  $k \in [i-1]$ .

**PROOF.** In order to show that  $\phi_i^{s_i \cdot \pi_i}$  is differentiable over  $\prod_{j \in [i-1]} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ , we will show that  $\sum_{j=1}^{i-1} a_{ij} v_j + c_i + s_i \cdot \pi_i$  is far from both 0 and 1 for all  $v \in \prod_{j \in [i-1]} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ . Indeed,  $\pi_i$  has been constructed in order to ensure this. Namely, we have

$$\left| \sum_{j=1}^{i-1} a_{ij} v_j + c_i + s_i \cdot \pi_i - 1 \right| \geq \left| \sum_{j=1}^{i-1} a_{ij} y_j + c_i + s_i \cdot \pi_i - 1 \right| - K\varepsilon_{i-1} \geq K\varepsilon_{i-1},$$

where we used the fact that  $\sum_{j=1}^{i-1} a_{ij} y_j + c_i + s_i \cdot \pi_i \notin (1 - 2K\varepsilon_{i-1}, 1 + 2K\varepsilon_{i-1})$ . A very similar argument shows that  $\sum_{j=1}^{i-1} a_{ij} v_j + c_i + s_i \cdot \pi_i$  is also far from 0. As a result, the first part of the claim follows, since  $\phi_i^{s_i \cdot \pi_i}$  restricted to  $\prod_{j \in [i-1]} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$  is a linear affine function.

We prove the second part of the claim by a case analysis.

– If  $\pi_i = 0$ , then  $\phi_i^{\pi_i} = \phi_i^{-\pi_i} = \phi_i^0$ . Thus, since  $\lambda_i + \lambda_{-i} = 1$ , we can write

$$\lambda_i \cdot \frac{\partial \phi_i^{\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) + \lambda_{-i} \cdot \frac{\partial \phi_i^{-\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = \frac{\partial \phi_i^0}{\partial x_k}(y_1, \dots, y_{i-1}).$$

Now, we have three subcases:

– If  $\sum_{j=1}^{i-1} a_{ij} y_j + c_i < -2K\varepsilon_{i-1}$ , then  $\frac{\partial \phi_i^0}{\partial x_k}(y_1, \dots, y_{i-1}) = 0$  and  $\lambda_i = 0$ . So, the desired equation holds.

- If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i > 1 + 2K\varepsilon_{i-1}$ , then  $\frac{\partial\phi_i^0}{\partial x_k}(y_1, \dots, y_{i-1}) = 0$  and  $\lambda_i = 0$ . So, the equation holds.
- If  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \in (2K\varepsilon_{i-1}, 1 - 2K\varepsilon_{i-1})$ , then  $\frac{\partial\phi_i^0}{\partial x_k}(y_1, \dots, y_{i-1}) = a_{ik}$  and  $\lambda_i = 1$ . So, the equation holds.
- If  $\pi_i = -4K\varepsilon_{i-1}$ , then, by construction of  $\pi_i$ , we must have  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \in [1 - 2K\varepsilon_{i-1}, 1 + 2K\varepsilon_{i-1}]$ . But this implies that  $\frac{\partial\phi_i^{\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = a_{ik}$  and  $\frac{\partial\phi_i^{-\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = 0$ . As a result, the desired equation again holds.
- If  $\pi_i = 4K\varepsilon_{i-1}$ , then, by construction of  $\pi_i$ , we must have  $\sum_{j=1}^{i-1} a_{ij}y_j + c_i \in [-2K\varepsilon_{i-1}, 2K\varepsilon_{i-1}]$ . But this implies that  $\frac{\partial\phi_i^{\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = a_{ik}$  and  $\frac{\partial\phi_i^{-\pi_i}}{\partial x_k}(y_1, \dots, y_{i-1}) = 0$ . So, the desired equation holds.

□

We are now ready to prove the technical lemma. We will prove a slightly stronger statement by induction. For this, we need some additional notation. For  $i \in [n] \setminus \{1\}$ , we let  $C_i^\pi$  denote the circuit  $C^\pi$  but where we have only kept the gates  $i+1, \dots, n$ . We think of  $C_i^\pi$  as having input variables  $x_1, x_2, \dots, x_i$  (even though some of those variables might be unused and thus not have any impact on the output of the circuit). We let  $f_i^\pi : \mathbb{R}^i \rightarrow \mathbb{R}$  denote the function represented by  $C_i^\pi$ . Note that  $f_2^\pi = f^\pi$  and  $f_n^\pi(x_1, \dots, x_n) = x_n$ .

**CLAIM 2.** *For any  $i \in [n] \setminus \{1\}$  we have*

- (1) *For any  $s \in \{+1, -1\}^{n-2}$ , the function  $f_i^{s, \pi}$  is differentiable (with respect to its inputs  $x_1, x_2, \dots, x_i$ ) over  $\prod_{j \in [i]} [y_j - \varepsilon_i, y_j + \varepsilon_i]$  and we have*

$$\frac{\partial f_i^{s, \pi}}{\partial x_k}(v_1, \dots, v_i) = \frac{\partial f_i^{s, \pi}}{\partial x_k}(y_1, \dots, y_i)$$

*for all  $k \in [i]$  and all  $v \in \prod_{j \in [i]} [y_j - \varepsilon_i, y_j + \varepsilon_i]$ .*

- (2) *For any  $k \in [i]$ , we have*

$$\frac{\partial p_i}{\partial y_k}(y, z) = \delta^{n+1} \sum_{s: s_j=1 \forall j \leq i} \left( \prod_{j=i+1}^n \lambda_{s_j, j} \right) \frac{\partial f_i^{s, \pi}}{\partial x_k}(y_1, \dots, y_i).$$

The technical lemma (Lemma 3.5) simply follows from the claim by noting that for  $i = 2$ , we have  $f_2^{s, \pi} = f^{s, \pi}$  and  $p_2 = p$ . Furthermore, note that for all  $i \in [n] \setminus \{1, 2\}$ , we have  $|\pi_i| \leq 4K\varepsilon_{i-1} \leq 4K\varepsilon_{n-1} = 8K^2\delta$ , as desired. It remains to prove the claim.

**PROOF.** We prove the claim by induction on  $i$ .

**Base case:**  $i = n$ . Recall that  $f_n^{s, \pi}(v_1, \dots, v_n) = v_n$  and  $p_n(y, z) = \delta^{n+1}y_n$ . Clearly,  $f_n^{s, \pi}$  is differentiable with  $\frac{\partial f_n^{s, \pi}}{\partial x_n}(v_1, \dots, v_n) = 1$  and  $\frac{\partial f_n^{s, \pi}}{\partial x_k}(v_1, \dots, v_n) = 0$  for all  $k \in [n-1]$  and for all  $v \in \mathbb{R}^n$ . Furthermore,  $\frac{\partial p_n}{\partial y_n}(y, z) = \delta^{n+1}$  and  $\frac{\partial p_n}{\partial y_k}(y, z) = 0$  for all  $k \in [n-1]$ . Thus, for all  $k \in [n]$  we have

$$\frac{\partial p_n}{\partial y_k}(y, z) = \delta^{n+1} \sum_{s: s_j=1 \forall j \leq n} \left( \prod_{j=n+1}^n \lambda_{s_j, j} \right) \frac{\partial f_n^{s, \pi}}{\partial x_k}(y_1, \dots, y_n),$$

where we note that  $\prod_{j=n+1}^n \lambda_{s_j, j} = 1$ , because it is an empty product, and that the sum has a single summand.

**Induction step.** Let  $i \in [n] \setminus \{1, 2\}$ , and assume that the induction hypothesis holds for  $i$ . We show that it also holds for  $i - 1$ . Fix any  $k \in [i - 1]$ . For any  $s \in \{+1, -1\}^{n-2}$ , by the definition of  $f_{i-1}^{s \cdot \pi}$ , we have that for all  $v \in \mathbb{R}^{i-1}$

$$f_{i-1}^{s \cdot \pi}(v_1, \dots, v_{i-1}) = f_i^{s \cdot \pi}(v_1, \dots, v_{i-1}, \phi_i^{s_i \cdot \pi_i}(v_1, \dots, v_{i-1})),$$

where we recall that  $\phi_i^{s_i \cdot \pi_i}(v_1, \dots, v_{i-1}) = T(\sum_{j=1}^{i-1} a_{ij}v_j + c_i + s_i \cdot \pi_i)$ . Now, for any  $v \in \prod_{j=1}^{i-1} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ , we have that

$$\begin{aligned} |y_i - \phi_i^{s_i \cdot \pi_i}(v_1, \dots, v_{i-1})| &= \left| y_i - T\left(\sum_{j=1}^{i-1} a_{ij}v_j + c_i + s_i \cdot \pi_i\right) \right| \\ &\leq \left| y_i - T\left(\sum_{j=1}^{i-1} a_{ij}y_j + c_i\right) \right| + K\varepsilon_{i-1} + |\pi_i| \\ &\leq (2K\delta)^{n+1-i} + K\varepsilon_{i-1} + 4K\varepsilon_{i-1} \\ &\leq 6K\varepsilon_{i-1} = 6K(2K\delta)\varepsilon_i \leq \varepsilon_i, \end{aligned}$$

where we used Lemma 3.4,  $(2K\delta)^{n+1-i} = \varepsilon_{i-1}$ ,  $\varepsilon_{i-1} = (2K\delta)\varepsilon_i$  and  $\delta < 1/16K^2$ . As a result, we have that  $(v_1, \dots, v_{i-1}, \phi_i^{s_i \cdot \pi_i}(v_1, \dots, v_{i-1})) \in \prod_{j=1}^i [y_j - \varepsilon_i, y_j + \varepsilon_i]$ . By the induction hypothesis and by Claim 1, it follows using the chain rule that  $f_{i-1}^{s \cdot \pi}$  is differentiable on  $\prod_{j=1}^{i-1} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ , and that its partial derivative with respect to  $x_k$  satisfies

$$\begin{aligned} \frac{\partial f_{i-1}^{s \cdot \pi}}{\partial x_k}(v) &= \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(v, \phi_i^{s_i \cdot \pi_i}(v)) + \frac{\partial \phi_i^{s_i \cdot \pi_i}}{\partial x_k}(v) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(v, \phi_i^{s_i \cdot \pi_i}(v)) \\ &= \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \frac{\partial \phi_i^{s_i \cdot \pi_i}}{\partial x_k}(y_{\leq i-1}) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}) \end{aligned}$$

for all  $v \in \prod_{j=1}^{i-1} [y_j - \varepsilon_{i-1}, y_j + \varepsilon_{i-1}]$ . Here, we used  $y_{\leq i}$  to denote  $(y_1, \dots, y_i)$ .

For any sign vector  $s = (s_i)_{i \in [n] \setminus \{1, 2\}}$  with  $s_i = +1$ , let  $s' = (s_3, \dots, s_{i-1}, -s_i, s_{i+1}, \dots, s_n)$ . We can write

$$\begin{aligned} \frac{\partial f_{i-1}^{s' \cdot \pi}}{\partial x_k}(y_{\leq i-1}) &= \frac{\partial f_i^{s' \cdot \pi}}{\partial x_k}(y_{\leq i}) + \frac{\partial \phi_i^{s'_i \cdot \pi_i}}{\partial x_k}(y_{\leq i-1}) \cdot \frac{\partial f_i^{s' \cdot \pi}}{\partial x_i}(y_{\leq i}) \\ &= \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \frac{\partial \phi_i^{-s_i \cdot \pi_i}}{\partial x_k}(y_{\leq i-1}) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}), \end{aligned}$$

where we used the fact that  $f_i^{s' \cdot \pi} = f_i^{s \cdot \pi}$ , since the  $i$ th gate is not included in the corresponding arithmetic circuits. As a result,

$$\begin{aligned} &\lambda_i \frac{\partial f_{i-1}^{s \cdot \pi}}{\partial x_k}(y_{\leq i-1}) + \lambda_{-i} \frac{\partial f_{i-1}^{s \cdot \pi}}{\partial x_k}(y_{\leq i-1}) \\ &= \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \left( \lambda_i \frac{\partial \phi_i^{\pi_i}}{\partial x_k}(y_{\leq i-1}) + \lambda_{-i} \frac{\partial \phi_i^{-\pi_i}}{\partial x_k}(y_{\leq i-1}) \right) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}) \\ &= \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \lambda_i \cdot a_{ik} \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}), \end{aligned} \tag{8}$$

where we used  $\lambda_{s_i \cdot i} + \lambda_{-s_i \cdot i} = 1$  and Claim 1. Thus, we can now write

$$\begin{aligned}
& \sum_{s: s_j=1 \forall j \leq i-1} \left( \prod_{j=i}^n \lambda_{s_j \cdot j} \right) \frac{\partial f_{i-1}^{s \cdot \pi}}{\partial x_k}(y_{\leq i-1}) \\
&= \sum_{s: s_j=1 \forall j \leq i} \left( \prod_{j=i+1}^n \lambda_{s_j \cdot j} \right) \cdot \left( \lambda_i \frac{\partial f_{i-1}^{s \cdot \pi}}{\partial x_k}(y_{\leq i-1}) + \lambda_{-i} \frac{\partial f_{i-1}^{s' \cdot \pi}}{\partial x_k}(y_{\leq i-1}) \right) \\
&= \sum_{s: s_j=1 \forall j \leq i} \left( \prod_{j=i+1}^n \lambda_{s_j \cdot j} \right) \cdot \left( \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \lambda_i \cdot a_{ik} \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}) \right) \\
&= \sum_{s: s_j=1 \forall j \leq i} \left( \prod_{j=i+1}^n \lambda_{s_j \cdot j} \right) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_k}(y_{\leq i}) + \lambda_i \cdot a_{ik} \cdot \sum_{s: s_j=1 \forall j \leq i} \left( \prod_{j=i+1}^n \lambda_{s_j \cdot j} \right) \cdot \frac{\partial f_i^{s \cdot \pi}}{\partial x_i}(y_{\leq i}) \\
&= \frac{1}{\delta^{n+1}} \cdot \frac{\partial p_i}{\partial y_k}(y, z) + \lambda_i \cdot a_{ik} \frac{1}{\delta^{n+1}} \frac{\partial p_i}{\partial y_i}(y, z),
\end{aligned}$$

where we used (8) and the induction hypothesis for  $i$ . Thus, it remains to show that

$$\frac{\partial p_i}{\partial y_k}(y, z) + \lambda_i \cdot a_{ik} \frac{\partial p_i}{\partial y_i}(y, z) = \frac{\partial p_{i-1}}{\partial y_k}(y, z)$$

to establish that the induction hypothesis also holds for  $i - 1$ . By definition of  $p_{i-1}$ , we have that

$$\frac{\partial p_{i-1}}{\partial y_k}(y, z) = \frac{\partial p_i}{\partial y_k}(y, z) + \delta^i \frac{\partial q_i}{\partial y_k}(y, z),$$

which means that it suffices to prove that

$$\frac{\partial q_i}{\partial y_k}(y, z) = \frac{1}{\delta^i} \cdot \lambda_i \cdot a_{ik} \frac{\partial p_i}{\partial y_i}(y, z).$$

We have

$$\frac{\partial q_i}{\partial y_k}(y, z) = -2a_{ik} \left( y_i + Kz_i^+ - Kz_i^- - \sum_{j=1}^{i-1} a_{ij}y_j - c_i \right) = -2a_{ik} \left( y_i - T \left( \sum_{j=1}^{i-1} a_{ij}y_j + c_i \right) \right),$$

where we also used Lemma 3.3. We can rewrite this as

$$\frac{\partial q_i}{\partial y_k}(y, z) = \frac{1}{\delta^i} \cdot \lambda_i \cdot a_{ik} \frac{\partial p_i}{\partial y_i}(y, z)$$

by using the fact that  $\lambda_i$  satisfies (7), that is,

$$y_i - T \left( \sum_{j=1}^{i-1} a_{ij}y_j + c_i \right) = -\frac{1}{2\delta^i} \cdot \frac{\partial p_i}{\partial y_i}(y, z) \cdot \lambda_i.$$

Thus, the induction hypothesis also holds for  $i - 1$  and the proof is complete.  $\square$

#### 4 CLS-hardness of 2D-LINEAR-KKT

In this section, we prove the following lower bound.

**PROPOSITION 4.1.** *The 2D-LINEAR-KKT problem is CLS-hard.*

The section is structured as follows. We begin with the formal description of the mesa functions (Section 4.1), and then prove some key properties about these functions (Section 4.2). Finally, we use these to provide a proof of Proposition 4.1. One key lemma (Lemma 4.5) in that proof, which

pertains to how the mesa functions can be implemented by circuits in a robust way, is deferred to Section 5.

#### 4.1 The Mesa Functions

Fix some side length  $\ell > 0$ , and a boundary slope  $\Gamma > 0$ . A mesa function is defined by a center-point  $p \in \mathbb{R}^2$ , a value  $a \in \mathbb{R}$ , and a gradient  $g \in \mathbb{R}^2$ . Formally, we define the mesa function  $M(\cdot; p, a, g) : \mathbb{R}^2 \rightarrow \mathbb{R}$  as the minimum of five linear (affine) functions, namely, for all  $x \in \mathbb{R}^2$

$$M(x; p, a, g) = \min \{P_c(x; p, a, g), P_r(x; p, a, g), P_t(x; p, a, g), P_l(x; p, a, g), P_b(x; p, a, g)\},$$

where these linear functions are defined as follows:

- (center) The linear function with gradient  $g = (g_1, g_2)$  that has value  $a$  at point  $p$ . Formally, this is the function

$$P_c(x; p, a, g) = (x_1 - p_1)g_1 + (x_2 - p_2)g_2 + a.$$

- (right) The linear function with gradient  $(-\Gamma + g_1, g_2)$  that has value  $P_c(p_1 + \ell/2, p_2 + \ell/2) = (g_1 + g_2)\ell/2 + a$  at point  $(p_1 + \ell/2, p_2 + \ell/2)$ . Formally, this is the function

$$\begin{aligned} P_r(x; p, a, g) &= (x_1 - p_1 - \ell/2)(-\Gamma + g_1) + (x_2 - p_2 - \ell/2)g_2 + (g_1 + g_2)\ell/2 + a \\ &= (x_1 - p_1)(-\Gamma + g_1) + (x_2 - p_2)g_2 + a + \Gamma\ell/2. \end{aligned}$$

- (top) The linear function with gradient  $(g_1, -\Gamma + g_2)$  that has value  $P_c(p_1 + \ell/2, p_2 + \ell/2) = (g_1 + g_2)\ell/2 + a$  at point  $(p_1 + \ell/2, p_2 + \ell/2)$ . Formally, this is the function

$$\begin{aligned} P_t(x; p, a, g) &= (x_1 - p_1 - \ell/2)g_1 + (x_2 - p_2 - \ell/2)(-\Gamma + g_2) + (g_1 + g_2)\ell/2 + a \\ &= (x_1 - p_1)g_1 + (x_2 - p_2)(-\Gamma + g_2) + a + \Gamma\ell/2. \end{aligned}$$

- (left) The linear function with gradient  $(\Gamma + g_1, g_2)$  that has value  $P_c(p_1 - \ell/2, p_2 - \ell/2) = -(g_1 + g_2)\ell/2 + a$  at point  $(p_1 - \ell/2, p_2 - \ell/2)$ . Formally, this is the function

$$\begin{aligned} P_l(x; p, a, g) &= (x_1 - p_1 + \ell/2)(\Gamma + g_1) + (x_2 - p_2 + \ell/2)g_2 - (g_1 + g_2)\ell/2 + a \\ &= (x_1 - p_1)(\Gamma + g_1) + (x_2 - p_2)g_2 + a + \Gamma\ell/2. \end{aligned}$$

- (bottom) The linear function with gradient  $(g_1, \Gamma + g_2)$  that has value  $P_c(p_1 - \ell/2, p_2 - \ell/2) = -(g_1 + g_2)\ell/2 + a$  at point  $(p_1 - \ell/2, p_2 - \ell/2)$ . Formally, this is the function

$$\begin{aligned} P_b(x; p, a, g) &= (x_1 - p_1 + \ell/2)g_1 + (x_2 - p_2 + \ell/2)(\Gamma + g_2) - (g_1 + g_2)\ell/2 + a \\ &= (x_1 - p_1)g_1 + (x_2 - p_2)(\Gamma + g_2) + a + \Gamma\ell/2. \end{aligned}$$

*Remark 1.* It will be convenient to abuse notation and to also define the function  $M(\cdot; p, A, g)$ , where  $A$  is a vector of values, namely,  $A = (a_c, a_r, a_t, a_l, a_b) \in \mathbb{R}^5$ . In that case, we let

$$M(x; p, A, g) = \min \{P_c(x; p, a_c, g), P_r(x; p, a_r, g), P_t(x; p, a_t, g), P_l(x; p, a_l, g), P_b(x; p, a_b, g)\}$$

that is, the values  $a$  used by the different linear pieces are not (necessarily) the same and are given by the vector  $A$ . This will be useful to analyze the effect of perturbations.

#### 4.2 Key Properties of the Mesa Construction

LEMMA 4.2. *For any  $\ell > 0$ ,  $\Gamma > 0$ ,  $p \in [0, 1]^2$ ,  $g \in [-1, 1]^2$ , and  $A = (a_c, a_r, a_t, a_l, a_b) \in [0, 1]^5$ , we have that for all  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \geq \ell/2 + 3/\Gamma$*

$$M(x; p, A, g) \leq 0.$$

For example, if  $\Gamma \geq 12/\ell$ , then,  $M(x; p, A, g) \leq 0$  for all  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \geq 3\ell/4$ .

PROOF. First, note that for any  $x \in [0, 1]^2$ , we have

$$\begin{aligned} P_r(x; p, a_r, g) &= (x_1 - p_1)(-\Gamma + g_1) + (x_2 - p_2)g_2 + a_r + \Gamma\ell/2 \\ &= -(x_1 - p_1 - \ell/2)\Gamma + (x_1 - p_1)g_1 + (x_2 - p_2)g_2 + a_r \\ &\leq -(x_1 - p_1 - \ell/2)\Gamma + 1 + 1 + 1 \end{aligned}$$

and thus  $P_r(x; p, a_r, g) \leq 0$  whenever  $x_1 \geq p_1 + \ell/2 + 3/\Gamma$ . Similarly, one can also show that for all  $x \in [0, 1]^2$

- $P_t(x; p, a_t, g) \leq 0$  whenever  $x_2 \geq p_2 + \ell/2 + 3/\Gamma$ ,
- $P_l(x; p, a_l, g) \leq 0$  whenever  $x_1 \leq p_1 - \ell/2 - 3/\Gamma$ ,
- $P_b(x; p, a_b, g) \leq 0$  whenever  $x_2 \leq p_2 - \ell/2 - 3/\Gamma$ .

Now, since the mesa function is the minimum of these four linear functions (and of the piece  $P_c$ ), it follows that  $M(x; p, A, g) \leq 0$  for all  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \geq \ell/2 + 3/\Gamma$ , as claimed.  $\square$

LEMMA 4.3. *Let the following be given:  $\ell = 1/N$  for some  $N \in \mathbb{N}$ ,  $\Gamma > 0$ , and for all  $p \in G$ ,  $A^p = (a_c^p, a_r^p, a_t^p, a_l^p, a_b^p) \in [0.4, 0.6]^5$  and  $g^p \in [-0.01, 0.01]^2$ , where  $G = \{0, \ell, 2\ell, \dots, 1 - \ell, 1\}^2$ . Then, we have that for all  $x \in [0, 1]^2$*

$$\max_{p \in G} M(x; p, A^p, g^p) \in [1/3, 2/3].$$

PROOF. We begin by proving the upper bound, namely,  $\max_{p \in G} M(x; p, A^p, g^p) \leq 2/3$ . We will show that for all  $p \in G$  and all  $x \in [0, 1]^2$ ,  $M(x; p, A^p, g^p) \leq 2/3$ . By construction of the mesa function, it suffices to show that  $P_c(x; p, a_c^p, g^p) \leq 2/3$ . For any  $x \in [0, 1]^2$ , we have

$$P_c(x; p, a_c^p, g^p) = (x_1 - p_1)g_1^p + (x_2 - p_2)g_2^p + a_c^p \leq |g_1^p| + |g_2^p| + a_c^p \leq 0.62 \leq 2/3.$$

For the lower bound, fix any  $x \in [0, 1]^2$  and let  $p \in G$  be such that  $\|x - p\|_\infty \leq \ell/2$ . (Such a  $p$  must necessarily exist.) It suffices to show that  $M(x; p, A^p, g^p) \geq 1/3$ , since we take the maximum over all mesas. Let  $a^* := \min\{a_c^p, a_r^p, a_t^p, a_l^p, a_b^p\}$ . Then, we must have  $M(x; p, A^p, g^p) \geq M(x; p, a^*, g^p)$ , so now it suffices to show that  $M(x; p, a^*, g^p) \geq 1/3$ . By construction, we have that since  $\|x - p\|_\infty \leq \ell/2$ ,  $M(x; p, a^*, g^p) = P_c(x; p, a^*, g^p)$  and then

$$P_c(x; p, a^*, g^p) = (x_1 - p_1)g_1^p + (x_2 - p_2)g_2^p + a^* \geq -|g_1^p| - |g_2^p| + a^* \geq 0.38 \geq 1/3,$$

which proves the lemma.  $\square$

LEMMA 4.4. *Let the following be given:  $\ell = 1/N$  for some  $N \in \mathbb{N}$ ,  $\Gamma \geq 6/\ell$ , and for all  $p \in G$ ,  $A^p = (a_c^p, a_r^p, a_t^p, a_l^p, a_b^p) \in [0.4, 0.6]^5$  and  $g^p \in [-0.01, 0.01]^2$ , where  $G = \{0, \ell, 2\ell, \dots, 1 - \ell, 1\}^2$ . Furthermore, assume that for all  $p, p' \in G$  that are adjacent (i.e.,  $\|p - p'\|_\infty \leq \ell$ ) we have*

- (1)  $\|g^p - g^{p'}\|_\infty \leq \ell$ ,
- (2) for all  $i, j \in \{c, r, t, l, b\}$ ,  $|a_i^{p'} - a_j^p - \langle 2g^p, p' - p \rangle| \leq \ell^2$ .

Define the function  $f : [0, 1]^2 \rightarrow \mathbb{R}$

$$f(x) := \max_{p \in G} M(x; p, A^p, g^p).$$

Then, for any  $p \in G$  and any  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \leq \ell/2$  such that  $f$  is differentiable at  $x$ , we have

- If  $g_1^p \geq 10\ell$ , then  $\frac{\partial f}{\partial x_1}(x) \geq g_1^p - \ell$ .
- If  $g_1^p \leq -10\ell$ , then  $\frac{\partial f}{\partial x_1}(x) \leq g_1^p + \ell$ .
- If  $g_2^p \geq 10\ell$ , then  $\frac{\partial f}{\partial x_2}(x) \geq g_2^p - \ell$ .
- If  $g_2^p \leq -10\ell$ , then  $\frac{\partial f}{\partial x_2}(x) \leq g_2^p + \ell$ .

PROOF. Let  $p^* \in G$  and  $x^* \in [0, 1]^2$  with  $\|x^* - p^*\|_\infty \leq \ell/2$  such that  $f$  is differentiable at  $x^*$ . We only consider the case where  $g_1^{p^*} \geq 10\ell$ . The other cases are handled analogously.

First, observe that for all  $x \in [0, 1]^2$  with  $\|x - p^*\|_\infty \leq \ell$  we have

$$f(x) = \max_{p \in G_{p^*}} M(x; p, A^p, g^p),$$

where  $G_{p^*} := \{p \in G : \|p - p^*\|_\infty \leq \ell\}$  is the set of grid points adjacent to  $p^*$ . Indeed, for any  $p \in G \setminus G_{p^*}$ , we have  $\|x - p\|_\infty \geq \ell \geq \ell/2 + 3/\Gamma$  and, thus, by Lemma 4.2, it must be that  $M(x; p, A^p, g^p) \leq 0$ . On the other hand, by Lemma 4.3, we have  $f(x) \in [1/3, 2/3]$ , which implies that  $f(x) > M(x; p, A^p, g^p)$  for all  $p \in G \setminus G_{p^*}$ .

Now, since  $f$  is differentiable at  $x^*$ ,  $\nabla f(x^*)$  must correspond to the gradient of one of the linear affine pieces of  $M(\cdot; p, A^p, g^p)$  for some  $p \in G_{p^*}$ . By construction of the mesa function  $M(\cdot; p, A^p, g^p)$ , its linear affine pieces  $P_c, P_t$ , and  $P_b$  all have gradient equal to  $g_1^p$  in the first coordinate. Furthermore, the linear affine piece  $P_l$  has gradient  $g_1^p + \Gamma \geq g_1^p$  in the first coordinate. Thus, as long as the piece  $P_r$  does not appear in  $f$  (i.e.,  $P_r(x; p, A^p, g^p) < f(x)$  whenever  $P_r(x; p, A^p, g^p) = M(x; p, A^p, g^p)$ ), we can deduce that  $\frac{\partial f}{\partial x_1}(x^*) \geq g_1^p$ . This implies the desired result that  $\frac{\partial f}{\partial x_1}(x^*) \geq g_1^{p^*} - \ell$ , since by assumption 1 we have  $\|g^{p^*} - g^p\|_\infty \leq \ell$ .

It remains to prove that for all  $p \in G_{p^*}$ , the piece  $P_r$  indeed does not appear in  $f$ . Since  $g_1^p \geq g_1^{p^*} - \ell \geq 9\ell$ , this is implied by the following slightly more general claim: If  $g_1^p \geq 9\ell$  for some  $p \in G$ , then for all  $x \in [0, 1]^2$  we have

$$P_r(x; p, a_r^p, g^p) = M(x; p, A^p, g^p) \implies P_r(x; p, a_r^p, g^p) < f(x).$$

In the remainder of this proof, we prove this claim.

Let  $p \in G$  with  $g_1^p \geq 9\ell$ . We begin by noting the following fact: For any  $p \in G$  and  $i, j \in \{c, r, t, l, b\}$ , we have

$$|a_i^p - a_j^p| \leq |a_i^{p'} - a_i^p - \langle 2g^p, p' - p \rangle| + |a_i^{p'} - a_j^p - \langle 2g^p, p' - p \rangle| \leq 2\ell^2 \quad (9)$$

by assumption 2, where  $p'$  is some arbitrary gridpoint adjacent to  $p$ .

Furthermore, note that since  $\Gamma \geq 6/\ell$ , Lemma 4.2 implies that  $M(x; p, A^p, g^p) \leq 0$  for all  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \geq \ell$ . On the other hand, by Lemma 4.3, we have  $f(x) \geq 1/3$  for all  $x \in [0, 1]^2$ . As a result, for any  $x \in [0, 1]^2$  with  $\|x - p\|_\infty \geq \ell$ , if  $P_r(x; p, a_r^p, g^p) = M(x; p, A^p, g^p)$ , then  $P_r(x; p, a_r^p, g^p) \leq 0 < 1/3 \leq f(x)$ , as desired. Thus, it remains to prove the claim for all  $x \in ([p_1 - \ell, p_1 + \ell] \times [p_2 - \ell, p_2 + \ell]) \cap [0, 1]^2 =: B_\ell(p) \cap [0, 1]^2$ .

First of all, note that if  $p$  lies on the right-hand boundary of  $G$ , that is,  $p_1 = 1$ , then  $P_r(x; p, a_r^p, g^p) > P_c(x; p, a_c^p, g^p) \geq M(x; p, A^p, g^p)$  for all  $x \in [0, 1]^2$ , and thus the claim holds trivially. Indeed

$$\begin{aligned} & P_r(x; p, a_r^p, g^p) - P_c(x; p, a_c^p, g^p) \\ &= (x_1 - p_1)(-\Gamma + g_1^p) + (x_2 - p_2)g_2^p + a_r^p + \Gamma\ell/2 - (x_1 - p_1)g_1^p - (x_2 - p_2)g_2^p - a_c^p \\ &= -(x_1 - p_1)\Gamma + \Gamma\ell/2 + a_r^p - a_c^p \\ &\geq \Gamma\ell/2 - 2\ell^2 = \ell(\Gamma/2 - 2\ell) > 0, \end{aligned}$$

where we used  $x_1 \leq 1 = p_1$ ,  $a_r^p - a_c^p \geq -2\ell^2$  by (9), and  $\Gamma \geq 6/\ell \geq 6 > 4\ell$ .

Now, consider the case where  $p$  does not lie on the right-hand boundary of  $G$ . Then, the point  $p' := (p_1 + \ell, p_2)$  lies on the grid  $G$ . We will show that for all  $x \in B_\ell(p)$ ,  $P_r(x; p, a_r^p, g^p) = M(x; p, A^p, g^p) \implies P_r(x; p, a_r^p, g^p) < M(x; p', A^{p'}, g^{p'})$ , which implies the claim, since  $M(x; p', A^{p'}, g^{p'}) \leq f(x)$ . For ease of notation, we drop the  $p, p'$  superscripts, and use  $A := A^p$ ,

$A' := A^p$ ,  $a_i := a_i^p$ ,  $a'_i := a_i^{p'}$ ,  $g := g^p$ ,  $g' := g^{p'}$ . We will prove the contrapositive, namely  $P_r(x; p, a_r, g) \geq M(x; p', A', g') \implies P_r(x; p, a_r, g) > M(x; p, A, g)$  for all  $x \in B_\ell(p)$ . We proceed by a case analysis. If  $x$  is such that  $P_r(x; p, a_r, g) \geq M(x; p', A', g')$ , then at least one of the following five cases must occur.

**Case 1:**  $P_r(x; p, a_r, g) \geq P_c(x; p', a'_c, g')$ . In this case, we show that  $P_r(x; p, a_r, g) > P_c(x; p, a_c, g) \geq M(x; p, A, g)$ . Indeed, we have

$$\begin{aligned} P_r(x; p, a_r, g) - P_c(x; p, a_c, g) &\geq P_c(x; p', a'_c, g') - P_c(x; p, a_c, g) \\ &= (x_1 - p_1 - \ell)(g'_1 - g_1) - \ell g_1 + (x_2 - p_2)(g'_2 - g_2) + a'_c - a_c \\ &\geq -2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| - \ell g_1 + a'_c - a_c \\ &\geq -3\ell^2 + \ell g_1 - \ell^2 = \ell(g_1 - 4\ell) > 0, \end{aligned}$$

where we used assumptions 1 and 2, as well as  $g_1 > 4\ell$ . Indeed, note that assumption 2 implies that  $a'_c - a_c - 2\ell g_1 \geq -\ell^2$  and thus  $a'_c - a_c - \ell g_1 \geq \ell g_1 - \ell^2$ , which we used here.

**Case 2:**  $P_r(x; p, a_r, g) \geq P_t(x; p', a'_t, g')$ . In this case, we show that  $P_r(x; p, a_r, g) > P_t(x; p, a_t, g) \geq M(x; p, A, g)$ . Indeed, we have

$$\begin{aligned} P_r(x; p, a_r, g) - P_t(x; p, a_t, g) &\geq P_t(x; p', a'_t, g') - P_t(x; p, a_t, g) \\ &= (x_1 - p_1 - \ell)(g'_1 - g_1) - \ell g_1 + (x_2 - p_2)(g'_2 - g_2) + a'_t - a_t \\ &\geq -2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| - \ell g_1 + a'_t - a_t \\ &\geq \ell(g_1 - 4\ell) > 0 \end{aligned}$$

as in the previous case.

**Case 3:**  $P_r(x; p, a_r, g) \geq P_b(x; p', a'_b, g')$ . In this case, we show that  $P_r(x; p, a_r, g) > P_b(x; p, a_b, g) \geq M(x; p, A, g)$ . Indeed, we have

$$\begin{aligned} P_r(x; p, a_r, g) - P_b(x; p, a_b, g) &\geq P_b(x; p', a'_b, g') - P_b(x; p, a_b, g) \\ &= (x_1 - p_1 - \ell)(g'_1 - g_1) - \ell g_1 + (x_2 - p_2)(g'_2 - g_2) + a'_b - a_b \\ &\geq -2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| - \ell g_1 + a'_b - a_b \\ &\geq \ell(g_1 - 4\ell) > 0 \end{aligned}$$

as in the previous cases.

**Case 4:**  $P_r(x; p, a_r, g) \geq P_r(x; p', a'_r, g')$ . This case is in fact impossible, since

$$\begin{aligned} P_r(x; p', a'_r, g') - P_r(x; p, a_r, g) &= (x_1 - p_1 - \ell)(g'_1 - g_1) + \Gamma\ell - \ell g_1 + (x_2 - p_2)(g'_2 - g_2) + a'_r - a_r \\ &\geq -2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| + \Gamma\ell - \ell g_1 + a'_r - a_r \\ &\geq \ell(\Gamma + g_1 - 4\ell) \geq \ell(g_1 - 4\ell) > 0 \end{aligned}$$

as in the previous cases.

**Case 5:**  $P_r(x; p, a_r, g) \geq P_l(x; p', a'_l, g')$ . In this case, we show that  $P_r(x; p, a_r, g) > P_c(x; p, a_c, g) \geq M(x; p, A, g)$ . Indeed, assume toward a contradiction that  $P_r(x; p, a_r, g) \leq P_c(x; p, a_c, g)$ . Then,

$$0 \leq P_c(x; p, a_c, g) - P_r(x; p, a_r, g) = (x_1 - p_1)\Gamma + a_c - a_r - \Gamma\ell/2,$$

which yields  $x_1 \geq (a_r - a_c)/\Gamma + \ell/2 + p_1$ . But then, we have

$$\begin{aligned} & P_l(x; p', a'_l, g') - P_r(x; p, a_r, g) \\ &= (x_1 - p_1 - \ell)(2\Gamma + g'_1 - g_1) + \Gamma\ell - \ell g_1 + (x_2 - p_2)(g'_2 - g_2) + a'_l - a_r \\ &\geq \Gamma(\ell + 2(x_1 - p_1 - \ell)) - 2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| - \ell g_1 + a'_l - a_r \\ &\geq 2(a_r - a_c) - 2\ell|g'_1 - g_1| - \ell|g'_2 - g_2| - \ell g_1 + a'_l - a_r \\ &\geq \ell(g_1 - 8\ell) > 0, \end{aligned}$$

where we used assumptions 1 and 2, as well as  $a_r - a_c \geq -2\ell^2$  by (9) and  $g_1 > 8\ell$ . We have thus obtained a contradiction and the claim holds in this case as well.  $\square$

### 4.3 Proof of Proposition 4.1

We reduce from the problem of computing an  $\varepsilon$ -KKT point of a continuously differentiable function  $h : [0, 1]^2 \rightarrow \mathbb{R}$  with  $L$ -Lipschitz gradient  $\nabla h$ , where both  $h$  and  $\nabla h$  are given as well-behaved arithmetic circuits. This problem is known to be CLS-complete [15]. We do not need to define well-behaved arithmetic circuits here, since we will only use the fact that they can be evaluated in polynomial time (see [15]).

Without loss of generality, we can assume that  $h$  satisfies

- $h(x) \in [0.49, 0.51]$  for all  $x \in [0, 1]^2$ ,
- $\nabla h(x) \in [-0.001, 0.001]^2$  for all  $x \in [0, 1]^2$ ,
- $L \leq 1/100$ .

Indeed, this is easy to achieve by scaling  $h$  and  $\varepsilon$  by some sufficiently small factor, and by adding a constant offset to  $h$ . This does not change the set of solutions.

**Construction.** Set  $\ell := 1/(2^n - 1)$  for some sufficiently large  $n \in \mathbb{N}$  such that  $\ell \leq \varepsilon/100$ . Define the grid  $G = \{0, \ell, 2\ell, \dots, 1 - \ell, 1\}^2$ .

Since  $h$  and  $\nabla h$  are given as well-behaved arithmetic circuits, we can construct the following two Boolean circuits in polynomial time:

- $a : G \rightarrow [0.45, 0.55]$  that satisfies  $|a(p) - h(p)| \leq \ell^2/100$  for all  $p \in G$ ,
- $g : G \rightarrow [-0.01, 0.01]^2$  that satisfies  $\|2g(p) - \nabla h(p)\|_\infty \leq \ell/100$  for all  $p \in G$ .

Note that this is where we use the “half-gradient” trick, namely, we let  $g(p)$  be close to  $\nabla h(p)/2$ , instead of  $\nabla h(p)$ .

Set  $\Gamma := 12/\ell$ , and  $\delta' := \ell^2/100$ . The following lemma states that we can now construct a circuit  $C$ , which implements the corresponding mesa construction in a robust manner.

**LEMMA 4.5.** *Let the following be given:  $\delta' > 0$ ,  $\ell = 1/(2^n - 1)$  for some  $n \in \mathbb{N}$ ,  $\Gamma \geq 12/\ell$ , and Boolean circuits<sup>13</sup> computing functions:*

- $a : G \rightarrow [0.45, 0.55]$ ,
- $g : G \rightarrow [-0.01, 0.01]^2$ ,

where  $G = \{0, \ell, 2\ell, \dots, 1 - \ell, 1\}^2$ . Then, in polynomial time, we can compute  $\delta > 0$  and construct a linear arithmetic circuit  $C$ , which implements the mesa construction described by  $(\ell, \Gamma, a, g)$  in a robust manner, that is, for every  $\pi \in [-\delta, \delta]^m$  (where  $m$  is the number of gates of  $C$ ), there exists a vector  $\tau \in [-\delta', \delta']^{G \times \{c, r, t, l, b\}}$  such that for all  $x \in [0, 1]^2$

$$C^\pi(x) = \max_{p \in G} M(x; p, A^{p, \tau}, g(p)),$$

<sup>13</sup>The outputs of the two Boolean circuits are represented in binary as described in Section 5.1.

where  $A^{p,\tau}$  is the following five-tuple

$$A^{p,\tau} := (a(p) + \tau_{p,c}, a(p) + \tau_{p,r}, a(p) + \tau_{p,t}, a(p) + \tau_{p,l}, a(p) + \tau_{p,b}).$$

We will now complete the proof of Proposition 4.1 assuming this lemma. The next section will then be dedicated to proving the lemma.

By Lemma 4.5, we obtain  $\delta > 0$  and a circuit  $C$  such that for all  $x \in [0, 1]^2$  and all perturbations  $\pi \in [-\delta, \delta]^m$  we have

$$C^\pi(x) = \max_{p \in G} M(x; p, A^p, g(p)),$$

where  $A^p = (a_c^p, a_r^p, a_t^p, a_l^p, a_b^p)$  satisfies  $|a_i^p - a(p)| \leq \delta'$  for all  $i$  and  $p$ . The following claim shows that the function computed by the circuit  $C^\pi$  satisfies the conditions of Lemma 4.4.

**CLAIM 3.** *For all adjacent  $p, p' \in G$ , we have*

- (1)  $\|g(p) - g(p')\|_\infty \leq \ell$ ,
- (2) *for all  $\tau \in [-\delta', \delta']^{G \times \{c, r, t, l, b\}}$  and all  $i, j \in \{c, r, t, l, b\}$* 
  - (a)  $a_i^p \in [0.4, 0.6]$ , and
  - (b)  $|a_i^{p'} - a_j^p - \langle 2g(p), p' - p \rangle| \leq \ell^2$ .

**PROOF.** For the first point, note that by the  $L$ -Lipschitz-continuity of  $\nabla h$ , we have  $\|\nabla h(p) - \nabla h(p')\|_\infty \leq L\ell \leq \ell/100$ , since  $L \leq 1/100$ . Thus, by construction of  $g(\cdot)$  we obtain

$$\|g(p) - g(p')\|_\infty \leq \|\nabla h(p)/2 - \nabla h(p')/2\|_\infty + 2 \cdot \ell/200 \leq \ell/200 + 2 \cdot \ell/200 \leq \ell.$$

For the second point, note that  $a_i^p \in [0.4, 0.6]$ , because  $a(p) \in [0.45, 0.55]$  by construction, and  $|a_i^p - a(p)| \leq \delta' \leq \ell^2/100 \leq 0.05$ . Next, by Taylor's theorem, we also have

$$|h(p') - h(p) - \langle \nabla h(p), p' - p \rangle| \leq \frac{L}{2} \|p' - p\|_2^2 \leq \frac{1}{100} \ell^2,$$

where we used  $L \leq 1/100$  and  $\|p' - p\|_2 \leq \sqrt{2}\ell$ . Now, for all  $i, j \in \{c, r, t, l, b\}$ , we can rewrite this as

$$|a_i^{p'} - a_j^p - \langle 2g(p), p' - p \rangle| \leq \frac{1}{100} \ell^2 + 2 \cdot \ell^2/50 + \ell^2/50 \leq \ell^2,$$

where we used  $|a_j^p - h(p)| \leq |a_j^p - a(p)| + |a(p) - h(p)| \leq \delta' + \ell^2/100 \leq \ell^2/50$  (since  $\delta' \leq \ell^2/100$ ) and similarly  $|a_i^{p'} - h(p')| \leq \ell^2/50$ , as well as

$$|\langle 2g(p) - \nabla h(p), p' - p \rangle| \leq \|2g(p) - \nabla h(p)\|_2 \cdot \|p' - p\|_2 \leq 2\ell \cdot \|2g(p) - \nabla h(p)\|_\infty \leq \ell^2/50$$

by construction of  $g(\cdot)$ . □

Now, we can use Lemma 4.4 to prove the following claim, which finishes the proof of Proposition 4.1.

**CLAIM 4.** *If  $x \in [0, 1]^2$  satisfies the  $\varepsilon/3$ -KKT conditions with respect to the  $\delta$ -generalized circuit gradient of  $C$ , then  $x$  also satisfies the  $\varepsilon$ -KKT conditions with respect to the gradient of  $h$ .*

**PROOF.** We prove the contrapositive. Let  $x \in [0, 1]^2$  be a point that does not satisfy the  $\varepsilon$ -KKT conditions with respect to the gradient of  $h$ . In other words, there exists  $j \in \{1, 2\}$  such that at least one of the two following statements holds:

- $-x_j > 0$  and  $\frac{\partial h}{\partial x_j}(x) > \varepsilon$ ,
- $-x_j < 1$  and  $\frac{\partial h}{\partial x_j}(x) < -\varepsilon$ .

In the remainder of this proof, we only consider the case where  $x_1 > 0$  and  $\frac{\partial h}{\partial x_1}(x) > \varepsilon$ . The other three cases are handled analogously. Our goal is to show that  $x$  cannot satisfy the  $\varepsilon/3$ -KKT conditions with respect to the  $\delta$ -generalized circuit gradient of  $C$ . Namely, we will show that all  $u \in \tilde{\partial}_\delta C(x)$  satisfy  $u_1 > \varepsilon/3$ . By the definition of  $\delta$ -generalized circuit gradient, it suffices to show that  $\frac{\partial f^\pi}{\partial x_1}(x) > \varepsilon/3$  for all  $\pi \in [-\delta, \delta]^m$  such that  $f^\pi$  is differentiable at  $x$ , where  $f^\pi$  denotes the function computed by  $C^\pi$ .

Let  $\pi \in [-\delta, \delta]^m$  be such that  $f^\pi$  is differentiable at  $x$ . Let  $p \in G$  be such that  $\|x - p\|_\infty \leq \ell/2$ . By the  $L$ -Lipschitz-continuity of  $h$  and the construction of  $g(\cdot)$  we have

$$\|2g(p) - \nabla h(x)\|_\infty \leq \|2g(p) - \nabla h(p)\|_\infty + \|\nabla h(p) - \nabla h(x)\|_\infty \leq \ell/100 + L \cdot \ell/2 \leq \ell,$$

where we used  $L \leq 1/100$ . As a result, we have

$$g_1(p) \geq \frac{1}{2} \cdot \frac{\partial h}{\partial x_1}(x) - \ell \geq \varepsilon/2 - \ell \geq 10\ell,$$

where we used  $\ell \leq \varepsilon/100$ . As argued above, the assumptions of Lemma 4.4 are satisfied and we can thus use it to deduce that  $\frac{\partial f^\pi}{\partial x_1}(x) \geq g_1(p) - \ell \geq \varepsilon/2 - 2\ell > \varepsilon/3$ , as desired.  $\square$

## 5 Construction of the Circuit

In this section, we prove Lemma 4.5, which was crucially used in the last section to obtain Proposition 4.1. In order to prove the lemma, we have to show that we can construct arithmetic circuits that implement the mesa construction from the last section in a robust manner.

In our description of the construction of the circuit, it will be convenient to extend the set of gates we can use. Consider an arithmetic circuit  $C$  with  $n$  inputs and one output, and using gates  $+, -, c, \times c, \min, \max, T_{[a,b]}$ .<sup>14</sup> Let  $m$  denote the number of  $\min, \max, T_{[a,b]}$  gates. For any perturbation vector  $\pi = (\pi_i)_{i \in [m]} \in \mathbb{R}^m$ , we let  $C^\pi$  denote the circuit perturbed by  $\pi$ , namely, for each  $i \in [m]$ , the  $i$ th gate of type  $\min, \max, T_{[a,b]}$  is perturbed as follows:

- $x_i := T_{[a,b]}(a_i x_j + b_i x_k + c_i)$  is replaced by  $x_i := T_{[a,b]}(a_i x_j + b_i x_k + c_i + \pi_i)$ .
- $x_i := \min\{x_j, x_k\}$  is replaced by  $x_i := \min\{x_j, x_k + \pi_i\}$ .
- $x_i := \max\{x_j, x_k\}$  is replaced by  $x_i := \max\{x_j, x_k + \pi_i\}$ .

Our goal now is to prove the following lemma.

**LEMMA 5.1.** *Let the following be given:  $\delta' > 0$ ,  $\ell = 1/(2^n - 1)$  for some  $n \in \mathbb{N}$ ,  $\Gamma \geq 12/\ell$ , and Boolean circuits computing functions:*

- $a : G \rightarrow [0.45, 0.55]$ ,
- $g : G \rightarrow [-0.01, 0.01]^2$ ,

where  $G = \{0, \ell, 2\ell, \dots, 1 - \ell, 1\}^2$ . Then, in polynomial time, we can compute  $\delta > 0$  and construct a linear arithmetic circuit  $C$  (using gates  $+, -, c, \times c, \min, \max, T_{[a,b]}$ ), which implements the mesa construction described by  $(\ell, \Gamma, a, g)$  in a robust manner, that is, for every  $\pi \in [-\delta, \delta]^m$  (where  $m$  is the number of  $\min, \max, T_{[a,b]}$  gates of  $C$ ), there exists a vector  $\tau \in [-\delta', \delta']^{G \times \{c, r, t, l, b\}}$  such that for all  $x \in [0, 1]^2$

$$C^\pi(x) = \max_{p \in G} M(x; p, A^{p,\tau}, g(p)),$$

where  $A^{p,\tau}$  is the following five-tuple

$$A^{p,\tau} := (a(p) + \tau_{p,c}, a(p) + \tau_{p,r}, a(p) + \tau_{p,t}, a(p) + \tau_{p,l}, a(p) + \tau_{p,b}).$$

<sup>14</sup> $T_{[a,b]}$  denotes truncation to the interval  $[a, b]$ .

Indeed, Lemma 5.1 implies Lemma 4.5, because of the following lemma, which is proved in Appendix B.

**LEMMA 5.2.** *Let  $C$  be an arithmetic circuit with  $n$  inputs and one output, and using gates  $+, -, c, \times c, \min, \max, T_{[a,b]}$ , and such that the output gate is a  $T_{[0,1]}$  gate. Then, we can construct in polynomial time a number  $K > 0$  and an arithmetic circuit  $\bar{C}$  with the same number of inputs and outputs as  $C$ , but that only uses  $T_{[0,1]}$  gates, and such that for any  $\delta \leq 1/K$  and any  $\delta$ -perturbation  $\pi$  of  $\bar{C}$ , there exists a  $\delta K$ -perturbation  $\sigma$  of  $C$  that satisfies*

$$f^\sigma(y) = \bar{f}^\pi(y)$$

for all  $y \in [0, 1]^n$ , where  $f^\sigma$  and  $\bar{f}^\pi$  are the functions computed by  $C^\sigma$  and  $\bar{C}^\pi$ , respectively.

In the remainder of this section, we present an overview of the proof of Lemma 5.1, followed by the formal proof.

**Rescaling the grid.** Lemma 5.1 uses the grid  $G = \{0, 1/(2^n - 1), \dots, 1\}^2$ . This grid discretizes each dimension into  $2^n$  points, and has points along the boundaries at 0 and 1. Since we will use a standard binary encoding to represent grid points, it will be more technically convenient to use the grid  $\tilde{G} = \{0, 1/2^n, \dots, (2^{n-1} - 1)/2^n\}$ , which discretizes each dimension into  $2^n$  points and has points along the boundaries at 0 and  $1 - 1/2^n$ .

This can be achieved via a straightforward rescaling. From Lemma 5.1, we are given Boolean circuits  $a : G \rightarrow [0.45, 0.55]$  and  $g : G \rightarrow [-0.01, 0.01]^2$ . We then build the following Boolean circuits.

- We build the circuit  $\tilde{a} : \tilde{G} \rightarrow [0.45, 0.55]$ . For each point  $x = (x_1/2^n, x_2/2^n)$ , we set

$$\tilde{a}(x) = a(x_1/(2^n - 1), x_2/(2^n - 1)).$$

That is,  $\tilde{a}$  simply looks up the corresponding point in  $x' \in G$  and outputs  $a(x')$ .

- We build the circuit  $\tilde{g} : \tilde{G} \rightarrow [-0.02, 0.02]^2$ . For each point  $x = (x_1/2^n, x_2/2^n)$ , we set

$$\tilde{g}(x) = \frac{1}{1 - 1/2^n} \cdot g(x_1/(2^n - 1), x_2/(2^n - 1)).$$

So,  $\tilde{g}$  looks up the corresponding point in  $G$  and outputs the corresponding gradient, but it also rescales that gradient to account for the smaller space.

Observe that  $\tilde{a}$  and  $\tilde{g}$  can both be constructed in polynomial time with respect to  $a$  and  $g$ .

In the rest of this section, we will build a linear circuit that outputs, for each point  $x \in [0, 1 - 1/2^n]^2$

$$\tilde{f}(x) = \max_{p \in \tilde{G}} M(x; p, \tilde{a}(p), \tilde{g}(p)).$$

We can then build the final circuit for Lemma 5.1 by undoing the rescaling in the following way:

$$f(x) = \tilde{f}((1 - 1/2^n) \cdot x).$$

Clearly  $f$  can be constructed in polynomial time once we have constructed  $\tilde{f}$ . Observe that, since we have undone the rescaling of the gradients in  $\tilde{g}$ , so we will have  $f(x) = \max_{p \in G} M(x; p, a(p), g(p))$ , as required.

**Important properties of the mesas.** We will use two properties of the mesa construction repeatedly.

- (1) From Lemma 4.3, we have that  $f(x) \in [1/3, 2/3]$  for all  $x \in [0, 1]$ . Since  $\tilde{f}$  is a simple rescaling of the domain of  $f$ , which does not change the output values, we therefore have  $\tilde{f}(x) \in [1/3, 2/3]$  for all  $x \in [0, 1 - 1/2^n]$  as well.

- (2) By Lemma 4.2 and the choice of  $\Gamma$  in the statement of Lemma 5.1, we have  $M(x; p, a(p), g(p)) \leq 0$  whenever  $\|x - y\|_\infty > 3/4 \cdot 1/(2^n - 1)$ . In the rescaled  $\tilde{f}(x)$ , we therefore have  $\|x - y\|_\infty > 3/4 \cdot 1/2^n$ , meaning that the region in which mesa takes positive values is contained within a region of radius  $3/4 \cdot 1/2^n$  around the center of the mesa.

**Overview.** The rest of this section is dedicated to building a linear circuit that outputs  $\tilde{f}$ . We begin by building linear circuits that implement some useful operations. In Section 5.1, we build linear circuits that can encode and decode values that are represented in binary, and we build a linear circuit that can simulate a Boolean circuit. In Section 5.2, we build linear circuits that, given a point  $p \in \tilde{G}$  encoded in binary, can output the five affine pieces of  $M(x; p, \tilde{a}(p), \tilde{g}(p))$ . To do this, we first build a linear circuit that can multiply a continuous variable by a binary variable, and then we use that operation to build a circuit that can output a specific affine function whose parameters are encoded in binary. Finally, in Section 5.3, we build  $\tilde{f}$  itself.

Along the way, we must consider how our circuits behave when they are evaluated with perturbations. Recall from Lemma 5.1 that the min, max, and T operations will be perturbed, and that we must show that these perturbations only additively affect the pieces of each of the mesas, without affecting their gradients.

To keep track of the effect of the perturbations, for each linear circuit  $C$ , we will use  $C^\pi$  to refer to the version of that circuit that is evaluated under the perturbation vector  $\pi$ . Along the way, we will prove lemmas that bound the deviation in the output of  $C$  as a function of the perturbation vector  $\pi$ .

## 5.1 Binary Variables

Throughout the construction, we will make use of variables that are encoded in binary. An  $n$ -bit binary variable can represent values from the set  $V := \{0, 1/2^n, 2/2^n, \dots, (2^n - 1)/2^n\}$ . For each value  $x \in V$ , we use  $x_1, x_2, \dots, x_n$  to denote the  $n$  bits used to represent that number, where  $x_1$  is the most significant bit.

We will also need to encode negative numbers in binary, and the most expedient way of doing this for our construction is to write an  $n$  bit number  $x$  using  $2n$  bits  $x_1^+$  through  $x_n^+$ , and  $x_1^-$  through  $x_n^-$ , where

- If  $x > 0$  then  $x_1^+$  through  $x_n^+$  hold a binary encoding of  $x$  and  $x_i^- = 0$  for all  $i$ .
- If  $x < 0$  then  $x_1^-$  through  $x_n^-$  hold a binary encoding of  $-x$  and  $x_i^+ = 0$  for all  $i$ .
- If  $x = 0$  then  $x_i^+ = x_i^- = 0$  for all  $i$ .

Throughout this section, we will use the binary variable  $x$  and the set of bits  $\{x_i^j\}_{1 \leq i \leq n, j \in \{-, +\}}$  interchangeably to refer to the value  $x$ .

When we use variables in a linear circuit to hold a binary variable  $x$ , it is possible that some bits used in the encoding of  $x$  may not be in  $\{0, 1\}$ . If it is the case that  $x_i^j \in \{0, 1\}$  for all  $i$  and  $j$ , then we say that  $x$  is a *correct* binary encoding.

**5.1.1 The DECODE Circuit: Decoding a Binary Variable.** The first circuit that we build is the standard *binary decoder* circuit, which takes a correct binary encoding of a variable, and outputs a continuous variable with that value. Given a binary variable  $x$  we define

$$\begin{aligned} \text{DECODE}(x) = & x_1^+ \cdot \frac{1}{2} + x_2^+ \cdot \frac{1}{4} + \cdots + x_n^+ \cdot \frac{1}{2^n} \\ & - x_1^- \cdot \frac{1}{2} - x_2^- \cdot \frac{1}{4} - \cdots - x_n^- \cdot \frac{1}{2^n}. \end{aligned}$$

This operation simply follows the definition of the binary encoding that we are using, and so if  $x$  is a correct binary variable, then the circuit will output a continuous variable holding  $x$ .

Recall that  $\text{DECODE}^\pi$  is the version of  $\text{DECODE}$  that is evaluated under the perturbation vector  $\pi$ . Since  $\text{DECODE}$  does not use min, max, or truncation, we have that  $\text{DECODE}^\pi(x) = \text{DECODE}(x)$  for all  $x$  and  $\pi$ .

**5.1.2 The EXTRACTBITS Circuit: Extracting Bits from a Continuous Variable.** Given a continuous variable  $x \in [0, 1]$ , an integer  $n \geq 1$ , and a positive rational constant  $L > 0$ , the  $\text{EXTRACTBITS}(x, n, L)$  circuit will output a binary variable  $b$  consisting of bits  $b_1^+, b_2^+, \dots, b_n^+$ , which are the first  $n$  bits of  $x$ . Since a linear circuit must compute a continuous function, it is not possible to correctly extract bits for every value of  $x \in [0, 1]$ . Thus, our circuit will fail for some inputs, and the constant  $L$  will control the values for which these failures occur.

We implement  $\text{EXTRACTBITS}$  in the following way. We fix  $x_0 = x$ , and we define

$$\begin{aligned} b_i^+ &= T_{[0,1]} \left( \left( x_{i-1} - \frac{1}{2^i} \right) \cdot \frac{1}{L} \right), \\ x_i &= x_{i-1} - \frac{b_i^+}{2^i}. \end{aligned}$$

Since we only require  $\text{EXTRACTBITS}$  to work for positive inputs, we can simply set  $b_i^- = 0$  for all  $i$ .

It is relatively straightforward to show that if  $x$  does not lie in a *bad region*  $[\frac{k}{2^n}, \frac{k}{2^n} + L]$  for any integer  $k \in \{0, 1, \dots, 2^n - 1\}$ , then this circuit will correctly extract the first  $n$  bits of  $x$ . This is because when  $x$  is not in a bad region, the value inside the truncation gate will either be less than or equal to 0, or greater than, or equal to 1, and thus the value will be truncated to either 0 or 1, meaning that the bit is correctly decoded.

On the other hand, if  $x$  does lie in a bad region, then the value inside the truncation gate may lie in the range  $(0, 1)$ , meaning that the bit  $b_i^+$  will not be correctly decoded, and all bits  $b_j^+$  with  $j > i$  may also have incorrect values.

Recall that  $\text{EXTRACTBITS}^\pi$  is the  $\text{EXTRACTBITS}$  circuit evaluated under the perturbation vector  $\pi$ . The following lemma shows that the circuit still functions correctly when evaluated with perturbations, but the bad regions are slightly expanded.

LEMMA 5.3. *If  $x$  does not lie in*

$$\left[ \frac{k}{2^n} - \frac{\max_i |\pi_i|}{L}, \quad \frac{k}{2^n} + L + \frac{\max_i |\pi_i|}{L} \right]$$

*for any integer  $k \in \{0, 1, \dots, 2^n - 1\}$ , then  $\text{EXTRACTBITS}^\pi(x, n, L)$  will correctly extract the leading  $n$  bits of  $x$ .*

PROOF. We will inductively prove for each  $i$  that each  $b_j^+$  with  $j < i$  correctly contains bit  $j$  of  $x$ , and that  $x_i = x - \sum_{j=1}^i \frac{b_j^+}{2^j}$ . The base case, when  $i = 0$ , is trivial.

For the inductive step, if  $\pi_i$  is the perturbation associated with the truncation operation used to define  $b_i^+$ , then we have

$$b_i^+ = T \left( \left( x_{i-1} - \frac{1}{2^i} \right) \cdot \frac{1}{L} + \pi_i \right).$$

Hence, if  $x_{i-1} - 1/2^i \geq L - \pi_i/L$ , then we will have  $b_i^+ = 1$ , while if  $x_{i-1} - 1/2^i \leq -\pi_i/L$ , then we will have  $b_i^+ = 0$ , and we note that one of these two cases must be true since  $x$  is not in a bad region. Thus,  $b_i^+$  is decoded correctly. Since  $x_{i-1} = x - \sum_{j=1}^{i-1} \frac{b_j^+}{2^j}$ , and since  $x_i = x_{i-1} - \frac{b_i^+}{2^i}$ , we have that  $x_i$  is computed correctly as well.  $\square$

**5.1.3 Evaluating a Boolean Circuit.** A Boolean circuit is defined by a tuple  $(V, X, Y, G)$ , where  $V$  is a set of variables,  $X \subset V$  is a set of *input variables*, and  $Y \subset V$  is a set of *output variables*. Each variable can hold a value from the set  $\{0, 1\}$ . We can therefore interpret the set  $X = \{x_1, x_2, \dots, x_k\}$  as a vector  $x \in \{0, 1\}^k$ , and the set  $Y = \{y_1, y_2, \dots, y_l\}$  as a vector  $y \in \{0, 1\}^l$ . The set  $G$  contains gates of the following form:

- **Not:**  $v_1 = 1 - v_2$ , where  $v_1, v_2 \in V$ .
- **And:**  $v_1 = 1$  if  $v_2 = v_3 = 1$ , and  $v_1 = 0$  otherwise, where  $v_1, v_2, v_3 \in V$ .

We insist that every non-input variable is the output of exactly one gate. We also require that the circuit be acyclic in the sense that one can topologically order the set  $V = \{v_1, v_2, \dots, v_n\}$  such that the gate that outputs a value to variable  $v_i$  only takes inputs from variables  $v_j$  with  $j < i$ .

Therefore, each Boolean circuit computes a function  $F : \{0, 1\}^k \rightarrow \{0, 1\}^l$ , where the value  $y = F(x)$  is the result of fixing  $x$  at the input variables of the circuit, and then evaluating the gates to obtain the output value  $y$ .

We can simulate a Boolean circuit via a linear circuit in the following way. For each not-gate  $v_i$  with input  $v_j$ , we set

$$v_i = 1 - v_j$$

in the linear circuit. For each and-gate  $v_i$  with inputs  $v_j$  and  $v_k$ , we set

$$v_i = T_{[0, 1]}(4 \cdot (v_j + v_k - 1.5)).$$

We use  $\text{EVAL}(G, x)$  to refer to a linear circuit that simulates  $G$  in this way on inputs given by the binary variable  $x$ .

Recall that  $\text{EVAL}^\pi(G, x)$  refers to the version of  $\text{EVAL}$  that is perturbed by the vector  $\pi$ . The following lemma states that, if perturbations in  $\pi$  are in the range  $(-1, 1)$ , then  $\text{EVAL}^\pi(G, x)$  correctly evaluates the Boolean circuit with no perturbations appearing in the output variables. When we prove Lemma 5.1, we will set  $\delta$  to be much smaller than 1, meaning that we will have  $\pi_i \in (-1, 1)$ , so the lemma implies that from now on we can assume that all Boolean circuits we evaluate will be correctly simulated.

**LEMMA 5.4.** *Let  $G$  be a Boolean circuit with input variables  $x_1$  through  $x_k$  and output variables  $y_1$  through  $y_l$ . If  $\pi \in (-1, 1)$ , each  $x_i \in \{0, 1\}$ , and we compute  $y_1$  through  $y_l$  using the linear circuit  $\text{EVAL}^\pi(G, x)$ , then each  $y_i$  will lie in  $\{0, 1\}$ , and will hold the value that would be computed by  $G$  on input  $x$ .*

**PROOF.** We will prove inductively that each gate in the Boolean circuit is computed correctly.

For each not-gate  $v_i$  with input  $v_j$ , the linear circuit directly computes  $v_i = 1 - v_j$ . Since by the inductive hypothesis, we have that  $v_j \in \{0, 1\}$  is the correct value for gate  $v_j$ , it is clear that  $v_i$  will be computed correctly.

For an and-gate  $v_i$  with inputs  $v_j$  and  $v_k$ , the inductive hypothesis implies that  $v_j, v_k \in \{0, 1\}$  and that both are computed correctly. If  $\pi_i$  is the perturbation that is associated with the truncation operation for  $v_i$ , then we have

$$v_i = T_{[0, 1]}(4 \cdot (v_j + v_k - 1.5) + \pi_i).$$

If at least one of the two inputs is equal to 0, then we have

$$\begin{aligned} 4 \cdot (v_j + v_k - 1.5) + \pi_i &\leq -2 + \pi_i \\ &< -1, \end{aligned}$$

where the last line uses the fact that  $\pi_i < 1$ . Hence, the value will be truncated to 0, so we will have  $v_i = 0$ .

On the other hand, if  $v_j = v_k = 1$ , then we have

$$\begin{aligned} 4 \cdot (v_j + v_k - 1.5) + \pi_i &= 2 + \pi_i \\ &> 1, \end{aligned}$$

where the last line uses the fact that  $\pi_i > -1$ . Hence, the value will be truncated to 1, so we will have  $v_i = 1$ .

Therefore, in both cases, we have that  $v_i \in \{0, 1\}$  and that  $v_i$  is correctly computed.  $\square$

Given an expression  $e$  over binary variables, we write  $E(e)$  to be the linear circuit that is defined in the following way. First, we construct a Boolean circuit  $G$  that evaluates  $e$ , then we evaluate that circuit using EVAL.

## 5.2 Computing a Mesa

We now build circuits that will allow us to output the five affine pieces of a mesa.

**5.2.1 The BITMULTIPLY Circuit: Multiplying by a Single Bit.** As mentioned in the technical overview, if we are to output an affine function with a particular gradient, we need to be able to multiply two variables together. Although it is, by definition, impossible to multiply two continuous variables in a linear circuit, we show that we can multiply a continuous variable with a variable encoded in binary.

We break this operation down into two steps. In this section, we implement the  $\text{BITMULTIPLY}(x, b)$  circuit, which multiplies a continuous variable  $x \in [-2, 2]$  with a bit  $b \in \{0, 1\}$ . We define the circuit  $\text{BITMULTIPLY}$  in the following way:

$$\text{BITMULTIPLY}(x, b) = 4 \cdot (1 - b) + \max(-4, x - 8 \cdot (1 - b)).$$

Recall that  $\text{BITMULTIPLY}^\pi$  is the  $\text{BITMULTIPLY}$  circuit perturbed by the vector  $\pi$ , and also recall that we will choose  $\delta$  such that  $\pi \in (-1, 1)$ . The following lemma shows that  $\text{BITMULTIPLY}^\pi$  is always correct when  $b = 0$ , and when  $b = 1$ , the circuit outputs the correct answer (additively) perturbed by a value that is at most  $\max_i |\pi_i|$ .

**LEMMA 5.5.** *If  $x \in [-2, 2]$  and each  $\pi_i \in (-1, 1)$ , then we have the following:*

- If  $b = 0$  then  $\text{BITMULTIPLY}^\pi(x, b) = 0$ .
- If  $b = 1$  then  $\text{BITMULTIPLY}^\pi(x, b) = x + \sigma_{bm}(\pi)$ , where  $\sigma_{bm}$  is a function satisfying  $|\sigma_{bm}(\pi)| \leq \max_i |\pi_i|$ .

**PROOF.** Throughout, we will assume that  $\pi_i$  is the perturbation associated with the max gate in the  $\text{BITMULTIPLY}$  circuit.

If  $b = 0$ , then we have

$$\begin{aligned} \text{BITMULTIPLY}^\pi(x, 0) &= 4 \cdot (1 - b) + \max(-4, x - 8 \cdot (1 - b) + \pi_i) \\ &= 4 + \max(-4, x - 8 + \pi_i) \\ &= 0, \end{aligned}$$

where the second line uses the fact that  $x \leq 2$  and  $\pi_i < 1$ , meaning that the  $-4$  term is the maximum.

If  $b = 1$ , then we have

$$\begin{aligned} \text{BITMULTIPLY}^\pi(x, 1) &= 4 \cdot (1 - b) + \max(-4, x - 8 \cdot (1 - b) + \pi_i) \\ &= \max(-4, x + \pi_i) \\ &= x + \pi_i, \end{aligned}$$

where the second line uses the fact that  $x \geq -2$  and  $\pi_i > -1$ , meaning that the  $x + \pi_i$  term is the maximum. So setting  $\sigma_{\text{bm}}(\pi) = \pi_i$  completes the proof.  $\square$

**5.2.2 The  $\text{CONTTIMESBIN}$  Circuit: Multiplying by a Binary Variable.** With the  $\text{BITMULTIPLY}$  circuit at hand, it is now straightforward to implement the  $\text{CONTTIMESBIN}(x, y)$  circuit, which multiplies a continuous variable  $x \in [-2, 2]$  with a binary variable  $y$ . Specifically, we just multiply each bit individually, and then sum the results. We use the following definition:

$$\begin{aligned} \text{CONTTIMESBIN}(x, y) = & \sum_{i=1}^n \left( \frac{1}{2^i} \cdot \text{BITMULTIPLY}(x, y_i^+) \right) \\ & - \sum_{i=1}^n \left( \frac{1}{2^i} \cdot \text{BITMULTIPLY}(x, y_i^-) \right). \end{aligned}$$

Recall that  $\text{CONTTIMESBIN}^\pi$  is the  $\text{CONTTIMESBIN}$  circuit perturbed by the vector  $\pi$ . The following lemma shows that  $\text{CONTTIMESBIN}^\pi$  introduces a perturbation of magnitude at most  $\max_i |\pi_i|$ .

**LEMMA 5.6.** *If  $x \in [-2, 2]$ , each  $y_i^j \in \{0, 1\}$ , and each  $\pi_i \in (-1, 1)$ , then we have*

$$\text{CONTTIMESBIN}^\pi(x, y_1, \dots, y_n) = x \cdot y + \sigma_{\text{ctb}}(\pi),$$

where  $\sigma_{\text{ctb}}$  is a function satisfying  $|\sigma_{\text{ctb}}(\pi)| \leq \max_i |\pi_i|$ .

**PROOF.** We will prove this for the case where  $y \geq 0$ . The case where  $y \leq 0$  is entirely symmetric.

Let  $\sigma_{\text{bm}}^i$  be the function associated with the  $\text{BITMULTIPLY}(x, y_i)$  operation. Observe that  $x$ , each  $y_i^+$ , and each  $\pi_i$  all satisfy the preconditions of Lemma 5.5. Thus, we have

$$\begin{aligned} \text{CONTTIMESBIN}^\pi(x, y_1, \dots, y_n) &= \sum_{i: y_i^+ = 1} \left( \frac{1}{2^i} (x + \sigma_{\text{bm}}^i(\pi)) \right) - 0 \\ &= x \cdot y + \sum_{i: y_i = 1} \left( \frac{1}{2^i} \cdot \sigma_{\text{bm}}^i(\pi) \right), \end{aligned}$$

where the first line uses the fact that  $y_i^- = 0$  for all  $i$ , so the negative summands introduce no extra perturbations. We set  $\sigma_{\text{ctb}}(\pi) = \sum_{i: y_i = 1} \frac{1}{2^i} \cdot \sigma_{\text{bm}}^i(\pi)$ . Since for each  $i$ , we have  $|\sigma_{\text{bm}}^i(\pi)| \leq \max_j \pi_j$ , and since  $\sum_{i: y_i = 1} \frac{1}{2^i} \leq 1$ , we have that  $|\sigma_{\text{ctb}}(\pi)| \leq \max_j \pi_j$ .  $\square$

**5.2.3  $\text{AFFINE}$ : Computing an Affine Function.** We now build a linear circuit that outputs an affine function that has value  $a$  at point  $p = (p^1, p^2) \in \mathbb{R}^2$  and has gradient  $g = (g^1, g^2)$ , where each of these values will be presented in binary to the circuit. Specifically, given a continuous variable  $x \in [0, 1]^2$ , we define the circuit  $\text{AFFINE}(x, p^1, p^2, a, g^1, g^2)$ , which will output the value of the affine function defined by  $p$ ,  $a$ , and  $g$  at the point  $x$ . We define  $\text{AFFINE}(x, p^1, p^2, a, g^1, g^2)$  to be

$$\begin{aligned} &\text{CONTTIMESBIN}(x_1 - \text{DECODE}(p_1^1, \dots, p_n^1), g_1^1, \dots, g_n^1) \\ &+ \text{CONTTIMESBIN}(x_2 - \text{DECODE}(p_1^2, \dots, p_n^2), g_1^2, \dots, g_n^2) \\ &+ \text{DECODE}(a_1, \dots, a_n). \end{aligned}$$

Recall that  $\text{AFFINE}^\pi$  is the  $\text{AFFINE}$  circuit perturbed by the vector  $\pi$ . The following lemma states that  $\text{AFFINE}^\pi$  correctly outputs an affine function defined by  $p$ ,  $a$ , and  $g$ , while introducing an additive perturbation of magnitude at most  $2 \cdot \max_i \pi_i$ . In particular, note that the gradient of the outputted function is exactly  $g$ .

LEMMA 5.7. If  $x_1 - p^1, x_2 - p^2 \in [-2, 2]$ , if each  $p_i^1, p_i^2, g_i^1, g_i^2, a_i \in \{0, 1\}$ , and if each  $\pi_i \in (-1, 1)$ , then

$$\text{AFFINE}^\pi(x_1, x_2, p^1, p^2, a, g^1, g^2) = (x_1 - p^1) \cdot g^1 + (x_2 - p^2) \cdot g^2 + a + \sigma_{\text{aff}}(\pi)$$

where  $\sigma_{\text{aff}}$  is a function satisfying  $|\sigma_{\text{aff}}(\pi)| \leq 2 \cdot \max |\pi_i|$ .

PROOF. Let  $\sigma_{\text{ctb}}^1$  and  $\sigma_{\text{ctb}}^2$  be the perturbation functions associated with the first and second `CONTTIMESBIN` operations, respectively. Since all of our parameters meet the preconditions of Lemma 5.6, we obtain

$$\text{AFFINE}^\pi(x_1, x_2, p^1, p^2, a, g^1, g^2) = (x_1 - p^1) \cdot g^1 + (x_2 - p^2) \cdot g^2 + a + \sigma_{\text{ctb}}^1(\pi) + \sigma_{\text{ctb}}^2(\pi).$$

We set  $\sigma_{\text{aff}}(\pi) = \sigma_{\text{ctb}}^1(\pi) + \sigma_{\text{ctb}}^2(\pi)$ . Since  $|\sigma_{\text{ctb}}^j(\pi)| \leq \max_i |\pi_i|$  for all  $j$ , we have  $|\sigma_{\text{aff}}(\pi)| \leq 2 \cdot \max_i |\pi_i|$ .  $\square$

We should remark at this point that it is important that we keep track of perturbations in the way that we have been doing so far. To prove Lemma 5.1, it is not enough to just show that  $\text{AFFINE}^\pi$  outputs the affine function perturbed by at most  $2 \cdot \max |\pi_i|$ , because this formulation could allow different perturbations to occur at different values of  $x$ , which would then affect the gradient of the affine function. Instead, our formulation encapsulates the perturbation in the function  $\sigma_{\text{aff}}(\pi)$ , which makes it clear that the perturbation *does not* depend on  $x$ . So, we get an affine function with exactly the gradient that we asked for that has been additively perturbed everywhere by  $\sigma_{\text{aff}}(\pi)$ .

**5.2.4 Constructing the Mesa Pieces.** We now build circuits that output the five affine functions defined in Section 4.1 that are used to define a mesa.

Recall that  $E(e)$  denotes the evaluation of an expression  $e$  over binary variables using the `EVAL` circuit. We will also make use of the  $\ell$  and  $\Gamma$  constants used in the statement of Lemma 5.1, and we note that both values can be represented in binary using polynomially many bits.

If we are given continuous variables  $x = (x_1, x_2)$  and binary encodings of  $p = (p_1, p_2)$ ,  $a$ , and  $g = (g_1, g_2)$ , we can build the five affine pieces given in Section 4.1 as follows:

$$\begin{aligned} P_c(x, p, a, g) &= \text{AFFINE}(x_1, x_2, p_1, p_2, a, g_1, g_2) \\ P_r(x, p, a, g) &= \text{AFFINE}(x_1, x_2, E(p_1 + \ell/2), E(p_2 + \ell/2), E((g_1 + g_2)\ell/2 + a), E(-\Gamma + g_1), g_2) \\ P_t(x, p, a, g) &= \text{AFFINE}(x_1, x_2, E(p_1 + \ell/2), E(p_2 + \ell/2), E((g_1 + g_2)\ell/2 + a), g_1, E(-\Gamma + g_2)) \\ P_l(x, p, a, g) &= \text{AFFINE}(x_1, x_2, E(p_1 - \ell/2), E(p_2 - \ell/2), E(-(g_1 + g_2)\ell/2 + a), E(\Gamma + g_1), g_2) \\ P_b(x, p, a, g) &= \text{AFFINE}(x_1, x_2, E(p_1 - \ell/2), E(p_2 - \ell/2), E(-(g_1 + g_2)\ell/2 + a), g_1, E(\Gamma + g_2)). \end{aligned}$$

These circuits simply implement the definitions given in Section 4.1.

The mesa itself is the minimum of these five functions, but we will not compute that minimum at this stage. This is because we will first apply the averaging trick to each of the pieces and then take the minimum. The reasons for this will be discussed later.

Recall that for each circuit  $P_i$ , we have that  $P_i^\pi$  is the version of that circuit perturbed by the vector  $\pi$ . Since each piece is implemented by a single call to `AFFINE`, we get that  $P_i^\pi$  correctly outputs the corresponding piece of the mesa with additive perturbation whose magnitude is at most  $2 \cdot \max_j |\pi_j|$ .

LEMMA 5.8. If  $\pi \in (-1, 1)$ ,  $x_1 - p_1 \pm \ell/2 \in [-2, 2]$ , and  $x_2 - p_2 \pm \ell/2 \in [-2, 2]$  then for each  $i \in \{c, r, t, l, b\}$ , we have  $P_i^\pi(x) = P_i(x) + \sigma_p^i(\pi)$  where  $|\sigma_p^i(\pi)| \leq 2 \cdot \max_j |\pi_j|$ .

PROOF. First note that Lemma 5.4 guarantees that the calls to `E` will be correctly computed. Hence, we can apply Lemma 5.7 to argue that each  $P_i$  will be computed to be the correct affine function perturbed by  $\sigma_{\text{aff}}$ . Since  $|\sigma_{\text{aff}}(\pi)| \leq 2 \cdot \max_j |\pi_j|$ , this completes the proof.  $\square$

### 5.3 The Construction

We now proceed to construct the circuit  $\tilde{f}$ , which for each  $x \in [0, 1 - 1/2^n]^2$  will output

$$\tilde{f}(x) = \max_{p \in \tilde{G}} M(x; p, \tilde{a}(p), \tilde{g}(p)).$$

As mentioned in the technical overview, we will divide the points in  $\tilde{G}$  into four sets,  $S_1$  through  $S_4$ , and we will build circuits  $g_1$  through  $g_4$  that output

$$g_i = \max_{p \in S_i} M(x; p, \tilde{a}(p), \tilde{g}(p)).$$

The sets themselves are shown on the right-hand side of Figure 4. They simply divide the points into sub-grids of double the width. We define this formally using the following points:

$$\begin{aligned} a^1 &= (0, 0) & a^2 &= (0, 1) \\ a^3 &= (1, 0) & a^4 &= (1, 1). \end{aligned}$$

Then, we define

$$S_i = \{y \in G : y = a^i + (2a/2^n, 2b/2^n) \text{ for some } a, b \in \mathbb{N}\}.$$

Once, we have built the  $g_i$  circuits, we will then build  $\tilde{f}$  in the following way:

$$\tilde{f}(x) = \max(g_1(x), \max(g_2(x), \max(g_3(x), g_4(x)))).$$

**The definition of the  $g_i$  circuits.** To complete the construction, we must specify how the  $g_i$  circuits are constructed. This circuit will make use of the averaging trick, and we fix  $k = 12$  to be the number of samples that we will average over.

The first step for computing  $g_i$  is to extract the bits of  $x$ . We do this using the EXTRACTBITS circuit, and we carefully select  $k$  points close to  $x$  and attempt to decode each of them. Formally, for each  $i \in \{1, 2, 3, 4\}$  and each  $j \in \{1, \dots, k\}$ , we extract the binary variable  $y^{i,j} = (y_1^{i,j}, y_2^{i,j})$  in the following way:

$$y_l^{i,j} = \text{EXTRACTBITS} \left( a_l^i + x_l + \frac{9/8}{2^n} - \frac{j/8k}{2^n}, n-1, \frac{1}{24k \cdot 2^n} \right).$$

Since we are looking to extract a point from  $S_i$ , we add  $a^i$  to the point, and extract only the first  $n-1$  bits, to ensure that we get a binary encoding of a point from  $S_i$ . The other parameters here are chosen so that there can be at most two values of  $j$  such that  $y^{i,j}$  fails to be decoded correctly.

The next step is to construct the five pieces of the mesa defined by  $y^{i,j}$ , which we do in the following operation. Recall that  $E(e)$  denotes the evaluation of an expression  $e$  over binary variables using the EVAL circuit. Then, for each  $d \in \{c, r, t, l, b\}$ , we define

$$p_d^{i,j}(x) = \min \left( 1, P_d(x, y^{i,j}, E(\tilde{a}(y^{i,j})), E(\tilde{g}(y^{i,j}))) \right).$$

Here, the minimum with 1 is used to prevent outputs from bad decodes from being excessively large. This cannot affect the final output because we know that  $\tilde{f}(x) \leq 2/3$ , and so any piece that is in the output will not be affected by the minimum operation.

Next, we average over the  $p_d^{i,j}(x)$  values to obtain a single averaged piece. Formally, for each  $d \in \{c, r, t, l, b\}$ , we define

$$p_d^i(x) = \sum_{j=1}^k \frac{1}{k} \cdot p_d^{i,j}(x).$$

Finally, we output the mesa by taking the minimum over the five averaged pieces.

$$g_i(x) = \min\left(p_c^i(x), \min\left(p_r^i(x), \min\left(p_t^i(x), \min\left(p_l^i(x), p_b^i(x)\right)\right)\right)\right).$$

It is worth remarking that we must first average each piece and then take the minimum over averaged pieces, as we have done here. The alternative approach of first constructing  $k$  different mesas, and then averaging over them, does not work. This is because each mesa would be perturbed slightly differently, meaning that the locations at which the pieces of the mesa meet would move slightly. So, when we then average over those mesas we could end up, for example, averaging the center piece of one mesa with the right piece of another, which would create a new undesired gradient.

**Correctness.** We now prove that this construction satisfies Lemma 5.1. We begin by proving that the  $g_i$  functions are correct. Recall that  $g_i^\pi$  is the circuit  $g_i$  perturbed by the vector  $\pi$ . We will prove two properties.

- If  $\|x - y\|_\infty \leq 3/4 \cdot 1/2^n$  for some  $y \in S_i$ , meaning that we are suitably close to a point in  $S_i$ , then  $g_i^\pi$  outputs the mesa defined by  $y$ , in which all pieces have been additively perturbed by a small amount. This is because all of the  $k$  points that we decoded gave us binary encodings of  $y$ , since  $y$  is not close to any of the bad regions of the EXTRACTBITS operation. Thus, each  $p_d^i$  averages over  $k$  pieces that all have identical gradients, but each of which has been perturbed slightly differently. The averaging operation therefore results in a mesa whose pieces have gradients that are exactly correct, but where each has been perturbed by the average of the perturbations from the  $k$  different  $p_d^{i,j}$  operations.
- If we are not close to a point in  $S_i$ , then  $g_i$  outputs a value strictly less than 1/4. In this case, the EXTRACTBITS operations may decode different points in  $S_i$ , however, since  $\|x - y\|_\infty > 3/4 \cdot 1/2^n$  for all points  $y \in S_i$ , any mesa from  $S_i$  that we evaluate at  $x$  will output a value that is less than or equal to 0. We may also have at most two points that are incorrectly decoded, and for these points, we bound the value at 1 using the min operation in  $p_d^{i,j}$ . Since  $k = 12$  we get that  $g_i(x) \leq 10/12 \cdot 0 + 2/12 \cdot 1 = 1/6$ . While perturbations may increase this value, we show that if each element of  $\pi$  is suitably small, then  $g_i^\pi(x) < 1/4$ .

The two points above are shown formally in the following lemma:

LEMMA 5.9. Suppose that  $\max |\pi_i| \leq (\frac{1}{24k \cdot 2^n})^2 / 2$ . For each  $i \in \{1, 2, 3, 4\}$ , we have

- If  $\|x - y\|_\infty \leq 3/4 \cdot 1/2^n$  for some  $y \in S_i$ , then

$$g_i^\pi(x) = M(x, y, A, \tilde{g}(y)),$$

where  $A = (\tilde{a}(y) + \sigma_g^c(\pi), \tilde{a}(y) + \sigma_g^r(\pi), \tilde{a}(y) + \sigma_g^t(\pi), \tilde{a}(y) + \sigma_g^l(\pi), \tilde{a}(y) + \sigma_g^b(\pi))$  and each  $|\sigma_g^j(\pi)| \leq 7 \cdot \max_I |\pi_I|$ .

- Otherwise  $g_i^\pi(x) < 1/4$ .

PROOF. We fix  $t_j = \frac{9/8}{2^n} - \frac{j/8k}{2^n}$  and  $L = \frac{1}{24k \cdot 2^n}$  to be the arguments to the EXTRACTBITS circuit used to extract  $y^{i,j}$ . By Lemma 5.3, we have that an EXTRACTBITS operation for  $x_1$  will fail only if it lies in the region

$$\left[ a_1^i + \frac{m}{2^{n-1}} - t_j - \frac{\max_i |\pi_i|}{L}, \frac{m}{2^{n-1}} - t_j + L + \frac{\max_i |\pi_i|}{L} \right]$$

for some integer  $m \in \{0, 1, \dots, 2^{n-1} - 1\}$ . By assumption we have that  $\max_l |\pi_l| \leq L^2/2$ , so we can define the following *bad regions*:

$$\begin{aligned} R_j^1 &= \left[ a_1^i + \frac{m}{2^{n-1}} - t_j - L/2, \frac{m}{2^{n-1}} - t_j + 3L/2 \right], \\ R_j^2 &= \left[ a_2^i + \frac{m}{2^{n-1}} - t_j - L/2, \frac{m}{2^{n-1}} - t_j + 3L/2 \right]. \end{aligned}$$

So, the EXTRACTBITS operations used in  $g_i^\pi$  can fail only if  $x_1 \in R_j^1$  for some  $m$ , or if  $x_2 \in R_j^2$  for some  $m$ .

For each  $l \in \{1, 2\}$ , we can show that the set of bad regions  $\{R_j^l\}_{j=1, \dots, k}$  do not overlap with the following chain of inequalities:

$$\begin{aligned} \frac{m}{2^n} - t_j + 3L/2 &= a_l^i + \frac{m}{2^{n-1}} - \frac{9/8}{2^n} + \frac{j/(8k)}{2^n} + \frac{3/(2 \cdot 24k)}{2^n} \\ &= a_l^i + \frac{m}{2^{n-1}} - \frac{9/8}{2^n} + \frac{6j/(48k) + 3/(48k)}{2^n} \\ &< a_l^i + \frac{m}{2^{n-1}} - \frac{9/8}{2^n} + \frac{6(j+1)/(48k)}{2^n} - \frac{1/(48k)}{2^n} \\ &= a_l^i + \frac{m}{2^{n-1}} - t_j - L/2. \end{aligned}$$

Hence, both  $x_1$  and  $x_2$  can each lie in at most one bad region.

Since all bad regions lie between  $a_l^i + \frac{m}{2^{n-1}} - \frac{9/8}{2^n}$  and  $a_l^i + \frac{m}{2^{n-1}} - \frac{7/8}{2^n}$ , if either  $x_1$  or  $x_2$  lies in a bad region, then  $\|x - y\|_\infty \geq 3/4 \cdot 1/2^n$  for all  $y \in S_i$ .

We can now prove the lemma via a case analysis.

– If there exists a  $y \in S_i$  with  $\|x - y\|_\infty \leq 3/4 \cdot 1/2^n$  then note that all of the EXTRACTBITS operations used in  $g_i^\pi$  circuits will each return  $y$ , since we would have to cross a bad region to output any other point, and there are no bad regions within distance  $3/4 \cdot 1/2^n$  of  $y$ .

Thus, for each  $j$  and  $d \in \{c, r, t, l, b\}$ , we have that the  $P_d$  operation used in  $p_d^{i,j}(x)$  will output  $P_d(x, y, \tilde{a}(y), \tilde{g}(y)) + \sigma_p^d(\pi)$ , where  $|\sigma_p^d(\pi)| \leq 2 \cdot \max_j |\pi_j|$  by Lemma 5.8. This value is then perturbed by the min gate used in  $p_d^{i,j}$ , which introduces a perturbation that we will call  $\pi_d^{i,j}$ . Each min gate used in the definition of  $g_i(x)$  can add an additional perturbation of magnitude at most  $\max_l |\pi_l|$ , and at most four such perturbations can be added when computing  $g_i$ . So, we have that

$$g_i^\pi(x) = \min_{d \in \{c, r, t, l, b\}} \left( \min(1, P_d(x, y, \tilde{a}(y), \tilde{g}(y)) + \sigma_p^d(\pi) + \pi_d^{i,j}) \right) + \sigma_{\text{mins}}(\pi),$$

where  $|\sigma_{\text{mins}}(\pi)| \leq 4 \max_l |\pi_l|$ .

We must show that the min with 1 used in each term above does not affect the output. For all  $d$ , we have

$$\begin{aligned} 2/3 &\geq f(x) \\ &\geq M(x; y, \tilde{a}(y), \tilde{g}(y)) \\ &\geq P_d(x, y, \tilde{a}(y), \tilde{g}(y)). \end{aligned}$$

Observe that

$$\begin{aligned} \sigma_p^d(\pi) + \pi_d^{i,j} &\leq 3 \cdot \max_l |\pi_l| \\ &\leq \left( \frac{1}{24k \cdot 2^n} \right)^2 / 2 \\ &< 1/3, \end{aligned}$$

where the final line holds for all  $n \geq 1$ . Hence, the minimum never affects the output of  $g_i^\pi(x)$ , and so we have

$$g_i^\pi(x) = \min_{d \in \{c, r, t, l, b\}} \left( P_d(x, y, \tilde{a}(y), \tilde{g}(y)) + \sigma_p^d(\pi) + \pi_d^{i,j} \right) + \sigma_{\text{mins}}(\pi).$$

When we account for all of the perturbations above, we have that each piece is additively perturbed by at most  $7 \cdot \max_l |\pi_l|$ , as required.

- On the other hand, if there is no  $y \in S_i$  such that  $\|x - y\|_\infty \leq 3/4 \cdot 1/2^n$ , then note that  $M(x; y, \tilde{a}(y), \tilde{g}(y)) \leq 0$  for all  $y \in S_i$ . Since  $x_1$  and  $x_2$  can each fall into at most one bad region, there will be at most two indices  $j$  for which the  $p_d^{i,j}$  circuits will receive incorrectly decoded points, and for these circuits, we will have  $p_d^{i,j}(x) \leq 1$  due to the min operation used in the definition of  $p_d^{i,j}$ . For every other index  $j$ , the EXTRACTBITS operation will output some valid point  $y \in S_i$ . If  $\pi_d^{i,j}$  is the perturbation associated with the min gate in  $p_d^{i,j}$ , then we can apply Lemma 5.8 to obtain

$$\begin{aligned} p_d^{i,j}(x) &\leq P_d(x, y, \tilde{a}(y), \tilde{g}(y)) + \sigma_p(\pi) + \pi_d^{i,j} \\ &\leq 0 + \sigma_p(\pi) + \pi_d^{i,j} \\ &\leq 3 \cdot \max_l |\pi_l|, \end{aligned}$$

where the first inequality uses the fact that  $\|x - y\|_\infty > 3/4 \cdot 1/2^n$ , which implies that the mesa associated with  $y$  has value less than or equal to 0 at  $x$ .

Since  $k = 12$ , the circuit  $p_d^i(x)$  averages over 10 values that are at most  $3 \cdot \max_l |\pi_l|$ , and 2 values that are at most 1, so we have

$$p_d^i(x) \leq \frac{10}{12} \cdot 3 \cdot \max_l |\pi_l| + \frac{2}{12}.$$

As argued in the first case, the  $g_i$  circuit introduces an additive perturbation of magnitude at most  $4 \cdot \max_l |\pi_l|$ . So, we have

$$g_i^\pi(x) \leq 1/6 + 7 \cdot \max_l |\pi_l|.$$

Since  $\max_l |\pi_l| \leq (\frac{1}{24k \cdot 2^n})^2 / 2 < 1/84$  for all  $n \geq 1$ , we have  $g_i^\pi(x) < 1/4$ , as required.  $\square$

The following final lemma now shows that  $\tilde{f}$  correctly computes the perturbed mesa function that is desired by Lemma 5.1. As mentioned in the technical overview, any spurious gradients that arise from bad bit decodes do not appear in the output. This is because the previous lemma assures that  $g_i(x) < 1/4$  whenever  $g_i$  fails to decode a point. Since  $\tilde{f}(x) \geq 1/3$  for all points  $x$ , there is therefore some other function  $g_j(x) \geq 1/3$  that outputs the mesa that defines  $\tilde{f}(x)$ , and thus all gradients from  $g_i$  do not appear in the final output of the function. The following lemma proves this, and also shows that if  $\pi$  is suitably small, then the perturbations do not affect this property.

LEMMA 5.10. *Suppose that  $\max |\pi_i| \leq (\frac{1}{24k \cdot 2^n})^2 / 2$ . For all  $x \in [0, 1 - 1/2^n]^2$  we have*

$$\tilde{f}^\pi(x) = M(x, y, A, \tilde{g}(y)),$$

where  $A = (\tilde{a}(y) + \sigma_f^c(\pi), \tilde{a}(y) + \sigma_f^r(\pi), \tilde{a}(y) + \sigma_f^t(\pi), \tilde{a}(y) + \sigma_f^l(\pi), \tilde{a}(y) + \sigma_f^b(\pi))$  and each  $|\sigma_f^j(\pi)| \leq 11 \cdot \max_l |\pi_l|$ .

PROOF. If  $\pi_a$ ,  $\pi_b$ , and  $\pi_c$  are the perturbations associated with the three min gates used to compute  $\tilde{f}$ , then we have

$$\tilde{f}^\pi(x) = \max(g_1^\pi(x), \max(g_2^\pi(x), \max(g_3^\pi(x), g_4^\pi(x) + \pi_a) + \pi_b) + \pi_c). \quad (10)$$

From Lemma 5.9, we have that either  $g_i^\pi(x)$  outputs a perturbed mesa for some  $y \in S_i$ , that is within distance  $3/4 \cdot 1/2^n$  of  $x$ , or that  $g_i(x) < 1/4$ .

We rule out the case in which  $g_i(x) < 1/4$  for all  $i$ . To do this, we first recall that by assumption we have  $f(x) \geq 1/3$ . Hence, there is a mesa defined by  $y$  such that  $M(x; y, \tilde{a}(y), \tilde{g}(y)) \geq 1/3$ , so we must also have  $\|x - y\|_\infty \leq 3/4 \cdot 1/2^n$ . Therefore, Lemma 5.9 implies that there is an index  $i$  and  $d \in \{c, r, t, l, b\}$  such that

$$\begin{aligned} g_i(x) &\geq M(x; y, \tilde{a}(y), \tilde{g}(y)) + \sigma_g^d(\pi) \\ &\geq M(x; y, \tilde{a}(y), \tilde{g}(y)) - 7 \cdot \max_i |\pi_i| \\ &> M(x; y, \tilde{a}(y), \tilde{g}(y)) - 1/12 \\ &\geq 1/4, \end{aligned}$$

where the third line uses the fact that  $\max |\pi_i| \leq (\frac{1}{24k \cdot 2^n})^2 / 2 < 1/84$  for all  $n$ , and the final line uses the fact that  $M(x; y, \tilde{a}(y), \tilde{g}(y)) \geq 1/3$ .

Finally note that the perturbations contributed by  $\pi_a$ ,  $\pi_b$ , and  $\pi_c$ , can perturb the final output by a value of magnitude at most  $4 \cdot \max_i |\pi_i|$ . So, once we incorporate the perturbations of magnitude at most  $7 \cdot \max_i |\pi_i|$  arising from  $\sigma_g(\pi)$ , each mesa piece can be additively perturbed by a value of magnitude at most  $11 \cdot \max_i |\pi_i|$ .  $\square$

To complete the proof of Lemma 5.1, it therefore suffices to pick  $\delta = \min(\delta'/11, (\frac{1}{24k \cdot 2^n})^2 / 2)$ .

## 6 Conclusions, Open Problems

Open problems include the question of whether our result continues to hold for restrictions of QPs such as the bilinear case, where there are no squared terms. Babichenko and Rubinstein[4] show that it is CLS-complete to find a KKT-point of a *multilinear* degree-five polynomial. They use this multilinearity to show CLS-completeness also for the problem of finding a mixed equilibrium of degree-five polytensor games, on the way to showing CLS-completeness for finding a mixed equilibrium of a congestion game. One of the main open problems arising from their work is whether it is CLS-hard to find a mixed equilibrium of a “network coordination” game, that is, a degree-2 polytensor game. This would follow if our result continued to hold without squared terms. Other special cases of interest include: no linear terms (the NP-hardness results of [2] apply to QPs that have no linear terms, i.e., quadratic forms). Our result also exploits exponential ratios between coefficients, leaving open the question of whether it should hold if coefficients are presented in unary. In connection with their discussion of KKT solutions, [29] also ask about the special case where there is a single local minimum (hence, the global minimum). This latter problem would have to be treated as a promise problem.

## Acknowledgments

We thank the anonymous reviewers for comments and suggestions that helped improve the presentation of the paper.

## References

- [1] Amir Ali Ahmadi and Jeffrey Zhang. 2022. Complexity aspects of local minima and related notions. *Advances in Mathematics* 397, (2022), 108119. DOI:<https://doi.org/10.1016/j.aim.2021.108119>

- [2] Amir Ali Ahmadi and Jeffrey Zhang. 2022. On the complexity of finding a local minimizer of a quadratic function over a polytope. *Mathematical Programming* 195, 1–2 (2022), 783–792. DOI : <https://doi.org/10.1007/s10107-021-01714-2>
- [3] Amar Andjouh and Mohand Ouamer Bibi. 2022. Adaptive global algorithm for solving box-constrained non-convex quadratic minimization problems. *Journal of Optimization Theory and Applications* 192, 1 (2022), 360–378. DOI : <https://doi.org/10.1007/s10957-021-01980-2>
- [4] Yakov Babichenko and Aviad Rubinstein. 2021. Settling the complexity of Nash equilibrium in congestion games. In *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC’21)*. 1426–1437. DOI : <https://doi.org/10.1145/3406325.3451039>
- [5] Mihir Bellare and Phillip Rogaway. 1995. The complexity of approximating a nonlinear program. *Mathematical Programming* 69, 1–3 (1995), 429–441. DOI : <https://doi.org/10.1007/bf01585569>
- [6] Immanuel M. Bomze, Mirjam Dür, Etienne de Klerk, Cornelis Roos, Arie J. Quist, and Tamás Terlaky. 2000. On copositive programming and standard quadratic optimization problems. *Journal of Global Optimization* 18, 4 (2000), 301–320. DOI : <https://doi.org/10.1023/A:1026583532263>
- [7] Sébastien Bubeck. 2014. Convex optimization: Algorithms and complexity. In *Foundations and Trends in Machine Learning* 8, 3–4 (2014), 231–357.
- [8] Samuel Burer and Adam N. Letchford. 2009. On nonconvex quadratic programming with box constraints. *SIAM Journal on Optimization* 20, 2 (2009), 1073–1089. DOI : <https://doi.org/10.1137/080729529>
- [9] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. 2009. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM* 56, 3 (2009), 14:1–14:57. DOI : <https://doi.org/10.1145/1516512.1516516>
- [10] Arka Rai Choudhuri, Pavel Hubáček, Chethan Kamath, Krzysztof Pietrzak, Alon Rosen, and Guy N. Rothblum. 2019. Finding a Nash equilibrium is no easier than breaking Fiat-Shamir. In *Proceedings of the 51st ACM Symposium on Theory of Computing (STOC’19)*. 1103–1114. DOI : <https://doi.org/10.1145/3313276.3316400>
- [11] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. 2009. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing* 39, 1 (2009), 195–259. DOI : <https://doi.org/10.1137/070699652>
- [12] Constantinos Daskalakis and Christos Papadimitriou. 2011. Continuous local search. In *Proceedings of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA’11)*. 790–804. DOI : <https://doi.org/10.1137/1.9781611973082.62>
- [13] Edith Elkind, Abheek Ghosh, and Paul W. Goldberg. 2022. Simultaneous contests with equal sharing allocation of prizes: Computational complexity and price of anarchy. In *Proceedings of the 15th International Symposium on Algorithmic Game Theory (SAGT’22)*. 133–150. DOI : [https://doi.org/10.1007/978-3-031-15714-1\\_8](https://doi.org/10.1007/978-3-031-15714-1_8)
- [14] Kousha Etessami and Mihalis Yannakakis. 2010. On the complexity of Nash equilibria and other fixed points. *SIAM Journal on Computing* 39, 6 (2010), 2531–2597. DOI : <https://doi.org/10.1137/080720826>
- [15] John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. 2022. The complexity of gradient descent:  $\text{CLS} = \text{PPAD} \cap \text{PLS}$ . *J. ACM* 70, 1 (2022), 7:1–7:74. DOI : <https://doi.org/10.1145/3568163>
- [16] Uriel Feige and Joe Kilian. 1994. Two prover protocols: Low error at affordable rates. In *Proceedings of the 26th ACM Symposium on Theory of Computing*. 172–183. DOI : <https://doi.org/10.1145/195058.195128>
- [17] Uriel Feige and László Lovász. 1992. Two-prover one-round proof systems: Their power and their problems (extended abstract). In *Proceedings of the 24th ACM Symposium on Theory of Computing*. 733–744. DOI : <https://doi.org/10.1145/129712.129783>
- [18] Abheek Ghosh and Alexandros Hollender. 2024. The complexity of symmetric bimatrix games with common payoffs. In *Proceedings of the 20th International Conference on Web and Internet Economics (WINE’24)*.
- [19] Pavel Hubáček and Eylon Yogev. 2020. Hardness of continuous local search: Query complexity and cryptographic lower bounds. *SIAM Journal on Computing* 49, 6 (2020), 1128–1172. DOI : <https://doi.org/10.1137/17M1118014>
- [20] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Zhang. 2021. SNARGs for bounded depth computations and PPAD hardness from sub-exponential LWE. In *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC’21)*. 708–721. DOI : <https://doi.org/10.1145/3406325.3451055>
- [21] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. 1988. How easy is local search? *Journal of Computer and System Sciences* 37, 1 (1988), 79–100. DOI : [https://doi.org/10.1016/0022-0000\(88\)90046-3](https://doi.org/10.1016/0022-0000(88)90046-3)
- [22] Mark W. Krentel. 1990. On finding and verifying locally optimal solutions. *SIAM Journal on Computing* 19, 4 (1990), 742–749. DOI : <https://doi.org/10.1137/0219052>
- [23] Nimrod Megiddo and Christos H. Papadimitriou. 1991. On total functions, existence theorems and computational complexity. *Theoretical Computer Science* 81, 2 (1991), 317–324. DOI : [https://doi.org/10.1016/0304-3975\(91\)90200-L](https://doi.org/10.1016/0304-3975(91)90200-L)
- [24] T. S. Motzkin and E. G. Strauss. 1965. Maxima for graphs and a new proof of a theorem of Turán. *Canadian Journal of Mathematics* 17 (1965), 553–540. DOI : <https://doi.org/10.4153/CJM-1965-053-6>
- [25] Katta G. Murty and Santosh N. Kabadi. 1987. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming* 39, 2 (1987), 117–129. DOI : <https://doi.org/10.1007/BF02599248>
- [26] Christos H. Papadimitriou. 1994. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences* 48, 3 (1994), 498–532. DOI : [https://doi.org/10.1016/S0022-0000\(05\)80063-7](https://doi.org/10.1016/S0022-0000(05)80063-7)

- [27] Panos M. Pardalos and G. Schnitger. 1988. Checking local optimality in constrained quadratic programming is NP-hard. *Operations Research Letters* 7, 1 (1988), 33–35. DOI : [https://doi.org/10.1016/0167-6377\(88\)90049-1](https://doi.org/10.1016/0167-6377(88)90049-1)
- [28] Panos M. Pardalos and Stephen A. Vavasis. 1991. Quadratic programming with one negative eigenvalue is NP-hard. *Journal of Global Optimization* 1, 1 (1991), 15–22. DOI : <https://doi.org/10.1007/BF00120662>
- [29] Panos M. Pardalos and Stephen A. Vavasis. 1992. Open questions in complexity theory for numerical optimization. *Mathematical Programming* 57 (1992), 337–339. DOI : <https://doi.org/10.1007/BF01581088>
- [30] Panos M. Pardalos, Yinyu Ye, and Chi-Geun Han. 1991. Algorithms for the solutions of quadratic knapsack problems. *Linear Algebra and Its Applications* 152 (1991), 69–91. DOI : [https://doi.org/10.1016/0024-3795\(91\)90267-Z](https://doi.org/10.1016/0024-3795(91)90267-Z)
- [31] Aviad Rubinstein. 2018. Inapproximability of Nash equilibrium. *SIAM Journal on Computing* 47, 3 (2018), 917–959. DOI : <https://doi.org/10.1137/15M1039274>
- [32] Sartaj Sahni. 1972. Some related problems from network flows, game theory and integer programming. In *Proceedings of the 13th Annual Symposium on Switching and Automata Theory (SWAT'72)*. 130–138. DOI : <https://doi.org/10.1109/SWAT.1972.23>
- [33] Sartaj Sahni. 1974. Computationally related problems. *SIAM Journal on Computing* 3, 4 (1974), 262–279. DOI : <https://doi.org/10.1137/0203021>
- [34] Emanuel Tewolde, Caspar Oesterheld, Vincent Conitzer, and Paul W. Goldberg. 2023. The computational complexity of single-player imperfect-recall games. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI'23)*. 2878–2887. DOI : <https://doi.org/10.24963/ijcai.2023/321>
- [35] Stephen A. Vavasis. 1990. Quadratic programming is in NP. *Information Processing Letters* 36, 2 (1990), 73–77. DOI : [https://doi.org/10.1016/0020-0190\(90\)90100-c](https://doi.org/10.1016/0020-0190(90)90100-c)
- [36] Stephen A. Vavasis and Richard Zippel. 1990. *Proving Polynomial-Time for Sphere-Constrained Quadratic Programming*. Technical Report TR90-1182. Cornell University. <https://hdl.handle.net/1813/7022>
- [37] Yinyu Ye. 1992. On affine scaling algorithms for nonconvex quadratic programming. *Mathematical Programming* 56, 1–3 (1992), 285–300. DOI : <https://doi.org/10.1007/bf01580903>
- [38] Yinyu Ye. 1998. On the complexity of approximating a KKT point of quadratic programming. *Mathematical Programming* 80, 2 (1998), 195–211. DOI : <https://doi.org/10.1007/BF01581726>
- [39] Yinyu Ye. 1999. Approximating quadratic programming with bound and quadratic constraints. *Mathematical Programming* 84, 2 (1999), 219–226. DOI : <https://doi.org/10.1007/s10107980012a>

## Appendices

### A From Approximate to Exact Solutions

The following shows that our hardness result also applies to the problem of computing an  $\varepsilon$ -KKT point of a quadratic polynomial over  $[0, 1]^2$ , where  $\varepsilon > 0$  is allowed to be exponentially small. The proof technique for this is standard, see, for example, [14].

**LEMMA A.1.** *Given an instance  $p$  of QUADRATIC-KKT, we can compute in polynomial time an  $\varepsilon > 0$  such that from any  $\varepsilon$ -KKT point of  $p$ , we can obtain an exact KKT point of  $p$  in polynomial time.*

**PROOF.** Let  $p$  be an instance of QUADRATIC-KKT over  $n$  variables. For any sets  $I_0, I_1 \subseteq [n]$  with  $I_0 \cap I_1 = \emptyset$ , we define a corresponding LP( $I_0, I_1$ ) over variables  $z, x_i$ :

$$\begin{array}{ll} \min & z \\ \text{s.t.} & z \geq \frac{\partial p}{\partial x_i}(x) \quad \forall i \in [n] \setminus I_0 \\ & z \geq -\frac{\partial p}{\partial x_i}(x) \quad \forall i \in [n] \setminus I_1 \\ & x_i = 0 \quad \forall i \in I_0 \\ & x_i = 1 \quad \forall i \in I_1 \\ & 0 \leq x_i \leq 1 \quad \forall i \in [n] \\ & z \geq 0 \end{array} .$$

Note that this is indeed an LP, and that it is feasible and admits an optimal solution. Since LPs have solutions with bit complexity that is polynomially bounded in their description length, and the description length of LP( $I_0, I_1$ ) for all  $I_0, I_1$  is bounded by a polynomial in the description length of  $p$ , we can compute in polynomial time a value  $\varepsilon > 0$  such that for all  $I_0, I_1$  the optimal value of LP( $I_0, I_1$ ) is 0 or strictly larger than  $\varepsilon$ .

Now, consider any  $\varepsilon$ -KKT point  $x^*$  of  $p$ . Letting  $I_0 := \{i \in [n] : x_i^* = 0\}$  and  $I_1 := \{i \in [n] : x_i^* = 1\}$ , observe that  $(\varepsilon, x^*)$  is feasible for  $\text{LP}(I_0, I_1)$  with objective function value  $\varepsilon$ . As a result, the  $\text{LP}(I_0, I_1)$  must have optimal value 0. Now, it suffices to solve this LP to obtain an optimal solution  $(0, x)$  and note that  $x$  must be an exact KKT point of  $p$ .  $\square$

Note that the proof also gives all the ingredients needed to show the existence of rational solutions with polynomially bounded bit complexity. Indeed, since a solution is guaranteed to exist (because  $p$  is a continuous function over a compact domain and thus admits a global minimum), there must exist  $I_0$  and  $I_1$  such that  $\text{LP}(I_0, I_1)$  has optimal value 0. Since all these LPs have solutions of polynomially bounded bit complexity, the same also holds for QUADRATIC-KKT. We note that both this proof sketch and the proof of the lemma can be extended to general compact domains of the form  $Ax \leq b$ .

## B Proof of Lemma 5.2

Let  $C$  be as in the statement of Lemma 5.2. Let  $x_1, \dots, x_N$  denote the variables computed by the circuit, where  $x_1, \dots, x_n$  are the inputs, and  $x_N$  is the output.

**Preprocessing.** We begin by noting that we can assume that  $C$  only consists of gates of the form  $x_i := ax_j + bx_k + c$  and of the form  $x_i := T_{[a, b]}(x_j)$ . Indeed, given  $C$ , we can construct  $\bar{C}$  that only uses this restricted set of gates as follows:

- A gate of type  $+$ ,  $-$ ,  $c$ , or  $\times c$  can be directly simulated by a single affine linear gate of the form  $x_i := ax_j + bx_k + c$ .
- A gate  $x_i := T_{[a, b]}(a_i x_j + b_i x_k + c_i)$  can be easily simulated by combining the two types of allowed gates.
- A gate  $x_i := \min\{x_j, x_k\}$  can be simulated by instead letting

$$x_i := x_j - T_{[0, 3B]}(x_j - x_k),$$

which can be computed using only the two types of allowed gates. Here,  $B \geq 1$  is a bound on the value of any variable in the perturbed circuit  $C^\sigma$  when  $\sigma \in [-1, 1]^m$  and the inputs lie in  $[0, 1]^n$ . Formally, for any  $j \in [N]$ ,  $B \geq 1$  must satisfy

$$\max_{\sigma \in [-1, 1]^m} \max_{y \in [0, 1]^n} f_j^\sigma(y) \in [-B, B],$$

where  $f_j^\sigma(y)$  denotes the value of the  $j$ th variable of circuit  $C^\sigma$  on input  $y$ . Note that such a bound  $B$  can be computed in polynomial time given the description of  $C$ ; see, e.g., [15].

- A gate  $x_i := \max\{x_j, x_k\}$  can be simulated by first replacing it by  $x_i := -\min\{-x_j, -x_k\}$ , and then simulating the min gate as shown above.

We let  $\bar{x}_1, \dots, \bar{x}_N$  denote the variables in circuit  $\bar{C}$  that correspond to the original variables  $x_1, \dots, x_N$  of  $C$ . Note that  $\bar{C}$  will also contain some additional variables that are needed to simulate the original gates. For any perturbation  $\sigma$  of  $C$ , we let  $x_1^\sigma, \dots, x_N^\sigma$  denote the variables of circuit  $C^\sigma$ . We define  $\bar{x}_1^\pi, \dots, \bar{x}_N^\pi$  similarly for  $\bar{C}^\pi$ . Finally, note that gates of the form  $\bar{x}_i := a\bar{x}_j + b\bar{x}_k + c$  are not perturbed, while gates of the form  $\bar{x}_i := T_{[a, b]}(\bar{x}_j)$  are perturbed as  $\bar{x}_i^\pi := T_{[a, b]}(\bar{x}_j^\pi + \pi_i)$ .

We can now check that the new circuit  $\bar{C}$  satisfies, for any  $\delta \leq 1$ : given any  $\delta$ -perturbation  $\pi$  of  $\bar{C}$ , there exists a  $\delta$ -perturbation  $\sigma$  of  $C$  such that  $x_i^\sigma = \bar{x}_i^\pi$  whenever the input to the circuit lies in  $[0, 1]^n$ . Indeed, this can be shown by induction by using the following observations:

- The simulation of a gate of type  $+$ ,  $-$ ,  $c$ , or  $\times c$  does not introduce any gates that are perturbed.
- The simulation of a gate of type  $x_i := T_{[a, b]}(a_i x_j + b_i x_k + c_i)$  yields that  $\bar{x}_i^\pi = T_{[a, b]}(a_i \bar{x}_j^\pi + b_i \bar{x}_k^\pi + c_i + \pi_i)$ , and thus  $\bar{x}_i^\pi = x_i^\sigma$ , if we set  $\sigma_i := \pi_i$ .

- The simulation of a gate of type  $x_i := \min\{x_j, x_k\}$  yields that  $\bar{x}_i^\pi = \bar{x}_j^\pi - T_{[0,3B]}(\bar{x}_j^\pi - \bar{x}_k^\pi + \pi_i)$ . We claim that this implies  $\bar{x}_i^\pi = \min\{\bar{x}_j^\pi, \bar{x}_k^\pi - \pi_i\}$ , and thus  $\bar{x}_i^\pi = \min\{x_j^\sigma, x_k^\sigma + \sigma_i\} = x_i^\sigma$  by setting  $\sigma_i := -\pi_i$ . Indeed, if  $\bar{x}_j^\pi \leq \bar{x}_k^\pi - \pi_i$ , then  $\bar{x}_i^\pi = \bar{x}_j^\pi - T_{[0,3B]}(\bar{x}_j^\pi - \bar{x}_k^\pi + \pi_i) = \bar{x}_j^\pi$ . On the other hand, if  $\bar{x}_j^\pi \geq \bar{x}_k^\pi - \pi_i$ , then  $\bar{x}_i^\pi = \bar{x}_j^\pi - T_{[0,3B]}(\bar{x}_j^\pi - \bar{x}_k^\pi + \pi_i) = \bar{x}_j^\pi - (\bar{x}_j^\pi - \bar{x}_k^\pi + \pi_i) = \bar{x}_k^\pi - \pi_i$  where we used the fact that  $\bar{x}_j^\pi - \bar{x}_k^\pi + \pi_i = x_j^\sigma - x_k^\sigma + \pi_i \leq 2B + \pi_i \leq 3B$ , by construction of  $B$ .
- Similarly, the simulation of a gate of type  $x_i := \max\{x_j, x_k\}$  yields that  $\bar{x}_i^\pi = -\min\{-x_j^\sigma, -x_k^\sigma - \pi_i\} = \max\{x_j^\sigma, x_k^\sigma + \pi_i\} = \max\{x_j^\sigma, x_k^\sigma + \sigma_i\} = x_i^\sigma$  by setting  $\sigma_i := \pi_i$ .

**Construction.** Consider now a circuit  $C$  that satisfies the conditions of the lemma, but only consists of  $x_i := ax_j + bx_k + c$  and  $x_i := T_{[a,b]}(x_j)$  gates. It remains to show that we can construct a circuit  $\bar{C}$  that only uses  $T_{[0,1]}$  gates and satisfies the lemma. We let  $T := T_{[0,1]}$ .

We begin by computing in polynomial time a bound  $B \geq 2$  such that for any  $i \in [N]$ ,  $B$  satisfies

$$\max_{\sigma \in [-1,1]^m} \max_{y \in [0,1]^n} f_j^\sigma(y) \in [-B, B].$$

We also make sure that we have  $B \geq a + b + 1$  for every gate  $x_i := ax_j + bx_k + c$ , and  $B \geq b - a$  for every gate  $x_i := T_{[a,b]}(x_j)$ .

Since all gates in  $\bar{C}$  can only output values in  $[0, 1]$ , we will encode values of the original circuit, which lie in  $[-B, B]$ , inside the interval  $[1/4, 3/4]$ . A value  $x \in [-B, B]$  can be encoded inside  $[1/4, 3/4]$  by using the map

$$\phi : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \frac{1}{2} + \frac{x}{4B}.$$

Note that this transformation maps  $[-2B, 2B]$  to  $[0, 1]$ . A value  $x \in [1/4, 3/4]$  can be decoded to  $[-B, B]$  by using the map

$$\phi^{-1} : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto 4B(x - 1/2).$$

The circuit  $\bar{C}$  is constructed as follows:

- For every input variable  $x_i$  of  $C$ , let  $\bar{z}_i$  denote the corresponding input variable of  $\bar{C}$ . We compute the variable  $\bar{x}_i$  of  $\bar{C}$  by encoding  $\bar{z}_i$  into  $[1/4, 3/4]$ , that is, by letting  $\bar{x}_i := T(\phi(\bar{z}_i)) = T(1/2 - \bar{z}_i/4B)$ .
- Every gate of the type  $x_i := ax_j + bx_k + c$  is replaced by the gate  $\bar{x}_i := T(\phi(a\phi^{-1}(\bar{x}_j) + b\phi^{-1}(\bar{x}_k) + c))$ . This corresponds to decoding  $\bar{x}_j$  and  $\bar{x}_k$ , applying the original gate operation, and then encoding again. Note that the expression inside the truncation operator is affine linear, so the gate is well defined.
- Every gate of the type  $x_i := T_{[a,b]}(x_j)$  is replaced by the gate  $\bar{z}_i := T(\psi(\phi^{-1}(\bar{x}_j)))$  and the gate  $\bar{x}_i := T(\phi(\psi^{-1}(\bar{z}_i)))$ , where  $\psi$  is a linear transformation mapping  $[a, b]$  to  $[0, 1]$ , that is,

$$\psi : \mathbb{R} \rightarrow \mathbb{R}, \quad x \mapsto \frac{x - a}{b - a}.$$

The first expression corresponds to decoding  $\bar{x}_j$ , then using a linear map that maps  $[a, b]$  to  $[0, 1]$ , and then truncating to  $[0, 1]$ . The second expression corresponds to taking the inverse linear map from  $[0, 1]$  to  $[a, b]$ , and then encoding again.

- Special case: For the output gate of  $C$ , which by assumption is of type  $x_N := T_{[0,1]}(x_j)$ , we just replace it by  $\bar{x}_N := T(\phi^{-1}(\bar{x}_j))$ . This corresponds to the same construction as above, except that (i)  $a = 0$  and  $b = 1$ , and (ii) we leave the output un-encoded.

**Correctness.** Let  $K := 4B^N$  and consider any  $\delta \leq 1/K$ . Let  $\pi$  be any perturbation vector for  $\bar{C}$  with  $|\pi_i| \leq \delta$  for all  $i$ . We will show that there exists a perturbation vector  $\sigma$  for  $C$  with  $|\sigma_i| \leq \delta K$  such that  $f^\sigma(y) = \bar{f}^\pi(y)$  for all  $y \in [0, 1]^n$ .

We prove below by induction that we can construct a  $\delta K$ -perturbation  $\sigma$  of  $C$  such that for all  $i \in [N - 1]$ , there exists some  $\varepsilon_i$  with  $|\varepsilon_i| \leq 4B^i \delta$  such that

$$\bar{x}_i^\pi = \phi(x_i^\sigma + \varepsilon_i) \quad (11)$$

whenever the input to the circuit lies in  $[0, 1]^n$ .

From this, we then obtain the desired statement as follows. Recall that the last gate of  $C$  is  $x_N := T(x_j)$  and that it is replaced by  $\bar{x}_N := T(\phi^{-1}(\bar{x}_j))$  in  $\bar{C}$ . As a result, we have that  $\bar{x}_N^\pi = T(\phi^{-1}(\bar{x}_j^\pi) + \pi(\bar{x}_N))$  where  $\pi(\bar{x}_i)$  denotes the entry of  $\pi$  which is used to perturb the gate computing  $\bar{x}_i$ . Using the fact that  $\bar{x}_j^\pi = \phi(x_j^\sigma + \varepsilon_j)$  by (11), we obtain

$$\bar{x}_N^\pi = T(x_j^\sigma + \varepsilon_j + \pi(\bar{x}_N)) = T(x_j^\sigma + \sigma(x_N)) = x_N^\sigma$$

by setting  $\sigma(x_N) := \varepsilon_j + \pi(\bar{x}_N)$ , which satisfies  $|\sigma(x_N)| \leq 4B^{N-1} \delta + \delta \leq 4B^N \delta = \delta K$ . As a result, we have that  $f^\sigma(y) = x_N^\sigma = \bar{x}_N^\pi = \bar{f}^\pi(y)$  for all  $y \in [0, 1]^n$ , as desired.

It remains to prove (11). It is easy to see that the statement holds for the input variables, that is, for  $i \in [n]$ . Indeed, by definition, we have  $x_i^\sigma = \bar{z}_i^\pi$  for all input variables, that is, for all  $i \in [n]$ . Thus, we can write

$$\bar{x}_i^\pi = T(\phi(\bar{z}_i^\pi) + \pi(\bar{x}_i)) = T(\phi(x_i^\sigma + 4B\pi(\bar{x}_i))) = T(\phi(x_i^\sigma + \varepsilon_i)) = \phi(x_i^\sigma + \varepsilon_i)$$

by setting  $\varepsilon_i := 4B\pi(\bar{x}_i)$ , which satisfies  $|\varepsilon_i| \leq 4B\delta \leq 4B^i \delta$ , as desired. We also used the fact that  $\phi(x_i^\sigma + \varepsilon_i) \in [0, 1]$ , since  $x_i^\sigma + \varepsilon_i \in [-2B, 2B]$ , because  $x_i^\sigma \in [-B, B]$  by construction of  $B$ , and  $|\varepsilon_i| \leq 1 \leq B$  since  $\delta \leq 1/K = 1/4B^N$ .

Now, consider any  $i \in [N - 1]$  with  $i > n$ , and assume that the induction hypothesis holds for smaller values of  $i$ . We handle the following two cases separately:

- If the  $i$ th gate is of the type  $x_i := ax_j + bx_k + c$ , then by construction we have that

$$\begin{aligned} \bar{x}_i^\pi &= T(\phi(a\phi^{-1}(\bar{x}_j^\pi) + b\phi^{-1}(\bar{x}_k^\pi) + c) + \pi(\bar{x}_i)) \\ &= T(\phi(ax_j^\sigma + ae_j + bx_k^\sigma + be_k + c) + \pi(\bar{x}_i)) \\ &= T(\phi(x_i^\sigma + ae_j + be_k + 4B\pi(\bar{x}_i))) \\ &= T(\phi(x_i^\sigma + \varepsilon_i)) = \phi(x_i^\sigma + \varepsilon_i), \end{aligned}$$

where we set  $\varepsilon_i := ae_j + be_k + 4B\pi(\bar{x}_i)$ , which satisfies  $|\varepsilon_i| \leq (a + b)4B^{i-1} \delta + 4B\delta \leq 4B^i \delta$  by construction of  $B$ , and where we also used the induction hypothesis, namely  $\bar{x}_j^\pi = \phi(x_j^\sigma + \varepsilon_j)$  and  $\bar{x}_k^\pi = \phi(x_k^\sigma + \varepsilon_k)$ . We also used the fact that  $\phi(x_i^\sigma + \varepsilon_i) \in [0, 1]$ , because  $x_i^\sigma + \varepsilon_i \in [-2B, 2B]$  as before.

- If the  $i$ th gate is of the type  $x_i := T_{[a, b]}(x_j)$ , then by construction, we have that

$$\begin{aligned} \bar{z}_i^\pi &= T(\psi(\phi^{-1}(\bar{x}_j^\pi)) + \pi(\bar{z}_i)) = T(\psi(\phi^{-1}(\bar{x}_j^\pi) + (b - a)\pi(\bar{z}_i))) \\ &= \psi(T_{[a, b]}(\phi^{-1}(\bar{x}_j^\pi) + (b - a)\pi(\bar{z}_i))) \\ &= \psi(T_{[a, b]}(x_j^\sigma + \varepsilon_j + (b - a)\pi(\bar{z}_i))) \\ &= \psi(T_{[a, b]}(x_j^\sigma + \sigma(x_i))) = \psi(x_i^\sigma), \end{aligned}$$

where we set  $\sigma(x_i) := \varepsilon_j + (b - a)\pi(\bar{z}_i)$ , which satisfies  $|\sigma(x_i)| \leq 4B^{i-1}\delta + B\delta \leq 4B^i\delta \leq \delta K$ , and where we also used the induction hypothesis  $\bar{x}_j^\pi = \phi(x_j^\sigma + \varepsilon_j)$ . Thus, it follows that

$$\begin{aligned}\bar{x}_i^\pi &= T(\phi(\psi^{-1}(\bar{z}_i^\pi)) + \pi(\bar{x}_i)) = T(\phi(x_i^\sigma) + \pi(\bar{x}_i)) \\ &= T(\phi(x_i^\sigma + 4B\pi(\bar{x}_i))) \\ &= T(\phi(x_i^\sigma + \varepsilon_i)) = \phi(x_i^\sigma + \varepsilon_i),\end{aligned}$$

where we set  $\varepsilon_i := 4B\pi(\bar{x}_i)$ , which satisfies  $|\varepsilon_i| \leq 4B\delta \leq 4B^i\delta$ , and where we also used the fact that  $\phi(x_i^\sigma + \varepsilon_i) \in [0, 1]$ , since  $x_i^\sigma + \varepsilon_i \in [-2B, 2B]$  as before.

Thus, (11) follows by induction, and the proof of Lemma 5.2 is complete.

Received 24 July 2024; revised 25 May 2025; accepted 26 May 2025

# Parameterized Inapproximability Hypothesis under ETH

**VENKATESAN GURUSWAMI**, University of California Berkeley, Berkeley, United States

**BINGKAI LIN**, Nanjing University, Nanjing, China

**XUANDI REN**, University of California Berkeley, Berkeley, United States

**YICAN SUN**, Peking University, Beijing, China

**KEWEN WU**, University of California Berkeley, Berkeley, United States

---

The Parameterized Inapproximability Hypothesis (PIH) asserts that no fixed parameter tractable (FPT) algorithm can distinguish a satisfiable CSP instance, parameterized by the number of variables, from one where every assignment fails to satisfy an  $\varepsilon$  fraction of constraints for some absolute constant  $\varepsilon > 0$ . PIH plays the role of the PCP theorem in parameterized complexity. However, PIH has only been established under the Gap Exponential Time Hypothesis (ETH), a very strong assumption with an inherent gap.

In this work, we prove PIH under the ETH. This is the first proof of PIH from a gap-free assumption. Our proof is self-contained and elementary. We identify an ETH-hard CSP whose variables take vector values, and constraints are either linear or of a special parallel structure. Both kinds of constraints can be checked with constant soundness via a “parallel PCP of proximity” based on the Walsh-Hadamard code.

**CCS Concepts:** • **Theory of computation → Fixed parameter tractability; Problems, reductions and completeness;**

**Additional Key Words and Phrases:** Fixed-parameter algorithms and complexity, Hardness of approximations, PCP theorems

**ACM Reference Format:**

Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, Yican Sun, and Kewen Wu. 2025. Parameterized Inapproximability Hypothesis under ETH. *J. ACM* 72, 5, Article 32 (October 2025), 40 pages. <https://doi.org/10.1145/3749982>

---

This is an invited paper to JACM from the 2024 ACM Symposium of Theory of Computing (STOC).

Venkatesan Guruswami research supported in part by NSF grants CCF-2228287 and CCF-2211972 and a Simons Investigator award.

Xuandi Ren supported in part by NSF grant CCF-2228287.

Kewen Wu supported by a Sloan Research Fellowship and NSF CAREER Award CCF-2145474.

Authors' Contact Information: Venkatesan Guruswami, University of California Berkeley, Berkeley, California, United States; e-mail: venkatg@berkeley.edu; Bingkai Lin, Nanjing University, Nanjing, Jiangsu, China; e-mail: lin@nju.edu.cn; Xuandi Ren, University of California Berkeley, Berkeley, California, United States; e-mail: xuandi\_ren@berkeley.edu; Yican Sun, Peking University, Beijing, Beijing, China; e-mail: sycpk@pku.edu.cn; Kewen Wu, University of California Berkeley, Berkeley, California, United States; e-mail: shlw\_kevin@hotmail.com.



[This work is licensed under a Creative Commons Attribution 4.0 International License.](#)

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART32

<https://doi.org/10.1145/3749982>

## 1 Introduction

A comprehensive understanding of NP-hard problems is an everlasting pursuit in the TCS community. Toward this goal, researchers have proposed many alternative hypotheses as strengthenings of the classic  $P \neq NP$  assumption to obtain more fine-grained lower bounds for NP-hard problems, for example, **Exponential Time Hypothesis (ETH)** [47], **Strong ETH (SETH)** [15, 47], **Gap ETH (Gap-ETH)** [25].

Besides a richer family of hypotheses, *approximation* and *fixed parameter tractability (FPT)* are also two orthogonal approaches to cope with NP-hardness.

- In the approximation setting, we consider optimization problem, where input instances are associated with a cost function and the goal is to find a solution with cost function value close to the optimum.
- In the **fixed parameter tractability (FPT)** setting, every instance is attached with a parameter  $k$  indicating specific quantities (e.g., the optimum, the treewidth) of the instance. This setting treats  $k$  as a parameter much smaller than the instance size  $n$ , i.e.,  $1 \leq k \ll n$ . Thus, the required runtime of the algorithm is relaxed from  $n^{O(1)}$  to  $f(k) \cdot n^{O(1)}$ , for any computable function  $f$ . The class FPT is the set of parameterized problems that admit an algorithm within this running time.

The seminal studies in this setting built up *parameterized complexity theory* [30, 31, 36]. In this theory, there are also alternative hypotheses as a strengthening of  $P \neq NP$ . For example,  $W[1] \neq FPT$ , which is equivalent to the statement that  $k$ -CLIQUE has no  $f(k) \cdot n^{O(1)}$ -time algorithm.

Recently, there has been an extensive study at the intersection of these two settings: the existence (or absence) of approximation algorithms that solve NP-hard problems in FPT time.

- On the algorithmic side, approximation algorithms with FPT runtime have been designed for various NP-complete problems. Examples include VERTEX-COLORING [23, 67], MIN- $k$ -CUT [40, 41, 54, 64],  $k$ -PATH-DELETION [55],  $k$ -CLUSTERING [1],  $k$ -MEANS /  $k$ -MEDIANs [2, 14, 17, 22, 49, 56], MAX  $k$ -HYPERGRAPH VERTEX COVER [65, 70], FLOW TIME SCHEDULING [72].
- In terms of computational hardness, the existence of such algorithms for certain NP-complete problems has also been ruled out under reasonable assumptions:  $k$ -SETCOVER [16, 20, 21, 51, 52, 58, 61],  $k$ -SETINTERSECTION [13, 57],  $k$ -STEINER ORIENTATION [73], MAX- $k$ -COVERAGE [66],  $k$ -SVP,  $k$ -MINDISTANCEPROBLEM and related problems [10, 11, 66].

An exciting recent line of work [16, 18, 21, 50, 59, 60, 62] shows that approximating  $k$ -CLIQUE is not FPT under Gap-ETH, ETH, and  $W[1] \neq FPT$ .

We refer to the survey by Feldmann, Karthik, Lee, and Manurangsi [34] for a detailed discussion.

*The Quest for Parameterized PCP-Type Theorems.* Despite all the recent progress in the study of parameterized inapproximability, the reductions presented in the papers above are often ad-hoc and tailored to the specific problems in question. Obtaining a unified and powerful machinery for parameterized inapproximability, therefore, becomes increasingly important.

A good candidate is to establish a *parameterized PCP-type theorem*. The PCP theorem [6, 7, 24], a cornerstone of modern complexity theory, gives a polynomial time reduction from an NP-hard problem like 3SAT to a gap version of 3SAT where the goal is to distinguish satisfiable instances from those for which every assignment fails to satisfy a  $\gamma$  fraction of clauses for some absolute constant  $\gamma > 0$ . This then serves as the starting point for a large body of inapproximability results for fundamental problems, including constraint satisfaction, graph theory, and optimization [5, 29, 32, 33].

As discussed in Reference [34], the current situation in the parameterized world is similar to that of the landscape of the traditional hardness of approximation *before* the celebrated PCP theorems was established. Given the similarity, the following folklore open problem has been recurring in the field of parameterized inapproximability:

*Can we establish a PCP-type theorem in parameterized complexity theory?*

In light of its rising importance, Lokshtanov, Ramanujan, Saurabh, and Zehavi [63] formalized and titled the above question as **Parameterized Inapproximability Hypothesis (PIH)**. Here, we present the following reformulation<sup>1</sup> of PIH due to Reference [34]:

**HYPOTHESIS 1.1 (PARAMETERIZED INAPPROXIMABILITY HYPOTHESIS).** *There is an absolute constant<sup>2</sup>  $\varepsilon > 0$ , such that no fixed parameter tractable algorithm which, takes as input a 2-CSP<sup>3</sup>  $G$  with  $k$  variables of size- $n$  alphabets, can decide whether  $G$  is satisfiable or at least  $\varepsilon$  fraction of constraints must be violated.*

Similar to the PCP theorem, PIH, if true, serves as a shared beginning for results in parameterized hardness of approximation:  $k$ -CLIQUE,  $k$ -SETCOVER,  $k$ -EXACTCOVER [45], SHORTEST VECTOR [10, 11], DIRECT ODD CYCLE TRANSVERSAL [63], DETERMINANT MAXIMIZATION and GRID TILING [68], Baby PIH [45],  $k$ -MAXCOVER [51], and more.

Prior to our work, PIH was only proved under the Gap-ETH assumption [25], the gap version of ETH. Since there is an inherent gap in Gap-ETH, the result can be obtained by a simple gap-preserving reduction (see, e.g., Reference [34]). Indeed, it is often recognized that gap-preserving reductions are much easier than gap-producing reductions [33]. A more desirable result is, analogous to the PCP theorem, to create a gap from a gap-free assumption:

*Can we prove PIH under an assumption without an inherent gap?*

## 1.1 Our Results

We answer the above question in the affirmative by proving the *first* result to base PIH on a gap-free assumption. We consider the celebrated ETH [47], a fundamental gap-free hypothesis in modern complexity theory and a weakening of the Gap-ETH assumption.<sup>4</sup>

**HYPOTHESIS (EXPONENTIAL TIME HYPOTHESIS (ETH), INFORMAL).** *Solving 3SAT needs  $2^{\Omega(n)}$  time.<sup>5</sup>*

Our main theorem can be stated concisely as:

**THEOREM 1.2 (MAIN).** *ETH implies PIH.*

In Theorem 3.1, we provide a quantitative version of Theorem 1.2 that presents an explicit runtime lower bound of  $f(k) \cdot n^{\Omega(\sqrt[4]{\log k})}$  for the problem in Hypothesis 1.1 under ETH.<sup>6</sup>

<sup>1</sup>The original statement of PIH in Reference [63] replaces the runtime bound by W[1]-hardness, and the reformulation by Reference [34] suffices for applications.

<sup>2</sup>The exact constant here is not important. Starting from a constant  $\varepsilon > 0$ , one can boost it to  $1 - \eta$  for any constant  $\eta > 0$  by standard reductions.

<sup>3</sup>See Section 2.1 for the formal definition of Constraint Satisfaction Problems.

<sup>4</sup>We thank Benny Applebaum for pointing his work [3] to us, which shows that smooth-ETH, a strengthening of ETH, implies Gap-ETH (and thus PIH). In contrast, we prove PIH directly from ETH, completely bypassing Gap-ETH.

<sup>5</sup>See Hypothesis 2.3 for the formal definition.

<sup>6</sup>In the conference version of this article [43], we got an  $f(k) \cdot n^{\Omega(\sqrt{\log \log k})}$  lower bound. Here, we improve it to  $f(k) \cdot n^{\Omega(\sqrt[4]{\log k})}$  by incorporating some ideas from Reference [42] to simplify the reduction.

As a byproduct of the above quantitative bound, we have the following probabilistic checkable proof version of the main theorem (see Theorem 3.2 for the full version). This can be seen as a PCP theorem in the parameterized world where the proof length depends only on  $k$  (which is supposed to be a small growing parameter), but the alphabet size is the significantly growing parameter. The runtime of the PCP verifier is in FPT.

**THEOREM 1.3.** *For any integer  $k \geq 1$ , 3SAT has a PCP verifier which can be constructed in time  $f(k) \cdot |\Sigma|^{O(1)}$  for some computable function  $f$ , makes two queries on a proof with length  $2^{O(k^4)}$  and alphabet size  $|\Sigma| = 2^{O(n/k)}$ , and has completeness 1 and soundness  $1 - \frac{1}{9600}$ .*

As mentioned, PIH serves as a unified starting point for many parameterized inapproximability results. Below, we highlight some ETH-hardness of approximation for fundamental parameterized problems obtained by combining our result and existing reductions from PIH.

*Application Highlight:* MAX  $k$ -COVERAGE. The MAX  $k$ -COVERAGE is the maximization variant of the  $k$ -SETCOVER problem, where we are given a universe  $U$  and a collection  $\mathcal{S}$  of subsets of  $U$ . The goal is to choose  $k$  sets with the maximum union size. Its approximation version, denoted by MAX  $(\rho, k)$ -COVERAGE, asks to distinguish between the following two cases:

- There exists  $k$  subsets from  $\mathcal{S}$  whose union equals  $U$ .
- Any  $k$  subsets from  $\mathcal{S}$  has the union size at most  $\rho \cdot |U|$ .

On the algorithmic side, there exists a simple greedy algorithm solving MAX  $(1 - \frac{1}{e}, k)$ -COVERAGE within polynomial runtime [46]. On the hardness side, the inapproximability of MAX  $k$ -COVERAGE implies inapproximability of fundamental clustering problems, namely,  $k$ -MEANS and  $k$ -MEDIAN [39]. The celebrated result of Feige [32] showed the NP-hardness of MAX  $(1 - \frac{1}{e} + \varepsilon, k)$ -COVERAGE for any  $\varepsilon > 0$ , thus proving a tight inapproximability result.

In the parameterized world, Cohen-Addad, Gupta, Kumar, Lee, and Li [22] showed that assuming Gap-ETH, MAX  $(1 - \frac{1}{e} + \varepsilon, k)$ -COVERAGE requires  $f(k)|\mathcal{S}|^{k^{\text{poly}(1/\varepsilon)}}$  runtime. Manurangsi [66] further improved this lower bound to the tightest  $f(k)|\mathcal{S}|^{O(k)}$  under Gap-ETH. Our work implies the tight FPT inapproximability of MAX  $(\rho, k)$ -COVERAGE under ETH.

**THEOREM 1.4.** *Assume ETH. For any constant  $\varepsilon \in (0, 1/e)$ , no FPT algorithm can decide MAX  $(1 - \frac{1}{e} + \varepsilon, k)$ -COVERAGE.*

The proof of Theorem 1.4 follows the classic approach from [32, 66] and is deferred to Appendix A.

*Application Highlight:*  $k$ -EXACTCOVER.  $k$ -EXACTCOVER (also known as  $k$ -UNIQUE SET COVER) is a variant of the famous  $k$ -SETCOVER problem. In the  $\rho$ -approximation version of this problem, denoted by  $(k, \rho \cdot k)$ -EXACTCOVER, we are given a universe  $U$  and a collection  $\mathcal{S}$  of subsets of  $U$ . The goal is to distinguish between the following two cases.

- There exists  $k$  disjoint subsets that cover the whole universe  $U$ , i.e., the union of the  $k$  subsets is exactly the whole universe  $U$ .
- Any  $\rho \cdot k$  subsets of  $\mathcal{S}$  cannot cover  $U$ .

Here, the parameter is the optimum  $k$ . Note that  $k$ -EXACTCOVER is an easier problem than  $k$ -SETCOVER due to the additional disjointness property, which makes its computational hardness even more difficult to prove. On the positive side, this additional structure also makes  $(k, \rho \cdot k)$ -EXACTCOVER an excellent proxy for subsequent reductions. We refer interested readers to previous works for details [4, 66].

For constant  $\rho > 1$ , the hardness of  $(k, \rho \cdot k)$ -EXACTCOVER was only proved under assumptions with inherent gaps [45, 66], imitating the reduction in the non-parameterized world [32]. It was still open whether we could derive the same result under a weaker and gap-free assumption. Combining its PIH hardness (see e.g., Reference [45]<sup>7</sup>) with our main theorem (Theorem 1.2), we prove the first inapproximability for  $k$ -EXACTCOVER under a gap-free assumption.

**COROLLARY 1.5.** *Assuming ETH, for any absolute constant  $\rho \geq 1$ , no FPT algorithm can decide  $(k, \rho \cdot k)$ -EXACTCOVER.*

*Application Highlight: Directed Odd Cycle Transversal.* Given a directed graph  $D$ , its directed odd cycle transversal, denoted by DOCT( $D$ ), is the minimum set  $S$  of vertices such that deleting  $S$  from  $D$  results in a graph with no directed odd cycles. The  $\rho$ -approximating version of the directed odd cycle transversal problem, denoted by  $(k, \rho \cdot k)$ -DOCT, is to distinguish directed graphs  $D$  with  $\text{DOCT}(D) \leq k$ , from those with  $\text{DOCT}(D) \geq \rho \cdot k$ . The parameter of this problem is the optimum  $k$ . This problem is a generalization of several well studied problems including DIRECTED FEEDBACK VERTEX SET and ODD CYCLE TRANSVERSAL. For a brief history of this problem, we refer to the previous work [63]. In their work, the authors prove the following hardness of  $(k, \rho \cdot k)$ -DOCT.

**THEOREM 1.6 ([63]).** *Assuming PIH, for some  $\rho \in (1, 2)$ , no FPT algorithm can decide  $(k, \rho \cdot k)$ -DOCT.*

Combining the theorem above with Theorem 1.2, we establish the first hardness of  $(k, \rho \cdot k)$ -DOCT under a gap-free assumption.

**COROLLARY 1.7.** *Assuming ETH, for some  $\rho \in (1, 2)$ , no FPT algorithm can decide  $(k, \rho \cdot k)$ -DOCT.*

## 1.2 Overview of Techniques

To prove our main theorem (Theorem 1.2), we present an efficient reduction from 3SAT formulas to PARAMETERIZED CSP on  $k$  variables with a constant gap.

To construct such a reduction, we follow the widely-used paradigm for proving PCP theorems [6, 7, 44]. Via this approach, we first arithmetize 3SAT into an intermediate CSP (usually a constant-degree polynomial system in the literature) with  $k$  variables and alphabet  $\Sigma_1$ . Then, we select a suitable locally testable and correctable code  $C : \Sigma_1^k \rightarrow \Sigma_2^{k'}$  (e.g., the quadratic code [6], the Reed-Muller code [7], or the long code [44]), and treat the proof  $\pi$  as an encoding of some assignment  $\sigma : [k] \rightarrow \Sigma_1$  to the intermediate CSP (viewed as a vector in  $\Sigma_1^k$ ). Leveraging the power of the local testability and correctability of  $C$ , we will check whether the input proof is (close to) the encoding of an assignment that satisfies the intermediate CSP.

*Our Strategy.* To follow the outline above and also factor in the runtime and the parameter blowup, our plan is as follows:

- (1) First, we need to design an appropriate intermediate CSP problem, which has some runtime lower bound under ETH. In the parameterized setting, the number of variables is a small parameter  $k$ , while the alphabet  $|\Sigma_1|$  holds the greatest order of magnitude.
- (2) Second, we need to construct an error correcting code  $C$ , which can be used to encode a solution of the intermediate CSP, and allows us to locally check its satisfiability. Here, the efficiency of the code is also measured in the parameterized sense. Specifically, the alphabet size is polynomial in  $|\Sigma_1|$ , and codeword length can be arbitrary in  $k$  but independent of  $|\Sigma_1|$ .

<sup>7</sup>Reference [45] proves the hardness of  $(k, \rho \cdot k)$ -EXACTCOVER under a weaker version of PIH, namely, Average Baby PIH with rectangular constraints.

However, the plan above confronts the following basic obstacle. The constructions in proving the PCP theorems usually require the proof length  $|\pi| = |\Sigma_1|^{\Omega(k)}$ . On the other hand, as illustrated in Item 2 above, we must eliminate  $|\Sigma_1|$  in the proof length to make sure the reduction is FPT.

*Vectorization.* We bypass this obstacle by applying *vectorization*, an idea also used in [62]. In detail, we enforce the alphabet  $\Sigma_1$  to be a vector space  $\mathbb{F}^d$ , where  $\mathbb{F}$  is a field of constant size. In this way, an assignment  $\sigma \in \Sigma_1^k = (\mathbb{F}^d)^k$  can be viewed as  $d$  parallel sub-assignments in  $\mathbb{F}^k$ . Thus, if we have a good code  $C : \mathbb{F}^k \rightarrow \mathbb{F}^{k'}$  that tests the validity of a sub-assignment, we can encode  $\sigma$  by separately encoding each sub-assignment and combining them as an element in  $(\mathbb{F}^{k'})^d = (\mathbb{F}^d)^{k'}$ . Since  $|\mathbb{F}|$  is a constant, this makes  $k'$  dependent only on  $k$  but not on the whole alphabet  $\Sigma_1 = \mathbb{F}^d$ .

Guided by the vectorization idea, we aim to design an ETH-hard intermediate CSP problem where the alphabet is a vector space. Furthermore, to facilitate the construction of the error correcting code  $C$  in the second step, we also hope that there are appropriate restrictions on the constraints of this intermediate CSP problem. The constraints should be neither too restrictive (which loses the ETH-hardness) nor too complicated (which hinders an efficient testing procedure). Following these intuitions, we define the following *Vector-Valued CSPs* as our intermediate problem.

*Vector-Valued CSPs.* Vector-Valued CSPs (Definition 3.3) are CSPs with the following additional features:

- The parameter  $k$  is the number of variables.
- The alphabet is a vector space  $\mathbb{F}^d$ , where  $\mathbb{F}$  is a finite field of characteristic two and constant size and  $d$  holds the greatest order of magnitude.
- Each constraint is either a (coordinate-wise) parallel constraint or a linear constraint, where:
  - A parallel constraint is defined by a sub-constraint  $\Pi^{sub} : \mathbb{F} \times \mathbb{F} \rightarrow \{0, 1\}$ . It checks  $\Pi^{sub}$  for every coordinate of the vector assignments.
  - A linear constraint is defined by a matrix  $M$ . It enforces that two vector assignments satisfy a linear equation specified by  $M$ .
- Each variable is related to at most one parallel constraint.

We emphasize that vector-valued CSPs will become fixed parameter tractable if all constraints are linear (resp., parallel). In detail, one can handle linear constraints by efficiently solving a system of linear equations, or handle parallel constraints by brute force enumeration individually for each coordinate. However, due to our reduction, one cannot solve vector-valued CSPs with both constraint types efficiently under ETH.

*3SAT to Vector-Valued CSPs.* In Theorem 3.4, we establish the ETH-hardness of vector-valued CSP instances by a series of standard transformations.

First, we reduce 3SAT to a PARAMETERIZED CSP instance. We partition the  $n$  variables and  $O(n)$  clauses of a 3SAT formula,<sup>8</sup> respectively, into  $k$  parts each. The  $2k$  parts are then built as CSP variables, which take assignments of the part of the clauses/variables. The alphabet is therefore a vector space.

Then, we impose constraints between clause parts and variable parts. Each constraint is a conjunction of clause validity and clause-variable consistency. These constraints ensure that the  $2k$  partial assignments correspond to a global satisfying assignment to the original 3SAT formula.

---

<sup>8</sup>Without loss of generality, we can assume the number of clauses is linear in the number of variables [47].

However, the constraints above are neither parallel nor linear. To make them parallel, we first appropriately split constraints, then duplicate each variable into several copies and spread out its constraints. After this procedure, each variable is related to exactly one constraint, and each constraint is the same sub-constraint applied in a matching way on the  $d$  coordinates of the related vector variables. We can thus add auxiliary variables to permute the  $d$  coordinates of each variable accordingly. As a result, the sub-constraints become coordinate-wise. However, the parallel sub-constraint is imposed on a subset of the coordinates rather than every coordinate. Thus, we do more duplications of variables to obtain the parallel constraint form that we desire. In addition, we need to check the consistency of the different duplicates. These checks can be done using equality constraints, a particular form of linear constraints.

*Vector-Valued CSPs to Constant-Gap CSPs.* In Theorem 3.5, we construct another FPT reduction from a vector-valued CSP to a general constant-gap PARAMETERIZED CSP in three steps.

- First, we split the vector-valued CSP instance into two by partitioning the constraints into linear and parallel parts.
- Next, we construct a randomized PCP verifier for each of the two sub-instances to check whether all constraints are satisfied. The verifier takes as input a parallel encoding of a solution. It then flips random coins, makes a constant number of queries based on the randomness, and decides whether to accept the input proof based on the query result. The verifier will have a constant soundness and a constant proximity parameter. In the traditional complexity theory, such verifiers are also known as **Probabilistic Checkable Proof of Proximity (PCPP)** verifiers.
- In our proof, the verifier is designed separately for linear and parallel constraints. The consistency of the two verifiers is guaranteed via a unified parallel Walsh-Hadamard encoding of the solution shared by both verifiers.
- Finally, we obtain a constant-gap CSP instance by a standard reduction from probabilistic checkable proof verifiers to CSPs.

The proof is then completed by combining the two reductions above. The crux of our proof is the design of the PCPP verifiers. Below, we present high-level descriptions of this part.

*PCPPs for Vector-Valued CSPs with Parallel Constraints.* Fix a vector-valued CSP instance  $G$  with only parallel constraints. The key observation in designing PCPPs for  $G$  is that, though the parallel sub-constraints can be arbitrary, different coordinates of the vector-variables are independent. Let  $k$  be the number of variables in  $G$ , and let  $d$  be the dimension of the vector variables.

Following the observation above, we can split  $G$  into  $d$  sub-instances  $G_1, \dots, G_d$  with respect to the  $d$  coordinates. Each  $G_i$  is a CSP instance with  $k$  variables and alphabet  $\mathbb{F}$ .

The key insight is that since every parallel constraint sets up the same sub constraint over all coordinates, the sub-instances  $G_1, \dots, G_d$  are exactly the same instance  $G_0^*$ . Thus, we can follow the classical construction [5, 6] of PCPPs to construct a verifier  $A^*$  to efficiently and locally check the satisfiability of  $G_0^*$ . Furthermore, since  $G_0^*$  is over  $k$  variables and has the constant alphabet  $\mathbb{F}$ , the proof length required for  $A^*$  depends only on  $k$ . Finally, we parallelize  $A^*$  into  $d$ -dimensions to obtain a single verifier  $A$  that works over the original alphabet  $\mathbb{F}^d$  with blowup dependent only on  $k$ , not  $d$ . See Section 5 for details.

*PCPPs for Vector-Valued CSPs with Linear Constraints.* To design a verifier for linear constraints, we leverage the power of the Walsh-Hadamard code to decode any linear combinations of the

messages. Fix a vector-valued CSP instance  $G$  with linear constraints only. For each linear constraint  $e = (u_e, v_e) \in E$ , we further denote its form by  $1_{u_e=M_e}v_e$ .

To test the conjunction of all linear constraints, it is natural to consider the linear combination of these constraints. In detail, we pick independently random  $\lambda_1, \dots, \lambda_{|E|} \in \mathbb{F}$ , and test whether:

$$\sum_{e \in E} \lambda_e u_e = \sum_{e \in E} \lambda_e \cdot M_e v_e \quad (1)$$

By the random subsum principle [5], if any one of the linear constraints is violated, the equation above does not hold with high probability.

Following this idea, we introduce auxiliary variables  $z_{v,e}$  for each variable  $v$  and constraint  $e$ , which is supposed to be  $M_e v$ . We set up the parallel version of the Walsh-Hadamard code over the assignments to the variables in  $G$  and the auxiliary variables  $z_{v,e}$ . In this way, we can decode both the LHS and RHS of Equation (1) above by two queries on the Walsh-Hadamard code, and then check whether the equation holds.

We need extra testing procedures to ensure  $z_{v,e}$  equals  $M_e v$ . This is again achieved by the random subsum principle. See Section 6 for details.

### 1.3 Discussions

Here we discuss related works and propose future directions.

*Related Works.* As mentioned above, prior to our work, PIH was only known to hold under Gap-ETH [16, 27]. The techniques there do not apply here since their proofs rely on an inherent gap from the assumption, which ETH does not have.

Using a different approach, Lin, Ren, Sun, and Wang [60, 62] proposed to prove PIH via a strong lower bound for constant-gap  $k$ -CLIQUE. This is reminiscent of [8], where the NP-hardness of constant-gap CLIQUE leads to a free-bit PCP. However, the construction in Reference [8] does not apply in the parameterized setting since the proof length will be too long. In addition, the framework of [62] only designs a weaker variant of PCPP, which can only locally test the validity of a single constraint rather than the conjunction of all constraints. Moreover, the boosting from weak PCPPs to standard PCPPs seems to meet an information-theoretic barrier from locally decodable codes [53]. In contrast, we successfully design PCPPs for special CSPs in this work, which is based on a key observation that CSPs remain ETH-hard even when the variables are vector-valued and the constraints are either parallel or linear.

Furthermore, a recent work by Guruswami, Ren, and Sandeep [45] established a weaker version of PIH called Baby PIH, under  $W[1] \neq FPT$ . However, they also gave a counterexample to show that the basic direct product approach underlying their reduction is not strong enough to establish PIH.

*Future Directions.* We highlight some interesting open directions.

- Starting from PIH and by our work, many previous parameterized hardness of approximation results can now be based on ETH (see Corollary 1.5 and 1.7 as representatives). However, there are still many basic problems whose parameterized inapproximability remains unknown, e.g.,  $k$ -BALANCED BICLIQUE [16, 34], DENSEST  $k$ -SUBGRAPH [16].

Can we discover more ETH-based parameterized inapproximability results? Since there is already a gap in PIH, we expect that reducing PIH to other parameterized problems would be easier than reducing directly from gap-free 3SAT.

- We have presented a gap-producing reduction from ETH to PIH. It is natural to ask whether we can prove PIH under the minimal hypothesis  $W[1] \neq FPT$ . Our article constructs an FPT reduction from vector-valued CSPs to gap CSPs. In light of this, all we need is to establish the  $W[1]$ -hardness for the *exact* version of vector-valued CSPs. We remark that our reduction starts from vector-valued CSP that is  $M[1]$ -complete [35] where  $M[1]$  is an intermediate complexity class between  $FPT$  and  $W[1]$ . Thus, unless  $M[1] = W[1]$ , our proof may not be directly generalized to prove PIH under  $W[1] \neq FPT$ . We refer interested readers to [19] for a detailed discussion of these complexity classes and hierarchies.
- It is also of great interest to get a more succinct parameterized PCP, to reach a tighter runtime lower bound of  $\varepsilon$ -GAP PARAMETERIZED CSP, as compared with Theorem 3.1. This will lead to tighter runtime lower bounds for approximating many parameterized problems (e.g.  $k$ -CLIQUE, MAX  $k$ -COVERAGE,  $k$ -EXACTCOVER, etc.) under ETH. This is studied by a follow-up work [42] by the same set of authors, where the quantitative runtime lower bounds are improved to almost optimal. The current article incorporates some (but not all) ideas in [42] to simplify the reduction—in particular, we have used the notion of a more structured form of vector CSP, but still employ the Hadamard based PCP instead of the (more complicated) succinct PCPs based on higher-degree polynomials. As a result, this article achieves better quantitative bounds than the conference version [43], but not optimal as [42].

*Paper Organization.* In Section 2, we define necessary notation and introduce useful tools. Then, the article is organized in a modular manner. First, in Section 3, we present the proof of our main result with the proofs of technical lemmas deferred to later sections. Then, in Section 4, we show how to obtain a vector-valued CSP instance with desired structures from 3SAT as needed in Section 3. Next, in Section 5, we design the probabilistic verifier for parallel constraints in the CSP instance, another building block needed in Section 3. Finally, in Section 6, we give the probabilistic verifier for linear constraints in the CSP instance, the last missing piece of Section 3.

## 2 Preliminaries

For a positive integer  $n$ , we use  $[n]$  to denote the set  $\{1, 2, \dots, n\}$ . We use  $\log$  to denote the logarithm with base 2. For an event  $\mathcal{E}$ , we use  $1_{\mathcal{E}}$  as the indicator function, which equals 1 if  $\mathcal{E}$  happens and equals 0 if otherwise. For disjoint sets  $S$  and  $T$ , we use  $S \dot{\cup} T$  to denote their union while emphasizing  $S \cap T = \emptyset$ . For a prime power  $q = p^t$  where  $p$  is a prime and  $t \geq 1$  is an integer, we use  $\mathbb{F}_q$  to denote the finite field of order  $p^t$  and characteristic  $p$ .

We use superscript  $\top$  to denote vector and matrix transpose. For two vectors  $u, v \in \mathbb{F}^d$ , we use  $\langle u, v \rangle$  to denote their inner product which equals  $u^\top v$  (or  $v^\top u$ ). For two matrices  $A, B \in \mathbb{F}^{d \times d}$ , we use  $\langle A, B \rangle = \sum_{i,j \in [d]} A_{i,j} B_{j,i}$  to denote their inner product.

Throughout the article, we use  $O(\cdot), \Theta(\cdot), \Omega(\cdot)$  to hide absolute constants that do not depend on any other parameter.

### 2.1 (Parameterized) Constraint Satisfaction Problems

*Constraint Satisfaction Problem.* In this article, we only focus on constraint satisfaction problems (CSPs) of arity two. Formally, a CSP instance  $G$  is a quadruple  $(V, E, \Sigma, \{\Pi_e\}_{e \in E})$ , where:

- $V$  is the set of variables.
- $E$  is the set of constraints. Each constraint  $e = \{u_e, v_e\} \in E$  has arity 2 and is related to two distinct variables  $u_e, v_e \in V$ .

The *constraint graph* is the undirected graph on vertices  $V$  and edges  $E$ . Note that we allow multiple constraints between every pair of variables, and thus, the constraint graph may have parallel edges.

- $\Sigma$  is the alphabet of each variable in  $V$ . For convenience, we sometimes have different alphabets for different variables, and we will view them as a subset of a larger alphabet  $\Sigma$  with some natural embedding.
- $\{\Pi_e\}_{e \in E}$  is the set of constraint validity functions. Given a constraint  $e \in E$ , the validity function  $\Pi_e(\cdot, \cdot) : \Sigma \times \Sigma \rightarrow \{0, 1\}$  checks whether the constraint  $e$  between  $u_e$  and  $v_e$  is satisfied.

We use  $|G| = (|V| + |E|) \cdot |\Sigma|$  to denote the *size* of a CSP instance  $G$ .

*Assignment and Satisfiability Value.* An *assignment* is a function  $\sigma : V \rightarrow \Sigma$  that assigns each variable some label in the alphabet. The *satisfiability value* for an assignment  $\sigma$ , denoted by  $\text{val}(G, \sigma)$ , is the fraction of constraints satisfied by  $\sigma$ , i.e.,  $\text{val}(G, \sigma) = \frac{1}{|E|} \sum_{e \in E} \Pi_e(\sigma(u_e), \sigma(v_e))$ . The satisfiability value for  $G$ , denoted by  $\text{val}(G)$ , is the maximum satisfiability value among all assignments, i.e.,  $\text{val}(G) = \max_{\sigma: V \rightarrow \Sigma} \text{val}(G, \sigma)$ . We say that an assignment  $\sigma$  is a *solution* to a CSP instance  $G$  if  $\text{val}(G, \sigma) = 1$ , and  $G$  is *satisfiable* iff  $G$  has a solution.

When the context is clear, we omit  $\sigma$  in the evaluation of a constraint, i.e.,  $\Pi_e(u_e, v_e)$  stands for  $\Pi(\sigma(u_e), \sigma(v_e))$ .

*Parameterization and Fixed Parameter Tractability.* For an instance  $G$ , the *parameterization* refers to attaching the parameter  $k := |V|$  (the size of the variable set) to  $G$  and treating the input as a  $(G, k)$  pair. We think of  $k$  as a growing parameter that is much smaller than the instance size  $n := |G|$ . A promise problem  $L_{\text{yes}} \dot{\cup} L_{\text{no}}$  is *fixed parameter tractable (FPT)* if it has an algorithm which, for every instance  $G$ , decides whether  $G \in L_{\text{yes}}$  or  $G \in L_{\text{no}}$  in  $f(k) \cdot n^{O(1)}$  time for some computable function  $f$ .

*FPT Reduction.* An *FPT reduction* from  $L_{\text{yes}} \dot{\cup} L_{\text{no}}$  to  $L'_{\text{yes}} \dot{\cup} L'_{\text{no}}$  is an algorithm  $\mathcal{A}$  which, on every input  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  outputs another instance  $G' = (V', E', \Sigma', \{\Pi'_e\}_{e \in E'})$  such that:

- **COMPLETENESS.** If  $G \in L_{\text{yes}}$ , then  $G' \in L'_{\text{yes}}$ .
- **SOUNDNESS.** If  $G \in L_{\text{no}}$ , then  $G' \in L'_{\text{no}}$ .
- **FPT.** There exist universal computable functions  $f$  and  $g$  such that  $|V'| \leq g(|V|)$  and the runtime of  $\mathcal{A}$  is bounded by  $f(|V|) \cdot |G|^{O(1)}$ .

*$\varepsilon$ -Gap Parameterized CSP.* We mainly focus on the gap version of the PARAMETERIZED CSP problem. Formally, an  $\varepsilon$ -GAP PARAMETERIZED CSP problem needs to decide whether a given CSP instance  $(G, |V|)$  with  $|V| = k$  satisfies  $\text{val}(G) = 1$  or  $\text{val}(G) < 1 - \varepsilon$ . The exact version is equivalent to 0-GAP PARAMETERIZED CSP.

*Parameterized Inapproximability Hypothesis (PIH).* PIH, first<sup>9</sup> formulated by Lokshtanov, Ramanujan, Saurabh, and Zehavi [63], is a central conjecture in parameterized complexity theory, which, if true, serves as a parameterized counterpart of the celebrated PCP theorem. Below, we present a slight reformulation of PIH, asserting fixed parameter intractability (rather than  $W[1]$ -hardness specifically) of gap CSP.

<sup>9</sup>As noted in Reference [63], prior to their work, this hypothesis was already informally stated by quite a few researchers as a natural formulation of the PCP theorem in parameterized complexity.

**HYPOTHESIS 2.1 (PIH).** *For an absolute constant  $0 < \epsilon < 1$ , no FPT algorithm, parameterized by  $k$ , can decide  $\epsilon$ -GAP PARAMETERIZED CSP.*

**Exponential Time Hypothesis (ETH).** ETH, first proposed by Impagliazzo and Paturi [47], is a widely-used strengthening of the  $P \neq NP$  hypothesis and provides a foundation for fine-grained understandings in modern complexity theory.

**Definition 2.2 (3SAT).** A 3CNF formula  $\varphi$  on  $n$  Boolean variables is a conjunction of  $m$  clauses, where each clause is a disjunction of three literals and each literal is a variable or its negation. The goal of the 3SAT problem is to decide whether  $\varphi$  is satisfiable or not.

The original ETH is stated in the general 3SAT problem. In this paper, for convenience, we use the following variant due to the sparsification lemma [48] and Tovey's reduction [71], which gives 3SAT additional structure.

**HYPOTHESIS 2.3 (ETH).** *There exists some  $\zeta > 0$ , such that every algorithm solving 3SAT instances with  $n$  variables requires runtime  $2^{\zeta n}$ . Furthermore, we can additionally assume each variable is contained in at most four clauses, and each clause contains exactly three distinct variables.*<sup>10</sup>

## 2.2 Parallel Walsh-Hadamard Code

As mentioned in Section 1.2, the key step to bypass the obstacle in previous constructions in literature is vectorization and parallel encoding of an error correcting code. In this article, we only consider the parallelization of the well-known *Walsh-Hadamard code*, a classic error correcting code that is locally testable and correctable. First, we recall standard notions in coding theory.

Given two words (aka strings)  $x, y \in \Sigma^K$  and same length  $K$ , their *relative distance*  $\Delta(x, y)$  is the fraction of coordinates that they differ, i.e.,  $\Delta(x, y) = \frac{1}{K}|\{i \in [K] : x_i \neq y_i\}|$ . We say  $x \in \Sigma^K$  is  $\delta$ -far (resp.,  $\delta$ -close) from a set of words  $S \subseteq \Sigma^K$  if  $\Delta(x, S) := \min_{y \in S} \Delta(x, y) \geq \delta$  (resp.,  $\leq \delta$ ).

**Definition 2.4 (Error Correcting Codes (ECCs)).** An error correcting code is the image of the encoding map  $C : \Sigma_1^k \rightarrow \Sigma_2^K$  with message length  $k$ , codeword length  $K$ . We say that the ECC has a relative distance  $\delta$  if  $\Delta(C(x), C(y)) \geq \delta$  holds for any distinct  $x, y \in \Sigma_1^k$ . We use  $\text{Im}(C)$  to denote the codewords of  $C$ .

**Definition 2.5 (Parallel Walsh-Hadamard Code).** Let  $\mathbb{F}$  be a finite field and  $(a_1, a_2, \dots, a_k) \in (\mathbb{F}^d)^k$  be a tuple of  $k$  vectors in  $\mathbb{F}^d$ . We view it as a matrix  $A = (a_1, a_2, \dots, a_k) \in \mathbb{F}^{d \times k}$  where the  $i$ -th column is the vector  $a_i$ .

The parallel Walsh-Hadamard encoding  $\text{PWH}(A)$  of  $A$  is a codeword indexed by  $\mathbb{F}^k$  where each entry is a vector in  $\mathbb{F}^d$ . Alternatively,  $\text{PWH}(A)$  is a function mapping  $\mathbb{F}^k$  to  $\mathbb{F}^d$  that enumerates linear combinations of the column vectors of  $A$ . Formally, for each  $b \in \mathbb{F}^k$ , we have  $\text{PWH}(A)[b] = Ab$ .

We remark that the parallel Walsh-Hadamard code is also known as interleaved Hadamard code and linear transformation code [26, 38].

In the notation of Definition 2.4,  $\text{PWH}$  has  $\Sigma_1 = \mathbb{F}^d$ ,  $\Sigma_2 = \mathbb{F}^d$ , and  $K = |\mathbb{F}|^k$ . Note that when  $d = 1$ , the parallel Walsh-Hadamard code coincides with the standard Walsh-Hadamard code. It is clear that  $\text{PWH}$  has the relative distance  $\delta = 1 - \frac{1}{|\mathbb{F}|}$ , which is at least  $\frac{1}{2}$  since  $|\mathbb{F}| \geq 2$  holds always.

---

<sup>10</sup>We say a variable  $x$  is contained in a clause  $C$  if the literal  $x$  or  $\neg x$  appears in  $C$ .

*Local Testability and Correctability.* Fix a word  $w \in (\mathbb{F}^d)^{\mathbb{F}^k}$  and treat it as a map  $f_w$  from  $\mathbb{F}^k$  to  $\mathbb{F}^d$  with  $f_w(a) = w[a]$  for all  $a \in \mathbb{F}^k$ . Note that any homomorphism from  $\mathbb{F}^k$  to  $\mathbb{F}^d$  corresponds to some  $w^* \in \text{Im}(\text{PWH})$ . To test whether  $w$  is close to a codeword of PWH, we perform the famous *BLR test* [12], which samples uniformly random  $a, b \in \mathbb{F}^k$  and accept if  $w[a] + w[b] = w[a + b]$  by three queries to  $w$ . The following theorem establishes the soundness of this test.

**THEOREM 2.6.** *If  $\Pr_{a,b \in \mathbb{F}^k} [w[a] + w[b] = w[a + b]] \geq 1 - \varepsilon$ , then  $\Delta(w, \text{Im}(\text{PWH})) \leq 6\varepsilon$ .*

We defer the proof of Theorem 2.6 to Appendix B.1.

Assume  $w$  is  $\eta$ -close to an actual codeword  $w^* \in \text{Im}(\text{PWH})$ . To obtain the value of  $w^*[x]$  for some  $x \in \mathbb{F}^k$ , we can draw a uniform  $a \in \mathbb{F}^k$  and compute  $w[x + a] - w[a]$  by two queries. The following fact concerns the soundness of this procedure.

**FACT 2.7.** *If  $w$  is  $\eta$ -close to some  $w^* \in \text{Im}(\text{PWH})$ , then  $\Pr_{a \in \mathbb{F}^k} [w[x + a] - w[a] = w^*[x]] \geq 1 - 2\eta$ .*

### 2.3 Probabilistic Checkable Proofs of Proximity

PCPP (PCPP, also known as assignment testers) [9, 28] are essential gadgets in proving the PCP theorem [5, 7, 24]. While PCP verifiers check whether an input belongs to a language by querying a few bits of the proof, PCPP verifiers strengthen this notion by determining whether the input is close to some word in the language.

In this article, we reformulate PCPP in the parameterized regime. Our reformulation is compatible with the parallel encoding. To conveniently combine different PCPPs, we specialize PCPPs into their PWH-based constructions.<sup>11</sup> Formally, we define the following *parallel PCPP PPCPP*.

**Definition 2.8** (( $q, \delta, \varepsilon, f, g$ )-PPCPPs). Let  $f$  and  $g$  be two computable functions. Given a finite field  $\mathbb{F}$  and a CSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  where  $\Sigma = \mathbb{F}^d$ . The  $(q, \delta, \varepsilon, f, g)$ -PPCPP associated with  $G$  is a randomized verifier  $A$  with the following workflow: Recall that  $k = |V|$  is the parameter of the CSP instance  $G$ .

- $A$  takes as input two blocks of proofs  $\pi_1 \circ \pi_2$  with alphabet  $\mathbb{F}^d$ , where:
  - $\pi_1$  has length  $|\mathbb{F}|^k$  with entries indexed by vectors in  $\mathbb{F}^k$ , which is supposed to be the parallel Walsh-Hadamard encoding of some assignment to  $V$ .
  - $\pi_2$  has length at most  $f(k)$ . It is an auxiliary proof enabling an efficient verification procedure.
- $A$  chooses a uniform  $r \in [R_A]$ , where  $R_A$  is at most  $g(k)$ , queries at most  $q$  positions in  $\pi_1 \circ \pi_2$  based on  $r$ , and decides to accept or reject the proof after getting the query result.
- The list of queries made by  $A$  can be generated in time at most  $h(k) \cdot |G|^{O(1)}$  for some computable function  $h$ .

The verifier  $A$  has the following properties.

- **COMPLETENESS.** For every solution  $\sigma$  of  $G$ , there exists a  $\pi_2$  such that  $\Pr[A \text{ accepts } \text{PWH}(\sigma) \circ \pi_2] = 1$ , where we treat an assignment  $\sigma: V \rightarrow \mathbb{F}^d$  as a vector in  $(\mathbb{F}^d)^{|V|}$ .
- **SOUNDNESS.** If  $\Pr[A \text{ accepts } \pi_1 \circ \pi_2] \geq 1 - \varepsilon$ , there exists some solution  $\sigma$  of  $G$  such that  $\Delta(\pi_1, \text{PWH}(\sigma)) \leq \delta$ .

<sup>11</sup>The choice of encoding is typically abstracted out in standard definitions of PCPPs.

Intuitively, PPCPPs check whether  $\pi_1$  is close to the Walsh-Hadamard encoding of some solution of  $G$ . Like the traditional PCPP, parallel PCPPs are also tightly connected with CSPs. The following standard reduction establishes the connection.

*Definition 2.9 (Reduction from PPCPPs to CSPs).* Given a  $(q, \delta, \varepsilon, f, g)$ -PPCPP verifier  $A$  for a CSP  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  with  $\Sigma = \mathbb{F}^d$ , we define a CSP instance  $G' = (V', E', \Sigma', \{\Pi'_e\}_{e \in E'})$ , where  $V' = V'_1 \dot{\cup} V'_2 \dot{\cup} V'_3$  and  $\Sigma' = (\mathbb{F}^d)^q$ , by the following steps:

- First, for  $i = 1, 2$ , we treat each position of  $\pi_i$  as a single variable in  $V'_i$  with alphabet  $\mathbb{F}^d$ . Note that  $|V'_1| = |\mathbb{F}|^k$  and  $|V'_2| \leq f(k)$ .
- Then, for each randomness  $r \in [R_A]$ , let  $S_r$  be the set of query positions over  $\pi_1 \circ \pi_2$  under randomness  $r$ ; and we add a supernode  $z_r$  to  $V'_3$  whose alphabet is  $(\mathbb{F}^d)^{|S_r|}$ , i.e., all possible configurations of the query result. Note that  $|V'_3| \leq g(k)$ .
- Finally, we add constraints between  $z_r$  and every query position  $i \in S_r$ . The constraint checks whether  $z_r$  is an accepting configuration, and the assignment of the position  $i$  is consistent with the assignment of  $z_r$ .

By construction, we can see that the completeness and soundness are preserved up to a factor of  $q$  under this reduction, where the loss comes from the construction where we split  $q$  queries into  $q$  consistency checks. In addition, since  $|\pi_1 \circ \pi_2| \leq |\mathbb{F}|^k + f(k)$ ,  $R_A \leq g(k)$ , and the list of queries made by  $A$  can be generated in time  $h(k) \cdot |G|^{O(1)}$ , the reduction from  $G$  to  $G'$  is a FPT reduction.

**FACT 2.10.** *The reduction described in Definition 2.9 is an FPT reduction. Recall that  $k = |V|$  is the parameter of  $G$  and  $\Sigma = \mathbb{F}^d$  is the alphabet of  $G$ . We have the following properties for  $G'$ :*

- *ALPHABET.* The alphabet of  $G'$  is  $\Sigma' = \mathbb{F}^{d \cdot q}$ .
- *PARAMETER BLOWUP.* The parameter of  $G'$  is  $|V'| \leq |\mathbb{F}|^k + f(k) + g(k)$ .
- *COMPLETENESS.* For every solution  $\sigma$  of  $G$ , there exists a solution  $\sigma'$  of  $G'$  assigning  $\text{PWH}(\sigma)$  to  $V'_1$ .
- *SOUNDNESS.* For any assignment  $\sigma'$  satisfying  $1 - \frac{\varepsilon}{q}$  fraction of the constraints in  $G'$ , there exists a solution  $\sigma$  of  $G$  such that  $\Delta(\sigma'(V'_1), \text{PWH}(\sigma)) \leq \delta$ .

For the self-containedness, we relegate the proof of this fact to Appendix B.2.

### 3 Proof of the Main Theorem

In this section, we prove the following quantitative version of our main theorem (Theorem 1.2). To depict a clear picture, we will treat some technical constructions as black boxes and relegate their proofs to subsequent sections.

**THEOREM 3.1.** *Assume ETH holds. Then, no algorithm can decide  $\frac{1}{9600}$ -GAP PARAMETERIZED CSP within runtime  $f(k) \cdot n^{o(\sqrt[4]{\log k})}$  for any computable function  $f$ .*

As a byproduct of the quantitative analysis, we also have the following PCP-style theorem, which can be viewed as a parameterized PCP theorem.

**THEOREM 3.2.** *For any integer  $k \geq 1$ , 3SAT has a PCP verifier which*

- *can be constructed in time  $f(k) \cdot |\Sigma|^{O(1)}$  for some computable function  $f$ ,*
- *makes two queries on a proof of length  $2^{O(k^4)}$  and alphabet size  $|\Sigma| = 2^{O(n/k)}$ ,*
- *has perfect completeness and soundness  $1 - \frac{1}{9600}$ .*

Our proof relies on an intermediate structured CSP, termed *vector-valued CSPs* (*VecCSP* for short).<sup>12</sup>

**Definition 3.3 (Vector-Valued CSP).** A CSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  is a VecCSP if the following additional properties hold.

- $\Sigma = \mathbb{F}^d$  is a  $d$ -dimensional vector space over a finite field  $\mathbb{F}$  with characteristic 2.
- For each constraint  $e = \{u, v\} \in E$  where  $u = (u_1, u_2, \dots, u_d)$  and  $v = (v_1, v_2, \dots, v_d)$ , the constraint validity function  $\Pi_e$  is classified as one of the following forms in order<sup>13</sup>:
- LINEAR. There exists a matrix  $M_e \in \mathbb{F}^{d \times d}$  such that

$$\Pi_e(u, v) = 1_{u=M_e v}$$

- PARALLEL. There exists a sub-constraint  $\Pi_e^{sub} : \mathbb{F} \times \mathbb{F} \rightarrow \{0, 1\}$  such that  $\Pi_e$  checks  $\Pi_e^{sub}$  for every coordinate, i.e.,

$$\Pi_e(u, v) = \bigwedge_{i \in [d]} \Pi_e^{sub}(u_i, v_i).$$

- Each variable is related to at most one parallel constraint.<sup>14</sup>

We refer to Figure 1 as an illustration of VecCSP.

Our reduction is accomplished as follows. First, in Section 3.1, we provide a reduction from 3SAT to VecCSPs. Second, in Section 3.2, we provide another reduction from VecCSPs to parameterized CSPs of constant gap. Finally, in Section 3.3, we show how to combine the two reductions above to prove Theorems 3.1 and 3.2.

### 3.1 Reduction I: From 3SAT to Vector-Valued CSPs

In this step, we reduce 3SAT to VecCSPs. By Hypothesis 2.3, we may assume 3SAT has some additional structure.

**THEOREM 3.4 (PROVED IN SECTION 4).** *There is a reduction such that the following holds. For any positive integer  $\ell$  and given as input a SAT formula  $\varphi$  of  $n$  variables and  $m$  clauses, where each variable is contained in at most four clauses and each clause contains exactly three distinct variables, the reduction produces a VecCSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  where:*

<sup>12</sup>The definition we use here is a special case of the definition in the conference version [43]. More precisely, the sub-constraint in VecCSP from [43] checks entries from a *subset* of coordinates. In later work [42], we found that using a special case of VecCSP in [43] whose parallel constraints have sub-constraints on every coordinate can simplify the proof. For the sake of simplicity, we adopt this definition.

<sup>13</sup>A constraint can be both linear and parallel (e.g., equality constraint). In this case, we classify it as linear instead of parallel, consistent with the order defined here.

<sup>14</sup>This is an additional feature from our reduction from 3SAT to VecCSP. However, this feature is not necessary when constructing PPCPPs.

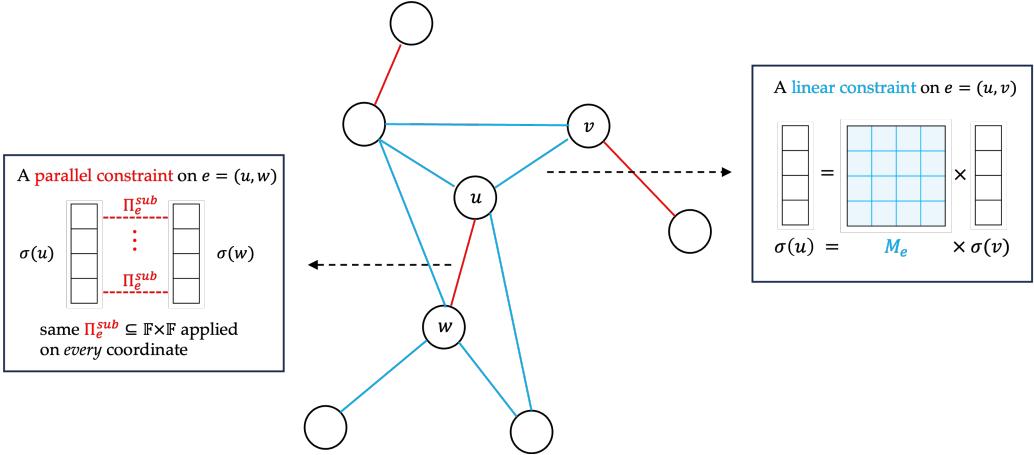


Fig. 1. An example of a vector-valued CSP.

**(S1) VARIABLES AND CONSTRAINTS.**  $|V| = O(\ell^2)$  and  $|E| = O(\ell^2)$ .

**(S2) RUNTIME.** The reduction runs in time  $\ell^{O(1)} \cdot 2^{O(n/\ell)}$ .

**(S3) ALPHABET.**  $\Sigma = \mathbb{F}_8^d$  where  $d = \max\{[m/\ell], [n/\ell]\}$ .

**(S4) COMPLETENESS AND SOUNDNESS.**  $G$  is satisfiable iff  $\varphi$  is satisfiable.

### 3.2 Reduction II: From Vector-Valued CSPs to Gap CSPs

Now we present our gap-producing reduction from VecCSPs to  $\varepsilon$ -GAP PARAMETERIZED CSP.

**THEOREM 3.5.** Fix an absolute constant  $\varepsilon^* = \frac{1}{9600}$ . There is a reduction that holds the following: Given as input a VecCSP instance  $G = (V, E, \Sigma = \mathbb{F}_8^d, \{\Pi_e\}_{e \in E})$  where

- $k = |V|$  is the parameter of  $G$ ,
- $|\mathbb{F}| = 2^t \leq h(k)$  for some computable function  $h$ ,
- $|E| \leq m(k)$  for some computable function  $m$  such that  $m(k) \geq 1$ ,

the reduction produces a CSP instance  $G^* = (V^*, E^*, \Sigma^* = \mathbb{F}^{4d}, \{\Pi_e^*\}_{e \in E^*})$  where:

- **FPT REDUCTION.** The reduction from  $G$  to  $G^*$  is an FPT reduction.
- **PARAMETER BLOWUP.** The parameter of  $G^*$  is  $|V^*| \leq |\mathbb{F}|^{O(k \cdot m(k))} \cdot 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$ .
- **COMPLETENESS.** If  $G$  is satisfiable, then  $G^*$  is satisfiable.
- **SOUNDNESS.** If  $G$  is not satisfiable, then  $\text{val}(G^*) < 1 - \varepsilon^*$ .

Below, we present our reduction and proof for Theorem 3.5. Fix a VecCSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  satisfying the conditions in Theorem 3.5. Our reduction is achieved in three steps.

**3.2.1 Step a: Instance Splitting.** Recall that  $G$  has two kinds of constraints: linear and parallel constraints. In this step, we partition the constraint set  $E$  into two parts  $E_L \cup E_P$ , where  $E_L$  and  $E_P$  consist of all linear and parallel constraints of  $E$ , and define  $G_L = (V, E_L, \Sigma, \{\Pi_e\}_{e \in E_L})$  and

$G_P = (V, E_P, \Sigma, \{\Pi_e\}_{e \in E_P})$  as the sub-CSP instance where the constraint set is  $E_L$  and  $E_P$ , respectively. Note that  $G_L$  and  $G_P$  are still VecCSPs with the same parameter  $k = |V|$ . Furthermore, we have the simple observation as follows.

FACT 3.6. *For every assignment  $\sigma$  over  $V$ ,  $\sigma$  is a solution of  $G$  if and only if it is the solution of both  $G_L$  and  $G_P$ .*

3.2.2 *Step b: Designing Parallel PCPPs for Sub-Instances.* In this step, we construct PPCPP verifiers  $A_L$  and  $A_P$  in FPT time to test whether all constraints in  $G_L$  and  $G_P$  are satisfied, respectively. We first handle parallel constraints and obtain  $A_P$ . Though the constraint graph of  $G_P$  has a maximum degree 1, our PPCPP construction works for every VecCSP with only parallel constraints.

PROPOSITION 3.7 (PPCPP FOR PARALLEL CONSTRAINTS. PROVED IN SECTION 5). *Let  $h$  be a computable function. Let  $G$  be a VecCSP instance with  $k$  variables where (1) the alphabet is  $\mathbb{F}^d$  and  $|\mathbb{F}| = 2^t \leq h(k)$ , and (2) all constraints are parallel constraints. Then for every  $\varepsilon \in (0, \frac{1}{600})$ , there is a  $(4, 36\varepsilon, \varepsilon, f(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}, g(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}})$ -PPCPP verifier for  $G$ , where  $f(k)$  is the length of the auxiliary proof, and  $g(k)$  is the number of random choices.*

Recall that the alphabet of  $G$  is  $\mathbb{F}^d$  where  $|\mathbb{F}| = 2^t$ , and  $G_P$  consists of parallel constraints of  $G$  only. Thus, by plugging  $\varepsilon = \frac{1}{1200}$  into the proposition above, we can obtain a  $(q_P = 4, \delta_P = \frac{3}{100}, \varepsilon_P = \frac{1}{1200}, f_P(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}, g_P(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}})$ -PPCPP verifier  $A_P$  for  $G_P$ . Now, we turn to linear constraints and obtain  $A_L$ .

PROPOSITION 3.8 (PPCPP FOR LINEAR CONSTRAINTS. PROVED IN SECTION 6). *Let  $h$  and  $m$  be two computable functions. Let  $G$  be a VecCSP instance with  $k$  variables where (1) the alphabet is  $\mathbb{F}^d$  and  $|\mathbb{F}| \leq h(k)$ , (2) all constraints are linear constraints, and (3) there are at most  $m(k)$  constraints. Then for every  $\varepsilon \in (0, \frac{1}{400})$ , there is a  $(4, 24\varepsilon, \varepsilon, f(k) = |\mathbb{F}|^{k \cdot m(k)}, g(k) = |\mathbb{F}|^{8k \cdot m(k)})$ -PPCPP verifier for  $G$ .*

By plugging  $\varepsilon = \frac{1}{600}$  into the proposition above, we can derive a  $(q_L = 4, \delta_L = \frac{1}{25}, \varepsilon_L = \frac{1}{600}, f_L(k) = |\mathbb{F}|^{k \cdot m(k)}, g_L(k) = |\mathbb{F}|^{8k \cdot m(k)})$ -PPCPP verifier  $A_L$  for  $G_L$ .

Now, we combine  $A_L$  and  $A_P$  into a single PPCPP  $A$  for the general VecCSP  $G$  from Theorem 3.5. In step c, we will convert  $A$  into a CSP instance with an inherent gap, completing the proof of Theorem 3.5.

Here  $A$  executes  $A_L$  and  $A_P$  as in a black-box way where  $A$  takes  $\pi_1 \circ \pi_L \circ \pi_P$  as a proof and with equal probability,  $A$  invokes  $A_L$  with proof  $\pi_1 \circ \pi_L$  or invokes  $A_P$  with proof  $\pi_1 \circ \pi_P$ .

Intuitively,  $\pi_1$  serves as a unified encoding of a solution of  $G$  via the parallel Walsh-Hadamard code PWH, and  $\pi_L$  and  $\pi_P$  are auxiliary proofs to convince  $A_L$  and  $A_P$  respectively. The following proposition shows that  $A$  is a PPCPP that efficiently checks all the constraints in  $G$ .

PROPOSITION 3.9 (COMBINED PCPP). *Given a VecCSP instance  $G$  satisfying the preconditions in Theorem 3.5, the verifier  $A$  described above is a  $(q = 4, \delta = \frac{1}{25}, \varepsilon = \frac{1}{2400}, f(k) = |\mathbb{F}|^{k \cdot m(k)} + 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}, g(k) = |\mathbb{F}|^{O(k \cdot m(k))} \cdot 2^{k^2 \cdot |\mathbb{F}|^{O(1)}})$ -PPCPP verifier for  $G$ .*

PROOF. Since  $A$  invokes either  $A_L$  or  $A_P$ , the number of queries is the maximum of  $q_L = 4$  and  $q_P = 4$ . The length of the auxiliary proof is

$$|\pi_L \circ \pi_P| \leq f_L(k) + f_P(k) = |\mathbb{F}|^{k \cdot m(k)} + 2^{k^2 \cdot |\mathbb{F}|^{O(1)}} = f(k).$$

For alignment, we pad the randomness of  $A_P$  and  $A_L$  to ensure that they have same amount

$$R \leq g_P(k)g_L(k) = |\mathbb{F}|^{8k \cdot m(k)} \cdot 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$$

of uniform choices. Thus the total number of uniform choices of  $A$  is  $2R \leq g(k)$ . Finally, we analyze the completeness and soundness.

*Completeness.* Let  $\sigma$  be a solution of  $G$ . We set  $\pi_1 = \text{PWH}(\sigma)$ . By Fact 3.6,  $\sigma$  is also a solution of  $G_L$  and  $G_P$ . As a result, by the definition of PPCPP (Definition 2.8), there exists  $\pi_L$  and  $\pi_P$  such that  $A_L$  and  $A_P$  always accept  $\pi_1 \circ \pi_L$  and  $\pi_1 \circ \pi_P$  respectively. Thus,  $A$  always accepts  $\pi_1 \circ \pi_L \circ \pi_P$ .

*Soundness.* Assume  $A$  accepts  $\pi_1 \circ \pi_L \circ \pi_P$  with probability at least  $1 - \varepsilon$ . By construction,  $A_L$  accepts  $\pi_1 \circ \pi_L$  with probability at least  $1 - 2\varepsilon \geq 1 - \varepsilon_L$ . Thus by the definition of PPCPP (Definition 2.8), there exists a solution  $\sigma_L$  of  $G_L$  such that  $\Delta(\pi_1, \text{PWH}(\sigma_L)) \leq \delta_L \leq \delta$ . Similarly for  $A_P$ , there exists a solution  $\sigma_P$  of  $G_P$  such that  $\Delta(\pi_1, \text{PWH}(\sigma_P)) \leq \delta$ . Thus

$$\Delta(\text{PWH}(\sigma_P), \text{PWH}(\sigma_L)) \leq 2\delta < \frac{1}{2}.$$

Recall from Subsection 2.2 that the relative distance of PWH is at least  $\frac{1}{2}$ . Hence we must have that  $\sigma_L = \sigma_P$ , which means  $\sigma_L = \sigma_P$  is a solution of  $G$  such that  $\Delta(\pi_1, \text{PWH}(\sigma_L)) \leq \delta$ .  $\square$

**3.2.3 Step c: Reducing Parallel PCPPs to Gap CSPs.** Finally, we complete the proof of Theorem 3.5 by converting the verifier  $A$  into a constant-gap PARAMETERIZED CSP  $G^*$ .

**PROOF OF THEOREM 3.5.** Set  $G^* = (V^*, E^*, \Sigma^*, \{\Pi_e^*\}_{e \in E^*})$  to be the CSP instance obtained by applying the reduction in Definition 2.9 on the verifier  $A$ . Then the claimed runtime of the reduction, as well as the alphabet size, completeness and soundness of  $G^*$ , follow immediately from combining Proposition 3.9 and Fact 2.10. Here we simply note the parameter blowup:

$$|V^*| \leq |\mathbb{F}|^k + f(k) + g(k) = |\mathbb{F}|^{O(k \cdot m(k))} \cdot 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}.$$

### 3.3 Putting Everything Together

In this part, we combine Theorems 3.4 and 3.5 to prove Theorem 3.1 and 3.2.

**PROOF OF THEOREM 3.1.** Assuming ETH (Hypothesis 2.3), there exists  $\xi > 0$  such that no  $2^{\xi n}$ -time algorithm can solve 3SAT formula  $\varphi$  where each variable is contained in at most four clauses and each clause contains exactly three distinct variables. For any such formula  $\varphi$ , we show how to reduce  $\varphi$  to a PARAMETERIZED CSP instance  $G^*$  with a constant inherent gap.

Let  $\ell$  be a parameter to be chosen later. We first invoke Theorem 3.4 and obtain a VecCSP instance  $G$  in  $\ell^{O(1)} \cdot 2^{O(n/\ell)}$  time where:

- There are  $k = 144\ell^2$  variables and their alphabet is  $\mathbb{F}_8^d$  with  $d = O(n/\ell)$ .
- $G$  is satisfiable iff  $\varphi$  is satisfiable.

Note that the size of  $G$  is  $|G| = \ell^{O(1)} \cdot 2^{O(n/\ell)}$ .

Then, we apply Theorem 3.5 on  $G$  with parameters  $h(k) = 8$  and  $m(k) = 2k$ . After this reduction, we obtain a CSP instance  $G^* = (V^*, E^*, \Sigma^*, \{\Pi_e^*\}_{e \in E^*})$  such that:

- The runtime and instance size  $N := |G^*|$  of the reduction are bounded by  $r_1(k) \cdot |G|^{O(1)} = r_2(\ell) \cdot 2^{O(n/\ell)}$  for some computable functions  $r_1, r_2$ .
- The parameter of  $G^*$  is  $K = |V^*| \leq 2^{O(k^2)} = 2^{O(\ell^4)}$ .

- If  $G$  is satisfiable, then  $G^*$  is satisfiable.
- If  $G$  is not satisfiable, then  $\text{val}(G^*) < 1 - \frac{1}{9600}$ .

Hence, the total runtime of the reduction above is bounded by  $(r_2(\ell) + \ell^{O(1)}) \cdot 2^{O(n/\ell)}$ , which is less than  $2^{\xi n}$  as long as  $\ell$  is sufficiently large.

Since the size of  $G^*$  is  $N \leq r_2(\ell) \cdot 2^{O(n/\ell)}$ , ETH (Hypothesis 2.3) rules out all algorithms with runtime  $f(K) \cdot N^{\xi'} \cdot \sqrt[4]{\log K}$  for some constant  $\xi' > 0$  and any computable function  $f$ . Thus Theorem 3.1 follows.  $\square$

The above quantitative analysis readily gives the PCP-style statement (Theorem 3.2).

**PROOF OF THEOREM 3.2.** Given a 3SAT formula of size  $n$ , we apply the sparsification lemma [48] and Tovey’s reduction [71] to obtain a 3SAT instance  $\varphi$  on  $n$  Boolean variables where each variable is contained in at most four clauses and each clause contains exactly three distinct variables. In addition, this reduction runs in  $n^{O(1)}$  time and preserves the satisfiability of the original 3SAT formula.

Let  $\ell \geq 1$  be a parameter. Then we apply the analysis of Theorem 3.1 on  $\varphi$  to obtain a CSP instance  $G^* = (V^*, E^*, \Sigma^*, \{\Pi_e^*\}_{e \in E^*})$  with  $|V^*| = K$  variables and alphabet size  $|\Sigma^*| \leq 2^{O(n/\ell)}$  in time  $r_2(\ell) \cdot 2^{O(n/\ell)}$ , where  $K = 2^{O(\ell^4)}$  and  $r_2$  is some computable function. In addition, if  $\varphi$  is satisfiable, then  $\text{val}(G^*) = 1$ ; otherwise  $\text{val}(G^*) < 1 - \frac{1}{9600}$ .

Now a PCP verifier takes a proof  $\pi: V^* \rightarrow \Sigma^*$ , viewed as a string of length  $K$  and alphabet  $\Sigma^*$ , and picks a uniform random constraint  $e \in E^*$  to check. Note that the runtime of this verifier is bounded by the size  $|G^*| \leq r_2(\ell) \cdot 2^{O(n/\ell)}$  of  $G^*$ . If the original 3SAT formula is satisfiable, then  $\varphi$  is satisfiable and thus there exists a proof that always passes the check. Otherwise, by the soundness guarantee of  $G^*$ , any proof will violate at least  $\frac{1}{9600}$  fraction of the constraints in  $E^*$ , which implies the soundness gap of the PCP verifier. Setting  $\ell = k$  and  $f(k) = r_2(k)$  completes the proof of Theorem 3.2.  $\square$

## 4 From 3SAT to Vector-Valued CSP

This section is devoted to the proof of Theorem 3.4, which shows how to obtain a VecCSP from a 3SAT instance.

Before going to the details, we mark the high level picture of the reduction as follows:

- (1) First, we divide the clauses and variables of the 3SAT instance  $\varphi$  into  $k$  parts and build a vector-valued variable (in the following, we will denote them as “vertices” in order to distinguish them from the variables in  $\varphi$ ) for each part of clauses and each part of variables. Then we apply tests to check the consistency between clauses and variables.
- (2) Next, we appropriately duplicate each vertex into several copies. We introduce equality constraints between these copies to ensure consistency. Then we split the “clause-variable” constraints and spread them out to different copies of the vertices, such that
  - each copy is related to at most one “clause-variable” constraint;
  - the sub-constraints inside each “clause-variable” constraint are of the same type, and these sub-constraints form a matching on the  $2d$  coordinates of the two endpoints.<sup>15</sup>

<sup>15</sup>The matching property is essential only for reducing 3SAT to VecCSPs, and isn’t required when designing PPCPPs.

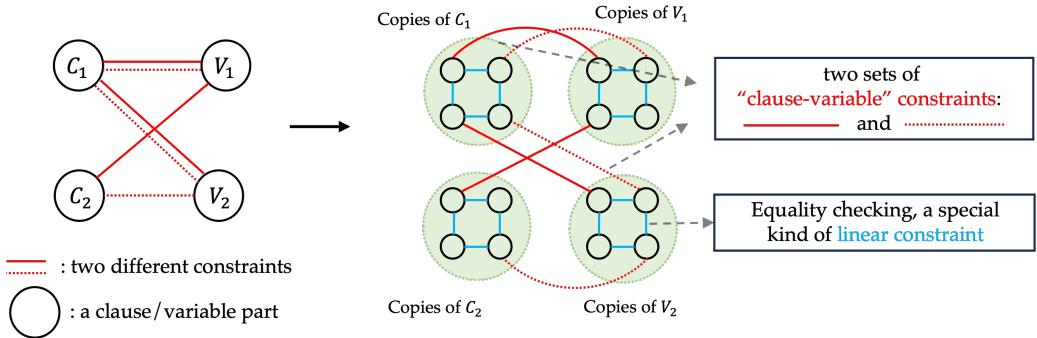


Fig. 2. An example showing the vertex duplicating and constraint splitting steps.

- (3) Finally, given the properties above, we can rearrange the  $d$  coordinates of each vertex according to its only constraint, to make the sub-constraints parallel. By introducing more auxiliary variables, we can also make sure that the sub-constraints span all coordinates.

We refer to Figure 2 for an informal illustration of the above process, where the set of “clause-variable” constraints is further demonstrated in Figure 3.

**THEOREM (THEOREM 3.4 RESTATED).** *There is a reduction with the following guarantee. For any positive integer  $\ell$  and given as input a SAT formula  $\varphi$  of  $n$  variables and  $m$  clauses, where each variable is contained in at most four clauses and each clause contains exactly three distinct variables, the reduction produces a VecCSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  in  $\ell^{O(1)} \cdot 2^{O(n/\ell)}$  time, where:*

**(S1) VARIABLES AND CONSTRAINTS.**  $|V| = O(\ell^2), |E| = O(\ell^2)$ .

**(S2) RUNTIME.** The reduction runs in time  $\ell^{O(1)} \cdot 2^{O(n/\ell)}$ .

**(S3) ALPHABET.**  $\Sigma = \mathbb{F}_8^d$  where  $d = \max \{[m/\ell], [n/\ell]\}$ .

**(S4) COMPLETENESS AND SOUNDNESS.**  $G$  is satisfiable iff  $\varphi$  is satisfiable.

Fix  $\varphi$  and  $\ell$  from Theorem 3.4. For each variable, we fix an arbitrary order of its (at most four) appearances in clauses. The order is used to construct parallel constraints.

We partition  $n$  variables of  $\varphi$  into  $\mathcal{V}_1, \dots, \mathcal{V}_\ell$  where each  $\mathcal{V}_i$  contains at most  $[n/\ell]$  variables, and similarly we partition  $m$  clauses of  $\varphi$  into  $\mathcal{C}_1, \dots, \mathcal{C}_\ell$  where each  $\mathcal{C}_i$  contains at most  $[m/\ell]$  clauses. For each variable  $x \in \mathcal{V}_1 \cup \dots \cup \mathcal{V}_\ell$ , we treat its assignment in  $\{0, 1\}$  as an element of  $\mathbb{F}_8$  via some canonical choice, i.e., we treat 0 as the zero element  $(0, 0, 0)$  in  $\mathbb{F}_8$ , and 1 as the unit element  $(0, 0, 1)$  in  $\mathbb{F}_8$ . For each clause  $C \in \mathcal{C}_1 \cup \dots \cup \mathcal{C}_\ell$ , we identify  $\mathbb{F}_8 = \{0, 1\}^3$  as the set of partial assignments to the clause  $C$ , where  $(0, 0, 0) \in \mathbb{F}_8$  is the only unsatisfying assignment.

Formally, given a clause  $C = y_{i_1} \vee y_{i_2} \vee y_{i_3}$ , where each literal  $y_{i_j}$  equals variable  $x_{i_j}$  or its negation  $\neg x_{i_j}$ , every  $\tau \in \mathbb{F}_8$ , viewed as an element in  $\{0, 1\}^3$ , corresponds to a unique assignment by setting  $y_{i_j} = \tau(j)$  for  $j \in [3]$ , which in turn assigns the value of  $x_{i_j}$ .

We now define six tests as sub-constraints to be used in constructing parallel constraints between two vertices. For  $j \in [3]$  and  $b \in \{0, 1\}$ , we define  $\Pi_{j,b} : \mathbb{F}_8 \times \mathbb{F}_8 \rightarrow \{0, 1\}$  by

$$\Pi_{j,b}(\tau, c) = 1_{c \in \{0,1\}} \cdot 1_{\tau \neq (0,0,0)} \cdot 1_{\tau(j)=c \oplus b},$$

Intuitively, these constraint checks that: (1) the variable assignment  $c$  is binary, (2) the clause assignment  $\tau$  is satisfying, and (3) the clause assignment and variable assignment are consistent, where  $b$  indicates whether the  $j$ th variable of the clause is negated.

We construct the desired VecCSP instance  $G$  in two steps. First, we split the “clause-variable” constraints into matching forms. This results in an intermediate PARAMETERIZED CSP instance  $G_0$  as follows.

*Vertices and Alphabets.* We first define the vertices and the alphabet of  $G_0$ . For each  $p \in [\ell], q \in [3], s \in [4], b \in \{0, 1\}$ , we introduce two vertices  $z_{p,q,j,s,b}$  and  $w_{p,q,j,s,b}$ , which are supposed to be the duplicates of assignments to  $\mathcal{C}_p$  and  $\mathcal{V}_q$ , respectively. Thus, we index entries of  $z_{p,q,j,s,b}$  by clauses in  $\mathcal{C}_p$  and entries of  $w_{p,q,j,s,b}$  by variables in  $\mathcal{V}_q$ . Since  $d = \max\{[m/\ell], [n/\ell]\} = \max\{|\mathcal{C}_p|, |\mathcal{V}_q|\}$ , some entries may be left unused.

*Constraints.* This part defines the constraints of  $G_0$ .

First, we ensure the vertices  $z_{p,\cdot,\cdot,\cdot,\cdot}$  correspond to the same assignment over  $\mathcal{C}_p$ . Similarly, we also need to check  $w_{q,\cdot,\cdot,\cdot,\cdot}$  corresponds to the same assignment for  $\mathcal{V}_q$ . To check this, for each  $p \in [\ell]$ , we connect  $\{z_{p,q,j,s,b} : q \in [\ell], j \in [3], s \in [4], b \in \{0, 1\}\}$  in the constraint graph  $G$  by an arbitrary cycle (denoted by the cycle of  $\mathcal{C}_p$ ), where every constraint in this cycle is a linear constraint. Specifically, for every two vertices  $\hat{z} = z_{p,q,j,s,b}$  and  $\tilde{z} = z_{p,q',j',b',s'}$  connected in the cycle of  $\mathcal{C}_p$ , we impose the linear constraint that  $1_{\hat{z}=\tilde{z}}$ . Similarly, for each  $q \in [\ell]$ , we also connect  $\{w_{p,q,j,s,b} : p \in [\ell], j \in [3], s \in [4], b \in \{0, 1\}\}$  in the constraint graph  $G$  by an arbitrary cycle (denoted by the cycle of  $\mathcal{V}_q$ ), and we impose equality constraints on adjacent pairs in this cycle.

After ensuring  $z_{p,\cdot,\cdot,\cdot,\cdot}$  and  $w_{q,\cdot,\cdot,\cdot,\cdot}$  are copies of the assignments of  $\mathcal{C}_p$  and  $\mathcal{V}_q$  (resp.), we then decompose the “clause-variable” checking over  $\mathcal{C}_p$  and  $\mathcal{V}_q$  into a collection of constraints in matching forms. To introduce these “clause-variable” constraints, we need some extra notations. For each  $p, q \in [\ell], j \in [3], s \in [4]$  (for simplicity we use  $\zeta$  to represent such a quadruple), we define

$$T_{\zeta,0} = \{(C, x) \in \mathcal{C}_p \times \mathcal{V}_q : \text{the } s\text{th appearance of variable } x \text{ is the } j\text{th literal in clause } C \text{ as } x\}$$

and

$$T_{\zeta,1} = \{(C, x) \in \mathcal{C}_p \times \mathcal{V}_q : \text{the } s\text{th appearance of variable } x \text{ is the } j\text{th literal in clause } C \text{ as } \neg x\}.$$

Note that for every  $b \in \{0, 1\}$ , any two distinct  $(C, x), (C', x') \in T_{\zeta,b}$  satisfy  $C \neq C'$  and  $x \neq x'$ . Thus,  $T_{\zeta,b}$  forms a (not necessarily perfect) matching over  $\mathcal{C}_p \times \mathcal{V}_q \subseteq [d] \times [d]$ . Hence, there exists a permutation  $\kappa_{\zeta,b} : [d] \rightarrow [d]$  such that  $\kappa_{\zeta,b}(C) = x$  for all  $(C, x) \in T_{\zeta,b}$ . Furthermore, we define  $Q_{\zeta,b} := \{C : \exists x \in \mathcal{V}_q, (C, x) \in T_{\zeta,b}\} \subseteq \mathcal{C}_p$  and  $R_{\zeta,b} := \{x : \exists C \in \mathcal{C}_p, (C, x) \in T_{\zeta,b}\} \subseteq \mathcal{V}_q$  as the projection of  $T_{\zeta,b}$  over the clauses and variables, respectively.

Having defined all notations, we are now ready to introduce the constraints. For each  $\zeta \in [\ell] \times [\ell] \times [3] \times [4]$  and  $b \in \{0, 1\}$ , we impose a “clause-variable” constraint between  $z_{\zeta,b}$  and  $w_{\zeta,b}$  as follows.

$$\bigwedge_{(C,x) \in T_{\zeta,b}} \Pi_{j,b}(z_{\zeta,b}[C], w_{\zeta,b}[x]).$$

Intuitively, the constraint checks (parts of) “clause-variable” consistency between  $\mathcal{C}_p$  and  $\mathcal{V}_q$ .

Since each variable  $x$  appears in no more than four clauses, and each appearance is either  $x$  or  $\neg x$ , the collection of all “clause-variable” constraints between  $z_{\zeta,b}$ 's and  $w_{\zeta,b}$ 's covers every consistency checking between  $\mathcal{C}_p$ 's and  $\mathcal{V}_q$ 's. Combined with the equality checking constructed above, which

further ensures the  $z_{p,\cdot,\cdot,\cdot}$  and  $w_{\cdot,q,\cdot,\cdot}$  are the same copy of some assignment over  $\mathcal{C}_p$  and  $\mathcal{V}_q$ , we can show that the original 3CNF formula  $\varphi$  is satisfiable if and only if the PARAMETERIZED CSP instance  $G_0$  is satisfiable. Moreover, note that

- each vertex  $z_{\zeta,b}$  and  $w_{\zeta,b}$  is adjacent to at most one “clause-variable” constraint;
- the “clause-variable” constraint between every pair of  $z_{\zeta,b}$  and  $w_{\zeta,b}$  is the conjunction of sub-constraints of the same type  $\Pi_{j,b}$ . In addition, since  $T_{\zeta,b}$  forms a matching over  $\mathcal{C}_p$  and  $\mathcal{V}_q$ , the sub-constraints form a matching over the  $2d$  coordinates of  $z_{\zeta,b}$  and  $w_{\zeta,b}$ .

However, the “clause-variable” constraints between  $z_{\zeta,b}$  and  $w_{\zeta,b}$  are in a matching form instead of the parallel form as defined in Definition 3.3. To fill this gap, we introduce more auxiliary vertices and constraints into  $G_0$  and obtain the final construction  $G$ .

*Auxiliary Vertices and Constraints.* For every  $\zeta \in [\ell] \times [\ell] \times [3] \times [4]$  and  $b \in \{0, 1\}$ , we introduce four auxiliary vertices  $z_{\zeta,b}^{(1)}, z_{\zeta,b}^{(2)}, w_{\zeta,b}^{(1)}$ , and  $w_{\zeta,b}^{(2)}$ , and split the original “clause-variable” constraints between  $z_{\zeta,b}$  and  $w_{\zeta,b}$  into five constraints, each is either linear or parallel. In detail,

- We introduce a linear constraint between  $z_{\zeta,b}$  and  $z_{\zeta,b}^{(1)}$ . The constraint checks whether for every clause  $C \in Q_{\zeta,b}$ ,  $z_{\zeta,b}^{(1)}[\kappa_{\zeta,b}(C)] = z_{\zeta,b}[C]$ .

Intuitively, this constraint rearranges the coordinates of  $z_{\zeta,b}$ . In this way, we can make the sub-constraints  $\Pi_{j,b}$  coordinate-wise.

- We introduce a linear constraint between  $z_{\zeta,b}^{(1)}$  and  $z_{\zeta,b}^{(2)}$  that checks whether

$$\forall C \in Q_{\zeta,b}, \quad z_{\zeta,b}^{(1)}[\kappa_{\zeta,b}(C)] = z_{\zeta,b}^{(2)}[\kappa_{\zeta,b}(C)].$$

Combined with the constraints between  $z_{\zeta,b}$  and  $z_{\zeta,b}^{(1)}$ , we can ensure that for every  $C \in Q_{\zeta,b}$ ,  $z_{\zeta,b}[C] = z_{\zeta,b}^{(2)}[\kappa_{\zeta,b}(C)]$ .

- We introduce a linear constraint between  $w_{\zeta,b}$  and  $w_{\zeta,b}^{(1)}$  that checks if  $\forall x \in R_{\zeta,b}, w_{\zeta,b}^{(1)}[x] = w_{\zeta,b}[x]$ , and introduce a linear constraint between  $w_{\zeta,b}^{(1)}$  and  $w_{\zeta,b}^{(2)}$  that checks whether  $\forall x \in R_{\zeta,b}, w_{\zeta,b}^{(1)}[x] = w_{\zeta,b}^{(2)}[x]$ .

These two constraints also serve as a consistency check over a subset of coordinates. They ensure that for every  $x \in R_{\zeta,b}, w_{\zeta,b}^{(2)}[x] = w_{\zeta,b}[x]$ .

- Finally, we introduce a parallel constraint between  $w_{\zeta,b}^{(2)}$  and  $z_{\zeta,b}^{(2)}$ , that constraint checks  $\Pi_{j,b}$  over all coordinates.

By the consistency constraints above, we can make sure that all sub-constraints between  $z_{\zeta,b}$  and  $w_{\zeta,b}$  are satisfied. Furthermore, since the consistency check is only imposed on a subset of entries, we can assign arbitrary satisfying assignments of  $\Pi_{j,b}$  to the entries of  $z_{\zeta,b}^{(2)}$  and  $w_{\zeta,b}^{(2)}$  irrelevant to consistency check, which enables us to extend the sub-constraint to all coordinates.

We present an illustration of the constraint above in Figure 3.

**CLAIM 4.1.**  $z_{\zeta,b}$  and  $w_{\zeta,b}$  pass all “clause-variable” checking in  $T_{\zeta,b}$  if and only if all five constraints above are satisfied.

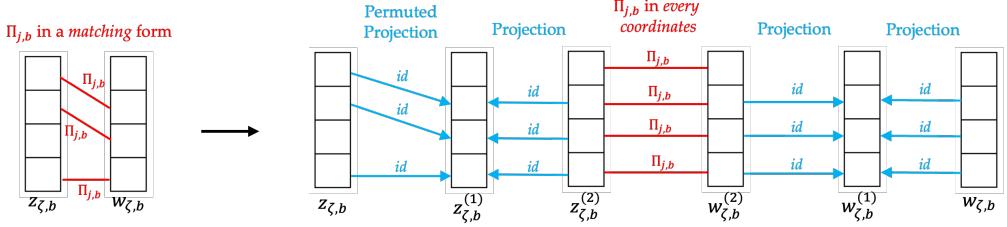


Fig. 3. An example showing how the “clause-variable” constraints are set up. Every assignment to the variable  $z_{\zeta,b}$  encodes an assignment to a group of clauses. Assignments to the variable  $w_{\zeta,b}$  encode assignments to a group of variables. The sub-constraint  $\Pi_{j,b}$  checks the consistency of values of a variable and the clause containing this variable and whether the clause is satisfied by the assignment. By duplicating variables and adding identity constraints, the original constraint between  $z_{\zeta,b}$  and  $w_{\zeta,b}$  is replaced by equivalent constraints that are either parallel or linear.

**PROOF.** For every  $(C, x) \in T_{\zeta,b}$ , we have that

$$z_{\zeta,b}^{(2)}[x] = z_{\zeta,b}^{(2)}[\kappa(C)] = z_{\zeta,b}^{(1)}[\kappa(C)] = z_{\zeta,b}[C] \quad \text{and} \quad w_{\zeta,b}^{(2)}[x] = w_{\zeta,b}^{(1)}[x] = w_{\zeta,b}[x]$$

by the construction of the linear constraints. Then, the parallel constraint between  $z_{\zeta,b}^{(2)}$  and  $w_{\zeta,b}^{(2)}$  checks the validity of “clause-variable” checking between  $C$  and  $x$ .  $\square$

Below, we establish the properties of our reduction.

*Runtime.* Since every variable of  $\phi$  is contained in at most four clauses and every clause contains three distinct variables, we have  $m \leq 4n/3$ . Hence  $|\Sigma| = 8^d = 2^{O(n/\ell)}$  and the construction of  $G$  above can be done in time  $\ell^{O(1)} \cdot 2^{O(n/\ell)}$ , showing Item (S2).

*Parameters.* It is obvious that  $G$  is an instance of VecCSP. Moreover, Item (S1) holds as we have  $O(\ell^2)$  total vertices and constraints in  $G$ ; and Item (S3) holds as all variables have alphabet  $\mathbb{F}_8^d$ .

*Completeness.* Assume  $\varphi$  is satisfiable by an assignment  $\sigma$  to the variables. This implies an assignment  $\tau$  to the literals in clauses. Then for each entry  $C$  of  $z_{p,q,j,s,b}$ , we assign it as  $\tau(C)$ , which is among  $\{0, 1\}^3 \setminus \{(0, 0, 0)\}$  as  $\sigma$  is a satisfying assignment. For each entry  $x$  of  $w_{p,q,j,s,b}$ , we assign it as  $\sigma(x)$ , and assign the value of  $z_{p,q,j,s,b}^{(1)}, z_{p,q,j,s,b}^{(2)}, w_{p,q,j,s,b}^{(1)}, w_{p,q,j,s,b}^{(2)}$  accordingly. By the construction of  $G$ , we can see that this is a valid solution.

*Soundness.* Assume  $G$  is satisfiable. By Claim 4.1, we have that for every  $p, q, j, s, b$ ,  $z_{p,q,j,s,b}$  and  $w_{p,q,j,s,b}$  passes all “clause-variable” checking over  $T_{p,q,j,s,b}$ . Since the assignment also passes all the consistency checking over the cycles of  $\mathcal{C}_p$  and  $\mathcal{V}_q$ , we can obtain an assignment  $\tau$  to the clauses of  $\varphi$  that satisfies each clause, and another assignment  $\sigma$  to each variable, such that all “clause-variable” checks are passed. Thus, we find a solution  $\sigma$  to  $\varphi$ .

## 5 Parallel PCPP for Vector-Valued CSP with Parallel Constraints

This section is devoted to proving Proposition 3.7, which is restated as follows.

**PROPOSITION (PROPOSITION 3.7 RESTATED).** *Let  $h$  be a computable function. Let  $G$  be a VecCSP instance with  $k$  variables where (1) the alphabet is  $\mathbb{F}^d$  and  $|\mathbb{F}| = 2^t \leq h(k)$ , and (2) all constraints are*

parallel constraints. Then for every  $\varepsilon \in (0, \frac{1}{800})$ , there is a  $(4, 48\varepsilon, \varepsilon, f(k) = 2^{k^2|\mathbb{F}|^{O(1)}}, g(k) = 2^{k^2|\mathbb{F}|^{O(1)}})$ -PPCPP verifier for  $G$ .

The construction of PPCPP in this section is a generalization of an assignment tester used in the proof of the classic exponential length PCP showing result  $\text{NP} \subseteq \text{PCP}[\text{poly}(n), O(1)]$  [6]. There, we first convert a Boolean circuit to a quadratic equation system (known as the QUADEQ problem) such that they share the same satisfiability, then we use the Walsh-Hadamard code to encode an assignment to the quadratic equation system and certify its satisfiability.

Our analysis relies on the random subsum principle (see e.g., [5]), demonstrated as follows.

**LEMMA 5.1 (RANDOM SUBSUM PRINCIPLE).** *Given a finite field  $\mathbb{F}$  and distinct  $M_1, M_2 \in \mathbb{F}^{\ell \times \ell'}$  with  $\ell, \ell' \geq 1$ , then  $\Pr_{x \in \mathbb{F}^\ell} [x^\top M_1 \neq x^\top M_2] \geq 1 - \frac{1}{|\mathbb{F}|}$ .*

## 5.1 An Exposition of the QUADEQ Problem

In the following, we first define the QUADEQ problem, and then introduce a PCP verifier for it. While this problem and the construction are standard in the literature (see, e.g., Reference [5]), we choose to give a brief exposition here for referencing purpose in our actual construction.

**Definition 5.2 (QUADEQ).** An instance  $\Gamma$  of the QUADEQ problem consists of  $q$  quadratic equations on  $c$  binary variables, written concisely as  $D_1, \dots, D_q \in \mathbb{F}_2^{c \times c}$  and  $b_1, \dots, b_q \in \mathbb{F}_2$ . The goal of the QUADEQ problem is to decide whether there exists a solution  $u \in \mathbb{F}_2^c$  such that  $u^\top D_i u = b_i$  holds for all  $i \in [q]$ .

The benefit of using the QUADEQ problem is that any Boolean circuit satisfiability problem can be efficiently reduced to QUADEQ by introducing dummy variables.

**FACT 5.3 (FOLKLORE, SEE E.G., [5]).** *Any Boolean circuit<sup>16</sup>  $\mathcal{C}$  with  $c$  gates (including input gates) can be converted into a QUADEQ instance  $\Gamma$  of  $c$  variables and  $q = O(c)$  equations in  $c^{O(1)}$  time such that  $\mathcal{C}$  is satisfiable iff  $\Gamma$  is satisfiable.*

Moreover, there is a one-to-one correspondence between entries of the assignment of  $\Gamma$  and gates of  $\mathcal{C}$  such that the following holds. Let  $u \in \mathbb{F}_2^c$  be an assignment of  $\Gamma$ . Then  $u$  is a solution of  $\Gamma$  iff  $\mathcal{C}$  represents a valid computation and outputs 1 after assigning values of entries of  $u$  to gates of  $\mathcal{C}$  by the above correspondence.

The QUADEQ problem also admits an efficient randomized verifier. Given instance  $\Gamma$ , the verifier will make at most four queries on a proof  $\pi$  and decide whether to accept or reject. If  $\Gamma$  is satisfiable, then there is a proof that it always accepts; otherwise, it rejects every proof with a constant probability.

For completeness and simplicity, we recall the standard (non-parallel) Walsh-Hadamard code over  $\mathbb{F}_2$ , which enjoys the same local testability (Theorem 2.6) and correctability (Fact 2.7) as PWH.

**Definition 5.4 (Walsh-Hadamard Code over  $\mathbb{F}_2$ ).** The Walsh-Hadamard encoding  $\text{WH}_2(a)$  of  $a \in \mathbb{F}_2^n$  enumerates linear combinations of entries of  $a$ . Formally,  $\text{WH}_2(a)[b] = a^\top b$  for each  $b \in \mathbb{F}_2^n$ .

The proof  $\pi$  for  $\Gamma$  consists of a length- $2^c$  binary string  $\pi_1$  and a length- $2^{c^2}$  binary string  $\pi_2$ . Here,  $\pi_1$  is supposed to be the Walsh-Hadamard encoding  $\text{WH}_2(u)$  of a solution  $u \in \mathbb{F}_2^c$ , and  $\pi_2$  is supposed

<sup>16</sup>A Boolean circuit consists of input gates, as well as AND, OR, NOT gates with fan-in (at most) two and fan-out unbounded. Here, we focus on Boolean circuits with a single output gate.

to be the Walsh-Hadamard encoding  $\text{WH}_2(w)$  of  $w = uu^\top \in \mathbb{F}_2^{c \times c}$  where we view matrix  $w$  as a length- $c^2$  vector. Then, the verifier checks one of the following tests with equal probability.

- (1) LINEARITY TEST. Perform BLR test (recall from Subsection 2.2) on  $\pi_1$  or  $\pi_2$  with equal probability. These tests make three queries.

By Theorem 2.6, if the test passes with high probability, then  $\pi_1$  and  $\pi_2$  are close to  $\text{WH}_2(u)$  and  $\text{WH}_2(w)$  of some  $u \in \mathbb{F}_2^c$  and  $w \in \mathbb{F}_2^{c \times c}$  respectively. By the local correctability (Fact 2.7), we can assume that we have access to  $\text{WH}_2(u), \text{WH}_2(w)$  via  $\pi_1, \pi_2$ .

- (2) TENSOR TEST. Test whether  $w$  equals to  $uu^\top$ . This is achieved by generating uniformly random vectors  $r, r' \in \mathbb{F}_2^c$  and making four queries, where two queries are used to obtain  $\text{WH}_2(u)[r]$  and  $\text{WH}_2(u)[r']$ , and the other two queries are used to locally correct  $\text{WH}_2(w)[r'r^\top]$  via Fact 2.7. The test accepts if  $\text{WH}_2(w)[r'r^\top] = r^\top wr'$  equals  $\text{WH}_2(u)[r] \cdot \text{WH}_2(u)[r'] = r^\top (uu^\top)r'$ .

By applying the random subsum principle (Lemma 5.1) twice, we know that if the test passes with high probability, then  $w = uu^\top$ . Now, we can assume that  $w = uu^\top$ .

- (3) CONSTRAINT TEST. Check whether  $u$  is a solution of  $\Gamma$ , i.e., whether  $u^\top D_i u = b_i$  holds for every  $i \in [q]$ . This is achieved by generating a uniform  $H \subseteq [q]$  and making two queries to locally correct  $\text{WH}_2(w)\left[\sum_{i \in H} D_i\right]$  via Fact 2.7. The test accepts if  $\text{WH}_2(w)\left[\sum_{i \in H} D_i\right] = \sum_{i \in H} u^\top D_i u$  equals  $\sum_{i \in H} b_i$ .

By Lemma 5.1, if the test passes with high probability, then  $u$  is indeed a solution.

## 5.2 From Parallel Constraints to Parallel QUADEQ

We will generalize the above verifier to the parallel setting to prove Proposition 3.7. To this end, we first need to convert the parallel constraints into the QUADEQ form. Here, we will have *parallel QUADEQ* since the alphabet of VecCSP is a vector space of dimension  $d$ .

Recall that we are given a VecCSP instance  $G$  from Proposition 3.7 with  $k$  variables and alphabet  $\mathbb{F}^d$ , where  $|\mathbb{F}| = 2^t$  and all constraints are parallel constraints. We use  $V = \{x_1, \dots, x_k\}$  to denote the variables in  $G$ , and use  $E = \{e_1, \dots, e_m\}$  to denote the constraints in  $G$ . Recall the definition of VecCSP (Definition 3.3). We know that each variable is related to at most one parallel constraint, which implies  $m \leq k/2$ . By relabeling, we assume without loss of generality that  $e_\ell$  connects  $x_{2\ell-1}$  and  $x_{2\ell}$  for each  $\ell \in [m]$ . We also recall that a parallel constraint  $e_\ell$  checks a specific sub-constraint  $\Pi_\ell : \mathbb{F} \times \mathbb{F} \rightarrow \{0, 1\}$  on all  $d$  coordinates simultaneously between  $x_{2\ell-1}$  and  $x_{2\ell}$ .

We will need the following additional notations:

- Let  $\chi : \mathbb{F} \rightarrow \mathbb{F}_2^t$  be a one-to-one map that flattens elements in  $\mathbb{F}$  into  $t$  bits. The map  $\chi$  preserves the addition operator, i.e.,  $\chi(a) + \chi(b) = \chi(a + b)$ .
- For each sub-constraint  $\Pi_\ell : \mathbb{F} \times \mathbb{F} \rightarrow \{0, 1\}$ , we define  $\bar{\Pi}_\ell : \mathbb{F}_2^t \times \mathbb{F}_2^t \rightarrow \{0, 1\}$  by setting  $\bar{\Pi}_\ell(a, b) = \Pi_e(\chi^{-1}(a), \chi^{-1}(b))$  for all  $a, b \in \mathbb{F}_2^t$ . In other words, we map sub-constraints with field inputs to sub-constraints with binary bits as input.

Note that we can represent each  $\bar{\Pi}_\ell$  as a Boolean circuit of size  $2^{O(t)}$  in time  $2^{O(t)}$ .

- We build a Boolean circuit  $\mathcal{C}$  to compute the conjunction of all the sub-constraints. Formally,  $\mathcal{C}$  is the Boolean function mapping  $\mathbb{F}_2^{k \cdot t}$  to  $\{0, 1\}$  such that

$$\mathcal{C}(y_1, \dots, y_k) = \bigwedge_{\ell \in [m]} \bar{\Pi}_\ell(y_{2\ell-1}, y_{2\ell}) = \bigwedge_{\ell \in [m]} \Pi_e(\chi^{-1}(y_{2\ell-1}), \chi^{-1}(y_{2\ell})),$$

where each  $y_i \in \mathbb{F}_2^t$  is the binary representation of a coordinate of the original  $x_i$  via additive isomorphism  $\chi$ .

Since the circuit representation of each  $\bar{\Pi}_t$  has size  $2^{O(t)}$ ,  $\mathcal{C}$  has  $c = k \cdot 2^{O(t)} = k \cdot |\mathbb{F}|^{O(1)}$  gates. In addition, by rearranging indices, we assume the first  $k \cdot t$  gates are input gates corresponding to  $(y_1, \dots, y_k)$ . The construction of  $\mathcal{C}$  can also be done in time  $k \cdot |\mathbb{F}|^{O(1)}$ .

We remark that the reason we convert  $\mathbb{F}$  into binary bits is that QUADEQ can only handle binary circuits and sticking with  $\mathbb{F}$  will require equation systems of higher degree to preserve the satisfiability, which complicates the analysis.

Now the satisfiability of the VecCSP instance  $G$  is equivalent to the satisfiability of the Boolean circuits  $\mathcal{C}_S$ 's. This is formalized in Claim 5.5. For convenience, for each assignment  $\sigma: V \rightarrow \mathbb{F}^d$  of  $G$  and each coordinate  $j \in [d]$ , we define  $\sigma^j: V \rightarrow \mathbb{F}$  as the sub-assignment of  $\sigma$  on the  $j$ -th coordinate of all variables in  $V$ . Note that  $\sigma$  and  $\sigma^j$  can be equivalently viewed as vectors in  $(\mathbb{F}^d)^k$  and  $\mathbb{F}^k$  respectively.

**CLAIM 5.5.** *Let  $\sigma: V \rightarrow \mathbb{F}^d$  be an assignment of  $V$ . Then  $\sigma$  is a solution of  $G$  iff  $\mathcal{C}(y_1^j, \dots, y_k^j) = 1$  holds for every  $j \in [d]$ , where each  $y_i^j = \chi(\sigma^j(x_i))$  is the binary representation of  $\sigma^j(x_i)$ .*

At this point, we appeal to the QUADEQ problem to further encode the satisfiability of  $\mathcal{C}$  as the satisfiability of a quadratic equation system. We construct a QUADEQ instance  $\Gamma$  by Fact 5.3, which consists of matrices  $D_1, \dots, D_q \in \mathbb{F}_2^{cx^c}$  and bits  $b_1, \dots, b_q \in \mathbb{F}_2$  with  $q = O(c)$ . In addition, by the “moreover” part of Fact 5.3, we assume the first  $k \cdot t$  bits in an assignment  $u \in \mathbb{F}_2^c$  correspond to the input gates of the circuit  $\mathcal{C}$ ; and the rest corresponds to values of other gates in  $\mathcal{C}$ .

Then Claim 5.6 establishes the conversion from  $G$  to a parallel QUADEQ instance.

**CLAIM 5.6.** *Let  $\sigma$  be an assignment of  $V$ . Recall that  $\sigma^j$  is the sub-assignment of  $\sigma$  on the  $j$ -th coordinate. Let  $(D_1, \dots, D_q, b_1, \dots, b_q)$  be the QUADEQ instance  $\Gamma$  for  $\mathcal{C}$ .*

*Then  $\sigma$  is a solution of  $G$  iff  $(u^j)^\top D_i u^j = b_i$  holds for all  $j \in [d]$  and  $i \in [q]$ , where each  $u^j \in \mathbb{F}_2^c$  is some vector with the first  $k \cdot t$  bits equal to  $\sigma^j$ . In addition, we have  $c = k \cdot |\mathbb{F}|^{O(1)}$  and  $q = k \cdot |\mathbb{F}|^{O(1)}$  here. Moreover,  $u^j$  represents the values of gates in  $\mathcal{C}$  given as input the first  $k \cdot t$  bits of  $u^j$ .*

We remark that the computation so far is very efficient and runs in FPT time since  $|\mathbb{F}| \leq h(k)$ .

### 5.3 Designing Parallel PCPPs for Parallel QUADEQ

In light of Claim 5.6, we now aim to generalize the PCP verifier of QUADEQ to the parallel setting to verify the computation on  $d$  coordinates simultaneously.

Recall that  $(D_1, \dots, D_q, b_1, \dots, b_q)$  is the QUADEQ instance  $\Gamma$  (see Claim 5.6) reduced from circuit  $\mathcal{C}$  (see Claim 5.5). We also recall that  $\sigma^j(x_i) \in \mathbb{F}$  is the  $j$ -th entry of  $\sigma(x_i) \in \mathbb{F}^d$ . For clarity, we use PWH<sub>2</sub> to denote the parallel Walsh-Hadamard encoding with field  $\mathbb{F}_2$  and reserve PWH for the parallel Walsh-Hadamard encoding with field  $\mathbb{F}$ .

The verifier  $A$  is defined as follows.

*Input of A.* The verifier  $A$  takes as input  $\pi_1 \circ \pi_2$ , where:

- $\pi_1$  has length  $|\mathbb{F}|^k$  and alphabet  $\mathbb{F}^d$ .  
It is supposed to be PWH( $\sigma$ ) for an assignment  $\sigma$  to the variables of  $G$ .
- $\pi_2$  consists of two parts: a  $2^c$ -length string  $\tau_1$  with alphabet  $\mathbb{F}_2^d$  and a  $2^{c^2}$ -length string  $\tau_2$  with alphabet  $\mathbb{F}_2^d$ .

$\tau_1$  and  $\tau_2$  are supposed to be  $\text{PWH}_2(\bar{u})$  and  $\text{PWH}_2(\bar{w})$  for some  $\bar{u} \in (\mathbb{F}_2^d)^c$  and  $\bar{w} \in (\mathbb{F}_2^d)^{c^2}$  constructed as follows: for each  $j \in [d]$ , we use  $u^j \in \mathbb{F}_2^c$ ,  $w^j \in \mathbb{F}_2^{c \times c}$  to denote the proof<sup>17</sup> that the binary representations of  $\sigma^j$  satisfy the circuit  $\mathcal{C}$ . For  $\bar{u}$  (resp.,  $\bar{w}$ ), we place  $u^j$  (resp.,  $w^j$ ) on the  $j$ -th coordinate.

We remark that the alphabet of the verifier  $A$  here has different alphabets ( $\mathbb{F}^d$  and  $\mathbb{F}_2^d$ ) for  $\pi_1$  and  $\pi_2$ . This is convenient for stating the tests and the analysis. To make it consistent with the definition of PPCPP (Definition 2.8), we can simply perform a black-box reduction that equips  $\pi_2$  with alphabet  $\mathbb{F}^d$  as well but rejects if any query result during the test is not from  $\{0, 1\}^d \cong \mathbb{F}_2^d$ .

*Verification Procedure of A.* The verifier  $A$  selects one of the following six tests with equal probability. For ease of understanding, we group the tests according to their functions.

— LINEARITY TEST.

(P1) Pick uniformly random  $\alpha, \beta \in \mathbb{F}^k$  and check if  $\pi_1[\alpha] + \pi_1[\beta] = \pi_1[\alpha + \beta]$  with three queries.

(P2) Pick uniformly random  $\alpha, \beta \in \mathbb{F}_2^c$  and check if  $\tau_1[\alpha] + \tau_1[\beta] = \tau_1[\alpha + \beta]$  with three queries.

(P3) Pick uniformly random  $\alpha, \beta \in \mathbb{F}_2^{c^2}$  and check if  $\tau_2[\alpha] + \tau_2[\beta] = \tau_2[\alpha + \beta]$  with three queries.  
These three tests ensure that  $\pi_1, \tau_1, \tau_2$  are close to  $\text{PWH}(\sigma)$ ,  $\text{PWH}_2(\bar{u})$ ,  $\text{PWH}_2(\bar{w})$  for some  $\sigma \in (\mathbb{F}^d)^k$ ,  $\bar{u} \in (\mathbb{F}_2^d)^c$  and  $\bar{w} \in (\mathbb{F}_2^d)^{c^2}$ , respectively.

— TENSOR TEST.

(P4) Pick uniformly random  $r, r' \in \mathbb{F}_2^c$  and  $y \in \mathbb{F}_2^{c \times c}$ , and check whether

$$\tau_1[r] \odot \tau_1[r'] = \tau_2[y] + \tau_2[y + rr'^\top] \quad (2)$$

with four queries, where  $\odot$  is the coordinate-wise multiplication.

This performs the TENSOR TEST (Item 2) of QUADEQ on all  $d$  coordinates simultaneously.

— CONSTRAINT TEST.

(P5) Pick a random subset  $H$  of  $[q]$  and uniformly random  $\beta \in \mathbb{F}_2^{c \times c}$ . Define  $\alpha = \sum_{z \in H} D_z \in \mathbb{F}_2^{c \times c}$ . Obtain  $y := \tau_2[\beta] + \tau_2[\alpha + \beta]$  by two queries and reject if for some  $j \in [d]$ , the  $j$ -th coordinate does not equal to  $\sum_{z \in H} b_z$ .

This performs the CONSTRAINT TEST (Item 3) of QUADEQ on all  $d$  coordinates simultaneously, where on the  $j$ -th coordinate we check the constraints with respect to  $\mathcal{C}_{\kappa(j)}$ .

— CONSISTENCY TEST.

(P6) Pick a random subset  $S$  of  $[k]$  and a uniformly random  $\beta \in \mathbb{F}^k$ . Pick a random linear function  $\psi: \mathbb{F}_2^t \rightarrow \mathbb{F}_2$  and uniformly random  $\xi \in \mathbb{F}_2^c$ . Define  $\alpha \in \mathbb{F}^k$  to be the indicator vector of  $S$ , i.e.,  $\alpha_i = 1$  for  $i \in S$  and  $\alpha_i = 0$  for  $i \notin S$ . Let

$$\gamma = (\psi(1, 0, \dots, 0), \psi(0, 1, \dots, 0), \dots, \psi(0, 0, \dots, 1)) \in \mathbb{F}_2^t$$

and

$$\eta = (\underbrace{\gamma_1, \dots, \gamma_k}_{k \text{ of } t \text{ bits}}, \underbrace{0, \dots, 0}_{\text{remaining } c - kt \text{ bits}}) \in \mathbb{F}_2^c \quad \text{where} \quad \gamma_i = \begin{cases} \gamma & \text{if } i \in S, \\ 0^t & \text{otherwise.} \end{cases}$$

Then check if

$$\psi \circ \chi(\pi_1[\beta] + \pi_1[\alpha + \beta]) = \tau_1[\xi] + \tau_1[\eta + \xi], \quad (3)$$

where  $\psi \circ \chi: \mathbb{F} \rightarrow \mathbb{F}_2$  is applied coordinate-wise.

<sup>17</sup>Technically this proof is for QUADEQ instance  $\Gamma$ . But due to the correspondence in Fact 5.3 (or Claim 5.6), we view it as a proof for the satisfiability of  $\mathcal{C}$ . In fact,  $u^j$  is the values of gates in  $\mathcal{C}$  and  $w^j = u^j(u^j)^\top$ .

This test checks if for every  $j \in [d]$ , the first  $k \cdot t$  bits in  $u^j$  equal to the binary representations of  $\sigma^j$  specified by the isomorphism  $\chi$ .

#### 5.4 Analysis of Parallel PCPPs

In this part, we prove Proposition 3.7 with the following three lemmas (Lemmas 5.7, 5.8, and 5.9), which are devoted to bound the parameters, and show completeness and soundness, respectively.

**LEMMA 5.7 (PARAMETERS).** *The verifier  $A$  takes as input two proofs  $\pi_1$  and  $\pi_2$ , where  $\pi_1$  has length  $|\mathbb{F}|^k$  and  $\pi_2$  has length at most  $f(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$ .  $A$  then uses at most  $g(k) = 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$  randomness, and queries at most four positions of the proofs. Furthermore, the list of queries made by  $A$  can be generated in FPT time.*

**PROOF.** The length of  $\pi_1$  is  $|\mathbb{F}|^k$  by definition, and the length of  $\pi_2$  is  $2^c + 2^{c^2} \leq 2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$ , where we recall that  $m \leq k/2$  and  $c = k \cdot |\mathbb{F}|^{O(1)}$ .

The amount of randomness is calculated as follows. Item (P1) has  $|\mathbb{F}|^{2k}$  uniform possibilities, Item (P2) has  $2^{2c}$ , Item (P3) has  $2^{2c^2}$ , Item (P4) has  $2^{2c} \cdot 2^{c^2}$ , Item (P5) has  $2^q \cdot 2^{c^2}$ , and Item (P6) has<sup>18</sup>  $2^k \cdot |\mathbb{F}|^k \cdot 2^t \cdot 2^c$ . Recall that  $2^t = |\mathbb{F}|$ ,  $m \leq k/2$ ,  $c = k \cdot |\mathbb{F}|^{O(1)}$ , and  $q = k \cdot |\mathbb{F}|^{O(1)}$ . Hence we may duplicate integer multiples for each of them and assume that they all have  $2^{k^2 \cdot |\mathbb{F}|^{O(1)}}$  uniform possibilities. Then the total randomness sums up to  $8 \cdot 2^{k^2 \cdot |\mathbb{F}|^{O(1)}} \leq g(k)$  as desired.

It's easy to see that  $A$  makes at most four queries in any case, and the list of queries under all randomness can be generated in FPT time.  $\square$

**LEMMA 5.8 (COMPLETENESS).** *Suppose there is a solution  $\sigma : V \rightarrow \mathbb{F}^d$  of  $G$ , then there is a proof  $\pi_1 \circ \tau_1 \circ \tau_2$  which  $A$  accepts with probability 1.*

**PROOF.** By Claim 5.5, for each  $j \in [d]$ ,  $\mathcal{C}$  outputs 1 when taking the binary representation of  $\sigma^j$ , i.e.,  $(\chi(\sigma^j(x_1)), \dots, \chi(\sigma^j(x_k)))$ , as input. Thus by Claim 5.6, for each  $j \in [d]$ , we have a solution  $u^j$  to the QUADEQ instance  $\Gamma$ , where the first  $k \cdot t$  bits of  $u^j$  equal to  $\chi(\sigma^j(x_1)), \dots, \chi(\sigma^j(x_k))$ .

We set  $\pi_1 = \text{PWH}(\sigma(x_1), \dots, \sigma(x_k))$ ,  $\tau_1 = \text{PWH}_2(\bar{u})$ , and  $\tau_2 = \text{PWH}_2(\bar{w})$  where  $\bar{u} \in (\mathbb{F}_2^d)^c$  and  $\bar{w} \in (\mathbb{F}_2^d)^{c^2}$  as follows.

- For every  $j \in [d]$ , the  $j$ th coordinate of  $\bar{u}$ , viewed as a length- $c$  binary string, is  $u^j$ .
- For every  $j \in [d]$ , the  $j$ th coordinate of  $\bar{w}$ , viewed as a length- $c^2$  binary string, is  $u^j(u^j)^\top$ .

Since  $\pi_1, \tau_1$  and  $\tau_2$  are all parallel Walsh-Hadamard codewords, they pass the linearity tests in Item (P1), Item (P2), Item (P3) naturally. To verify Item (P4) and Item (P5), we simply observe that

- For every  $j \in [d]$  and  $r, r' \in \mathbb{F}_2^c$ , we have  $\langle w^j, rr'^\top \rangle = \langle u^j(u^j)^\top, rr'^\top \rangle = (r^\top u^j)(r'^\top u^j)$ .
- For every  $j \in [d]$  and  $H \subseteq [q]$ , we have

$$\left\langle w^j, \sum_{z \in H} D_z \right\rangle = (u^j)^\top \left( \sum_{z \in H} D_z \right) u^j = \sum_{z \in H} (u^j)^\top D_z u^j = \sum_{z \in H} b_z,$$

since  $u^j$  is a solution to the QUADEQ instance  $\Gamma$ .

<sup>18</sup>The second  $2^t$  comes from the randomness in  $\psi$ , which is a random linear function from  $\mathbb{F}_2^t$  to  $\mathbb{F}_2$ .

For Item (P6), on the left hand side of Equation (3), we have

$$\psi \circ \chi(\pi_1[\beta] + \pi_1[\alpha + \beta]) = \psi \circ \chi(\pi_1[\alpha]) = \psi \circ \chi\left(\sum_{i \in S} \sigma(x_i)\right) = \sum_{i \in S} \psi(\chi(\sigma(x_i))),$$

where  $\psi \circ \chi$  is applied coordinate-wise and the second equality is due to the linearity of  $\psi$  and the fact that  $\chi$  is an additive isomorphism. On the right hand side of Equation (3), by our choice of  $\eta$  and the fact that the first  $k \cdot t$  bits of each  $u^j$  are just  $(\chi(\sigma^j(x_1)), \dots, \chi(\sigma^j(x_k)))$ , we also get  $\sum_{i \in S} \psi(\chi(\sigma(x_i)))$ .  $\square$

**LEMMA 5.9 (SOUNDNESS).** *Suppose there is a proof  $\pi_1 \circ \tau_1 \circ \tau_2$  which  $A$  accepts with probability at least  $1 - \varepsilon$ , then there is a solution  $\sigma$  to  $G$  such that  $\Delta(\pi_1, \text{PWH}(\sigma)) \leq 36\varepsilon$ .*

**PROOF.** Given such a proof, each individual test passes with probability at least  $1 - 6\varepsilon$ . By the soundness of BLR testing (Theorem 2.6), passing the linearity test in Item (P1) with probability at least  $1 - 6\varepsilon$  implies there exists  $\sigma \in (\mathbb{F}_2^d)^k$  such that  $\Delta(\pi_1, \text{PWH}(\sigma)) \leq 36\varepsilon$ . Similarly for Items (P2) and (P3),  $\tau_1, \tau_2$  are  $(36\varepsilon)$ -close to  $\text{PWH}_2(\bar{u})$  and  $\text{PWH}_2(\bar{w})$  for some  $\bar{u} \in (\mathbb{F}_2^d)^c$  and  $\bar{w} \in (\mathbb{F}_2^d)^{c^2}$  respectively.

At this point, for each  $j \in [d]$ , define  $u^j \in \mathbb{F}_2^c$  (resp.,  $w^j \in \mathbb{F}_2^{c \times c}$ ) to be the  $j$ -th coordinate of  $\bar{u}$  (resp.,  $\bar{w}$ ). By the analysis above, any query  $\tau_1[\alpha]$  gives us a vector  $v \in \mathbb{F}_2^d$ , whose  $j$ -th coordinate stores  $\langle u^j, \alpha \rangle$ ; and the same holds for  $\tau_2$ . We then prove that  $w^j = u^j(u^j)^\top$  holds for every  $j \in [d]$ , and  $u^j{}^\top D_z u^j = b_z$  holds for every  $j \in [d], z \in [q]$ . This shows that they form a solution of the quadratic equation system in Claim 5.6.

– If for some  $j \in [d]$ , we have  $w^j \neq u^j \otimes u^j$ . Then by Lemma 5.1 twice, for at least  $\frac{1}{4}$  fraction of choices of  $(r, r')$ , we have  $r^\top w^j r' \neq (r^\top u^j)(r'^\top u^j)$ . Suppose the four queried points in Item (P4) are indeed as if on  $\text{PWH}_2(\bar{u}), \text{PWH}_2(\bar{w})$ , which happens with probability at least  $1 - 144\varepsilon$ . Then the left hand side of Equation (2) is a vector  $v$  whose  $j$ -th coordinate is  $(r^\top u^j)(r'^\top u^j)$ , while the right hand side of Equation (2) is a vector  $v'$  whose  $j$ -th coordinate is  $r^\top w^j r'$ . Thus  $v \neq v'$  and Item (P4) rejects. Now that the rejection probability is at least  $\frac{1}{4} - 144\varepsilon$ , it is greater than  $6\varepsilon$  when  $\varepsilon < \frac{1}{600}$ , which provides a contradiction.

– If for some  $j \in [d], z \in [q]$ , we have  $u^j{}^\top D_z u^j \neq b_z$ . By Lemma 5.1, for  $\frac{1}{2}$  fraction of choices of  $H \subseteq [q]$ , we have  $\sum_{z \in H} u^j{}^\top D_z u^j \neq \sum_{z \in H} b_z$ . Suppose the two queried points in Item (P5) are indeed as if on  $\text{PWH}_2(\bar{w})$ , which happens with probability at least  $1 - 72\varepsilon$ . Then, Item (P5) rejects. Now that the rejection probability is at least  $\frac{1}{2} - 72\varepsilon$ , it is greater than  $6\varepsilon$  as  $\varepsilon < \frac{1}{156}$ , which provides a contradiction.

Finally, we prove that for every  $j \in [d]$ , the first  $k \cdot t$  bits in  $u^j$ , which we denote as  $(u_1^j, \dots, u_k^j) \in \mathbb{F}_2^{k \cdot t}$ , equal to  $(\chi(\sigma^j(x_1)), \dots, \chi(\sigma^j(x_k)))$ . This shows that the binary representation of  $\sigma$  certifies the satisfiability of the circuit  $\mathcal{C}$  as well as the QUADEQ instances  $\Gamma$  by Claims 5.5 and 5.6, which means  $\sigma$  is a solution of  $G$ .

Suppose for some  $i \in [k], j \in [d]$ , we have  $\chi(\sigma^j(x_i)) \neq u_i^j$ . Since  $\psi$  is a random linear function mapping to  $\mathbb{F}_2$ , with probability  $\frac{1}{2}$ , they still differ after  $\psi$  applied on. Then by Lemma 5.1, for  $\frac{1}{2}$  of the choices of  $S \subseteq [k]$ , we have

$$\sum_{i \in S} \psi(\chi(\sigma^j(x_i))) \neq \sum_{i \in S} \psi(u_i^j).$$

Suppose the four queried points in Item (P6) are indeed as if on  $\text{PWH}(\sigma)$ ,  $\text{PWH}_2(\bar{u})$ , which happens with probability at least  $1 - 144\epsilon$ . Then the left hand side of Equation (3) is

$$\psi \circ \chi \left( \sum_{i \in S} \sigma(x_i) \right) = \sum_{i \in S} \psi(\chi(\sigma(x_i))),$$

where  $\psi \circ \chi$  is applied coordinate-wise and the equality holds due to the linearity of  $\psi$  and the fact that  $\chi$  is an additive isomorphism. The right hand side of Equation (3) is  $\sum_{i \in S} \psi(u_i^j)$  by our construction of  $\eta$ . Therefore, Item (P6) rejects with probability at least  $\frac{1}{4} - 144\epsilon$ , which is greater than  $6\epsilon$  when  $\epsilon < \frac{1}{600}$ , leading to a contradiction again.

In conclusion, we have shown that if  $A$  accepts with probability at least  $1 - \epsilon$ , then  $\pi_1$  must be  $(36\epsilon)$ -close to  $\text{PWH}(\sigma)$ , where  $\sigma$  is a solution of  $G$ .  $\square$

Proposition 3.7 follows from a combination of Lemmas 5.7, 5.8, and 5.9.

## 6 Parallel PCPP for Vector-Valued CSP with Linear Constraints

This section is devoted to proving Proposition 3.8, which we recall below.

**PROPOSITION (PROPOSITION 3.8 RESTATED).** *Let  $h$  and  $m$  be two computable functions. Let  $G$  be a VecCSP instance with  $k$  variables where (1) the alphabet is  $\mathbb{F}^d$  and  $|\mathbb{F}| \leq h(k)$ , (2) all constraints are linear constraints, and (3) there are at most  $m(k)$  constraints. Then for every  $\epsilon \in (0, \frac{1}{400})$ , there is a  $(4, 24\epsilon, \epsilon, f(k) = |\mathbb{F}|^{k \cdot m(k)}, g(k) = |\mathbb{F}|^{8k \cdot m(k)})$ -PPCPP verifier for  $G$ .*

### 6.1 Construction of Parallel PCPPs

Fix a VecCSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  from Proposition 3.8. Recall that  $k = |V|$  and we set  $m = |E| \leq m(k)$ . By Definition 3.3, since all constraints are linear, for each constraint  $e \in E$  we denote

- its two endpoints by  $u_e$  and  $v_e$ ,
- the matrix for this linear constraint by  $M_e \in \mathbb{F}^{d \times d}$ ,
- the semantics of this constraint by  $\Pi_e(u_e, v_e) = 1_{u_e = M_e v_e}$ .

For ease of presentation, we call  $u_e$  the *head* of the constraint  $e$ , and  $v_e$  the *tail* of  $e$ , respectively.

Our construction of the PPCPP verifier  $A$  is similar to the Walsh-Hadamard-based one in [9], with an additional introduction of some subtle auxiliary variables.

*Auxiliary Variables.* Label variables  $V$  by  $\{1, 2, \dots, k\}$  and constraints by  $\{1, 2, \dots, m\}$ . For every  $p \in V$  and  $e \in E$ , we define an auxiliary variable  $z_{p,e}$  with alphabet  $\mathbb{F}^d$ . Given an assignment  $\sigma(p)$  to the variable  $p$ , the assignment to  $z_{p,e}$  should equal  $z_{p,e} = M_e \sigma(p)$ <sup>19</sup>.

Note that we introduce an auxiliary variable for every pair  $(p, e) \in V \times E$ , even if  $e$  is not adjacent to  $p$ . This way, we can check both the inner constraints  $z_{p,e} = M_e \sigma(p)$  and the conjunction of all linear constraints  $\sigma(u_e) = z_{v_e, e}$  with constant queries, soundness, and proximity.

Below, we describe the details of the PPCPP verifier  $A$  for  $G$ .

<sup>19</sup>Here, we abuse the notation and use  $z_{p,e}$  also to denote the value assigned to it.

*Input of A.* The verifier  $A$  takes as input  $\pi_1 \circ \pi_2$ , where:

- $\pi_1$  is indexed by vectors in  $\mathbb{F}^k$  and has alphabet  $\mathbb{F}^d$ . It is supposed to be  $\text{PWH}(\sigma)$ , the parallel Walsh-Hadamard encoding of an assignment  $\sigma$  to  $V$ .
- $\pi_2$  is indexed by vectors in  $\mathbb{F}^{km}$  and has alphabet  $\mathbb{F}^d$ . It is supposed to be the parallel Walsh-Hadamard encoding of the collection  $\{z_{p,e}\}_{p \in V, e \in E}$ , treated as a vector of  $(\mathbb{F}^d)^{km}$ .

*Verification Procedure of A.* Here is how  $A$  verifies whether  $\pi_1$  is close to  $\text{PWH}(\sigma)$  for some solution  $\sigma$  of  $G$ . With equal probability,  $A$  selects one of the following four tests:

– LINEARITY TEST.

- (L1) Pick uniformly random  $a_1, a_2 \in \mathbb{F}^k$  and check  $\pi_1[a_1] + \pi_1[a_2] = \pi_1[a_1 + a_2]$  by three queries.  
 (L2) Pick uniformly random  $b_1, b_2 \in \mathbb{F}^{km}$  and check  $\pi_2[b_1] + \pi_2[b_2] = \pi_2[b_1 + b_2]$  by three queries.

Intuitively, Items (L1) and (L2) ensure that both  $\pi_1$  and  $\pi_2$  are close to a codeword of  $\text{PWH}$ .

– MATRIX TEST.

- (L3) Pick uniformly random  $\lambda \in \mathbb{F}^k$  and  $\mu \in \mathbb{F}^m$  and set  $\gamma = (\lambda_1\mu_1, \lambda_1\mu_2, \dots, \lambda_k\mu_m) \in \mathbb{F}^{km}$ . Assume  $\mu$  is indexed by constraints  $e \in E$  and define matrix  $M_0 = \sum_{e \in E} \mu_e M_e$ . Note that we can compute  $M_0$  efficiently without any query. Then pick uniformly random  $a \in \mathbb{F}^k, b \in \mathbb{F}^{km}$ , query  $\pi_1[a], \pi_1[a + \lambda], \pi_2[b], \pi_2[b + \gamma]$ , and check if

$$\pi_2[b + \gamma] - \pi_2[b] = M_0(\pi_1[a + \lambda] - \pi_1[a]). \quad (4)$$

Intuitively, Item (L3) ensures that  $\pi_2$  encodes the collection  $\{z_{p,e}\}_{p \in V, e \in E}$  where all inner constraints  $z_{p,e} = M_e \sigma(p)$  are satisfied.

– CONSTRAINT TEST.

- (L4) Pick uniformly random  $\mu \in \mathbb{F}^m$  and assume  $\mu$  is indexed by constraints  $e \in E$ . Define a vector  $\lambda \in \mathbb{F}^k$  by setting  $\lambda_p = \sum_{e \in E: u_e=p} \mu_e$  for  $p \in V$ , where we assume that  $\lambda$  is indexed by vertices  $p \in V$ . In other words,  $\lambda_p$  is the sum of  $\mu_e$ 's for constraint  $e \in E$  whose head is  $p$ .

In addition, define a vector  $\gamma \in \mathbb{F}^{km}$ , indexed by a vertex-constraint pair  $(p, e) \in V \times E$ , by

$$\gamma_{p,e} = \begin{cases} \mu_e & v_e = p, \\ 0 & \text{otherwise.} \end{cases}$$

In other words,  $\gamma_{p,e}$  stores  $\mu_e$  if the tail of the constraint  $e$  is vertex  $p$ .

Note that the two vectors  $\mu$  and  $\gamma$  can be computed efficiently without any query.

Then pick uniformly random  $a \in \mathbb{F}^k, b \in \mathbb{F}^{km}$ , query  $\pi_1[a], \pi_1[a + \lambda], \pi_2[b], \pi_2[b + \gamma]$ , and check if

$$\pi_2[b + \gamma] - \pi_2[b] = \pi_1[a + \lambda] - \pi_1[a].$$

Intuitively, Item (L4) ensures  $\sigma(u_e) = z_{v_e, e}$  for every constraint  $e \in E$ .

## 6.2 Analysis of Parallel PCPPs

In this part, we prove Proposition 3.8 with the following three lemmas (6.1, 6.2, and 6.3), which are devoted to bounding the parameters, and establishing the completeness and soundness of the verifier, respectively.

LEMMA 6.1 (PARAMETERS). *The verifier A takes as input two proofs  $\pi_1$  and  $\pi_2$ , where  $\pi_1$  has length  $|\mathbb{F}|^k$  and  $\pi_2$  has length  $f(k) = |\mathbb{F}|^{km}$ . A then uses at most  $g(k) = |\mathbb{F}|^{8km}$  randomness, and queries at most four positions of the proofs. Furthermore, the list of queries made by A can be generated in FPT time.*

PROOF. The length of  $\pi_1$  and  $\pi_2$  are  $|\mathbb{F}|^k$  and  $|\mathbb{F}|^{km}$  respectively by definition.

As for randomness, Item (L1) has  $|\mathbb{F}|^{2k}$  uniform possibilities, Item (L2) has  $|\mathbb{F}|^{2km}$ , Item (L3) has  $|\mathbb{F}|^{2k+m+km}$ , and Item (L4) has  $|\mathbb{F}|^{k+m+km}$ . Note that we may duplicate integer multiples for each of them and assume that they all have  $|\mathbb{F}|^{4km}$  uniform possibilities. Then the total randomness sums up to  $4 \cdot |\mathbb{F}|^{4km} \leq g(k)$  as desired.

It's easy to see that A makes at most four queries in any case, and the list of queries under all randomness can be generated in FPT time.  $\square$

LEMMA 6.2 (COMPLETENESS). *Suppose there is a solution  $\sigma : V \rightarrow \mathbb{F}^d$  of G, then there is a proof  $\pi_1 \circ \pi_2$  which A accepts with probability 1.*

PROOF. Fix such a solution  $\sigma$ . We assign the value  $M_e\sigma(p)$  to the auxiliary variable  $z_{p,e}$  and treat  $\{z_{p,e}\}_{p \in V, e \in E}$  as a  $km$ -dimensional vector. We set  $\pi_1$  as  $\text{PWH}(\sigma)$  and  $\pi_2$  as  $\text{PWH}(\{z_{p,e}\}_{p \in V, e \in E})$ . Since both  $\pi_1$  and  $\pi_2$  are codewords of  $\text{PWH}$ , they naturally pass the tests in Item (L1) and Item (L2).

For Item (L3), note that for every  $\lambda \in \mathbb{F}^k, \mu \in \mathbb{F}^m, a \in \mathbb{F}^k, b \in \mathbb{F}^{km}$ , we have

$$\begin{aligned} \pi_2[b + \gamma] - \pi_2[b] &= \pi_2[\gamma] = \sum_{p \in V, e \in E} \lambda_p \mu_e z_{p,e} && \text{(by the definitions of } \pi_2 \text{ and } \gamma) \\ &= \sum_{p, e} \lambda_p \mu_e \cdot M_e \sigma(p) = \left( \sum_{e \in E} \mu_e M_e \right) \left( \sum_{p \in V} \lambda_p \sigma(p) \right) && \text{(by the definition of } z_{p,e}) \\ &= M_0 \pi_1[\lambda] = M_0 (\pi_1[a + \lambda] - \pi_1[a]), && \text{(by the definitions of } M_0 \text{ and } \pi_1) \end{aligned}$$

which means that the test in Item (L3) passes.

Finally, we turn to Item (L4). For every  $\mu \in \mathbb{F}^m, a \in \mathbb{F}^k, b \in \mathbb{F}^{km}$ , we have

$$\begin{aligned} \pi_2[b + \gamma] - \pi_2[b] &= \pi_2[\gamma] = \sum_{e \in E} \mu_e z_{v_e, e} = \sum_{e \in E} \mu_e \cdot M_e \sigma(v_e) && \text{(by the definitions of } \pi_2, \gamma, z_{v_e, e}) \\ &= \sum_{e \in E} \mu_e \sigma(u_e) && \text{(by } M_e \sigma(v_e) = \sigma(u_e) \text{ as } \sigma \text{ is a solution)} \\ &= \sum_{p \in V} \sigma(p) \cdot \left( \sum_{e \in E : u_e = p} \mu_e \right) && \text{(by rearranging the summation)} \\ &= \sum_{p \in V} \lambda_p \sigma(p) && \text{(by the definition of } \lambda) \\ &= \pi_1[\lambda] = \pi_1[a + \lambda] - \pi_1[a], && \text{(by the definition of } \pi_1) \end{aligned}$$

which passes the test in Item (L4). In all, A accepts  $\pi_1 \circ \pi_2 = \text{PWH}(\sigma) \circ \pi_2$  with probability 1.  $\square$

LEMMA 6.3 (SOUNDNESS). *Suppose there is a proof  $\pi_1 \circ \pi_2$  which A accepts with probability at least  $1 - \epsilon$ , then there is a solution  $\sigma$  to G such that  $\Delta(\pi_1, \text{PWH}(\sigma)) \leq 24\epsilon$ .*

PROOF. First, note that  $\pi_1$  fails at most  $4\epsilon$  fraction of tests in Item (L1). By the soundness of BLR testing (Theorem 2.6), there exists some  $\sigma \in (\mathbb{F}^d)^k$  such that  $\Delta(\pi_1, \text{PWH}(\sigma)) \leq 24\epsilon$ . Similarly

by Item (L2), we can find some  $\sigma_2 \in (\mathbb{F}^d)^{km}$  such that  $\Delta(\pi_2, \text{PWH}(\sigma_2)) \leq 24\epsilon$ . Below, we treat  $\sigma$  as a mapping from  $V$  to  $\mathbb{F}^d$ , and  $\sigma_2$  as a mapping from  $V \times E \rightarrow \mathbb{F}^d$ .

Now we prove that for every  $p \in V, e \in E$ , we have  $\sigma_2(p, e) = M_e \sigma(p)$ . Recall the notation from Item (L3). For any fixed  $\lambda, \mu$  (and thus  $\gamma, M_0$  are also fixed), by Fact 2.7 and a union bound, with probability at least  $1 - 96\epsilon$  over random  $a, b$ , both of the following two equations hold

$$\pi_2[b + \gamma] - \pi_2[b] = \sum_{p \in V, e \in E} \lambda_p \mu_e \sigma_2(p, e), \quad (5)$$

$$\pi_1[a + \lambda] - \pi_1[a] = \sum_{p \in V} \lambda_p \sigma(p). \quad (6)$$

Recall the definition of  $M_0$ . By taking a difference between the LHS and RHS of Equation (4) and plugging in Equations (5) and (6), we can deduce that, with probability at least  $1 - 96\epsilon$ ,

$$(\pi_2[b + \gamma] - \pi_2[b]) - M_0(\pi_1[a + \lambda] - \pi_1[a]) = \sum_{p \in V, e \in E} \lambda_p \mu_e (\sigma_2(p, e) - M_e \sigma(p)) = \lambda^\top (M_1 - M_2) \mu,$$

where we define matrix  $M_1$  by setting the  $(p, e)$ -th entry as  $\sigma_2(p, e)$  and matrix  $M_2$  by setting the  $(p, e)$ -th entry as  $M_e \sigma(p)$ .

If  $\sigma_2(p, e) \neq M_e \sigma(p)$  for some  $p \in V, e \in E$ , then  $M_1 \neq M_2$ . By Lemma 5.1 with  $\ell = |V|$  and  $\ell' = |E|$ , we have  $\Pr_\lambda [\lambda^\top M_1 \neq \lambda^\top M_2] \geq 1 - \frac{1}{|\mathbb{F}|}$ . Then by another round of Lemma 5.1 with  $\ell = |E|$  and  $\ell' = 1$ , we have  $\Pr_{\lambda, \mu} [\lambda^\top M_1 \mu \neq \lambda^\top M_2 \mu] \geq (1 - \frac{1}{|\mathbb{F}|})^2 \geq \frac{1}{4}$  as  $|\mathbb{F}| \geq 2$ . By a union bound, for at least  $\frac{1}{4} - 96\epsilon$  fraction of the tests in Item (L3), both Equations (5) and (6) hold, yet the difference above is non-zero. This means at least  $\frac{1}{4} - 96\epsilon$  fraction of tests in Item (L3) are violated. On the other hand, since  $A$  accepts  $\pi_1 \circ \pi_2$  with probability at least  $1 - \epsilon$ , at most  $4\epsilon$  fraction of the tests in Item (L3) can be violated. This gives a contradiction as  $\epsilon < \frac{1}{400}$ .

Finally, we prove that  $\sigma$  is a solution of  $G$ . We focus on tests in Item (L4) and recall the notation there. By a union bound, with probability at least  $1 - 96\epsilon$  over random  $a, b$  for any fixed  $\mu$  (and thus  $\lambda, \gamma$  are also fixed), both equations below hold:

$$\pi_2[b + \gamma] - \pi_2[b] = \sum_{e \in E} \mu_e \sigma_2(v_e, e) = \sum_{e \in E} \mu_e M_e \sigma(v_e), \quad (7)$$

$$\pi_1[a + \lambda] - \pi_1[a] = \sum_{e \in E} \mu_e \sigma(u_e), \quad (8)$$

where the second equality in Equation (7) follows from our analysis for Item (L3) above. If  $\sigma(u_e) \neq M_e \sigma(v_e)$  for some  $e \in E$ , then by Lemma 5.1, we have

$$\Pr_\mu [\pi_2[b + \gamma] - \pi_2[b] \neq \pi_1[a + \lambda] - \pi_1[a]] \geq 1 - \frac{1}{|\mathbb{F}|} \geq \frac{1}{2}.$$

By a union bound, for at least  $\frac{1}{2} - 96\epsilon$  fraction of tests in Item (L4), both Equations (7) and (8) hold, yet their difference is non-zero. This means  $\pi_1 \circ \pi_2$  violates at least  $\frac{1}{2} - 96\epsilon$  fraction of tests in Item (L4), contradicting the fact that  $\pi_1 \circ \pi_2$  can only violate at most  $4\epsilon$  fraction of tests, recalling again our assumption that  $\epsilon < \frac{1}{400}$ . Thus  $\sigma$  is indeed a solution of  $G$ , completing the proof.  $\square$

Proposition 3.8 immediately follows from the combination of Lemmas 6.1, 6.2, and 6.3.

## References

- [1] Fateme Abbasi, Sandip Banerjee, Jaroslaw Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. 2023. Parameterized approximation schemes for clustering with general norm objectives. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*. IEEE, 1377–1399. <https://doi.org/10.1109/FOCS57990.2023.00085>
- [2] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. 2020. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM J. Comput.* 49, 4 (2020). DOI : <https://doi.org/10.1137/18M1171321>
- [3] Benny Applebaum. 2017. Exponentially-hard gap-csp and local PRG via local hardcore functions. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 836–847.
- [4] Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. 1997. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.* 54, 2 (1997), 317–331. DOI : <https://doi.org/10.1006/JCSS.1997.1472>
- [5] Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity - A Modern Approach*. Cambridge University Press. Retrieved from <http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264>
- [6] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof verification and the hardness of approximation problems. *J. ACM* 45, 3 (1998), 501–555. DOI : <https://doi.org/10.1145/278298.278306>
- [7] Sanjeev Arora and Shmuel Safra. 1998. Probabilistic checking of proofs: A new characterization of NP. *J. ACM* 45, 1 (1998), 70–122. DOI : <https://doi.org/10.1145/273865.273901>
- [8] Mihir Bellare, Oded Goldreich, and Madhu Sudan. 1998. Free bits, PCPs, and nonapproximability-towards tight results. *SIAM J. Comput.* 27, 3 (1998), 804–915.
- [9] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. 2006. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput.* 36, 4 (2006), 889–974. DOI : <https://doi.org/10.1137/S0097539705446810>
- [10] Huck Bennett, Mahdi Cheraghchi, Venkatesan Guruswami, and João Ribeiro. 2023. Parameterized inapproximability of the minimum distance problem over all fields and the shortest vector problem in all  $\ell_p$  norms. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, Barna Saha and Rocco A. Servedio (Eds.). ACM, 553–566. DOI : <https://doi.org/10.1145/3564246.3585214>
- [11] Arnab Bhattacharyya, Édouard Bonnet, László Egri, Suprovat Ghoshal, Karthik C. S., Bingkai Lin, Pasin Manurangsi, and Dániel Marx. 2021. Parameterized intractability of even set and shortest vector problem. *J. ACM* 68, 3 (2021), 16:1–16:40. DOI : <https://doi.org/10.1145/3444942>
- [12] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. 1993. Self-testing/correcting with applications to numerical problems. *J. Comput. System Sci.* 47, 3 (1993), 549–595.
- [13] Boris Bukh, Karthik C. S., and Bhargav Narayanan. 2021. Applications of random algebraic constructions to hardness of approximation. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 237–244. DOI : <https://doi.org/10.1109/FOCS52979.2021.00032>
- [14] Jaroslaw Byrka, Thomas W. Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. 2017. An improved approximation for k-median and positive correlation in budgeted optimization. *ACM Trans. Algorithms* 13, 2 (2017), 23:1–23:31. DOI : <https://doi.org/10.1145/2981561>
- [15] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. 2009. The complexity of satisfiability of small depth circuits. In *Parameterized and Exact Computation*, Jianer Chen and Fedor V. Fomin (Eds.). Springer, Berlin, 75–85.
- [16] Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. 2020. From Gap-Exponential Time Hypothesis to Fixed Parameter Tractable Inapproximability: Clique, Dominating Set, and More. *SIAM J. Comput.* 49, 4 (2020), 772–810. DOI : <https://doi.org/10.1137/18M1166869> arXiv:<https://doi.org/10.1137/18M1166869>
- [17] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. 2002. A constant-factor approximation algorithm for the  $k$ -median problem. *J. Comput. Syst. Sci.* 65, 1 (2002), 129–149. DOI : <https://doi.org/10.1006/jcss.2002.1882>
- [18] Yijia Chen, Yi Feng, Bundit Laekhanukit, and Yanlin Liu. 2025. Simple Combinatorial Construction of the  $k^{o(1)}$ -Lower Bound for Approximating the Parameterized  $k$ -Clique. 263–280. DOI : <https://doi.org/10.1137/1.9781611978315.21> arXiv:<https://pubs.siam.org/doi/pdf/10.1137/1.9781611978315.21>
- [19] Yijia Chen and Martin Grohe. 2007. An isomorphism between subexponential and parameterized complexity theory. *SIAM J. Comput.* 37, 4 (2007), 1228–1258.
- [20] Yijia Chen and Bingkai Lin. 2019. The constant inapproximability of the parameterized dominating set problem. *SIAM J. Comput.* 48, 2 (2019), 513–533. DOI : <https://doi.org/10.1137/17M1127211>
- [21] Rajesh Hemant Chitnis, MohammadTaghi Hajaghayi, and Guy Kortsarz. 2013. Fixed-parameter and approximation algorithms: A new look. In *Parameterized and Exact Computation - 8th International Symposium, IPEC 2013, Sophia Antipolis, France, September 4-6, 2013, Revised Selected Papers (Lecture Notes in Computer Science)*, Gregory Z. Gutin and Stefan Szeider (Eds.), Vol. 8246. Springer, 110–122. DOI : [https://doi.org/10.1007/978-3-319-03898-8\\_11](https://doi.org/10.1007/978-3-319-03898-8_11)

- [22] Vincent Cohen-Addad, Anupam Gupta, Amit Kumar, Euiwoong Lee, and Jason Li. 2019. Tight FPT approximations for  $k$ -median and  $k$ -means. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9–12, 2019, Patras, Greece (LIPIcs)*, Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi (Eds.), Vol. 132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 42:1–42:14. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2019.42>
- [23] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken-ichi Kawarabayashi. 2005. Algorithmic Graph Minor Theory: Decomposition, Approximation, and Coloring. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23–25 October 2005, Pittsburgh, PA, USA, Proceedings*. IEEE Computer Society, 637–646. DOI : <https://doi.org/10.1109/SFCS.2005.14>
- [24] Irit Dinur. 2007. The PCP theorem by gap amplification. *J. ACM* 54, 3 (2007), 12.
- [25] Irit Dinur. 2016. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. Technical Report TR16-128. <https://eccc.weizmann.ac.il/report/2016/128>
- [26] Irit Dinur, Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. 2008. Decodability of group homomorphisms beyond the Johnson bound. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 275–284.
- [27] Irit Dinur and Pasin Manurangsi. 2018. ETH-hardness of approximating 2-CSPs and directed steiner network. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11–14, 2018, Cambridge, MA, USA (LIPIcs)*, Anna R. Karlin (Ed.), Vol. 94. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 36:1–36:20. DOI : <https://doi.org/10.4230/LIPIcs.ITCS.2018.36>
- [28] Irit Dinur and Omer Reingold. 2006. Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM J. Comput.* 36, 4 (2006), 975–1024.
- [29] I. Dinur and D. Steurer. 2014. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*. 624–633.
- [30] Rodney G. Downey and Michael R. Fellows. 1995. Fixed-Parameter Tractability and Completeness I: Basic Results. *SIAM J. Comput.* 24, 4 (1995), 873–921. DOI : <https://doi.org/10.1137/S0097539792228228>
- [31] Rodney G. Downey and Michael R. Fellows. 1995. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science* 141, 1–2 (1995), 109–131.
- [32] Uriel Feige. 1998. A threshold of  $\ln n$  for approximating set cover. *Journal of the ACM (JACM)* 45, 4 (1998), 634–652.
- [33] Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. 1996. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM (JACM)* 43, 2 (1996), 268–292.
- [34] Andreas Emil Feldmann, Karthik C. S., Euiwoong Lee, and Pasin Manurangsi. 2020. A survey on approximation in parameterized complexity: Hardness and algorithms. *Algorithms* 13, 6 (2020), 146.
- [35] Michael R. Fellows. 2003. Blow-ups, win/win’s, and crown rules: Some new directions in FPT. In *Graph-Theoretic Concepts in Computer Science: 29th International Workshop, WG 2003, Elspeet, The Netherlands, June 19–21, 2003. Revised Papers* 29. Springer, 1–12.
- [36] Jörg Flum and Martin Grohe. 2006. *Parameterized Complexity Theory*. Springer.
- [37] Oded Goldreich. 2017. *Introduction to Property Testing*. Cambridge University Press. DOI : <https://doi.org/10.1017/9781108135252>
- [38] Parikshit Gopalan, Venkatesan Guruswami, and Prasad Raghavendra. 2011. List Decoding Tensor Products and Interleaved Codes. *SIAM J. Comput.* 40, 5 (2011), 1432–1462.
- [39] Sudipto Guha and Samir Khuller. 1999. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms* 31, 1 (1999), 228–248. DOI : <https://doi.org/10.1006/jagm.1998.0993>
- [40] Anupam Gupta, Euiwoong Lee, and Jason Li. 2018. Faster exact and approximate algorithms for  $k$ -cut. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 113–123.
- [41] Anupam Gupta, Euiwoong Lee, and Jason Li. 2018. An FPT algorithm beating 2-approximation for  $k$ -cut. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2821–2837.
- [42] Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, Yican Sun, and Kewen Wu. 2025. Almost optimal time lower bound for approximating parameterized clique, CSP, and more, under ETH. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23–27, 2025*, Michal Koucký and Nikhil Bansal (Eds.). ACM, 2136–2144. <https://doi.org/10.1145/3717823.3718130>
- [43] Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, Yican Sun, and Kewen Wu. 2024. Parameterized Inapproximability Hypothesis under Exponential Time Hypothesis. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 24–35.
- [44] Venkatesan Guruswami, Jakub Opršal, and Sai Sandeep. 2020. Revisiting alphabet reduction in Dinur’s PCP. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020) (Leibniz International Proceedings in Informatics (LIPIcs))*, Jarosław Byrka and Raghu Meka (Eds.), Vol. 176. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 34:1–34:14. DOI : <https://doi.org/10.4230/LIPIcs.APPROX/RANDOM.2020.34>

- [45] Venkatesan Guruswami, Xuandi Ren, and Sai Sandeep. 2024. Baby PIH: Parameterized inapproximability of Min CSP. In *39th Computational Complexity Conference, CCC 2024, July 22-25, 2024, Ann Arbor, MI, USA (LIPIcs)*, Rahul Santhanam (Ed.), Vol. 300. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 27:1–27:17. <https://doi.org/10.4230/LIPICS.CCC.2024.27>
- [46] Dorit S. Hochbaum. 1997. Approximation algorithms for NP-hard problems. *SIGACT News* 28, 2 (jun 1997), 40–52. DOI : <https://doi.org/10.1145/261342.571216>
- [47] Russell Impagliazzo and Ramamohan Paturi. 2001. On the complexity of k-SAT. *J. Comput. System Sci.* 62, 2 (2001), 367–375. <https://doi.org/10.1006/JCSS.2000.1727>
- [48] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. 2001. Which problems have strongly exponential complexity? *J. Comput. System Sci.* 63, 4 (2001), 512–530.
- [49] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. 2004. A local search approximation algorithm for  $k$ -means clustering. *Comput. Geom.* 28, 2-3 (2004), 89–112. <https://doi.org/10.1016/j.comgeo.2004.03.003>
- [50] Karthik C. S. and Subhash Khot. 2022. Almost polynomial factor inapproximability for parameterized  $k$ -clique. In *37th Computational Complexity Conference (CCC 2022)*, Vol. 234.
- [51] Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. 2019. On the parameterized complexity of approximating dominating set. *J. ACM* 66, 5 (2019), 33:1–33:38. DOI : <https://doi.org/10.1145/3325116>
- [52] Karthik C. S. and Inbal Livni Navon. 2021. On hardness of approximation of parameterized set cover and label cover: Threshold graphs from error correcting codes. In *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, Hung Viet Le and Valerie King (Eds.). SIAM, 210–223. DOI : <https://doi.org/10.1137/1.9781611976496.24>
- [53] Jonathan Katz and Luca Trevisan. 2000. On the Efficiency of Local Decoding Procedures for Error-Correcting Codes. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing (STOC '00)*. Association for Computing Machinery, New York, NY, USA, 80–86. DOI : <https://doi.org/10.1145/335305.335315>
- [54] Ken-ichi Kawarabayashi and Bingkai Lin. 2020. A nearly 5/3-approximation FPT algorithm for min- $k$ -cut. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 990–999.
- [55] Euiwoong Lee. 2019. Partitioning a graph into small pieces with applications to path transversal. *Math. Program.* 177, 1-2 (2019), 1–19. DOI : <https://doi.org/10.1007/s10107-018-1255-7>
- [56] Shi Li and Ola Svensson. 2016. Approximating  $k$ -median via pseudo-approximation. *SIAM J. Comput.* 45, 2 (2016), 530–547. DOI : <https://doi.org/10.1137/130938645>
- [57] Bingkai Lin. 2018. The parameterized complexity of the  $k$ -biclique problem. *Journal of the ACM (JACM)* 65, 5 (2018), 1–23.
- [58] Bingkai Lin. 2019. A simple gap-producing reduction for the parameterized set cover problem. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece (LIPIcs)*, Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi (Eds.), Vol. 132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 81:1–81:15. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2019.81>
- [59] Bingkai Lin. 2021. Constant approximating  $k$ -clique is W[1]-hard. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, Samir Khuller and Virginia Vassilevska Williams (Eds.). ACM, 1749–1756. DOI : <https://doi.org/10.1145/3406325.3451016>
- [60] Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. 2022. On lower bounds of approximating parameterized  $k$ -clique. In *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France (LIPIcs)*, Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff (Eds.), Vol. 229. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 90:1–90:18. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2022.90>
- [61] Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. 2023. Constant approximating parameterized  $k$ -SETCOVER is W[2]-hard. In *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, Nikhil Bansal and Viswanath Nagarajan (Eds.). SIAM, 3305–3316. DOI : <https://doi.org/10.1137/1.9781611977554.ch126>
- [62] Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. 2023. Improved hardness of approximating  $k$ -clique under ETH. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*. IEEE, 285–306. <https://doi.org/10.1109/FOCS57990.2023.00025>
- [63] Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. 2020. Parameterized complexity and approximability of directed odd cycle transversal. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, Shuchi Chawla (Ed.). SIAM, 2181–2200.
- [64] Daniel Lokshtanov, Saket Saurabh, and Vaishali Surianarayanan. 2020. A parameterized approximation scheme for min  $k$ -cut. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, Sandy Irani (Ed.). IEEE, 798–809.

- [65] Pasin Manurangsi. 2019. A note on max k-vertex cover: Faster FPT-AS, smaller approximate kernel and improved approximation. In *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA (OASIcs)*, Jeremy T. Fineman and Michael Mitzenmacher (Eds.), Vol. 69. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 15:1–15:21. DOI : <https://doi.org/10.4230/OASIcs.SOSA.2019.15>
- [66] Pasin Manurangsi. 2020. Tight running time lower bounds for strong inapproximability of maximum  $k$ -coverage, unique set cover and related problems (via  $t$ -wise agreement testing theorem). In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 62–81.
- [67] Dániel Marx. 2008. Parameterized complexity and approximation algorithms. *Comput. J.* 51, 1 (2008), 60–78. DOI : <https://doi.org/10.1093/comjnl/bxm048>
- [68] Naoto Ohsaka. 2022. On the parameterized intractability of determinant maximization. In *33rd International Symposium on Algorithms and Computation (ISAAC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.
- [69] Anup Rao. 2011. Parallel repetition in projection games and a concentration bound. *SIAM J. Comput.* 40, 6 (2011), 1871–1891. DOI : <https://doi.org/10.1137/080734042> arXiv:<https://doi.org/10.1137/080734042>
- [70] Piotr Skowron and Piotr Faliszewski. 2017. Chamberlin-courant rule with approval ballots: Approximating the maxcover problem with bounded frequencies in FPT time. *J. Artif. Intell. Res.* 60 (2017), 687–716. DOI : <https://doi.org/10.1613/jair.5628>
- [71] Craig A. Tovey. 1984. A simplified NP-complete satisfiability problem. *Discret. Appl. Math.* 8, 1 (1984), 85–89. [https://doi.org/10.1016/0166-218X\(84\)90081-7](https://doi.org/10.1016/0166-218X(84)90081-7)
- [72] Andreas Wiese. 2018. Fixed-parameter approximation schemes for weighted flowtime. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA (LIPIcs)*, Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer (Eds.), Vol. 116. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 28:1–28:19. <https://doi.org/10.4230/LIPIcs.APPROX-RANDOM.2018.28>
- [73] Michał Włodarczyk. 2020. Parameterized inapproximability for steiner orientation by gap amplification. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

## Appendices

### A Inapproximability of Max $k$ -Coverage

To prove the inapproximability of MAX  $k$ -COVERAGE (Theorem 1.4), we need to construct a reduction from PARAMETERIZED CSP to MAX  $k$ -COVERAGE. We follow the classic approach from [32, 66].

- First, we obtain the inapproximability of the parameterized (bi-regular) LABELCOVER problem with respect to the weak agreement value.
- Second, we obtain the hardness of approximating MAX  $k$ -COVERAGE via a gap-preserving reduction (Definition A.4) from LABELCOVER.

To adapt the parameterized regime, we need to generalize [32, Lemma 2.3.1] to the parameterized case.

For convenience we introduce some definitions.

*Definition A.1 (Parameterized LABELCOVER).* The parameterized (bi-regular) LABELCOVER consists of all 2-CSP instances  $G = (V = V_L \cup V_R, E, \Sigma_L, \Sigma_R, \{\Pi_e\}_{e \in E})$  such that:

- The parameter is  $|V_L|$ .
- The variables in  $V_L$  have the alphabet  $\Sigma_L$ , whereas the variables in  $V_R$  has the alphabet  $\Sigma_R$ .
- The constraint graph is a *bipartite graph* between  $V_L$  and  $V_R$ , and each constraint  $\Pi_e$  is a *projection constraint* from some variable  $x_L \in V_L$  to some variable  $x_R \in V_R$ . In other words, there exists a function  $f_e : \Sigma \rightarrow \Sigma$  such that  $\Pi_e(\sigma(x_L), \sigma(x_R)) = 1[\sigma(x_R) = f_e(\sigma(x_L))]$ . Furthermore, the constraint graph is *bi-regular*, i.e., every variable in  $V_L$  has the same degree  $d_L$ , and every variable in  $V_R$  has the same degree  $d_R$ .

We use  $N(x)$  to denote the set of neighbors of a variable  $x \in V$ . The *weak agreement value*  $\text{weakval}(G)$  of  $G$  is defined as the maximum fraction of the right variables  $v_r \in V_R$  such that there is a pair of adjacent left variables that projects the same value on  $v_r$ . Formally,

$$\text{weakval}(G) := \max_{\sigma: V_L \rightarrow \Sigma_L} \Pr_{x_R \in V_R} [\exists \text{ distinct } x_1, x_2 \in N(x_R), f_{(x_1, x_R)}(\sigma(x_1)) = f_{(x_2, x_R)}(\sigma(x_2))].$$

*Definition A.2 (The WEAKLABELCOVER Problem).* The gap parameterized (bi-regular) LABELCOVER problem with respect to the weak agreement value, denoted by  $(\delta, k)$ -WEAKLABELCOVER, has the input an label cover instance  $G$  with the parameter  $|V_L| = k$ , and requires to distinguish whether  $\text{val}(G) = 1$  or  $\text{weakval}(G) < \delta$ .

We need a lemma from [66] to reduce the right alphabet  $\Sigma_R$ .

LEMMA A.3 ([66]). *For every constant  $\delta_0 > 0$ , there is a parameter-preserving FPT reduction from  $(\delta, k)$ -WEAKLABELCOVER to  $(\delta + \delta_0, k)$ -WEAKLABELCOVER such that the output instance has the following property.*

- The parameter  $k$  and the right degree  $d_R$  are preserved after reduction.
- The size of the right alphabet  $|\Sigma_R| \leq \frac{2d_R^2}{\delta_0}$ .

We also need a classic parameter-preserving reduction from  $(\delta, k)$ -WEAKLABELCOVER to MAX k-COVERAGE [32], presented as follows.

*Definition A.4.* Given a LABELCOVER instance  $G = (V = V_L \dot{\cup} V_R, E, \Sigma_L, \Sigma_R, \{\Pi_e\}_{e \in E})$ , the output MAX  $k'$ -COVERAGE instance is  $\mathcal{I} = (\mathcal{S}, \mathcal{U})$ , such that:

- $k' = |V_L|$ .
- $\mathcal{U} = V_R \times \{g : \Sigma_R \rightarrow [d_R]\}$ , i.e., the universe is partitioned into  $|V_R|$  parts, each part consists of all functions from  $\Sigma_R$  to  $[d_R]$ .
- $\mathcal{S}$  is defined by

$$\mathcal{S} := \{S_{x_L, w} \mid x_L \in V_L \wedge w \in \Sigma_L\},$$

where

$$S_{x_L, w} := \{(x_R, g) \mid g(f_{(x_L, x_R)}(w)) = i \text{ and } x_L \text{ is the } i\text{-th smallest element in } N(x_R)\}.$$

This reduction has been analyzed in [32], resulting in two lemmas as follows.

LEMMA A.5 ([32]). *The reduction above has the following properties.*

- (Completeness) If  $\text{val}(G) = 1$ , then there exist  $k'$  sets in  $\mathcal{S}$  that cover all elements in  $\mathcal{U}$ .
- (Soundness) For every collection  $\mathcal{S}_0$  of  $k'$  subsets in  $\mathcal{S}$ , the fraction of covered elements in  $\mathcal{U}$  is at most:

$$1 - \mathbb{E}_{x_R \in V_R} \left[ \prod_{w \in \Sigma_R} \left( 1 - \frac{|N_\Sigma(x_R, w)|}{d_R} \right) \right],$$

where

$$N_\Sigma(x_R, w) := \{x_L \in N(x_R) \mid \exists w' \text{ such that } S(x_L, w') \in \mathcal{S}_0 \wedge f_{(x_L, x_R)}(w') = w\},$$

i.e., all neighbors  $x_L$  consisting of some set  $S(x_L, w')$  where  $(w', w)$  passes the constraint  $(x_L, x_R)$ .

- The reduction takes the runtime  $O(|G| \cdot d_R^{|\Sigma_R|})$  and preserves the parameter.

LEMMA A.6 ([32]). *Given a LABELCOVER instance  $G$  with the right degree  $d_R$ . Let  $\mathcal{I} = (\mathcal{U}, \mathcal{S})$  be the MAX  $k$ -COVERAGE instance obtained after applying the reduction in Definition A.4. If  $\text{weakval}(G) < \delta$ , then any  $k$  sets from  $\mathcal{S}$  can cover at most  $(1 - 1/e + \lambda)$ -fraction of elements in  $\mathcal{U}$ , where  $\lambda := 3d_R^{2/3}\delta^{1/3}$ .*

Lemma A.6 offers us a way to obtain near-optimal soundness when  $\text{weakval}(G)$  is very small. However, when  $\text{weakval}(G)$  is very close to 1, Lemma A.6 cannot obtain anything non-trivial.

Now, we are ready to present our reduction. For this reduction, we need a combinatorial gadget as follows.

PROPOSITION A.7. *For every constant integer  $t \geq 1$ , define  $r = 1 + \lfloor \log(t) \rfloor$  and  $\ell = 2^r$ . Then, there exists a matrix  $M \in \{0, 1\}^{\ell \times \ell}$  such that for every two distinct rows  $i_1, i_2$ , there exists  $\ell/4$  columns  $j$  satisfying  $M[i_1, j] = 1$  and  $M[i_2, j] = 0$ .*

PROOF. We identify the set of columns  $[\ell]$  as the subsets over  $[r]$ . For each row  $i \in [t]$ , we assign a non-empty set  $A_i \subseteq [r]$ . Since  $t \leq \ell$ , we can make  $\{A_i\}_{i \in [t]}$  all distinct. Finally, we construct  $M[i, j]$  as the parity of  $|A_i \cap j|$ , where we recall that  $j$  is the column index viewed as a subset over  $[r]$ . It is straightforward to see that  $M$  satisfies the condition above via the properties of the Hadamard code.  $\square$

We also need parallel repetition distribution [69] to define our reduction.

Definition A.8 (Parallel Repetition Distribution). Given an integer  $\ell \in \mathbb{N}$  and a CSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$ , the  $\ell$ -parallel repetition distribution  $\mathcal{D}_\ell$  is a probability distribution over  $(E \times V)^\ell$ . Each sample of  $\mathcal{D}_\ell$  can be represented as an  $\ell$ -tuple  $\{(e_i, v_i)\}_{i=1}^\ell$ . For every  $i \in [\ell]$ ,  $e_i$  is an independent sample from the uniform distribution over  $E$ , and  $v_i \in V$  is a uniform endpoint of  $e_i$ . We use  $Q_\ell$  to denote the support of  $\mathcal{D}_\ell$ .

Now, we are ready to present the reduction from 2-CSP to WEAKLABELCOVER, which implies the second item of Theorem 1.4.

PROPOSITION A.9. *Fix some constant  $\varepsilon \in (0, 1)$ . For every sufficiently large integer  $t$ , define  $\ell = 2^{\lfloor \log(t) \rfloor}$ . There exist a universal constant  $\alpha > 0$  and a reduction that takes on input a sparse CSP instance  $G = (V, E, \Sigma, \{\Pi_e\}_{e \in E})$  with  $|E| = O(|V|)$ , and outputs a LABELCOVER instance  $G' = (V' = V'_L \dot{\cup} V'_R, E', \Sigma'_L, \Sigma'_R, \{\Pi'_e\}_{e \in E'})$  such that:*

- $V'_L = [t] \times Q_\ell, V'_R = Q_\ell$ .
- The parameter is  $|V'_L| \leq t \cdot (2|E|)^{2t}$ . The alphabet size is  $|\Sigma'_L| \leq |\Sigma|^{2t}$  and  $|\Sigma_R| = O_{\varepsilon, t}(1)$ . The right degree is  $d_R = t$ .
- If  $\text{val}(G) = 1$ , then  $\text{val}(G') = 1$ .
- If  $\text{val}(G) < 1 - \varepsilon$ , then  $\text{weakval}(G') < 2t^2(1 - \varepsilon/2)^{-\alpha\varepsilon\ell}$ .

PROOF. We construct the matrix  $M$  as in Proposition A.7. Then, we construct an intermediate LABELCOVER instance  $G_0 = (V_0 = V_{L,0} \dot{\cup} V_{R,0}, E_0, \Sigma_{L,0}, \Sigma_{R,0}, \{\Pi_{e,0}\}_{e \in E_0})$  as follows. We set  $V_{L,0} = [t] \times Q_\ell, V_{R,0} = Q_\ell$ . In other words, we can identify the left variables in  $G_0$  as all pairs  $(i, r)$  where  $i \in [t]$  and  $r$  are elements  $\in Q_\ell$ , and the right variables in  $G_0$  as all elements  $r \in Q_\ell$ .

Fix an element  $r$  in  $Q_\ell$ , which corresponds to an  $\ell$ -tuple  $\{(e_i, v_i)\}_{i=1}^\ell$  over  $(E \times V)^\ell$ . We set the constraint in  $G_0$  as follows.

- For every  $i \in [t]$ , the alphabet of the left variable  $(i, r)$  consists of all the  $\ell$ -tuple  $(z_1, \dots, z_\ell)$  such that, for every  $j \in [\ell]$ , if  $M[i, j] = 1$ , then  $z_j$  is a satisfying assignment for the constraint  $e_j$ , otherwise  $z_j$  is a value in  $\Sigma$  that represents the value of  $v_j$ .
- The alphabet of the right variable  $r$  consists of all  $\ell$ -tuple  $(z_1, \dots, z_\ell)$  such that, for every  $j \in [\ell]$ ,  $z_j$  is a value in  $\Sigma$  that represents the value of  $v_j$ .
- There is a projection constraint from every  $(i, r)$  to  $r$  that checks the consistency on  $\{v_j\}_{j=1}^\ell$ .

It is clear that if  $\text{val}(G) = 1$ , then  $\text{val}(G_0) = 1$ . If  $\text{val}(G) < 1 - \varepsilon$ , below we show that  $\text{weakval}(G_0) < t^2(1 - \varepsilon/2)^{-\alpha\varepsilon\ell}$ . Fix any assignment  $\sigma : V_{L,0} \rightarrow \Sigma_{L,0}$ . Then,

$$\begin{aligned} & \Pr_{r \in V_{R,0}} [\text{there is a weak agreement at } r] \\ & \leq \mathbb{E}_{r \in R_\ell} \left[ \sum_{i,j \in [t], i \neq j} 1[\sigma(i, r) \text{ and } \sigma(j, r) \text{ projects to the same value}] \right] \\ & = \sum_{i,j \in [t], i \neq j} \Pr_{r \in Q_\ell} [\sigma(i, r) \text{ and } \sigma(j, r) \text{ projects to the same value}] \\ & \leq t^2(1 - \varepsilon/2)^{\alpha\varepsilon\ell}, \end{aligned}$$

where the last inequality follows from the following fact. By the construction of  $M$ , there will be  $\ell/4$  columns  $k$  such that  $M[i, k] = 1$  but  $M[j, k] = 0$ . Thus, the probability that  $\sigma(i, r)$  and  $\sigma(j, r)$  project to the same value is no more than the winning probability of playing  $(\ell/4)$  rounds of parallel repetition over projection games, which is  $(1 - \varepsilon/2)^{\alpha\varepsilon\ell}$  for some universal constant  $\alpha > 0$  [69].

The final instance  $G'$  is the instance after applying the right alphabet reduction Lemma A.3 to  $G_0$  with  $\delta_0 = t^2(1 - \varepsilon/2)^{\alpha\varepsilon\ell}$ . All items in this proposition follow via Lemma A.3 and the construction of  $G_0$ .  $\square$

Then, Theorem 1.4 follows from Theorem 1.2 and the reduction combining Proposition A.9 with Definition A.4. The completeness and the running time lower bound follows from Lemma A.5. The soundness part follows from Lemma A.6, by observing that if we choose  $t$  sufficiently large, the maximum fraction of the covered elements is at most

$$1 - \frac{1}{e} + O\left(t^{4/3}(1 - \varepsilon/2)^{-\alpha\varepsilon t/3}\right) \rightarrow 1 - \frac{1}{e} \quad \text{as } t \rightarrow \infty.$$

## B Missing Proof in Section 2

### B.1 Proof of Theorem 2.6

To prove Theorem 2.6, we first rephrase the statement of the BLR test theorem from [37, Theorem 2.3] as follows.

**THEOREM B.1.** *For every  $\varepsilon \in (0, \frac{1}{6})$ , every group  $G, H$ , and every function  $f : G \rightarrow H$  such that:*

$$\Pr_{x,y \in G} [f(x) + f(y) = f(x+y)] \geq 1 - \varepsilon,$$

*there exists a group homomorphism  $g : G \rightarrow H$  such that is close to  $f$ , i.e.,*

$$\Pr_{x \in G} [f(x) \neq g(x)] \leq 2\varepsilon.$$

**THEOREM 2.6.** *If  $\Pr_{a,b \in \mathbb{F}^k} [w[a] + w[b] = w[a+b]] \geq 1 - \varepsilon$ , then  $\Delta(w, \text{Im}(\text{PWH})) \leq 6\varepsilon$ .*

PROOF. If  $\varepsilon < 1/6$ , we apply Theorem B.1 with  $f \leftarrow f_w, G \leftarrow \mathbb{F}^k, H \leftarrow \mathbb{F}^d$  and obtain  $\Delta(w, \text{Im}(\text{PWH})) \leq 2\varepsilon \leq 6\varepsilon$ . Otherwise  $\varepsilon \geq 1/6$  and we naturally have  $\Delta(w, \text{Im}(\text{PWH})) \leq 1 \leq 6\varepsilon$ .  $\square$

## B.2 Proof of Fact 2.10

FACT 2.10. *The reduction described in Definition 2.9 is an FPT reduction. Recall that  $k = |V|$  is the parameter of  $G$  and  $\Sigma = \mathbb{F}^d$  is the alphabet of  $G$ . We have the following properties for  $G'$ :*

- *ALPHABET.* The alphabet of  $G'$  is  $\Sigma' = \mathbb{F}^{d \cdot q}$ .
- *PARAMETER BLOWUP.* The parameter of  $G'$  is  $|V'| \leq |\mathbb{F}|^k + f(k) + g(k)$ .
- *COMPLETENESS.* For every solution  $\sigma$  of  $G$ , there exists a solution  $\sigma'$  of  $G'$  assigning  $\text{PWH}(\sigma)$  to  $V'_1$ .
- *SOUNDNESS.* For any assignment  $\sigma'$  satisfying  $1 - \frac{\varepsilon}{q}$  fraction of the constraints in  $G'$ , there exists a solution  $\sigma$  of  $G$  such that  $\Delta(\sigma'(V'_1), \text{PWH}(\sigma)) \leq \delta$ .

PROOF. The calculation of the alphabet and the parameter blowup is straightforward. Below, we show the completeness and the soundness.

For the completeness part, let  $x$  be any arbitrary solution of  $G$ , then by the completeness of PPCPP (Definition 2.8), there exists  $\pi_2$  such that the verifier accepts  $\text{PWH}(\sigma) \circ \pi_2$  with probability 1. Thus, we assign  $\text{PWH}(\sigma)$  to  $V'_1$  and  $\pi_2$  to  $V'_2$ , and for each randomness  $r \in [R_A]$  where the verifier queries the positions  $S_r$ , we assign  $z_r$  the projection of  $\text{PWH}(\sigma) \circ \pi_2$  over  $S_2$ . It is straightforward to check this assignment satisfies all constraints.

For the soundness part, fix some assignment  $\sigma'$  that satisfies  $\geq 1 - \frac{\varepsilon}{q}$  fraction of the constraints. Let  $\pi_{1,2,3}$  be the assignment to  $V'_{1,2,3}$ , respectively. Then, we have that.

$$\Pr_{z \in [R_A], i \in S_r} [\pi_3(r)|_i \neq (\pi_1 \circ \pi_2)_i] \leq \frac{\varepsilon}{q}$$

Thus, we have the following by union bound.

$$\Pr_{z \in [R_A]} [\exists i \in S_r, \pi_3(r)|_i \neq (\pi_1 \circ \pi_2)_i] \leq q \cdot \frac{\varepsilon}{q} \leq \varepsilon.$$

By construction of  $G'$ ,  $\pi_3$  only stores accepting configurations of  $A$ , thus  $\pi_1 \circ \pi_2$  will be accepted by  $A$  with probability at least  $1 - \varepsilon$ , the theorem follows by the soundness of PPCPP verifiers (Definition 2.8).  $\square$

Received 11 December 2024; revised 27 April 2025; accepted 29 June 2025

# Coverability in VASS Revisited: Improving Rackoff's Bounds to Obtain Conditional Optimality

MARVIN KÜNNEMANN, Karlsruhe Institute of Technology, Karlsruhe, Germany

FILIP MAZOWIECKI, University of Warsaw, Warszawa, Poland

LIA SCHÜTZE, Max Planck Institute for Software Systems (MPI-SWS), Kaiserslautern, Germany

HENRY SINCLAIR-BANKS\*, Department of Computer Science, University of Warwick, Coventry, United Kingdom of Great Britain and Northern Ireland and Mathematics, Informatics and Mechanics, University of Warsaw, Warszawa, Poland

KAROL WĘGRZYCKI, Max Planck Institute for Informatics, Saarbrücken, Germany

---

Seminal results establish that the coverability problem for Vector Addition Systems with States (VASS) is in EXPSPACE (Rackoff, '78) and is EXPSPACE-hard already under unary encodings (Lipton, '76). More precisely, Rosier and Yen later utilise Rackoff's bounding technique to show that if coverability holds then there is a run of length at most  $n^{2^{\mathcal{O}(d \log(d))}}$ , where  $d$  is the dimension and  $n$  is the size of the given unary VASS. Earlier, Lipton showed that there exist instances of coverability in  $d$ -dimensional unary VASS that are only witnessed by runs of length at least  $n^{2^{\mathcal{O}(d)}}$ . Our first result closes this gap. We improve the upper bound by removing the twice-exponentiated  $\log(d)$  factor, thus matching Lipton's lower bound. This closes the corresponding gap for the exact space required to decide coverability. This also yields a deterministic  $n^{2^{\mathcal{O}(d)}}$ -time algorithm for coverability. Our second result is a matching lower bound, that there does not exist a deterministic  $n^{2^{\mathcal{O}(d)}}$ -time algorithm, conditioned upon the exponential time hypothesis.

---

\*<http://henry.sinclair-banks.com>

Research of Marvin Künnemann is partially supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) grant number 462679611.

Filip Mazowiecki is supported by Polish National Science Centre SONATA BIS-12 grant number 2022/46/E/ST6/00230.

Lia Schütze is supported by the European Union (ERC, FINABIS, 101077902). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Henry Sinclair-Banks was supported by EPSRC Standard Research Studentship (DTP), grant number EP/T5179X/1 and is supported by the ERC grant INF SYS, agreement no. 950398.

Research of Karol Węgrzycki is partially supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) grant number 559177164.

Authors' Contact Information: Marvin Künnemann, Karlsruhe Institute of Technology, Karlsruhe, Baden-Württemberg, Germany; e-mail: marvin.kuennemann@kit.edu; Filip Mazowiecki, University of Warsaw, Warszawa, Poland; e-mail: f.mazowiecki@uw.edu.pl; Lia Schütze, Max Planck Institute for Software Systems (MPI-SWS), Kaiserslautern, Rheinland-Pfalz, Germany; e-mail: lschuetze@mpi-sws.org; Henry Sinclair-Banks, Department of Computer Science, University of Warwick, Coventry, United Kingdom of Great Britain and Northern Ireland and Mathematics, Informatics and Mechanics, University of Warsaw, Warszawa, Poland; e-mail: hsb@mimuw.edu.pl; Karol Węgrzycki, Max Planck Institute for Informatics, Saarbrücken, Saarland, Germany; e-mail: kwegrzyc@mpi-inf.mpg.de.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART33

<https://doi.org/10.1145/3762178>



When analysing coverability, a standard proof technique is to consider VASS with bounded counters. Bounded VASS make for an interesting and popular model due to strong connections with timed automata. Withal, we study a natural setting where the counter bound is linear in the size of the VASS. Here the trivial exhaustive search algorithm runs in  $\mathcal{O}(n^{d+1})$  time. We give evidence to this being near-optimal. We prove that in dimension one this trivial algorithm is conditionally optimal, by showing that  $n^{2-o(1)}$  time is required under the  $k$ -cycle hypothesis. In general, for any fixed dimension  $d \geq 4$ , we show that  $n^{d-2-o(1)}$  time is required under the 3-uniform hyperclique hypothesis.

CCS Concepts: • Theory of computation → Models of computation;

Additional Key Words and Phrases: Vector Addition System, Coverability, Reachability, Fine-Grained Complexity, Exponential Time Hypothesis,  $k$ -Cycle Hypothesis, Hyperclique Hypothesis

**ACM Reference Format:**

Marvin Künemann, Filip Mazowiecki, Lia Schütze, Henry Sinclair-Banks, and Karol Węgrzycki. 2025. Coverability in VASS Revisited: Improving Rackoff’s Bounds to Obtain Conditional Optimality. *J. ACM* 72, 5, Article 33 (October 2025), 27 pages. <https://doi.org/10.1145/3762178>

---

## 1 Introduction

**Vector Addition Systems with States (VASS)** are a popular model of concurrency with a number of applications in database theory [10], business processes [59], and more (see the survey [55]). A  $d$ -dimensional VASS ( $d$ -VASS) consists of a finite automaton equipped with  $d$  non-negative integer counters that can be updated by transitions. A configuration in a  $d$ -VASS consists of a state and a  $d$ -dimensional vector over the naturals. One of the central decision problems for VASS is the *coverability problem*, that asks whether there is a run from a given initial configuration to some configuration with at least the counter values of a given target configuration. Coverability finds applications in the verification of safety conditions, which often equate to whether or not a particular state can be reached without any precise counter values [17, 30]. Roughly speaking, one can use VASS as a modest model for concurrent systems where the dimension corresponds with the number of locations a process can be in and each counter value corresponds with the number of processes in a particular location [27, 31].

In 1978, Rackoff [53] showed that coverability is in EXPSPACE, by proving that if coverability holds then there exists a run of double-exponential length. Following, Rosier and Yen [54] analysed and discussed Rackoff’s ideas in more detail and argued that if coverability holds then it is witnessed by a run of length at most  $n^{2^{\mathcal{O}(d\log(d))}}$ , where  $n$  is the size of the given unary encoded  $d$ -VASS. Furthermore, this yields a  $2^{\mathcal{O}(d\log(d))} \cdot \log(n)$ -space algorithm for coverability. Prior to this in 1976, Lipton [44] proved that coverability is EXPSPACE-hard even when VASS is encoded in unary, by constructing an instance of coverability witnessed only by a run of double-exponential length  $n^{2^{\Omega(d)}}$ . Rosier and Yen [54] also presented a proof that generalised Lipton’s construction to show that  $2^{\Omega(d)} \cdot \log(n)$  space is required for coverability. Although this problem is EXPSPACE-complete in terms of classical complexity, a gap was left open for the exact space needed for coverability [54, Section 1]. By using an approach akin to Rackoff’s argument, we close this fourty-year-old gap by improving the upper bound to match Lipton’s lower bound.

**Result 1:** If coverability holds then there exists a run witnessing coverability of length at most  $n^{\mathcal{O}(d \cdot 2^d)}$  (Theorem 3.3). Accordingly, we obtain an optimal non-deterministic  $2^{\mathcal{O}(d)} \cdot \log(n)$ -space algorithm that decides coverability (Corollary 3.4).

Apart from closing the gap for the exact space needed to decide coverability, we would like to highlight the further relevance of this result. The upper bound of Result 1 relies on careful

analysis of the minimal length of runs. In doing so, we introduce the notion of thin configurations (Definition 3.6). This enriches the ever-growing collection of notions and techniques used when tackling problems about infinite-state systems. We note that thin configurations have already been used in a recent paper [56] to improve the running time of other algorithms for coverability [39] and to improve the upper bound (and close the complexity gap) for coverability in invertible affine VASS [6]. We state this, in part, to emphasise that our main contribution is the conceptual notion of thin configurations, a notion we only discovered by considering the intersection of fine-grained complexity with traditional problems from formal methods. We believe that further investigation into this infrequently studied intersection will be fruitful for finding new concepts, techniques, and points-of-view.

Our bound also implies the existence of a deterministic  $n^{2^{\mathcal{O}(d)}}$ -time algorithm for coverability. We complement this with a matching lower bound on the deterministic running time, that is, conditioned upon the **Exponential Time Hypothesis (ETH)**.

**Result 2:** Under ETH, there is no deterministic  $n^{2^{o(d)}}$ -time algorithm deciding coverability in unary  $d$ -VASS (Theorem 4.2).

One can see that the algorithm, that is, obtained from the upper bound (Result 1) is just a simple exhaustive search in a directed graph with a doubly exponential number of nodes. The lower bound (Result 2) shows that, under ETH, this simple algorithm is essentially optimal.

While our results establish a fast-increasing, conditionally optimal exponent of  $2^{\Theta(d)}$  in the time complexity of the coverability problem, they rely on careful constructions that enforce the observation of large counter values. In certain settings, however, it is natural to instead consider a restricted version of coverability, where all counter values remain *bounded*. This yields one of the simplest models, fixed dimension bounded unary VASS, for which we obtain even tighter results. Decision problems for  $B$ -bounded VASS, where  $B$  forms part of the input, have been studied due to their strong connections to timed automata [28, 33, 49]. We consider linearly-bounded unary VASS, that is, when the maximum counter value is bounded above by a constant multiple of the size of the VASS. Interestingly, coverability and reachability are equivalent in linearly-bounded unary VASS. The trivial algorithm that employs depth-first search on the space of configurations runs in  $\mathcal{O}(n^{d+1})$  time for both coverability and reachability. We provide evidence that the trivial algorithm is optimal. In the following two results, the  $o(1)$  terms are sub-constants that may only depend on the size of the VASS  $n$  (and not the dimension  $d$ , which is fixed).

**Result 3:** Reachability in linearly-bounded unary 1-VASS requires  $n^{2-o(1)}$  time, subject to the  $k$ -cycle hypothesis (Theorem 5.4).

This effectively demonstrates that the trivial algorithm is optimal in the one-dimensional case. For the case of large dimensions, we show that the trivial algorithm only differs from an optimal deterministic-time algorithm by at most an  $n^{3+o(1)}$  factor.

**Result 4:** Reachability in linearly-bounded unary  $d$ -VASS requires  $n^{d-2-o(1)}$  time, subject to the 3-uniform  $k$ -hyperclique hypothesis (Theorem 5.8).

Broadly speaking, these results add a time complexity perspective to the already known result about the space complexity of coverability: for any fixed dimension  $d \geq 0$ , coverability in unary  $d$ -VASS is NL-complete [53].

## Organisation and Overview

Section 3 contains our first main result, the improved upper bound on the space required for coverability. Most notably, in Theorem 3.3, we show that if coverability holds then there exists

a run of length at most  $n^{\mathcal{O}(d \cdot 2^d)}$ . Then, in Corollary 3.4 we are able to obtain a non-deterministic  $\mathcal{O}(d^2 \cdot 2^d \cdot \log(n))$ -space algorithm and a deterministic  $n^{\mathcal{O}(d^2 \cdot 2^d)}$ -time algorithm for coverability. In much of the same way as Rackoff, we proceed by induction on the dimension. The difference is in the inductive step; Rackoff's inductive hypothesis dealt with a case where all counters are bounded by the same well-chosen value. Intuitively speaking, the configurations are bounded within a  $d$ -hypercube. This turns out to be suboptimal. This is due to the fact that the volume of a  $d$ -hypercube with sides of length  $\ell$  is  $\ell^d$ ; unrolling the induction steps gives a bound of roughly  $n^{d \cdot (d-1) \cdots 1} = n^{d!} = n^{2^{\mathcal{O}(d \log(d))}}$ , hence the twice-exponentiated  $\log(d)$  factor. The key ingredient in our proof is to replace the  $d$ -hypercubes with a collection of hyperrectangles with greatly reduced volume, thus reducing the number of configurations in a run witnessing coverability.

Section 4 contains our second main result, the matching lower bound on the time required for coverability, that is, conditioned upon ETH. In Lemma 4.3, we first reduce from finding a  $k$ -clique in a graph to an instance of coverability in bounded unary 2-VASS with zero tests. Then, via Lemma 4.4, we implement the aforementioned technique of Rosier and Yen to, when there is a counter bound, remove the zero tests at the cost of increasing to a  $d$ -dimensional unary VASS. Then, in Theorem 4.2, by carefully selecting a value of  $k = 2^{\Theta(d)}$ , we are able to conclude that if ETH holds, then there does not exist a deterministic  $n^{2^{\mathcal{O}(d)}}$ -time algorithm for coverability in unary  $d$ -VASS. This is because ETH implies that there is no  $f(k) \cdot r^{o(k)}$ -time algorithm for finding a  $k$ -clique in a graph with  $r$  vertices (Theorem 4.1).

Section 5 contains our other results where we study bounded fixed dimension unary VASS. Firstly, Theorem 5.4 states that under the  $k$ -cycle hypothesis (Hypothesis 5.2), there does not exist a deterministic  $n^{2-o(1)}$ -time algorithm deciding reachability in linearly-bounded unary 1-VASS. Further, we conclude in Corollary 5.5, if the  $k$ -cycle hypothesis is assumed then there does not exist a deterministic  $n^{2-o(1)}$ -time algorithm for coverability in (not bounded) unary 2-VASS. Following, we prove Theorem 5.8, that shows there does not exist a deterministic  $n^{d-o(1)}$ -time algorithm for reachability in linearly-bounded unary  $(d+2)$ -VASS under the 3-uniform  $k$ -hyperclique hypothesis (Hypothesis 5.7). We achieve this with two components. First, in Lemma 5.9, we reduce from finding a  $4d$ -hyperclique to an instance of reachability in a bounded unary  $(d+1)$ -VASS with a fixed number of zero tests. Second, via Lemma 5.10, we use the recently developed “controlling counter technique” [21] to remove the fixed number of zero tests at the cost of increasing the dimension by one.

## Related Work

Despite being studied since the seventies, structural properties of the coverability problem for VASS still receive active attention. The set of configurations from which the target can be covered is upwards-closed, meaning that coverability still holds if the initial counter values are increased. An alternative approach, the *backwards algorithm* for coverability, relies on this phenomenon. Starting from the target configuration, one computes the set of configurations from which it can be covered [1]. Thanks to the upwards-closed property, it suffices to maintain the collection of minimal configurations. The backwards algorithm terminates due to Dickson's lemma, however, using Rackoff's bound one can show it runs in double-exponential time [11]. This technique has been analysed both for coverability in VASS and some extensions [29, 39]. Subsequently to our work, it was shown that thinness arises inherently in the configurations explored by the backwards coverability algorithm. Accordingly, an  $n^{2^{\mathcal{O}(d)}}$  upper bound on the running time is attained [56]. Despite the high worst-case complexity, there are many implementations of coverability algorithms relying on the backwards algorithm that work well in practice. Intuitively, the idea is to prune the set of configurations, using relaxations that can be efficiently implemented in SMT solvers [8, 27].

A recently tested family of algorithms that explore the configuration graph (akin to the simple algorithm arising from Result 1) rely on heuristics such as A\* and greedy best-first search [9].

Another central decision problem for VASS is the *reachability problem*, asking whether a run from a given initial configuration to a given target configuration exists. Reachability is a provably harder problem. In essence, reachability differs from coverability by allowing one zero test to each counter. Counter machines, well-known to be equivalent to Turing machines [51], can be seen as VASS with the ability to arbitrarily zero-test counters; coverability and reachability are equivalent here and are undecidable. In 1981, Mayr proved that reachability in VASS is decidable [47], making VASS one of the richest decidable variants of counter machines. Only recently, after decades of work, has the complexity of reachability in VASS been determined to be Ackermann-complete [20, 21, 41, 42]. A widespread technique for obtaining lower bounds for coverability and reachability problems in VASS is to simulate counter machines with some restrictions. Our overall approach to obtaining lower bounds follows suit; we first reduce finding cliques in graphs, finding cycles in graphs, and finding hypercliques in hypergraphs to various intermediate instances of coverability in VASS with additional restrictions or capabilities, such as bounded counters or a fixed number of zero tests. These VASS, which in some sense are restricted counter machines, are then simulated by standard higher-dimensional VASS. Such simulations are brought about by the two previously developed techniques. Rosier and Yen leverage Lipton's construction to obtain VASS that can simulate counter machines with bounded counters [54]. Czerwiński and Orlowski have shown that the presence of an additional counter in a VASS, with carefully chosen transition effects and reachability condition, can be used to implicitly perform a limited number of zero tests [21].

Another studied variant, *bidirected* VASS, has the property that for every transition  $(p, x, q)$  the reverse transition  $(q, -x, p)$  is also present. The reachability problem in bidirected VASS is equivalent to the uniform word problem in commutative semigroups, both of which are EXPSPACE-complete [48]; not to be confused with the reversible (or mutual) reachability problem in general VASS which is also EXPSPACE-complete [40]. In 1982, Meyer and Mayr listed an open problem that stated, in terms of commutative semigroups, the best-known upper bound for coverability in general VASS [53], the best-known lower bound for coverability in bidirected VASS [44], and asked for improvements to these bounds [48, Section 8, Problem 3]. Subsequently, Rosier and Yen refined the upper bound for coverability in general VASS to  $2^{\mathcal{O}(d\log(d))} \cdot \log(n)$  space [54]. Finally, Koppenhagen and Mayr showed that the coverability problem in bidirected VASS can be decided in  $2^{\mathcal{O}(n)}$  space [37], matching the lower bound.

Recently, some work has been dedicated to the coverability problem for low-dimensional VASS [3, 50]. Furthermore, reachability in low-dimensional VASS has been given plenty of attention, in particular for 1-VASS [32, 58] and for 2-VASS [7, 35]. In the restricted class of flat VASS, other fixed dimensions have also been studied [16, 19, 22].

The Dyck reachability problem has been studied from the fine-grained complexity perspective [12, 13, 38, 46]). Coverability in unary 1-VASS is a special case of Dyck-1 reachability; in fact reachability in unary 1-VASS is equivalent to Dyck-1 reachability. Our lower bounds (in Section 5) for reachability in linearly-bounded unary 1-VASS also applies to the Dyck-1 reachability problem. The complexity of coverability in 1-VASS is also closely related to a recently studied problem about reachability for electric cars [25].

## 2 Preliminaries

We use  $\mathbb{Z}$  to denote the set of integers,  $\mathbb{N}$  to denote the set of non-negative integers, and  $\mathbb{R}_+$  to denote the set of positive real numbers. Throughout, we assume that  $\log$  has base 2. We say that function  $f(x) = \mathcal{O}(g(x))$  for some functions  $f, g : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  if there exist constants  $c$  and  $N$  such that  $f(x) \leq c \cdot g(x)$  for all  $x > N$ . Similarly,  $f(x) = o(g(x))$  if for every  $c > 0$  there exists an  $N$

such that, for all  $x > N$ ,  $f(x) < c \cdot g(x)$  [57]. We use  $\text{poly}(n)$  to denote  $n^{\mathcal{O}(1)}$ . We use bold font for vectors. We index the  $i$ th component of a vector  $v$  by writing  $v[i]$ . Given two vectors  $u, v \in \mathbb{Z}^d$  we write  $u \leq v$  if  $u[i] \leq v[i]$  for each  $1 \leq i \leq d$ . For every  $1 \leq i \leq d$ , we write  $e_i \in \mathbb{Z}^d$  to represent the  $i$ th standard basis vector that has  $e_i[i] = 1$  and  $e_i[j] = 0$  for all  $j \neq i$ . Given a vector  $v \in \mathbb{Z}^d$  we define  $\|v\| = \max\{1, |v[1]|, \dots, |v[d']|\}$ . For convenience, given two vectors  $u \in \mathbb{Z}^d$  and  $v \in \mathbb{Z}^{d'}$ , we use  $(u, v)$  to denote the  $(d + d')$ -dimensional vector  $(u[1], \dots, u[d], v[1], \dots, v[d'])$ . Through this article, we analyse the running time of algorithms in the standard “word RAM model” with  $O(\log(n))$ -bit words, where  $n$  is the size of the input (see [34] for a survey).

A  $d$ -dimensional Vector Addition System with States ( $d$ -VASS)  $\mathcal{V} = (Q, T)$  consists of a non-empty finite set of states  $Q$  and a non-empty set of transitions  $T \subseteq Q \times \mathbb{Z}^d \times Q$ . A *configuration* of a  $d$ -VASS is a pair  $(q, v) \in Q \times \mathbb{N}^d$  consisting of the current state  $q$  and current counter values  $v$ , denoted  $q(v)$ . Given two configurations  $p(u), q(v)$ , we write  $p(u) \rightarrow q(v)$  if there exists  $t = (p, x, q) \in T$  where  $u + x = v$ . We may refer to  $x$  as the *effect* or *update* of a transition and may also write  $p(u) \xrightarrow{t} q(v)$  to emphasise the transition  $t$  taken.

A *path* in a VASS is a (possibly empty) sequence of transitions  $((p_1, x_1, q_1), \dots, (p_\ell, x_\ell, q_\ell))$ , where  $(p_i, x_i, q_i) \in T$  for all  $1 \leq i \leq \ell$  and such that the start and end states of consecutive transitions match  $q_i = p_{i+1}$  for all  $1 \leq i \leq \ell - 1$ . The *effect* of a path is the sum of the effects of the transitions in the path. A *run*  $\pi$  in a VASS is a sequence of configurations  $\pi = (q_0(v_0), \dots, q_\ell(v_\ell))$  such that  $q_i(v_i) \rightarrow q_{i+1}(v_{i+1})$  for all  $1 \leq i \leq \ell - 1$ . We denote the length of the run by  $\text{len}(\pi) := \ell + 1$ ; we denote the effect of the run by  $\text{eff}(\pi) := v_\ell - v_0$ . If there is such a run  $\pi$ , we can write  $q_0(v_0) \xrightarrow{\pi} q_\ell(v_\ell)$ . We may also write  $p(s) \xrightarrow{*} q(t)$  if there exists a run from  $p(s)$  to  $q(t)$ . The *underlying path* of a run  $\pi = (q_0(v_0), \dots, q_\ell(v_\ell))$  is the sequence of transitions  $(t_1, \dots, t_\ell)$  such that, for every  $0 \leq i \leq \ell - 1$ ,  $q_i(v_i) \xrightarrow{t_i} q_{i+1}(v_{i+1})$ .

We do allow for zero-dimensional VASS, that is, VASS with no counters, which can be seen as just directed graphs. A *hypergraph* is a generalisation of the graph. Formally, a hypergraph is a tuple  $H = (V, E)$  where  $V$  is a set of vertices and  $E$  is a collection of non-empty subsets of  $V$  called *hyperedges*. For an integer  $\mu$ , a hypergraph is  $\mu$ -uniform if each hyperedge has cardinality  $\mu$ . Note that a 2-uniform hypergraph is a standard graph.

We study the complexity of the *coverability problem*. An instance  $(\mathcal{V}, p(s), q(t))$  of coverability asks whether there is a run in the given VASS  $\mathcal{V}$  from the given initial configuration  $p(s)$  to a configuration  $q(t')$  with counter values  $t' \geq t$ . At times, we also consider the *reachability problem* that additionally requires  $t' = t$  so that the target configuration is reached exactly.

To measure the complexity of these problems we need to discuss the encoding used. In *unary encoding*, a  $d$ -VASS  $\mathcal{V} = (Q, T)$  has size  $\|\mathcal{V}\| = |Q| + \sum_{(p, x, q) \in T} \|x\|$ . An instance of coverability in a unary-encoded  $d$ -VASS  $(\mathcal{V}, p(s), q(t))$  has size  $\|\mathcal{V}\| + \|s\| + \|t\|$ . We define a *unary  $d$ -VASS*  $\mathcal{U} = (Q', T')$  to have restricted transitions  $T' \subseteq Q' \times \{-1, 0, 1\}^d \times Q'$ , the size is therefore  $\|\mathcal{U}\| = |Q'| + |T'|$ . For any unary encoded  $d$ -VASS  $\mathcal{V}$  there exists an equivalent unary  $d$ -VASS  $\mathcal{U}$  such that  $\|\mathcal{U}\| = \|\mathcal{V}\|$ .

It is well known that coverability in  $d$ -VASS can be reduced, in logarithmic space, to the reachability problem. Indeed, for an instance  $(\mathcal{V}, p(s), q(t))$  of coverability, define  $\mathcal{V}' = (Q, T')$  that has additional decrementing transitions at the target state  $q$ , precisely  $T' = T \cup \{(q, -e_i, q) : 1 \leq i \leq d\}$ . It is clear that  $p(s) \xrightarrow{*} q(t')$ , for some  $t' \geq t$ , in  $\mathcal{V}$  if and only if  $p(s) \xrightarrow{*} q(t)$  in  $\mathcal{V}'$ .

A *B-bounded  $d$ -VASS*, in short  $(B, d)$ -VASS, is given as an integer upper bound on the counter values  $B \in \mathbb{N}$  and  $d$ -VASS  $\mathcal{V}$ . A configuration in a  $(B, d)$ -VASS is a pair  $q(v) \in Q \times \{0, \dots, B\}^d$ . The notions of paths and runs in bounded VASS remain the same as for VASS, but are accordingly adapted for the appropriate bounded configurations. We note that one should think that  $B$  forms

part of the problem statement, not the input, as it will be given implicitly by a function depending on the size of the VASS. For example, we later consider *linearly-bounded d-VASS*, in which  $B = \mathcal{O}(\|\mathcal{V}\|)$ .

A *d-dimensional Vector Addition System (d-VAS)*  $\mathcal{U}$  is a system without states, consisting only of a non-empty collection of transitions  $\mathcal{U} \subseteq \mathbb{Z}^d$ . All definitions, notations, and problems carry over for VAS except that, for simplicity, we drop the states across the board. For example, a configuration in a VAS is just a vector  $v \in \mathbb{N}^d$ . A well-known result from the seventies by Hopcroft and Pansiot is that one can simulate the states of a VASS at the cost of three extra dimensions in a VAS [35].

**LEMMA 2.1** (SEE [35, LEMMA 2.1]). *Let  $\mathcal{V} = (Q, T)$  be a d-VASS. There exists a  $(d + 3)$ -VAS  $\mathcal{U}$  that simulates  $\mathcal{V}$  and such that  $\|\mathcal{U}\| = \text{poly}(\|\mathcal{V}\|)$ . Precisely, there exist an injective function  $f : Q \rightarrow \mathbb{N}^3$  such that there is a run  $p(s) \xrightarrow{\pi} q(t)$  in  $\mathcal{V}$  if and only if there is a run  $(f(p), s) \xrightarrow{\rho} (f(q), t)$  in  $\mathcal{U}$ . Moreover,  $f$  can be computed in  $\text{poly}(\|\mathcal{V}\|)$  time and  $\text{len}(\rho) = 3 \cdot \text{len}(\pi)$ .*

Roughly speaking, the VAS obtained has an equivalent reachability relation between configurations; a configuration  $q(x)$  in the original VASS corresponds with a configuration  $((a, b, c), x)$  in the VAS, where  $a, b, c \leq 2|Q|^2$  are carefully chosen values that together represent the state  $q$ . Each transition of the  $d$ -VASS  $\mathcal{V}$  is split into a triplet of vectors that get added to the  $(d + 3)$ -VAS  $\mathcal{U}$  where the combined effect of such a triplet is indeed the same as the effect of the original transition in  $\mathcal{V}$ . Importantly, during a run in  $\mathcal{U}$ , each triplet of vectors must be taken in sequence so a run of length  $\ell$  in  $\mathcal{V}$  corresponds to a run of length  $3\ell$  in  $\mathcal{U}$  and vice versa.

### 3 Improved Bounds on the Maximum Counter Value

This section is devoted to our improvement of the seminal result of Rackoff. Throughout, we fix our attention to the arbitrary instance  $(\mathcal{V}, p(s), q(t))$  of the coverability problem in a  $d$ -VASS  $\mathcal{V} = (Q, T)$  from the initial configuration  $p(s)$  to a configuration  $q(t')$  with at least the counter values of the target configuration  $q(t)$ . We denote  $n = \|\mathcal{V}\| + \|s\| + \|t\|$ . The following two theorems follow from Rackoff's technique and subsequent work by Rosier and Yen, in particular see [53, Lemma 3.4 and Theorem 3.5] and [54, Theorem 2.1 and Lemma 2.2].

**THEOREM 3.1** (COROLLARY OF [53, LEMMA 3.4] AND [54, THEOREM 2.1]). *Suppose  $p(s) \xrightarrow{*} q(t')$  for some  $t' \geq t$ . Then there exists a run  $\pi$  such that  $p(s) \xrightarrow{\pi} q(t'')$  for some  $t'' \geq t$  and  $\text{len}(\pi) \leq n^{2^{\mathcal{O}(d\log(d))}}$ .*

**THEOREM 3.2** (CF. [53, THEOREM 3.5]). *For a given  $d$ -VASS  $\mathcal{V}$ , integer  $\ell$ , and two configurations  $p(s)$  and  $q(t)$ , there is an algorithm that determines the existence of a run  $\pi$  of length  $\text{len}(\pi) \leq \ell$  that witnesses coverability, so  $p(s) \xrightarrow{\pi} q(t')$  for some  $t' \geq t$ . The algorithm can be implemented to run in non-deterministic  $\mathcal{O}(d\log(n \cdot \ell))$  space or deterministic  $2^{\mathcal{O}(d\log(n \cdot \ell))}$  time.*

**PROOF.** In runs whose length is bounded by  $\ell$ , the observed counter values are trivially bounded by  $n \cdot \ell$ . Notice that every configuration can be written in  $\mathcal{O}(d\log(n \cdot \ell))$  space. A non-deterministic algorithm can therefore decide coverability by guessing a run on-the-fly by maintaining the current configuration and the length of the run (using a step counter). The algorithm accepts if and only if  $t$  is covered by the final configuration.

The second part follows from the standard construction that if a problem can be solved in  $S(n)$  non-deterministic space then it can be solved in  $2^{\mathcal{O}(S(n))}$  deterministic time. Indeed, one can construct the graph of all configurations and check whether there is a path from the initial configuration to the final configuration. Since there are at most  $2^{\mathcal{O}(d\log(n \cdot \ell))}$  many configurations, this can be completed in  $2^{\mathcal{O}(d\log(n \cdot \ell))}$  time.  $\square$

Note that Theorem 3.1 combined with Theorem 3.2 yield non-deterministic  $2^{\mathcal{O}(d \log(d))} \cdot \log(n)$ -space and deterministic  $n^{2^{\mathcal{O}(d \log(d))}}$ -time algorithms for coverability. Our result improves this by an  $\mathcal{O}(\log(d))$  factor in the second exponent.

**THEOREM 3.3.** *Suppose  $p(s) \xrightarrow{*} q(t')$  for some  $t' \geq t$ . Then there exists a run  $\pi$  such that  $p(s) \xrightarrow{\pi} q(t'')$  for some  $t'' \geq t$  and  $\text{len}(\pi) \leq n^{\mathcal{O}(d \cdot 2^d)}$ .*

Combining Theorem 3.2 with Theorem 3.3 yields the following corollary.

**COROLLARY 3.4.** *Coverability in  $d$ -VASS can be decided by both a non-deterministic  $\mathcal{O}(d^2 \cdot 2^d \cdot \log(n))$ -space algorithm and a deterministic  $n^{\mathcal{O}(d^2 \cdot 2^d)}$ -time algorithm.*

**PROOF.** Let  $\ell = n^{c \cdot d \cdot 2^d}$  for some constant  $c$  such that Theorem 3.3 tells us that if  $p(s) \xrightarrow{*} q(t')$  for some  $t' \geq t$ , then there exists a run  $\pi$  such that  $p(s) \xrightarrow{\pi} q(t'')$  for some  $t'' \geq t$  and  $\text{len}(\pi) \leq \ell$ . By Theorem 3.2, we know that there exists a non-deterministic algorithm that decides coverability between  $p(s)$  and  $q(t)$  in the following space.

$$\begin{aligned} \mathcal{O}(d \log(n \cdot \ell)) &= \mathcal{O}(d \log(n) + d \log(\ell)) = \mathcal{O}\left(d \log(n) + d \log\left(n^{c \cdot d \cdot 2^d}\right)\right) \\ &= \mathcal{O}\left(d \log(n) + d \log(n) \cdot c \cdot d \cdot 2^d\right) \\ &= \mathcal{O}\left(d^2 \cdot 2^d \cdot \log(n)\right). \end{aligned}$$

Also by Theorem 3.2, we know that there exists a deterministic algorithm that decides coverability between  $p(s)$  and  $q(t)$  in  $2^{\mathcal{O}(d \log(n \cdot \ell))} = 2^{\mathcal{O}(d^2 \cdot 2^d \cdot \log(n))} = n^{\mathcal{O}(d^2 \cdot 2^d)}$  time.  $\square$

The rest of this section is dedicated to the proof of Theorem 3.3. Theorem 3.3 is a corollary of Lemma 2.1 and the following lemma (Lemma 3.5); a formal proof can be found at the end of this section. By Lemma 2.1, instead of handling a given VASS, we may instead handle an equivalent VAS with three additional counters whose size is polynomial in  $n$ . Recall that, as there are no states, a  $d$ -VAS consists only of a set of vectors in  $\mathbb{Z}^d$  which we still refer to as transitions. A configuration is just a vector in  $\mathbb{N}^d$ . Accordingly, we may fix our attention on the instance  $(\mathcal{V}, s, t)$  of the coverability problem in a  $d$ -VAS  $\mathcal{V} = \{v_1, \dots, v_m\}$  from the initial configuration  $s$  to a configuration  $t'$ , that is, at least as great as the target configuration  $t$ . Although Theorem 3.3 is a stronger statement than Theorem 3.1, we imitate the structure of Rackoff's proof; we proceed by induction on the dimension  $d$ .

**LEMMA 3.5.** *Define  $L_i := n^{i \cdot 2^i}$ , and let  $t \in \mathbb{N}^d$  such that  $\|t\| \leq n$ . For any  $s \in \mathbb{N}^d$ , if  $s \xrightarrow{*} t'$  for some  $t' \geq t$  then there exists a run  $\pi$  such that  $s \xrightarrow{\pi} t''$  for some  $t'' \geq t$  and  $\text{len}(\pi) \leq L_d$ .*

The base case is  $d = 0$ . In a 0-dimensional VAS, the only possible configuration is the empty vector  $\epsilon$  and therefore there is only the trivial run  $\epsilon \xrightarrow{*} \epsilon$ . This trivially satisfies the lemma.

For the inductive step, when  $d \geq 1$ , we assume that Lemma 3.5 holds for all lower dimensions  $0, \dots, d - 1$ . Let  $\pi = (c_0, c_1, \dots, c_\ell)$  be a run with minimal length such that  $s \xrightarrow{\pi} t'$  for some  $t' \geq t$ , so in particular,  $c_0 = s$  and  $c_\ell = t'$ . Our objective is to prove that  $\text{len}(\pi) = \ell + 1 \leq L_d$ . Observe that configurations  $c_i$  need to be distinct, else  $\pi$  could be shortened trivially. We introduce the notion of a *thin configuration*.

**Definition 3.6 (Thin Configuration).** In a  $d$ -VAS, we say that a configuration  $c \in \mathbb{N}^d$  is *thin* if there exists a permutation  $\sigma$  of  $\{1, \dots, d\}$  such that  $c[\sigma(i)] < M_i$  for every  $i \in \{1, \dots, d\}$ , where  $M_0 := n$  and for  $i \geq 1$ ,  $M_i := L_{i-1} \cdot n$ .

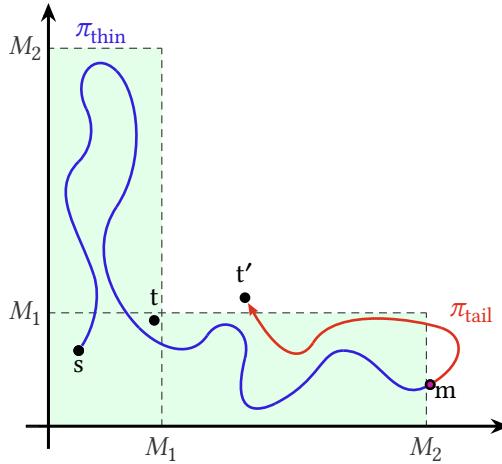


Fig. 1. The schematic view of Claims 3.7 and 3.8, restricted to the two-dimensional case. Here,  $s$  is the initial configuration and  $t$  is the target configuration. The shortest run witnessing coverability (to  $t' \geq t$ ) is drawn. Every configuration inside the green shaded polygon is thin, where each rectangular component of the green shaded polygon corresponds to a permutation of the indices. Observe that  $m$  is the first configuration, just outside the green shaded polygon, that is, not thin. The prefix of the run from  $s$  to  $m$  is  $\pi_{\text{thin}}$  (drawn in blue); Claim 3.7 bounds the maximum possible length of  $\pi_{\text{thin}}$  by the volume of the green polygon. The suffix of the run from  $m$  to  $t'$  is  $\pi_{\text{tail}}$  (drawn in red); Claim 3.8 bounds the maximum possible length of  $\pi_{\text{tail}}$  by  $L_{d-1}$ .

Recall, from above, the run  $\pi = (c_0, c_1, \dots, c_\ell)$ . Let  $t \in \{0, \dots, \ell\}$  be the first index where  $c_t$  is not thin, otherwise let  $t = \ell + 1$  if every configuration in  $\pi$  is thin. We decompose the run about the  $t$ th configuration  $\pi = \pi_{\text{thin}} \cdot \pi_{\text{tail}}$ , where  $\pi_{\text{thin}} := (c_0, \dots, c_{t-1})$  and  $\pi_{\text{tail}} := (c_t, \dots, c_\ell)$ . Note that  $\pi_{\text{thin}}$  or  $\pi_{\text{tail}}$  can be empty. Subsequently, we individually analyse the lengths of  $\pi_{\text{thin}}$  and  $\pi_{\text{tail}}$  (see Figure 1). We also denote  $m = c_t$  to be the first configuration, that is, not thin.

**CLAIM 3.7.**  $\text{len}(\pi_{\text{thin}}) \leq d! \cdot n^d \cdot L_{d-1} \cdot \dots \cdot L_0$ .

**PROOF.** By definition, every configuration in  $\pi_{\text{thin}}$  is thin. Moreover, since  $\pi$  has a minimal length, no configurations in  $\pi$  repeat; the same is therefore true for the prefix  $\pi_{\text{thin}}$ . We now count the number of possible thin configurations. There are  $d!$  many permutations of  $\{1, \dots, d\}$ . For a given permutation  $\sigma$  and an index  $i \in \{1, \dots, d\}$ , we know that for a thin configuration  $c$ ,  $0 \leq c[\sigma(i)] < M_i$ , so there are at most  $M_i = L_{i-1} \cdot n$  many possible values on the  $\sigma(i)$ th counter. Hence the total number of thin configurations is at most  $d! \cdot \prod_{i=1}^d (L_{i-1} \cdot n) = d! \cdot n^d \cdot L_{d-1} \cdot \dots \cdot L_0$ .  $\square$

**CLAIM 3.8.**  $\text{len}(\pi_{\text{tail}}) \leq L_{d-1}$ .

**PROOF.** Consider  $m \in \mathbb{N}^d$ , the first configuration of  $\pi_{\text{tail}}$ . Let  $\sigma$  be a permutation such that  $m[\sigma(1)] \leq m[\sigma(2)] \leq \dots \leq m[\sigma(d)]$ . Given that  $m$  is not thin, for every permutation  $\sigma'$  there exists an  $i \in \{1, \dots, d\}$  such that  $m[\sigma'(i)] \geq M_i$ ; in particular, this holds for  $\sigma$ . Note that this also implies  $M_i \leq m[\sigma(i+1)] \leq \dots \leq m[\sigma(d)]$ .

We construct an  $(i-1)$ -VASS  $\mathcal{U}$  from  $\mathcal{V}$  by ignoring the counters  $\sigma(i), \dots, \sigma(d)$ . Formally,  $u \in \mathcal{U}$  if there is  $v \in \mathcal{V}$  such that  $u[j] = v[\sigma(j)]$  for each  $1 \leq j \leq i-1$ . In such a case we say  $u$  is the *projection* of  $v$  via  $\sigma$ . We will use the inductive hypothesis to show that there is a short path  $\rho'$  in  $\mathcal{U}$  from (the projection of)  $m$  covering (the projection of)  $t$ . We will then show that the remaining components of  $m$  are large enough that the embedding of  $\rho'$  into  $\mathcal{V}$  maintains its covering status.

Recall that  $t'$  is the final configuration of the run  $\pi$ . Note that the run  $\pi_{\text{tail}}$  induces a run  $\pi'_{\text{tail}}$  in  $\mathcal{U}$  by permuting and projecting every configuration. More precisely,  $(m[\sigma(1)], \dots, m[\sigma(i-1)]) \xrightarrow{\pi'_{\text{tail}}} (t'[\sigma(1)], \dots, t'[\sigma(i-1)])$ . By the inductive hypothesis there exists a run  $\rho'$  in  $\mathcal{U}$  such that  $(m[\sigma(1)], \dots, m[\sigma(i-1)]) \xrightarrow{\rho'} (t''[\sigma(1)], \dots, t''[\sigma(i-1)])$ , such that  $(t''[\sigma(1)], \dots, t''[\sigma(i-1)]) \geq (t[\sigma(1)], \dots, t[\sigma(i-1)])$  and  $\text{len}(\rho') \leq L_{i-1}$ .

Let  $(u_1, \dots, u_{\text{len}(\rho')})$  be the underlying path of the run  $\rho'$ , that is, the sequence of transitions in  $\mathcal{U}$  that are sequentially added to form the run  $\rho'$ . By construction, each transition vector  $u_i \in \mathcal{U}$  has a corresponding transition vector  $v_i \in \mathcal{V}$  where  $u_i$  is the projection of  $v_i$  via  $\sigma$ . We will now show that the following run witnesses coverability of  $t$ .

$$\rho = \left( m, m + v_1, m + v_1 + v_2, \dots, m + \sum_{j=1}^{\text{len}(\rho')} v_j \right).$$

To this end, we verify that (i)  $\rho$  is a run, that is, all configurations lie in  $\mathbb{N}^d$ , and (ii) the final configuration indeed covers  $t$ . For components  $\sigma(1), \dots, \sigma(i-1)$ , this follows directly from the inductive hypothesis. For all other components we will show that *all* configurations of  $\rho$  are covering  $t$  in these components. This satisfies both (i) and (ii).

Let  $j$  be any of the remaining components. Recall that by the choice of  $m$ ,  $m[j] \geq M_i = n \cdot L_{i-1}$ . Since  $n > \|\mathcal{V}\| \geq \|v_j\|$  for every  $1 \leq j \leq \text{len}(\rho')$ , this means that in a single step, the value of a counter can change by at most  $(n-1)$ . Given that  $\text{len}(\rho) = \text{len}(\rho') \leq L_{i-1}$ , the value on each of the remaining components must be at least  $n$  for every configuration in  $\rho$ . In particular, observing that  $\|t\| \leq n$ , the final configuration of  $\rho$  satisfies

$$m + \sum_{j=1}^{\text{len}(\rho')} v_j \geq t.$$

Finally, observe that  $\text{len}(\rho) = \text{len}(\rho') \leq L_{i-1} \leq L_{d-1}$ . □

Now, we can prove that Lemma 3.5 follows from Claims 3.7 and 3.8.

**PROOF OF LEMMA 3.5.** From Claims 3.7 and 3.8,

$$\begin{aligned} \text{len}(\pi) &\leq \text{len}(\pi_{\text{thin}}) + \text{len}(\pi_{\text{tail}}) \leq d! \cdot n^d \cdot L_{d-1} \cdot \dots \cdot L_0 + L_{d-1} \\ &\leq 2 \cdot d! \cdot n^d \cdot L_{d-1} \cdot \dots \cdot L_0. \end{aligned}$$

Recall that  $n \geq 2$  and  $d! \leq n^d$ ; observe that  $2 \cdot d! \cdot n^d \leq n^{d^2+1}$ . Hence,

$$\text{len}(\pi) \leq n^{d^2+1} \cdot L_{d-1} \cdot \dots \cdot L_0.$$

Next, we use the definition of  $L_i := n^{i \cdot 2^i}$  to show

$$\text{len}(\pi) \leq n^{d^2+1} \cdot \prod_{i=0}^{d-1} n^{i \cdot 2^i} \leq n^{(d^2+1+\sum_{i=0}^{d-1} i \cdot 2^i)}.$$

Next, we use the fact that  $\sum_{i=0}^{d-1} i \cdot 2^i = d \cdot 2^d - 2^{d+1} + 2$ , so when  $d \geq 1$ ,  $d^2 + 1 + \sum_{i=0}^{d-1} i \cdot 2^i \leq d \cdot 2^d$ . Therefore,

$$\text{len}(\pi) \leq n^{d \cdot 2^d} = L_d. □$$

To conclude this section, we now transfer the upper bound to VASS to prove Theorem 3.3.

**PROOF OF THEOREM 3.3.** Let  $(\mathcal{V}, p(s), q(t))$  be an instance of  $d$ -VASS coverability of size  $n$ . By Lemma 2.1, we can construct a  $(d+3)$ -dimensional VAS  $\mathcal{U}$  of size  $\text{poly}(\|\mathcal{V}\|)$  and vectors  $p, q \in \mathbb{N}^3$  such that  $\|p\|, \|q\| \leq \text{poly}(\|\mathcal{V}\|)$  and, for some  $t' \geq t$ , there is a run from  $p(s)$  to  $q(t')$  in  $\mathcal{V}$  if and only if there is a run from  $(p, s)$  to  $(q, t')$  in  $\mathcal{U}$ .

Suppose  $(p, s) \xrightarrow{\rho} (q, t')$  is the minimal length run in  $\mathcal{U}$  such that  $t' \geq t$ . By Lemma 3.5, we know that  $\text{len}(\rho) \leq \text{poly}(n)^{(d+3)\cdot 2^{d+3}} = n^{\mathcal{O}(1)\cdot(d+3)\cdot 8\cdot 2^d} = n^{\mathcal{O}(d\cdot 2^d)}$ . As stated at the end of Lemma 2.1, there exists a corresponding run  $p(s) \xrightarrow{\pi} q(t')$  in  $\mathcal{V}$  where  $t' \geq t$  and  $\text{len}(\pi) = \frac{\text{len}(\rho)}{3}$ . We therefore obtain a run from  $p(s)$  that covers  $q(t)$  in  $\mathcal{V}$  of length  $\ell = \frac{\text{len}(\rho)}{3} \leq n^{\mathcal{O}(d\cdot 2^d)}$ .  $\square$

#### 4 Conditional Time Lower Bound for Coverability

In this section, we present a conditional lower bound based on the ETH [36]. Roughly speaking, ETH is a conjecture that a  $\lambda$ -variable instance of 3-SAT cannot be solved by a deterministic  $2^{o(\lambda)}$ -time algorithm (for a modern survey, see [45]). In our reductions, it will be convenient for us to reduce via the  $k$ -clique problem instead of reducing directly from 3-SAT. In the  $k$ -clique problem, the input is a graph  $G = (V, E)$ ,  $k$  is a parameter, and the task is to decide whether there is a set of  $k$  pairwise adjacent vertices in  $V$ . For a graph with  $r$  nodes, the naive algorithm for  $k$ -clique runs in  $\mathcal{O}(r^k)$  time. Even though the exact constant in the dependence on  $k$  can be improved [52], ETH implies that the exponent must have a linear dependence on  $k$ .

**THEOREM 4.1** ([14, THEOREM 4.2], [15, THEOREM 4.5], AND [18, THEOREM 14.21]). *Assuming the ETH, there is no algorithm running in  $f(k) \cdot r^{o(k)}$  time for the  $k$ -clique problem for any computable function  $f$ . Moreover, one can assume that  $G$  is  $k$ -partite, i.e.  $G = (V_1 \cup \dots \cup V_k, E)$  and  $E \subseteq \{\{u, v\} : u \in V_i, v \in V_j, i \neq j\}$ .*

We will use Theorem 4.1 to show the following conditional lower bound for coverability in unary  $d$ -VASS, which is proved at the end of this section.

**THEOREM 4.2.** *Assuming the ETH, there does not exist an  $n^{2^{o(d)}}$ -time algorithm deciding coverability in a  $d$ -VASS of size  $n$ .*

We remark, as corollary of Theorem 4.2, that there does not exist an  $n^{2^{o(d)}}$ -time algorithm for coverability in  $d$ -VAS under ETH.<sup>1</sup> Here, to be precise,  $n = \sum_{x \in \mathcal{V}} \|x\|$  refers to the size of a  $d$ -VAS  $\mathcal{V}$  encoded in unary. This follows from the fact that VAS can simulate VASS [35, Lemma 2.1]; see the text surrounding Lemma 2.1 for a brief discussion about this simulation.

To prove Theorem 4.2, we first reduce the  $k$ -clique problem to coverability in bounded 2-VASS with the ability to perform a fixed number of zero tests. We will then leverage a result by Rosier and Yen to construct an equivalent, with respect to coverability,  $(\mathcal{O}(\log(k)))$ -VASS without zero tests.

**LEMMA 4.3.** *Given a  $k$ -partite graph  $G = (V_1 \cup \dots \cup V_k, E)$  with  $r$  vertices, there exists a unary  $(\mathcal{O}(r^{2k}), 2)$ -VASS  $\mathcal{T}$  with  $\mathcal{O}(k^2)$  zero tests and two designated states  $q_I, q_F$  such that there is a  $k$ -clique in  $G$  if and only if there exists a run from  $q_I(0)$  to  $q_F(t)$  in  $\mathcal{T}$ , for some  $t \geq 0$ . Moreover,  $\|\mathcal{T}\| \leq \text{poly}(r+k)$  and  $\mathcal{T}$  can be constructed in  $\text{poly}(r+k)$  time.*

**PROOF.** Without loss of generality, we may assume that each of the  $k$  vertex subsets in the graph has the same size  $|V_1| = \dots = |V_k| = \ell$ . Thus  $r = k \cdot \ell$ . For convenience, we denote  $V = \{v_{i,j} : i \in \{1, \dots, k\}, j \in \{1, \dots, \ell\}\}$ .

<sup>1</sup>Recall, from Section 2, that a  $d$ -VAS is a  $d$ -dimensional vector addition system (without states).

---

**ALGORITHM 1:** A counter program for a VASS with zero tests with two counters  $x$  and  $y$ .

---

```

input:  $x = 0, y = 0$ 
 $x += 1$ 
for  $i \leftarrow 1$  to  $k$  do
|   guess  $j \in \{1, \dots, \ell\}$ 
|   Multiply[ $x, p_{i,j}$ ]
end
for  $(i, j) \in \{1, \dots, k\}^2, i < j$  do
|   guess  $e \in \{\{u, v\} \in E : u \in V_i, v \in V_j\}$ 
|   Edge[ $e$ ]
end

```

---

We begin by sketching the main ideas behind the reduction before they are implemented. We start by finding the first  $r = k \cdot \ell$  primes and associating a distinct prime  $p_{i,j}$  to each vertex  $v_{i,j} \in V$ . Note that a product of  $k$  different primes uniquely corresponds to a selection of  $k$  vertices. Thus the idea is to guess such a product and then test whether the corresponding vertices form a  $k$ -clique.

We present an overview of our construction in Algorithm 1. To simplify the presentation, we present VASS also as counter programs, inspired by Esparza's presentation of Lipton's lower bound [26, Section 7]. The program is non-deterministic and contains zero tests. The zero tests are not immediately obvious in Algorithm 1 because they are used in the subprocedures (including **Multiply**). Note that counter  $y$  is used only by subprocedures. Also note that the variable  $i$  in the first loop and variables  $i$  and  $j$  in the second loop are just syntactic sugar for copying similar code multiple times. The variable  $j$  in the first loop and the variable  $e$  in the second loop allow us to neatly represent non-determinism in the VASS corresponding to the program.

At the start of the program presented in Algorithm 1, both counters  $x$  and  $y$  are initialised to 0 (as specified in the statement of this lemma) and we are interested in the existence of a coverability run that simply reaches the final control state (with any counter values). One should think that coverability holds if and only if there is a run through the program that does not execute an instruction that would take a counter below zero (colloquially, without "getting stuck"). It turns out that a run could only get stuck in the **Edge**[ $e$ ] subprocedure, which will be explained later.

Algorithm 1 uses the **Multiply**[ $x, p$ ] and **Edge**[ $e$ ] subprocedures. These two subprocedures will be implemented later. The subprocedure **Multiply**[ $x, p$ ] takes a counter  $x$  as input as we later reuse this subprocedure when there is more than one counter subject to multiplication. The intended behaviour of **Multiply**[ $x, p$ ] is that it can be performed if and only if as a result we get  $x = x \cdot p$  from an initial value of  $x = x$  (despite the fact that VASS can only additively increase and decrease counters). The subprocedure **Edge**[ $e$ ] can be performed if and only if both vertices of the edge  $e$  are encoded in the value of counter  $x$ . Overall, Algorithm 1 is designed so that in the first part  $x$  is multiplied by  $p_{i,j}$ , where for every  $i$ , one  $j$  is guessed. This equates to selecting one vertex from each  $V_i$ . Then the second part the algorithm checks whether between every pair of selected vertices from  $V_i$  and  $V_j$  there is an edge. Clearly, there is a run through the program that does not get stuck if and only if there is  $k$ -clique in  $G$ .

In Figure 2 we present a VASS with zero tests implementing Algorithm 1. The construction will guarantee that  $q_F(0)$  can be covered from  $q_I(0)$  if and only if there is a  $k$ -clique in  $G$ .

It remains to define the subprocedures. One should think that every call of a subprocedure corresponds to a unique part of the VASS, like a gadget of sorts. To enter and leave the subprocedure one needs to add trivial transitions that do not change the counter values. All subprocedures rely on the invariant  $y = 0$  at the beginning and admit the invariant at the end.

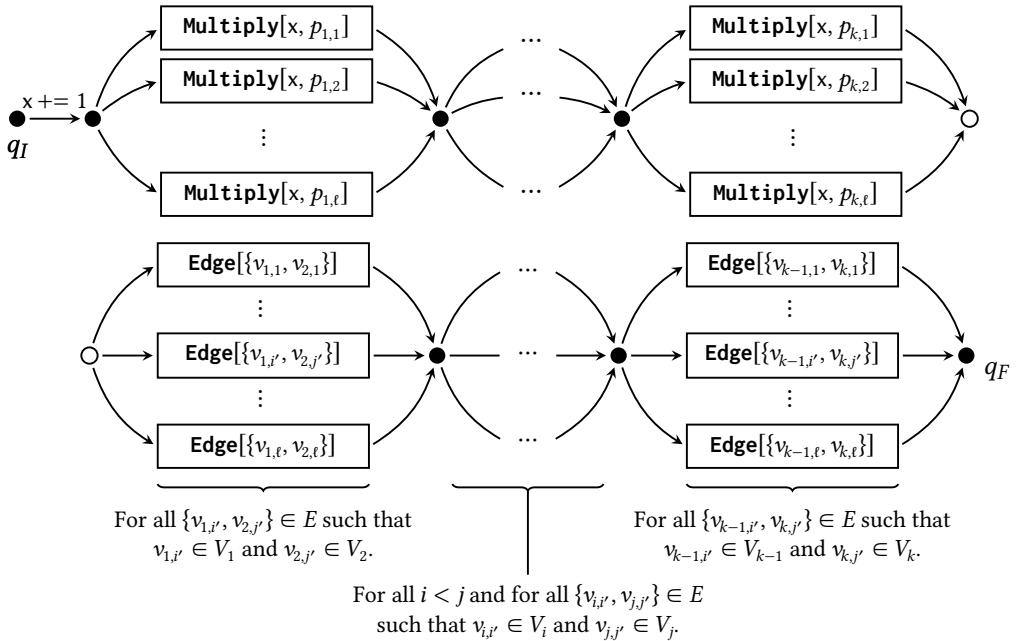


Fig. 2. The top part of the VASS implements the first line and the first loop in Algorithm 1; counter  $x$  is multiplied by  $k$  non-deterministically chosen primes  $p_{i,j}$  each corresponding to a vertex in  $V_i$ . The bottom part of the VASS implements the second loop in Algorithm 1; for every pair  $i \neq j$  the VASS non-deterministically chooses  $\{v_{i,i'}, v_{j,j'}\} \in E$  such that  $v_{i,i'} \in V_i$  and  $v_{j,j'} \in V_j$  and invokes the subprocedure  $\text{Edge}[e]$ .

We start with  $\text{Multiply}[x, p]$  and  $\text{Divide}[x, p]$ , which indeed multiply and divide  $x$  by  $p$ , respectively. See Algorithm 2 for the counter program and VASS implementations. Note that **loop** statements correspond to self-loop transitions in VASS. These loops can be taken any non-negative number of times so long as the counters remain non-negative; however, the subsequent zero tests implicitly enforce these loops to be iterated exhaustively. In the  $\text{Multiply}[x, p]$  gadget, it is easy to see that a run passes through the procedure if and only if  $x$  is multiplied by  $p$ . Similarly, in the  $\text{Divide}[x, p]$  gadget, it is easy to see that a run passes through the procedure if and only if  $x$  is divided by  $p$  wholly. Indeed, the division procedure would get stuck if  $p \nmid x$  because it will be impossible to exit the first loop.

The procedure  $\text{Edge}[v_{i,i'}, v_{j,j'}]$  is simply a sequence of four subprocedures, as shown in Algorithm 3. Indeed, to check if the vertices  $v_{i,i'}$  and  $v_{j,j'}$  are encoded in  $x$  we simply check whether  $x$  is divisible by the corresponding primes  $p_{i,i'}$  and  $p_{j,j'}$ .

Afterwards we multiply  $x$  with the same primes so that the value does not change and it is ready for future edge checks.

It remains to analyse the size of the VASS and its construction time in this reduction. In every run from  $q_I(0)$  to  $q_F(t)$ , for some  $t \geq 0$ , the greatest counter value observable can be bounded above by  $p^k$  where  $p$  is the  $r$ th prime. By the Prime Number Theorem (for example, see [61]), we know that  $p^k = \mathcal{O}((r \log(r))^k) = \mathcal{O}(r^{2k})$  is an upper bound on the counter values observed. Hence  $\mathcal{T}$  is an  $\mathcal{O}(r^{2k})$ -bounded unary 2-VASS.

Now, we count the number of zero tests performed in each run from  $q_I(0)$  to  $q_F(t)$ , for some  $t \geq 0$ . The only zero tests occur in the instances of the **Multiply** and **Divide** subprocedures, each performing two zero tests. In the first part of  $\mathcal{T}$ , a run will encounter  $k$  many **Multiply**

---

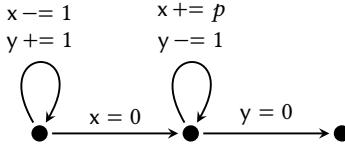
**ALGORITHM 2:** The counter program of **Multiply**[ $x, p$ ] above its VASS implementation (left) and the counter program of **Divide**[ $x, p$ ] above its VASS implementation (right).

---

```

input : $x = x, y = 0$ 
output: $x = x \cdot p, y = 0$ 
loop  $x -= 1, y += 1$ 
zero-test[ $x$ ]
loop  $x += p, y -= 1$ 
zero-test[ $y$ ]

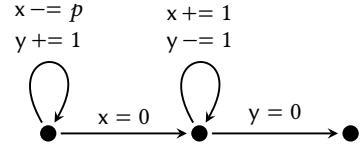
```



```

input : $x = x \cdot p, y = 0$ 
output: $x = x, y = 0$ 
loop  $x -= p, y += 1$ 
zero-test[ $x$ ]
loop  $x += 1, y -= 1$ 
zero-test[ $y$ ]

```




---

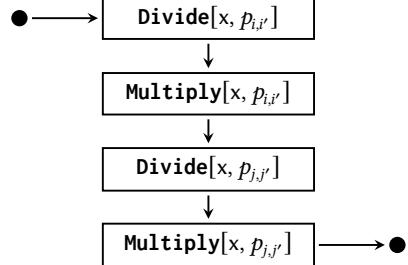
**ALGORITHM 3:** The counter program for **Edge**[ $\{v_{i,i'}, v_{j,j'}\}$ ] and its VASS implementation.

---

```

input : $x = x, y = 0$ 
output: $x = x, y = 0$ 
Divide[ $x, p_{i,i'}$ ]
Multiply[ $x, p_{i,i'}$ ]
Divide[ $x, p_{j,j'}$ ]
Multiply[ $x, p_{j,j'}$ ]

```



subprocedures, contributing  $2k$  many zero tests. In the second part of  $\mathcal{T}$ , a run will encounter  $\binom{k}{2}$  many **Edge** subprocedures, each containing two **Multiply** subprocedures and two **Divide** subprocedures, in total contributing  $8\binom{k}{2}$  many zero tests. Together, every run encounters exactly  $2k + 8\binom{k}{2} = 2k(2k-1)$  many zero tests. Hence  $\mathcal{T}$  is an  $\mathcal{O}(r^{2k})$ -bounded unary 2-VASS with  $2k(2k-1)$  zero tests.

Finally, the **Multiply** and **Divide** subprocedures contain three states and five transitions. Since the  $r$ th prime is bounded above by  $\mathcal{O}(r \log(r))$ , we can unfold the updates in these procedures into an  $\mathcal{O}(r \log(r))$ -length sequence of unary ( $-1, 0$ , or  $+1$ ) updates. Analysing Algorithm 1, it is easy to see that overall the number of states is polynomial in  $r$ . Finally, the first  $r$  primes can be found in  $r^{1+o(1)}$  time [2]. Therefore, in total  $\mathcal{T}$  has size  $\|\mathcal{T}\| = \text{poly}(r+k)$  and can be constructed in  $\text{poly}(r+k)$  time.  $\square$

To attain conditional lower bounds for coverability we must replace the zero tests. We make use of a technique of Rosier and Yen [54] that relies on the construction of Lipton [44]. They show that a  $2^{2^d}$ -bounded counter machine with finite state control can be simulated by a unary  $\mathcal{O}(d)$ -VASS. As Rosier and Yen detail after their proof, it is possible to apply this technique to multiple counters with zero tests at once [54, Page 127]. This accordingly results in the number of VASS counters increasing, but we instantiate this with just two counters. We remark that the VASS constructed in

Lemma 4.3 is structurally bounded, so for any initial configuration there is a limit on the largest observable counter value, as is the case in the VASS that Lipton constructed [44].

**LEMMA 4.4 (COROLLARY OF [54, LEMMA 4.3]).** *For parameters  $a$  and  $d$ , let  $\mathcal{T}$  be an  $m$ -state unary  $(a^{2^d}, 2)$ -VASS with zero tests and two designated states  $q_I, q_F$ . Then there exists a  $\text{poly}(a \cdot m)$ -state  $(4d)$ -VASS  $\mathcal{V}$  with two designated states  $q'_I, q'_F$  such that there is a run from  $q_I(0)$  to  $q'_F(v)$ , for some  $v \geq 0$ , in  $\mathcal{T}$  if and only if there is a run from  $q'_I(0)$  to  $q'_F(w)$ , for some  $w \geq 0$ , in  $\mathcal{V}$ . Moreover,  $\mathcal{V}$  has size  $\text{poly}(a \cdot \|\mathcal{T}\|)$  and can be constructed in the same time.*

**PROOF.** This essentially follows from Rosier and Yen's original construction [54], hence we briefly describe how to ensure that the dimension of the resulting VASS is not too large. Here, we will follow Esparza's detailed analysis [26]. In particular on [26, Page 411], all counters are listed with their properties (this also implies the dimension of the output VASS). Altogether, their construction consists of the following counters with the following properties:

- $x, y, s$  and  $\bar{x}, \bar{y}, \bar{s}$ ; after initialisation, the invariants  $x + \bar{x} = 2^{2^d}$ ,  $y + \bar{y} = 2^{2^d}$ , and  $s + \bar{s} = 2^{2^d}$  are satisfied and the values of  $x, y, s$  are bounded by  $2^{2^d}$ , and
- $y_i, z_i$  and  $\bar{y}_i, \bar{z}_i$  for each  $0 \leq i \leq d - 1$ ; after initialisation, for all  $0 \leq i \leq d - 1$ , the invariants  $y_i + \bar{y}_i = 2^{2^i}$  and  $z_i + \bar{z}_i = 2^{2^i}$  are satisfied and the values of  $y_i, z_i$  are bounded by  $2^{2^i}$ .

There are two nuances with the construction. First, the counters  $x, y$  are bounded by  $2^{2^d}$  and the counters  $y_i, z_i$  are bounded by  $2^{2^i}$ ; we require the bound  $a^{2^d}$ . This is straightforward to fix. Initially,  $y_0$  and  $z_0$  are set to 0 and  $\bar{y}_0, \bar{z}_0$  are set to 2. Hence in our construction, it suffices to change the aforementioned 2 to  $a$ , which using unary updates requires  $a$  additional states and transitions. The sequence of initialisations guarantees that  $y_i + \bar{y}_i$  and  $z_i + \bar{z}_i$  are bounded by  $(y_{i-1} + \bar{y}_{i-1})^2 = (z_{i-1} + \bar{z}_{i-1})^2 = (a^{2^{i-1}})^2 = a^{2^i}$ . Then, by construction, we will also get  $x + \bar{x} = y + \bar{y} = s + \bar{s} = a^{2^d}$ , this also provides an  $a^{2^d}$  bound on  $x$  and  $y$ .

The second issue, is that the number of counters is  $4d + 6$ ; we require  $4d$ . However, by construction the counters  $y_i, z_i$  and  $\bar{y}_i, \bar{z}_i$  are always bounded by  $a^{2^i}$ . Thus, we can do away with the six counters  $y_0, z_0, \bar{y}_0, \bar{z}_0, y_1$ , and  $z_1$  by encoding their value in the states. This only multiplies the number of states, and therefore the size of the VASS, by a polynomial in  $a$ .  $\square$

With this, we can conclude this section with the proof of its main theorem.

**PROOF OF THEOREM 4.2.** Let  $k = \frac{1}{2} \cdot 2^{d/4}$  and let  $G$  be an arbitrary  $k$ -partite graph with  $r$  nodes. By Lemma 4.3, there exists a unary  $(\mathcal{O}(r^{2k}), 2)$ -VASS with zero tests  $\mathcal{T}$  such that  $G$  contains a  $k$ -clique if and only if there is a run from  $q_I(0)$  to  $q_F(t)$ , for some  $t \geq 0$ , in  $\mathcal{T}$ .

Given the bound on the counters in  $\mathcal{T}$  is  $\mathcal{O}(r^{2k}) = \mathcal{O}(r^{2^{d/4}}) = (\mathcal{O}(r))^{2^{d/4}}$ , and the size of  $\mathcal{T}$  is  $\|\mathcal{T}\| = \text{poly}(r + k)$ , we can apply Lemma 4.4 to  $\mathcal{T}$ . There exists a unary  $d$ -VASS  $\mathcal{V}$  such that  $G$  contains a  $k$ -clique if and only if there is a run from  $q'_I(0)$  to  $q'_F(w)$ , for some  $w \geq 0$ , in  $\mathcal{V}$ . The size of  $\mathcal{V}$  is  $\|\mathcal{V}\| = \text{poly}(r + k)$ , and since  $k \leq r$ , we have  $\|\mathcal{V}\| = n \leq r^c$  for some constant  $c$ .

Assume, for the sake of contradiction, that an  $n^{2^{o(d)}}$ -time algorithm for coverability in unary  $d$ -VASS exists. By the above reduction, it would decide any given  $k$ -clique instance in time  $n^{2^{o(d)}} = n^{o(2^{d/4})} = n^{o(k)} = r^{o(k)}$ , where we used  $k = O(2^{d/4})$  in the second equation and  $n = \text{poly}(r)$  in the last equation. By Theorem 4.1, such an  $r^{o(k)}$ -time algorithm for  $k$ -clique would refute ETH.  $\square$

**Remark 4.5.** The reduction used to prove Theorem 4.2 actually excludes  $f(d) \cdot n^{o(2^{d/4})}$ -time algorithms for coverability for any computable function  $f$ , assuming ETH. Recall, from Corollary 3.4,

Table 1. Conditional Lower Bounds and Upper Bounds of the Time Complexity of Coverability and Reachability in Unary  $(\mathcal{O}(n), d)$ -VASS

$d$	Lower Bound	Upper Bound
0	$\Omega(n)$ (trivial)	$\mathcal{O}(n)$
1	$n^{2-o(1)}$ (Theorem 5.4)	$\mathcal{O}(n^2)$
2	$n^{2-o(1)}$ (from above)	$\mathcal{O}(n^3)$
3	$n^{2-o(1)}$ (from above)	$\mathcal{O}(n^4)$
$d \geq 4$	$n^{d-2-o(1)}$ (Theorem 5.8)	$\mathcal{O}(n^{d+1})$

For clarity, we remark that Theorem 5.4 is subject to Hypothesis 5.2 and that Theorem 5.8 is subject to Hypothesis 5.7. The  $o(1)$  terms in the exponents of the lower bounds are sub-constant terms that may only depend on  $n$ ; the lower bounds are more precisely formulated in Theorem 5.4 and Theorem 5.8. Note that the lower bounds for dimensions  $d = 2$  and  $d = 3$  follow from Theorem 5.4 by just adding components consisting of only zeros. All upper bounds follow from Observation 5.1.

that coverability can be decided in  $n^{\mathcal{O}(d^2 \cdot 2^d)}$  time. Note that our conditional lower bound only differs from the (unconditional) upper bound by a constant multiplicative factor of 4 in the second exponent; indeed,  $\mathcal{O}(d^2 \cdot 2^d) = \mathcal{O}(2^{(1+\varepsilon) \cdot d})$  for any positive constant  $\varepsilon > 0$ .

## 5 Coverability and Reachability in Bounded Unary VASS

In this section, we give even tighter bounds for coverability in *bounded* fixed dimension unary VASS. Specifically, for a non-decreasing function  $B : \mathbb{N} \rightarrow \mathbb{N}$ , the coverability problem in  $(B(n), d)$ -VASS asks, for a given  $(B(n), d)$ -VASS  $\mathcal{V} = (Q, T)$  of size  $n$  as well as configurations  $p(s), q(t)$ , whether there is a run in  $\mathcal{V}$  from  $p(s)$  to  $q(t')$  for some  $t' \geq t$  such that each counter value remains in  $\{0, \dots, B(n)\}$  throughout. We would like to clarify the fact that the bound is not part of the input to the problem. We focus on the natural setting of linearly-bounded fixed dimension VASS, that is,  $(\mathcal{O}(n), d)$ -VASS. There is a simple algorithm, presented in the proof of Observation 5.1, that yields an immediate  $\mathcal{O}(n^{d+1})$  upper bound for the time needed to decide the coverability problem. We accompany this observation with closely matching lower bounds, see Table 1 for an overview.

**OBSERVATION 5.1.** *Coverability in a unary  $(B(n), d)$ -VASS of size  $n$  can be decided in  $\mathcal{O}(n(B(n)+1)^d)$  time.*

**PROOF.** Since all configurations in a  $(B(n), d)$ -VASS belong to the finite set  $Q \times \{0, \dots, B(n)\}^d$ , we can exhaustively explore all configurations reachable from  $p(s)$  using a straightforward depth-first search. Each state  $q \in Q$  and each transition  $t \in T$  will be considered at most once for each admissible vector in  $\{0, \dots, B(n)\}^d$ . In total, the algorithm takes at most  $\mathcal{O}(n(B(n)+1)^d)$  time since  $|Q|, |T| \leq n$ . We accept the instance if and only if we have ever witnessed a configuration  $q(t')$  for some  $t' \geq t$ .  $\square$

## Lower Bounds for Coverability in Linearly-Bounded VASS

Now, we consider lower bounds for the coverability problem in linearly-bounded fixed dimension unary VASS. Firstly, in dimension one, we show that quadratic running time is conditionally optimal under the  $k$ -cycle hypothesis. Secondly, in any fixed dimension  $d \geq 4$ , under the 3-uniform hyperclique hypothesis, we show that a running time of  $\Omega(n^{d-2-\varepsilon})$  is required for every  $\varepsilon > 0$ . Together, this provides evidence that the simple  $\mathcal{O}(n^{d+1})$ -time algorithm for coverability in  $(\mathcal{O}(n), d)$ -VASS is close to optimal, as summarised in Table 1.

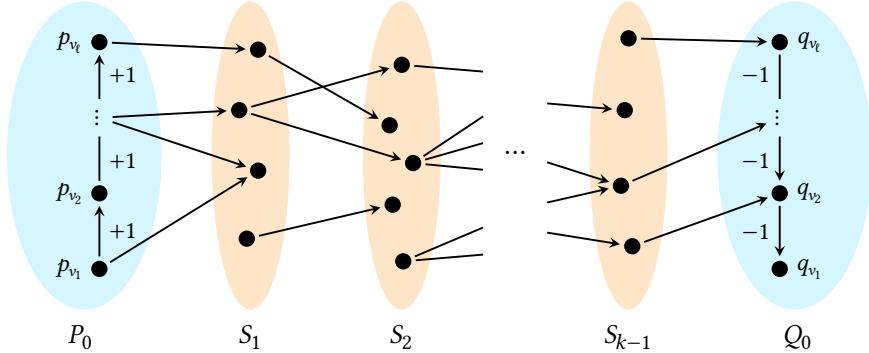


Fig. 3. The  $(\mathcal{O}(n), 1)$ -VASS  $\mathcal{V}$  of size  $n = \mathcal{O}(m)$  for detecting a  $k$ -cycle in a  $k$ -circle-layered graph with  $m$  edges. The transitions that are unlabelled have zero effect. Observe that the structure of the graph is mostly copied into the states and transitions of the linearly-bounded 1-VASS. Importantly, two copies of  $V_0$  are created ( $P_0$  and  $Q_0$ ). Consider a run with initial configuration  $p_{v_1}(0)$ . First, in  $P_0$ , a vertex from  $V_0$  belonging to a  $k$ -cycle can be selected by adding a value corresponding to that vertex to the counter. Then the configuration  $q_{v_1}(0)$  can be reached if the state first observed in  $Q_0$  corresponds to the vertex originally selected in  $P_0$ . Accordingly, there is a run from  $p_{v_1}(0)$  to  $q_{v_1}(0)$  if and only if there exists a  $k$ -cycle, since the states visited in the underlying path of the run correspond to the vertices of the  $k$ -cycle.

**HYPOTHESIS 5.2 ( $k$ -CYCLE HYPOTHESIS).** *For every  $\varepsilon > 0$  and  $m \in \mathbb{N}$ , there exists an integer  $k$  such that there does not exist an  $\mathcal{O}(m^{2-\varepsilon})$ -time algorithm for detecting whether there is a  $k$ -cycle in a directed graph with  $m$  edges.*

The  $k$ -cycle hypothesis arises from the challenge of improving upon the state-of-the-art  $\mathcal{O}(m^{2-\frac{c}{k}})$ -time algorithms for the  $k$ -cycle problem [4, 24, 60]; here  $c$  is some constant. It has been previously used as an assumption for hardness results, for example, see [5, 23, 43]. A standard observation, due to colour-coding arguments, is that we may without loss of generality assume that the given directed graph is  $k$ -circle-layered [43, Lemma 2.2]. Specifically, we can assume that the input graph  $G = (V, E)$  has vertex partition  $V = V_0 \cup \dots \cup V_{k-1}$  such that each edge  $(u, v) \in E$  is in  $V_i \times V_{i+1} \pmod k$  for some  $0 \leq i < k$ . We may also assume that  $|V| \leq |E|$ .

**LEMMA 5.3.** *Given a  $k$ -circle-layered graph  $G = (V_0 \cup \dots \cup V_{k-1}, E)$  with  $m$  edges, there exists a unary  $(\mathcal{O}(n), 1)$ -VASS  $\mathcal{V}$  such that there is a  $k$ -cycle in  $G$  if and only if there exists a run from  $p(0)$  to  $q(0)$  in  $\mathcal{V}$ . Moreover,  $\mathcal{V}$  has size  $n = \mathcal{O}(m)$  and can be constructed in  $\mathcal{O}(m)$  time.*

**PROOF.** Consider the unary  $(\mathcal{O}(m), 1)$ -VASS  $\mathcal{V} = (Q, T)$ , that is, defined as follows (see also Figure 3). For ease of construction let us number the vertices in  $V_0$ , so suppose that  $V_0 = \{v_1, \dots, v_t\}$ . Let us first define the set of states  $Q$ . There are two copies of the vertex subset  $V_0$ , namely  $P_0 = \{p_{v_1}, \dots, p_{v_t}\}$  and  $Q_0 = \{q_{v_1}, \dots, q_{v_t}\}$ . There are also copies of each of the vertex subsets  $V_1, V_2, \dots, V_{k-1}$ , namely  $S_i = \{s_v : v \in V_i\}$  for each  $1 \leq i \leq k-1$ .

$$Q = P_0 \cup S_1 \cup S_2 \cup \dots \cup S_{k-1} \cup Q_0.$$

Now, we define the set of transitions  $T$ . There are three kinds of transitions, the initial *vertex selection* transitions  $T_I$ , the intermediate transitions  $T_E$ , and the final *vertex checking* transitions  $T_F$ .

$$T = T_I \cup T_E \cup T_F.$$

The initial transitions connect states in  $P_0$  sequentially. Each transition increments the counter. Intuitively speaking, the counter takes a value corresponding to the vertex in  $V_0$  that will belong to the  $k$ -cycle in  $G$ .

$$T_I = \{(p_{v_i}, 1, p_{v_{i+1}}) : 1 \leq i < \ell\}.$$

The intermediate transitions are directed copies of the edges in the original graph. The only difference is that edges between  $V_0$  and  $V_1$  are now transitions from  $P_0$  to  $S_1$  and edges between  $V_{k-1}$  and  $V_0$  become transitions from  $S_{k-1}$  to  $Q_0$ .

$$T_E = \{(p_u, 0, s_v) : (u, v) \in (V_0 \times V_1) \cap E\} \cup \{(s_u, 0, q_v) : (u, v) \in (V_{k-1} \times V_0) \cap E\} \cup \{(s_u, 0, s_v) : (u, v) \in (V_i \times V_{i+1}) \cap E \text{ for some } 1 \leq i < k-1\}.$$

The final transitions connect the states in  $Q_0$  sequentially. Each such transition decrements the counter. Intuitively speaking, if the state reached in  $Q_0$  matches the counter that has a value corresponding to the vertex in  $V_0$  then the final state  $q_{v_1}$  can be reached with counter value zero.

$$T_F = \{(q_{v_{i+1}}, -1, q_{v_i}) : 1 \leq i < \ell\}.$$

Importantly, there is a run from the initial configuration  $p_{v_1}(0)$  to the target configuration  $q_{v_1}(0)$  in  $\mathcal{V}$  if and only if there is a  $k$ -cycle in the  $k$ -circle-layered graph  $G$ . In closing, observe that  $|\mathcal{Q}| \leq 2|V|$  and  $|\mathcal{T}| \leq 2|V| + |E|$ . Therefore,  $\mathcal{V}$  has size  $\mathcal{O}(m)$ . We remark that the greatest possible counter value is trivially bounded above by  $|\mathcal{Q}|$ , hence  $\mathcal{V}$  is a unary  $(\mathcal{O}(m), 1)$ -VASS of size  $\mathcal{O}(m)$  that can be constructed in  $\mathcal{O}(m)$  time.  $\square$

**THEOREM 5.4.** *Assuming the  $k$ -cycle hypothesis, there does not exist an  $\varepsilon > 0$  and an  $\mathcal{O}(n^{2-\varepsilon})$ -time algorithm that decides reachability (or coverability) in unary  $(\mathcal{O}(n), 1)$ -VASS of size  $n$ .*

**PROOF.** Assume, for the sake of contradiction, that reachability in a unary  $(\mathcal{O}(n), 1)$ -VASS of size  $n$  can be solved in  $\mathcal{O}(n^{2-\varepsilon})$  time for some  $\varepsilon > 0$ . By the  $k$ -cycle hypothesis (Hypothesis 5.2), there exists a  $k$  such that the problem of detecting a  $k$ -cycle in a  $k$ -circle layered graph with  $m$  vertices cannot be solved in  $\mathcal{O}(m^{2-\varepsilon})$  time. Via the reduction presented above in Lemma 5.3, we create a  $(\mathcal{O}(n), 1)$ -VASS  $\mathcal{V}$  of size  $n = \mathcal{O}(m)$  together with an initial configuration  $p(0)$  and a target configuration  $q(0)$ , such that deciding reachability from  $p(0)$  to  $q(0)$  in  $\mathcal{V}$  determines the existence of a  $k$ -cycle in  $G$ . Thus the  $\mathcal{O}(n^{2-\varepsilon})$ -time algorithm for reachability would yield an  $\mathcal{O}(m^{2-\varepsilon})$ -time algorithm for detecting  $k$ -cycles, contradicting the  $k$ -cycle hypothesis.

By the equivalence of coverability and reachability in unary  $(\mathcal{O}(n), 1)$ -VASS in Lemma 5.6, the same lower bound holds for coverability.  $\square$

We can now use Theorem 5.4 to obtain a conditional lower bound on the time required to decide coverability in unary 2-VASS.

**COROLLARY 5.5.** *Assuming the  $k$ -cycle hypothesis, there does not exist an  $\varepsilon > 0$  and an  $\mathcal{O}(n^{2-\varepsilon})$ -time algorithm that decides coverability in unary 2-VASS of size  $n$ .*

**PROOF.** Consider a standard modification of the reduction presented for Lemma 5.3, that is, to increase the dimension of  $\mathcal{V} = (Q, T)$  by one by adding an opposite counter of sorts, yielding a 2-VASS  $\mathcal{W} = (Q, T')$ . For every transition  $(p, x, q) \in T$ , create a transition also modifying the opposite counter,  $(p, (x, -x), q) \in T'$ . Now, the instance  $(\mathcal{W}, p(0, n), q(0, n))$  of coverability holds if and only if the instance  $(\mathcal{V}, p(0), q(0))$  of reachability holds. The rest follows by Theorem 5.4.  $\square$

## Equivalence of Coverability and Reachability in Linearly-Bounded VASS

Reachability in  $(\mathcal{O}(n), d)$ -VASS can be decided in  $\mathcal{O}(n(B(n) + 1)^d)$  time using the simple algorithm for Observation 5.1 with a trivially modified acceptance condition. It turns out that coverability and reachability are equivalent in unary  $(\mathcal{O}(n), d)$ -VASS. This is true in the sense that it may hold that, for example, coverability in a  $(100n, d)$ -VASS may be reduced in linear time to reachability in a  $(3n, d)$ -VASS. Conversely, reachability in some linearly-bounded  $d$ -VASS can be reduced in linear time to a corresponding instance of coverability in a linearly-bounded  $d$ -VASS. Note that perversely, it appears plausible that instances of coverability in a  $(100n, d)$ -VASS could in fact be simpler to solve than in a  $(3n, d)$ -VASS.

**LEMMA 5.6.** *For a  $(B(n), d)$ -VASS, let  $C^{B(n)}(n)$  and  $R^{B(n)}(n)$  denote the optimal running times for coverability and reachability, respectively. For any  $\gamma > 0$ , there exists some  $\delta > 0$  such that  $C^{\gamma n}(n) \leq \mathcal{O}(R^{\delta n}(n))$ . Conversely, for any  $\gamma > 0$ , there exists some  $\delta > 0$  such that  $R^{\gamma n}(n) \leq \mathcal{O}(C^{\delta n}(n))$ .*

**PROOF.** Given an instance  $(\mathcal{V}, p(s), q(t))$ , of size  $n$ , of coverability in  $(B(n), d)$ -VASS  $\mathcal{V}$ . We construct a  $(B(n), d)$ -VASS  $\mathcal{V}'$  from  $\mathcal{V}$  by adding transitions  $(q, -e_i, q)$  for every  $1 \leq i \leq d$ . It is easy to see that there exists a run from  $p(s)$  to  $q(t)$  in  $\mathcal{V}'$  if and only if there exists a run from  $p(s)$  to  $q(t')$  in  $\mathcal{V}$  for some  $t' \geq t$ . Since  $\|\mathcal{V}'\| = \mathcal{O}(n)$ , for  $B(n) = \gamma \cdot n$  we can ensure (by possibly adding any number of unreachable states) that  $B(n) = \delta \cdot \|\mathcal{V}'\|$  for an appropriately selected  $\delta$ . Thus,  $C^{\gamma n}(n) = \mathcal{O}(R^{\delta n}(\mathcal{O}(n))) = \mathcal{O}(R^{\delta n}(n))$ .

Conversely, consider an instance  $(\mathcal{V}, p(s), q(t))$ , of size  $n$ , of reachability in a  $(B(n), d)$ -VASS  $\mathcal{V}$ , again denote  $n = \|\mathcal{V}\|$ . We construct the  $(B(n), d)$ -VASS  $\mathcal{V}'$  from  $\mathcal{V}$  by adding a path from  $q$  to a new state  $r$  whose transitions update the counters by  $-t$ . This is easily implementable by a path of length at most  $B(n)$ , for if  $\|t\| > B(n)$  this instance is trivially false. We then append a path from  $r$  to a new state  $r'$  whose transitions add  $B(n)$  to every counter. It is easy to see that there is a run from  $p(s)$  to  $r'((B(n), \dots, B(n)))$  in  $\mathcal{V}'$  if and only if there exists a run from  $p(s)$  to  $q(t)$  in  $\mathcal{V}$ . Since  $\|\mathcal{V}'\| = \mathcal{O}(n + B(n))$ , for  $B(n) = \gamma \cdot n$  we can ensure that  $B(\|\mathcal{V}'\|) = \delta \cdot \|\mathcal{V}'\|$  for some  $\delta$ . Thus,  $R^{\gamma n}(n) = \mathcal{O}(C^{\delta n}(\mathcal{O}(n))) = \mathcal{O}(C^{\delta n}(n))$ .  $\square$

## Lower Bounds for Reachability in Linearly-Bounded VASS

To obtain further lower bounds for the coverability problem in  $(\mathcal{O}(n), d)$ -VASS, by Lemma 5.6, we can, equivalently, find lower bounds for the reachability problem in  $(\mathcal{O}(n), d)$ -VASS. In Theorem 5.8, we will assume a well-established hypothesis concerning the time required to detect a hyperclique in a 3-uniform hypergraph. In fact, Lincoln, Vassilevska Williams, and Williams state and justify an even stronger hypothesis about  $\mu$ -uniform hypergraphs for every  $\mu \geq 3$  [43, Hypothesis 1.4]. We will use this computational complexity hypothesis to expose precise lower bounds on the time complexity of reachability in linearly-bounded fixed dimension unary VASS.

**HYPOTHESIS 5.7 (K-HYPERCLIQUE HYPOTHESIS [43, HYPOTHESIS 1.4]).** *Let  $k \geq 3$  be a fixed integer. There does not exist an  $\varepsilon > 0$  and an  $\mathcal{O}(r^{k-\varepsilon})$ -time algorithm for detecting whether there is a  $k$ -hyperclique in a 3-uniform hypergraph with  $r$  vertices.*

For the remainder of this section, we focus on the proof of the following theorem.

**THEOREM 5.8.** *Let  $d \geq 1$  be a fixed integer. Assuming Hypothesis 5.7, there does not exist an  $\varepsilon > 0$  and an  $\mathcal{O}(n^{d-\varepsilon})$ -time algorithm that decides reachability in a unary  $(\mathcal{O}(n), d+2)$ -VASS of size  $n$ .*

The lower bound is obtained via reduction from detecting hyperclique in 3-uniform hypergraphs, hence it is subject to the  $k$ -hyperclique hypothesis. We present our reduction in two steps. The first step is an intermediate step, in Lemma 5.9 we offer a reduction to an instance of reachability in

unary VASS with a limited number of zero tests. The second step extends the first, in Lemma 5.10 we modify the reduction by adding a counter so zero tests are absented. This extension leverages the recently developed *controlling counter technique* of Czerwiński and Orlikowski [21]. This technique allows for implicit zero tests to be performed in the presence of a dedicated counter whose transition effects and reachability condition ensure the implicit zero tests were indeed performed correctly.

It has been shown that we may assume that the hypergraph is  $k$ -partite for  $k$ -hyperclique Hypothesis 5.7 [43, Theorem 3.1]. Thus, we may assume that the vertices can be partitioned into  $k$  disjoint subsets  $V = V_1 \cup \dots \cup V_k$  and every hyperedge  $\{u, v, w\}$  comprises of three vertices from distinct subsets  $u \in V_{i_1}$ ,  $v \in V_{i_2}$ , and  $w \in V_{i_3}$  for some  $1 \leq i_1 < i_2 < i_3 \leq k$ .

**LEMMA 5.9.** *Let  $d \geq 1$  be a fixed integer. Given a  $4d$ -partite 3-uniform hypergraph  $H = (V_1 \cup \dots \cup V_{4d}, E)$  with  $r$  vertices, there exists a unary ( $\mathcal{O}((r \log(r))^4)$ ,  $d + 1$ )-VASS with  $\mathcal{O}(d^3)$  zero tests  $\mathcal{T}$  such that there is a  $4d$ -hyperclique in  $H$  if and only if there is a run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{T}$ . Moreover,  $\mathcal{T}$  can be constructed in  $\text{poly}(d) \cdot (r \log(r))^4$  time.*

**PROOF.** We will re-employ some of the ideas already used in the constructions of the proof of Lemma 4.3. In particular, we will use **Multiply** and **Divide** subprocedures, see Algorithm 2. Let us denote the  $d + 1$  counters  $x_1, \dots, x_d, y$ . The collective role of  $x_1, \dots, x_d$  is to maintain a representation of the  $4d$  vertices forming the  $4d$ -hyperclique. The role of  $y$  is to ensure multiplications and divisions are completed correctly. Just as previously seen, before the execution of **Multiply** or **Divide**, we require  $y = 0$ . We will combine these subprocedures to construct new subprocedures for unary  $(d + 1)$ -VASS with zero tests to verify properties related to 3-uniform hypergraphs.

We start by finding the first  $r$  primes. We associate a distinct prime  $p_v$  to each vertex  $v \in V$ . Now we encode the chosen  $4d$  vertices that will form the  $4d$ -clique by storing, on  $d$  many counters, products of four primes corresponding to four of the selected vertices. Therefore, after the initial guessing part, the value of counter  $x_i$  will be  $p_t \cdot p_u \cdot p_v \cdot p_w$  for some vertices  $t, u, v, w \in V$ . Roughly speaking, we store the product of four primes on one counter so that the maximum observable counter value matches the size of the resulting VASS with zero tests.

*Guessing part.* The VASS presented in Algorithm 4 implements the following algorithm: Guess  $4d$  vertices, not necessarily distinct, and check whether they form a  $4d$ -hyperclique. Note that this algorithm is correct because it does not help us to repeatedly guess the same vertex or repeatedly guess vertices that belong to the same vertex subset. In contrast to Algorithm 1, the main difference is that the guessed vertices are encoded as quadruple products of primes across counters  $x_1, \dots, x_d$ . We do not store the entire product of  $4d$  primes explicitly, only the values of each counter.

*Checking part.* In the second part, we verify that we have selected a  $4d$ -hyperclique by testing for each of the  $\binom{4d}{3}$  hyperedges. This is achieved in essentially the same way as **Edge**[ $e$ ] was implemented in Algorithm 3. We check that between every triplet of vertex subsets there is a hyperedge that has all of the vertices selected in the first part. At the end of the checking part, there are  $d + 1$  self-loops that each decrement one of the  $d + 1$  counters. These loops can be used to reach the target configuration  $q_F(0)$  once all hyperedge checks have been succeeded.

For ease of presentation and as previously mentioned, we introduce the **VertexSelected** subprocedure that checks whether a given vertex has been selected; see Figure 4.

We also implement the **HyperEdge** subprocedure for checking whether the vertices, in a given hyperedge, have been selected; see Figure 5. This subprocedure checks that the three primes corresponding to the three vertices in the hyperedge can divide one of the values stored in  $x_1, \dots, x_d$ .

Now, we use the **HyperEdge** subprocedure to construct the checking part of  $\mathcal{T}$ . This part consists of a sequence of  $\binom{4d}{3}$  non-deterministic branching sections, one for each triplet of vertex subsets  $V_{i_1}, V_{i_2}$ , and  $V_{i_3}$  where  $1 \leq i_1 < i_2 < i_3 \leq 4d$ . In each branching section, there is an instance of the

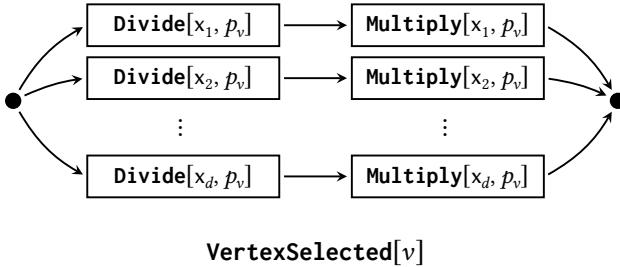


Fig. 4. The **VertexSelected** subprocedure implemented in a unary  $(d+1)$ -VASS with zero tests. To instantiate this subprocedure, a vertex  $v$  is specified so that the  $d$  counters can be checked for divisibility by the prime  $p_v$ , with the effect of checking whether the vertex  $v$  has been selected. The counter  $y$  is used by the **Divide** and **Multiply** subprocedures to ensure the division and multiplications are completed correctly.

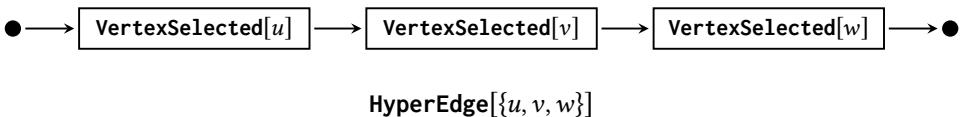


Fig. 5. The **HyperEdge** subprocedure implemented in a unary  $(d+1)$ -VASS with zero tests. Note that the three vertices of the given hyperedge may be stored across any of the  $d$  counters  $x_1, \dots, x_d$ . Therefore, we make use of the **VertexSelected** subprocedure three times to check if indeed  $u, v$ , and  $w$  have been selected.

---

**ALGORITHM 4:** A counter program representing the VASS with zero tests. As seen earlier, the variables in **for** loops are just syntactic sugar for repeating similar lines of code which correspond to states in the VASS. Similarly, the variables in **guess** statements represent non-deterministic branching transitions in a VASS.

---

```

input:  $x_1, \dots, x_d, y = 0$ 
for  $i \leftarrow 1$  to  $d$  do
|    $x_i \leftarrow 1$ 
|   for  $g \leftarrow 1$  to  $4$  do
|   |   guess  $j \in \{1, \dots, r\}$ 
|   |   Multiply[ $x_i, p_j$ ]
|   |   end
|   end
|   for  $(i_1, i_2, i_3) \in \{1, \dots, 4d\}^3, i_1 < i_2 < i_3$  do
|   |   guess  $e \in E \cap \{(u, v, w) : u \in V_{i_1}, v \in V_{i_2}, w \in V_{i_3}\}$ 
|   |   HyperEdge[ $e$ ]
|   end
|   for  $i \leftarrow 1$  to  $d$  do
|   |   loop  $x_i \leftarrow x_i - 1$ 
|   end
|   loop  $y \leftarrow y - 1$ 

```

---

**HyperEdge** subprocedure for each of the hyperedges  $\{u, v, w\} \in E$  such that  $u \in V_{i_1}, v \in V_{i_2}$ , and  $w \in V_{i_3}$ .

Figure 6 implements the check that the selected vertices indeed form a hyperclique: In order to reach the final state  $q_F$ , there must be a hyperedge between each of the  $4d$  vertices selected in the

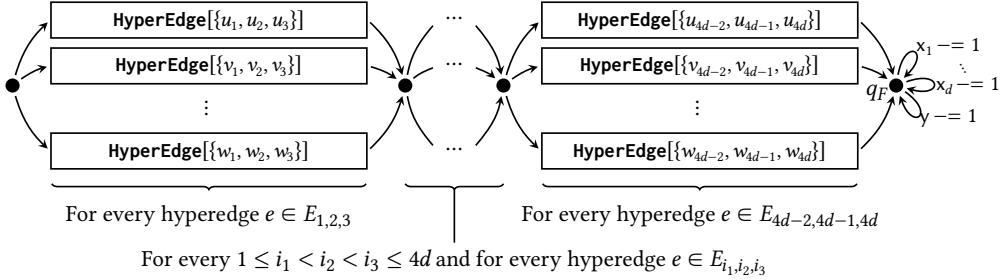


Fig. 6. The check part of the unary  $(d+1)$ -VASS with zero tests  $\mathcal{T}$  for detecting a  $4d$ -hyperclique in  $4d$ -partite hypergraph. The set  $E_{i_1,i_2,i_3}$  is shorthand for the subset of hyperedges containing vertices belonging to  $V_{i_1}$ ,  $V_{i_2}$ , and  $V_{i_3}$ ; precisely  $E_{i_1,i_2,i_3} = \{ \{u, v, w\} \in E : u \in V_{i_1}, v \in V_{i_2}, w \in V_{i_3} \}$ .

first part of  $\mathcal{T}$ . Thus, there is a  $4d$ -hyperclique in  $H$  if and only if there is a run from  $q_I(0)$  to  $q_F(t)$ , for some  $t \geq 0$  in  $\mathcal{T}$ . Given that there are self-loops decrementing each of the counters, there is always a run from  $q_F(t)$  to  $q_F(0)$ . Therefore,  $H$  contains a  $4d$ -hyperclique if and only if there is a run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{T}$ .

We are now able to finalize the proof. We will carefully analyse the maximum counter value observed on any run, count the number of zero tests performed on any run, and lastly evaluate the size of  $\mathcal{T}$ .

The highest counter value observed by each counter  $x_1, \dots, x_d$  is the product of four primes. The highest counter value observed by  $y$  is equal to the highest counter value observed by any of the other counters  $x_1, \dots, x_d$ . Therefore the bound on the highest value observed, altogether can be bounded about by  $p^4$  where  $p$  is the  $r$ th prime. By the Prime Number Theorem (for example, see [61]) we know that  $p \in \mathcal{O}(r \log(r))$ . Therefore, every run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{T}$  is  $\mathcal{O}((r \log(r))^4)$ -bounded.

Zero tests are only performed by the **Multiply** and **Divide** subprocedures, each instance of these subprocedures containing two zero tests. We therefore count the number of calls to these subprocedures on any given run. In the guessing part, there is one call to **Multiply** for each of the  $4d$  vertices selected to form a  $4d$ -hyperclique. In the checking part, there is a sequence of  $\binom{4d}{3}$  many calls to **HyperEdge**. Each **HyperEdge** call contains three invocations of **VertexSelected**, which, in turn, executes **Divide** and **Multiply** once each. In total, there are  $2(4d + 6\binom{4d}{3}) \in \mathcal{O}(d^3)$  many zero tests are performed in any run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{T}$ .

Finally, each instance of the **Multiply** and **Divide** subprocedures has size  $\mathcal{O}(r \log(r))$ . Note that the first  $r$  primes can be found in  $r^{1+o(1)}$  time [2]. In the guessing part, there are  $4d \cdot r$  instances of the **Multiply** subprocedure. In the checking part, there is an instance of the **HyperEdge** subprocedure for each edge in the hypergraph. The **HyperEdge** subprocedures themselves appear in  $\binom{4d}{3}$  many collections, one for each triplet of vertex subsets. Each **HyperEdge** subprocedure contains three instances of the **VertexSelected** subprocedure, which contains  $d$  instances of the **Multiply** subprocedure and  $d$  instances of the **Divide** subprocedure. Therefore, in total  $\mathcal{T}$  has polynomial size and can be constructed in  $\mathcal{O}(d \cdot r^2 \log(r) + m \cdot \binom{4d}{3} \cdot d \cdot r \log(r))$  time, where  $m \in \mathcal{O}(r^3)$  is the total number of hyperedges.  $\square$

Consider our earlier described two-step approach towards proving Theorem 5.8 by first obtaining a unary  $(d+1)$ -VASS  $\mathcal{T}$  with zero tests and then obtaining a unary  $(d+2)$ -VASS  $\mathcal{V}$  by increasing the dimension by one and removing the zero tests. In actuality, both steps occur together to prove Theorem 5.8. Ultimately, the  $d+2$  counters of  $\mathcal{V}$  have the following roles. The counters  $x_1, \dots, x_d$  are used to store the products of primes corresponding to vertices of hyperclique. The counter  $y$  is used

to complete multiplications and divisions. In the remainder of this section, we add the  $(d + 2)$ -nd counter, that is, used to ensure the (implicit) zero tests are performed faithfully. We achieve this by leveraging the *controlling counter technique* that was developed by Czerwiński and Orlikowski and first formalised in [21]. The following is the restatement of their technique, their lemma has been restricted to our scenario and rewritten using the notation of this article.

**LEMMA 5.10 ([21, LEMMA 10]).** *Let  $\rho$  be a run in a  $(d + 2)$ -VASS such that  $q_I(0) \xrightarrow{\rho} q_F(0)$ . Further, let  $q_0(v_0), q_1(v_1), \dots, q_r(v_r)$  be some distinguished configurations observed along the run  $\rho$  with  $q_0(v_0) = q_I(0)$  and  $q_r(v_r) = q_F(0)$  and let  $\rho_j$  be the segment of  $\rho$ , that is, between  $q_{j-1}(v_{j-1})$  and  $q_j(v_j)$ , so  $\rho$  can be described as:*

$$q_I(0) = q_0(v_0) \xrightarrow{\rho_1} q_1(v_1) \rightarrow \dots \rightarrow q_{r-1}(v_{r-1}) \xrightarrow{\rho_r} q_r(v_r) = q_F(0).$$

*Let  $S_1, \dots, S_d, S_{d+1} \subseteq \{0, 1, \dots, r\}$  be the sets of indices of the distinguished configurations where zero tests could be performed on counters  $x_1, \dots, x_d, x_{d+1}$ , respectively. Let  $t_{j,i} = |\{s \geq j : s \in S_i\}|$  be the number of zero test for the counter  $x_i$  in the remainder of the run  $\rho_{j+1} \dots \rho_r$ . Given that  $v_0 = 0$  and  $v_r = 0$ , if*

$$\text{eff}(\rho_j)[d+2] = \sum_{i=1}^{d+1} t_{j,i} \cdot \text{eff}(\rho_j)[i], \quad (1)$$

*then for every  $i \in \{1, \dots, d, d+1\}$  and  $j \in S_i$ , we know that  $v_j[i] = 0$ .*

With Lemma 5.10 in hand, we can ensure that the  $\mathcal{O}(d^3)$  zero tests performed by  $\mathcal{T}$  from Lemma 5.9 are executed correctly. We conclude this section with a proof of Theorem 5.8.

**PROOF OF THEOREM 5.8.** Consider the reduction, presented in Lemma 5.9, from detecting a  $4d$ -hyperclique in a  $4d$ -partite 3-uniform hypergraph  $H$  to reachability in unary  $(\mathcal{O}(r \log(r))^4, d+1)$ -VASS with  $\mathcal{O}(d^3)$  zero tests. Now, given Lemma 5.10, we will add a controlling counter to  $\mathcal{T}$  so that the zero tests on the  $d+1$  counters  $x_1, \dots, x_d, y$  are instead performed implicitly. To this end, we introduce another counter  $z$  that receives updates on transitions, consistent with Equation (1), whenever any of the other counters are updated. Note that counters  $y$  and  $z$ , for the sake of a succinct and consistent description, are, respectively, referred to as counters  $x_{d+1}$  and  $x_{d+2}$  in the statement of Lemma 5.10. Moreover, notice that the maximum value of  $z$  is bounded by  $\text{poly}(d) \cdot (\sum_{i=1}^{d+1} x_i) \in \text{poly}(d) \cdot (r \log(r))^4$ .

Therefore, we have constructed a unary  $(\text{poly}(d) \cdot (r \log(r))^4, d+2)$ -VASS  $\mathcal{V}$  with the property that there  $H$  contains a  $4d$ -hyperclique if and only if there is a run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{V}$ . Such a  $(\text{poly}(d) \cdot (r \log(r))^4, d+2)$ -VASS  $\mathcal{V}$  has size  $\mathcal{O}(t \cdot |\mathcal{T}|)$  where  $t \in \text{poly}(d)$  is the number of zero tests performed on the run from  $q_I(0)$  to  $q_F(0)$  in  $\mathcal{T}$ . Moreover,  $\mathcal{V}$  can be constructed in  $\text{poly}(d) \cdot (r \log(r))^4$  time. Hence, if reachability in unary  $(\mathcal{O}(n), d+2)$ -VASS of size  $n$  can be solved in  $n^{d-\varepsilon}$  time for some  $\varepsilon > 0$ , then one can decide whether there is a  $4d$ -hyperclique in a 3-uniform hypergraph with  $r$  vertices in  $r^{4d-\varepsilon'}$  time for some  $\varepsilon' > 0$ , contradicting Hypothesis 5.7.  $\square$

## 6 Conclusion

### Summary

In this article, we have revisited a classical problem of coverability in  $d$ -VASS. We have closed the gap left by Rosier and Yen [54] on the length of runs witnessing instances of coverability in  $d$ -VASS. We have lowered the upper bound of  $n^{2^{\mathcal{O}(d \log(d))}}$ , from Rackoff's technique [53], to  $n^{\mathcal{O}(d \cdot 2^d)}$  (Theorem 3.3), matching the  $n^{2^{\mathcal{O}(d)}}$  lower bound from Lipton's construction [44]. This accordingly closes the gap on the exact space required for the coverability problem and yields a deterministic

$n^{\mathcal{O}(d^2 \cdot 2^d)}$ -time algorithm for coverability in  $d$ -VASS (Corollary 3.4). We complement this with a matching lower bound conditional on ETH; there does not exist a deterministic  $n^{2^{o(d)}}$ -time algorithm for coverability (Theorem 4.2). As mentioned in Remark 4.5, the difference between the constant multiplicative factors in the second exponent between our upper bound and our conditional lower bound only differ by a factor 4. By and large, this settles the exact space and time complexity of coverability in VASS.

In addition, we study linearly-bounded unary  $d$ -VASS. Here, coverability and reachability are equivalent and the trivial exhaustive search  $\mathcal{O}(n^{d+1})$  algorithm is near-optimal. We prove that reachability in linearly-bounded 1-VASS requires  $n^{2-o(1)}$  time under the  $k$ -cycle hypothesis (Theorem 5.4), matching the trivial upper bound. We further prove that reachability in linearly-bounded  $(d+2)$ -VASS requires  $n^{d-o(1)}$  time under the 3-uniform hyperclique hypothesis (Theorem 5.8).

## Open Problems

The *boundedness problem*, a problem closely related to coverability, asks whether, from a given initial configuration, the set of all reachable configurations is finite. This problem was also studied by Lipton and Rackoff and is EXPSPACE-complete [44, 53]. Boundedness was further analysed by Rosier and Yen [54, Theorem 2.1] and the same gap also exists for the exact space required. We leave the same improvement, to eliminate the same twice-exponentiated  $\log(d)$  factor, as an open problem.

Our lower bounds for the time complexity of coverability and reachability in linearly-bounded unary  $d$ -VASS, for  $d \geq 2$ , leave a gap of up to  $n^{3+o(1)}$ , see Table 1. We leave it as an open problem to either improve upon the upper bound  $\mathcal{O}(n^{d+1})$  given by the trivial algorithm, or to raise our conditional lower bounds.

## Acknowledgments

We thank Łukasz Orlikowski for the time he spent checking and discussing a final draft of this article. We would also like to thank the reviewers for their comments and for their time.

## References

- [1] Parosh Aziz Abdulla, Karlis Cerans, Bengt Jonsson, and Yih-Kuen Tsay. 2000. Algorithmic analysis of programs with well quasi-ordered domains. *Inf. Comput.* 160, 1-2 (2000), 109–127. DOI : <https://doi.org/10.1006/inco.1999.2843>
- [2] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. 2004. PRIMES is in P. *Ann. Math.* 160, 2 (2004), 781–793. DOI : <https://doi.org/10.4007/annals.2004.160.781>
- [3] Shaull Almagor, Nathann Cohen, Guillermo A. Pérez, Mahsa Shirmohammadi, and James Worrell. 2020. Coverability in 1-VASS with disequality tests. In *Proceedings of the 31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference) (LIPIcs)*, Igor Konnov and Laura Kovács (Eds.), Vol. 171. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 38:1–38:20. DOI : <https://doi.org/10.4230/LIPIcs.CONCUR.2020.38>
- [4] Noga Alon, Raphael Yuster, and Uri Zwick. 1997. Finding and counting given length cycles. *Algorithmica* 17, 3 (1997), 209–223. DOI : <https://doi.org/10.1007/BF02523189>
- [5] Bertie Ancona, Monika Henzinger, Liam Roditty, Virginia Vassilevska Williams, and Nicole Wein. 2019. Algorithms and hardness for diameter in dynamic graphs. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece (LIPIcs)*, Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi (Eds.), Vol. 132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 13:1–13:14. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2019.13>
- [6] Michael Benedikt, Timothy Duff, Aditya Sharad, and James Worrell. 2017. Polynomial automata: Zeroness and applications. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavík, Iceland, June 20-23, 2017*. IEEE Computer Society, 1–12. DOI : <https://doi.org/10.1109/LICS.2017.8005101>
- [7] Michael Blondin, Matthias Englert, Alain Finkel, Stefan Göller, Christoph Haase, Ranko Lazic, Pierre McKenzie, and Patrick Totzke. 2021. The reachability problem for two-dimensional vector addition systems with states. *J. ACM* 68, 5 (2021), 34:1–34:43. DOI : <https://doi.org/10.1145/3464794>
- [8] Michael Blondin, Alain Finkel, Christoph Haase, and Serge Haddad. 2016. Approaching the coverability problem continuously. In *Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems - 22nd International*

- Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings (Lecture Notes in Computer Science)*, Marsha Chechik and Jean-François Raskin (Eds.), Vol. 9636. Springer, 480–496. DOI : [https://doi.org/10.1007/978-3-662-49674-9\\_28](https://doi.org/10.1007/978-3-662-49674-9_28)
- [9] Michael Blondin, Christoph Haase, and Philip Offtermatt. 2021. Directed reachability for infinite-state systems. In *Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems - 27th International Conference, TACAS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings, Part II (Lecture Notes in Computer Science)*, Jan Friso Groote and Kim Guldstrand Larsen (Eds.), Vol. 12652. Springer, 3–23. DOI : [https://doi.org/10.1007/978-3-030-72013-1\\_1](https://doi.org/10.1007/978-3-030-72013-1_1)
  - [10] Mikołaj Bojańczyk, Claire David, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. 2011. Two-variable logic on data words. *ACM Trans. Comput. Log.* 12, 4 (2011), 27:1–27:26. DOI : <https://doi.org/10.1145/1970398.1970403>
  - [11] Laura Bozzelli and Pierre Ganty. 2011. Complexity analysis of the backward coverability algorithm for VASS. In *Proceedings of the Reachability Problems - 5th International Workshop, RP 2011, Genoa, Italy, September 28-30, 2011. Proceedings (Lecture Notes in Computer Science)*, Giorgio Delzanno and Igor Potapov (Eds.), Vol. 6945. Springer, 96–109. DOI : [https://doi.org/10.1007/978-3-642-24288-5\\_10](https://doi.org/10.1007/978-3-642-24288-5_10)
  - [12] Karl Bringmann, Allan Grønlund, Marvin Künnemann, and Kasper Green Larsen. 2024. The NFA acceptance hypothesis: Non-combinatorial and dynamic lower bounds. In *Proceedings of the 15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA (LIPIcs)*, Venkatesan Guruswami (Ed.), Vol. 287. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 22:1–22:25. DOI : <https://doi.org/10.4230/LIPIcs.ITCS.2024.22>
  - [13] Krishnendu Chatterjee, Bhavya Choudhary, and Andreas Pavlogiannis. 2017. Optimal dyck reachability for data-dependence and alias analysis. *Proc. ACM Program. Lang.* 2, POPL (2017), 1–30.
  - [14] Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David W. Juedes, Iyad A. Kanj, and Ge Xia. 2005. Tight lower bounds for certain parameterized NP-hard problems. *Inf. Comput.* 201, 2 (2005), 216–231. DOI : <https://doi.org/10.1016/j.ic.2005.05.001>
  - [15] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. 2006. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.* 72, 8 (2006), 1346–1367. DOI : <https://doi.org/10.1016/j.jcss.2006.04.007>
  - [16] Dmitry Chistikov, Wojciech Czerwiński, Filip Mazowiecki, Lukasz Orlowski, Henry Sinclair-Banks, and Karol Węgrzycki. 2024. The tractability border of reachability in simple vector addition systems with states. In *Proceedings of the 65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*. IEEE, 1332–1354. DOI : <https://doi.org/10.1109/FOCS61266.2024.00086>
  - [17] Hubert Comon and Yan Jurski. 1998. Multiple counters automata, safety analysis and presburger arithmetic. In *Proceedings of the Computer Aided Verification, 10th International Conference, CAV '98, Vancouver, BC, Canada, June 28 - July 2, 1998, Proceedings (Lecture Notes in Computer Science)*, Alan J. Hu and Moshe Y. Vardi (Eds.), Vol. 1427. Springer, 268–279. DOI : <https://doi.org/10.1007/BFb0028751>
  - [18] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms*. Springer. DOI : <https://doi.org/10.1007/978-3-319-21275-3>
  - [19] Wojciech Czerwiński, Sławomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. 2020. Reachability in fixed dimension vector addition systems with states. In *Proceedings of the 31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference) (LIPIcs)*, Igor Konnov and Laura Kovács (Eds.), Vol. 171. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 48:1–48:21. DOI : <https://doi.org/10.4230/LIPIcs.CONCUR.2020.48>
  - [20] Wojciech Czerwiński, Sławomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. 2021. The reachability problem for petri nets is not elementary. *J. ACM* 68, 1 (2021), 7:1–7:28. DOI : <https://doi.org/10.1145/3422822>
  - [21] Wojciech Czerwiński and Łukasz Orlowski. 2021. Reachability in vector addition systems is ackermann-complete. In *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 1229–1240. DOI : <https://doi.org/10.1109/FOCS52979.2021.00120>
  - [22] Wojciech Czerwiński and Łukasz Orlowski. 2022. Lower bounds for the reachability problem in fixed dimensional VASSes. In *Proceedings of the LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, Christel Baier and Dana Fisman (Eds.). ACM, 40:1–40:12. DOI : <https://doi.org/10.1145/3531130.3533357>
  - [23] Mina Dalirrooyfard, Ce Jin, Virginia Vassilevska Williams, and Nicole Wein. 2022. Approximation algorithms and hardness for n-pairs shortest paths and all-nodes shortest cycles. In *Proceedings of the 63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*. IEEE, 290–300. DOI : <https://doi.org/10.1109/FOCS54457.2022.00034>
  - [24] Mina Dalirrooyfard, Thuy Duong Vuong, and Virginia Vassilevska Williams. 2021. Graph pattern detection: Hardness for all induced patterns and faster noninduced cycles. *SIAM J. Comput.* 50, 5 (2021), 1627–1662. DOI : <https://doi.org/10.1137/20M1335054>
  - [25] Dani Dorfman, Haim Kaplan, Robert E. Tarjan, and Uri Zwick. 2023. Optimal energetic paths for electric cars. In *Proceedings of the 31st Annual European Symposium on Algorithms, ESA 2023 (LIPIcs)*, Vol. 274. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 42:1–42:17. DOI : <https://doi.org/10.4230/LIPIcs.ESA.2023.42>

- [26] Javier Esparza. 1996. Decidability and complexity of petri net problems - an introduction. In *Proceedings of the Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996 (Lecture Notes in Computer Science)*, Wolfgang Reisig and Grzegorz Rozenberg (Eds.), Vol. 1491. Springer, 374–428. DOI : [https://doi.org/10.1007/3-540-65306-6\\_20](https://doi.org/10.1007/3-540-65306-6_20)
- [27] Javier Esparza, Ruslán Ledesma-Garza, Rupak Majumdar, Philipp J. Meyer, and Filip Niksic. 2014. An SMT-based approach to coverability analysis. In *Proceedings of the Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings (Lecture Notes in Computer Science)*, Armin Biere and Roderick Bloem (Eds.), Vol. 8559. Springer, 603–619. DOI : [https://doi.org/10.1007/978-3-319-08867-9\\_40](https://doi.org/10.1007/978-3-319-08867-9_40)
- [28] John Fearnley and Marcin Jurdziński. 2013. Reachability in two-clock timed automata is PSPACE-complete. In *Proceedings of the Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II (Lecture Notes in Computer Science)*, Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg (Eds.), Vol. 7966. Springer, 212–223. DOI : [https://doi.org/10.1007/978-3-642-39212-2\\_21](https://doi.org/10.1007/978-3-642-39212-2_21)
- [29] Diego Figueira, Santiago Figueira, Sylvain Schmitz, and Philippe Schnoebelen. 2011. Ackermannian and primitive-recursive bounds with dickson's lemma. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011, June 21-24, 2011, Toronto, Ontario, Canada*. IEEE Computer Society, 269–278. DOI : <https://doi.org/10.1109/LICS.2011.39>
- [30] Pierre Ganty and Rupak Majumdar. 2012. Algorithmic verification of asynchronous programs. *ACM Trans. Program. Lang. Syst.* 34, 1 (2012), 6:1–6:48. DOI : <https://doi.org/10.1145/2160910.2160915>
- [31] Steven M. German and A. Prasad Sistla. 1992. Reasoning about systems with many processes. *J. ACM* 39, 3 (1992), 675–735. DOI : <https://doi.org/10.1145/146637.146681>
- [32] Christoph Haase, Stephan Kreutzer, Joël Ouaknine, and James Worrell. 2009. Reachability in succinct and parametric one-counter automata. In *Proceedings of the CONCUR 2009 - Concurrency Theory, 20th International Conference, CONCUR 2009, Bologna, Italy, September 1-4, 2009. Proceedings (Lecture Notes in Computer Science)*, Mario Bravetti and Gianluigi Zavattaro (Eds.), Vol. 5710. Springer, 369–383. DOI : [https://doi.org/10.1007/978-3-642-04081-8\\_25](https://doi.org/10.1007/978-3-642-04081-8_25)
- [33] Christoph Haase, Joël Ouaknine, and James Worrell. 2012. On the relationship between reachability problems in timed and counter automata. In *Proceedings of the Reachability Problems - 6th International Workshop, RP 2012, Bordeaux, France, September 17-19, 2012. Proceedings (Lecture Notes in Computer Science)*, Alain Finkel, Jérôme Leroux, and Igor Potapov (Eds.), Vol. 7550. Springer, 54–65. DOI : [https://doi.org/10.1007/978-3-642-33512-9\\_6](https://doi.org/10.1007/978-3-642-33512-9_6)
- [34] Torben Hagerup. 1998. Sorting and searching on the word RAM. In *Proceedings of the STACS 98: 15th Annual Symposium on Theoretical Aspects of Computer Science Paris*. Springer, 366–398.
- [35] John E. Hopcroft and Jean-Jacques Pansiot. 1979. On the Reachability Problem for 5-Dimensional Vector Addition Systems. *Theor. Comput. Sci.* 8, 2 (1979), 135–159. DOI : [https://doi.org/10.1016/0304-3975\(79\)90041-0](https://doi.org/10.1016/0304-3975(79)90041-0)
- [36] Russell Impagliazzo and Ramamohan Paturi. 2001. On the Complexity of k-SAT. *J. Comput. Syst. Sci.* 62, 2 (2001), 367–375. DOI : <https://doi.org/10.1006/jcss.2000.1727>
- [37] Ulla Koppenhagen and Ernst W. Mayr. 2000. Optimal algorithms for the coverability, the subword, the containment, and the equivalence problems for commutative semigroups. *Inf. Comput.* 158, 2 (2000), 98–124. DOI : <https://doi.org/10.1006/inco.1999.2812>
- [38] Paraschos Koutris and Shaleen Deep. 2023. The fine-grained complexity of CFL reachability. *Proc. ACM Program. Lang.* 7, POPL, Article 59 (jan 2023), 27 pages. DOI : <https://doi.org/10.1145/3571252>
- [39] Ranko Lazic and Sylvain Schmitz. 2021. The ideal view on Rackoff's coverability technique. *Inf. Comput.* 277 (2021), 104582. DOI : <https://doi.org/10.1016/j.ic.2020.104582>
- [40] Jérôme Leroux. 2013. Vector addition system reversible reachability problem. *Log. Methods Comput. Sci.* 9, 1 (2013). DOI : [https://doi.org/10.2168/LMCS-9\(1:5\)2013](https://doi.org/10.2168/LMCS-9(1:5)2013)
- [41] Jérôme Leroux. 2021. The reachability problem for petri nets is not primitive recursive. In *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 1241–1252. DOI : <https://doi.org/10.1109/FOCS52979.2021.00121>
- [42] Jérôme Leroux and Sylvain Schmitz. 2019. Reachability in vector addition systems is primitive-recursive in fixed dimension. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*. IEEE, 1–13. DOI : <https://doi.org/10.1109/LICS.2019.8785796>
- [43] Andrea Lincoln, Virginia Vassilevska Williams, and R. Ryan Williams. 2018. Tight hardness for shortest cycles and paths in sparse graphs. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, Artur Czumaj (Ed.). SIAM, 1236–1252. DOI : <https://doi.org/10.1137/1.9781611975031.80>
- [44] Richard Lipton. 1976. The reachability problem requires exponential space. *Dep. Comput. Sci., Yale Univ.* 62 (1976), 1–16.
- [45] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. 2013. Lower bounds based on the exponential time hypothesis. *Bulletin of EATCS* 3, 105 (2013), 47–72.

- [46] Anders Alnor Mathiasen and Andreas Pavlogiannis. 2021. The fine-grained and parallel complexity of andersen's pointer analysis. *Proc. ACM Program. Lang.* 5, POPL (2021), 1–29. DOI : <https://doi.org/10.1145/3434315>
- [47] Ernst W. Mayr. 1984. An algorithm for the general petri net reachability problem. *SIAM J. Comput.* 13, 3 (1984), 441–460. DOI : <https://doi.org/10.1137/0213029>
- [48] Ernst W. Mayr and Albert R. Meyer. 1982. The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. Math.* 46, 3 (1982), 305–329. DOI : [https://doi.org/10.1016/0001-8708\(82\)90048-2](https://doi.org/10.1016/0001-8708(82)90048-2)
- [49] Filip Mazowiecki and Michał Pilipczuk. 2019. Reachability for bounded branching VASS. In *Proceedings of the 30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands (LIPIcs)*, Wan J. Fokkink and Rob van Glabbeek (Eds.), Vol. 140. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 28:1–28:13. DOI : <https://doi.org/10.4230/LIPIcs.CONCUR.2019.28>
- [50] Filip Mazowiecki, Henry Sinclair-Banks, and Karol Węgrzycki. 2023. Coverability in 2-VASS with one unary counter is in NP. In *Proceedings of the Foundations of Software Science and Computation Structures*, Orna Kupferman and Paweł Sobociński (Eds.). Springer Nature Switzerland, 196–217. DOI : [https://doi.org/10.1007/978-3-031-30829-1\\_10](https://doi.org/10.1007/978-3-031-30829-1_10)
- [51] Marvin L. Minsky. 1967. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc.
- [52] Jaroslav Nešetřil and Svatopluk Poljak. 1985. On the complexity of the subgraph problem. *Comment. Math. Univ. Carol.* 26, 2 (1985), 415–419.
- [53] Charles Rackoff. 1978. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.* 6, 2 (1978), 223–231. DOI : [https://doi.org/10.1016/0304-3975\(78\)90036-1](https://doi.org/10.1016/0304-3975(78)90036-1)
- [54] Louis E. Rosier and Hsu-Chun Yen. 1986. A multiparameter analysis of the boundedness problem for vector addition systems. *J. Comput. Syst. Sci.* 32, 1 (1986), 105–135. DOI : [https://doi.org/10.1016/0022-0000\(86\)90006-1](https://doi.org/10.1016/0022-0000(86)90006-1)
- [55] Sylvain Schmitz. 2016. The complexity of reachability in vector addition systems. *ACM SIGLOG News* 3, 1 (2016), 4–21. Retrieved from <https://dl.acm.org/citation.cfm?id=2893585>
- [56] Sylvain Schmitz and Lia Schütze. 2024. On the length of strongly monotone descending chains over  $\mathbb{N}^d$ . In *Proceedings of the 51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia (LIPIcs)*, Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson (Eds.), Vol. 297. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 153:1–153:19. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2024.153>
- [57] Michael Sipser. 1996. Introduction to the theory of computation. *ACM Sigact News* 27, 1 (1996), 27–29.
- [58] Leslie G. Valiant and Mike Paterson. 1975. Deterministic one-counter automata. *J. Comput. Syst. Sci.* 10, 3 (1975), 340–350. DOI : [https://doi.org/10.1016/S0022-0000\(75\)80005-5](https://doi.org/10.1016/S0022-0000(75)80005-5)
- [59] Wil M. P. van der Aalst. 1997. Verification of workflow nets. In *Proceedings of the Application and Theory of Petri Nets 1997, 18th International Conference, ICATPN '97, Toulouse, France, June 23-27, 1997, Proceedings (Lecture Notes in Computer Science)*, Pierre Azéma and Gianfranco Balbo (Eds.), Vol. 1248. Springer, 407–426. DOI : [https://doi.org/10.1007/3-540-63139-9\\_48](https://doi.org/10.1007/3-540-63139-9_48)
- [60] Raphael Yuster and Uri Zwick. 2004. Detecting short directed cycles using rectangular matrix multiplication and dynamic programming. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, J. Ian Munro (Ed.). SIAM, 254–260. Retrieved from <https://dl.acm.org/citation.cfm?id=982792.982828>
- [61] Don Zagier. 1997. Newman's short proof of the prime number theorem. *Am. Math. Mon.* 104, 8 (1997), 705–708.

Received 19 March 2024; revised 8 April 2025; accepted 29 June 2025



# Envy-Free Cake-Cutting for Four Agents

ALEXANDROS HOLLENDER, All Souls College, University of Oxford, Oxford, United Kingdom

AVIAD RUBINSTEIN, Stanford University, Stanford, United States

---

In the envy-free cake-cutting problem, we are given a resource, usually called a cake and represented as the  $[0, 1]$  interval, and a set of  $n$  agents with heterogeneous preferences over pieces of the cake. The goal is to divide the cake among the  $n$  agents such that no agent is envious of any other agent. Even under a very general preferences model, this fundamental fair division problem is known to always admit an exact solution where each agent obtains a connected piece of the cake; we study the complexity of *finding* an approximate solution, i.e., a connected  $\varepsilon$ -envy-free allocation.

For *monotone* valuations of cake pieces, Deng, Qi, and Saberi (2012) gave an efficient ( $\text{poly}(\log(1/\varepsilon))$  queries) algorithm for three agents and posed the open problem of four (or more) monotone agents. Even for the special case of *additive* valuations, Brânzei and Nisan (2022) conjectured an  $\Omega(1/\varepsilon)$  lower bound on the number of queries for four agents. We provide the first efficient algorithm for finding a connected  $\varepsilon$ -envy-free allocation with four monotone agents.

We also prove that as soon as valuations are allowed to be *non-monotone*, the problem becomes hard: it becomes PPAD-hard, requires  $\text{poly}(1/\varepsilon)$  queries in the black-box model, and even  $\text{poly}(1/\varepsilon)$  *communication complexity*. This constitutes, to the best of our knowledge, the first intractability result for any version of the cake-cutting problem in the communication complexity model.

CCS Concepts: • **Theory of computation → Problems, reductions and completeness; Algorithmic game theory; Communication complexity;**

Additional Key Words and Phrases: cake cutting, envy-free, communication complexity, query complexity, PPAD

**ACM Reference Format:**

Alexandros Hollender and Aviad Rubinstein. 2025. Envy-Free Cake-Cutting for Four Agents. *J. ACM* 72, 5, Article 34 (October 2025), 54 pages. <https://doi.org/10.1145/3765615>

---

## 1 Introduction

The field of fair division studies ways of dividing and allocating a resource among different agents, while ensuring that the division is *fair* in some sense. The perhaps most famous, and certainly most well-studied problem in fair division is *cake-cutting*. In this problem, first introduced by

---

A preliminary version of this article appeared at FOCS 2023.

The first author was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number MB22.00026. The second author was supported by NSF CCF-2112824, and a David and Lucile Packard Fellowship. Authors' Contact Information: Alexandros Hollender, All Souls College, University of Oxford, Oxford, United Kingdom; e-mail: alexandros.hollender@cs.ox.ac.uk; Aviad Rubinstein, Stanford University, Stanford, California, United States; e-mail: aviad@stanford.edu.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART34

<https://doi.org/10.1145/3765615>

Steinhaus [38], the resource is a “cake”. The cake serves as a metaphor for modeling settings where the resource is divisible—namely, it can be divided arbitrarily—and heterogeneous—different parts of the resource have different attributes and can thus be valued differently by different agents. Examples where the resource can be modeled as a cake include, among other things, division of land, time, and other natural resources. The problem has been extensively studied in mathematics and economics [10, 32] and more recently in computer science [30].

More formally, the cake is usually modeled as the interval  $[0, 1]$ , and there are  $n$  agents, each with their own preferences over different pieces of the cake. The preferences of each agent  $i$  are represented by a valuation function  $v_i$  which assigns a value to each piece of the cake. The goal is to divide the cake into  $n$  pieces  $A_1, \dots, A_n$  and assign one piece to each agent in a *fair* manner. Different notions of fairness have been proposed and studied. Steinhaus [38], with the help of Banach and Knaster, originally studied a fairness notion known as *proportionality*. An allocation where agent  $i$  is assigned piece  $A_i$  is proportional, if  $v_i(A_i) \geq v_i([0, 1])/n$  for all agents  $i$ . While proportionality ensures that each agent obtains a fair share of the total cake, some agents might be unhappy, because they prefer a piece  $A_j$  allocated to some other agent to their own piece  $A_i$ .

A stronger notion of fairness which tries to address this is *envy-freeness*. An allocation is envy-free, if no agent is envious of another agent’s piece. Formally, we require that  $v_i(A_i) \geq v_i(A_j)$  for all agents  $i, j$ . The problem of envy-free cake-cutting was popularized by Gamow and Stern [19] and has since been studied extensively. Stromquist [39] and Woodall [44] have shown that an envy-free allocation always exists under some mild continuity assumptions on the valuations. Moreover, this envy-free allocation is also *connected* (or *contiguous*), meaning that every agent is allocated a piece that is a single interval. In other words, the cake is only cut at  $n - 1$  positions.

The existence proofs of Stromquist and Woodall, as well as the more recent proof of Su [41], all rely on tools such as Brouwer’s fixed point theorem, or Sperner’s lemma. These powerful tools prove the existence of envy-free allocations, but they do not yield an efficient algorithm for finding one. Given the practical importance of fair division problems such as cake-cutting, this is a serious drawback to the existence result. Mathematicians and economists have tried to address this by proposing so-called *moving-knife* protocols to solve the problem. Unfortunately, the definition of a moving-knife protocol is mostly informal and thus, not well-suited for theoretical investigation (see [11] for a discussion). In more recent years, research in theoretical computer science has started studying these questions in formal models of computation. However, despite extensive efforts on the envy-free cake-cutting problem, its complexity is still poorly understood.

In this article, we study the envy-free cake-cutting problem with *connected pieces*. A simple algorithm is known for two agents: the *cut-and-choose* protocol. The first agent cuts the cake in two equal parts, according to its own valuation, and then the second agent picks its favorite piece, and the first agent receives the remaining piece. This simple algorithm can be implemented in the standard Robertson–Webb query model [43], and yields an envy-free allocation. For three players or more, Stromquist [40] has shown that no algorithm using Robertson–Webb queries exists for finding envy-free allocations. In order to bypass this impossibility result, Brânzei and Nisan [14] propose to study the complexity of finding approximate envy-free allocations instead. An allocation is  $\varepsilon$ -envy-free, if  $v_i(A_i) \geq v_i(A_j) - \varepsilon$  for all agents  $i, j$ . In this setting, an  $\varepsilon$ -envy-free allocation can be found by brute force using  $\text{poly}(1/\varepsilon)$  queries when the number of agents is constant. An *efficient* algorithm is one that instead only uses  $\text{poly}(\log(1/\varepsilon))$  queries.

With this relaxation in place, the problem with three agents can be solved efficiently [17]. As shown by Brânzei and Nisan [14], moving-knife algorithms due to Barbanel and Brams [6] and Stromquist [39] can also be efficiently simulated in the query model. All of these algorithms require that the valuations be *monotone*, a standard assumption in many works on the topic. A valuation function  $v_i$  is monotone if  $v_i(A) \geq v_i(B)$ , whenever  $A$  is a superset of  $B$ .

For four agents, the problem has remained open, even for moving-knife procedures. Barbanel and Brams [6] provided a moving-knife procedure for four agents but using five cuts, improving upon an existing result of Brams et al. [11] which used 11 cuts, and asked whether the minimal number of cuts could be achieved. The problem of finding an algorithm for connected  $\varepsilon$ -envy-free cake-cutting with four agents was explicitly posed by Deng et al. [17]. More recently it was conjectured by Brânzei and Nisan [14] that such an algorithm might not exist:

“We conjecture that unlike equitability, which remains logarithmic in  $1/\varepsilon$  for any number of players, computing a contiguous  $\varepsilon$ -envy-free allocation for  $n = 4$  players [...] will require  $\Omega(1/\varepsilon)$  queries.”

**Our contribution.** Our first contribution is to disprove this conjecture by providing an efficient algorithm that finds an  $\varepsilon$ -envy-free allocation using  $O(\log^3(1/\varepsilon))$  value queries.<sup>1</sup> As for existing approaches for three agents [6, 14, 17, 39], our algorithm also relies on the monotonicity assumption.

In the second part of our work, we investigate whether monotonicity is necessary for obtaining efficient algorithms. We prove that this is indeed the case, in a very strong sense. Namely, we show that the *communication complexity* of finding an  $\varepsilon$ -envy-free allocation with four *non-monotone* agents is  $\Omega(\text{poly}(1/\varepsilon))$ . To the best of our knowledge, this is the first intractability result for any version of the cake-cutting problem in the communication model. For the case of agents with identical non-monotone valuations, our reduction also yields an  $\Omega(\text{poly}(1/\varepsilon))$  query lower bound, as well as a PPAD-hardness result in the standard Turing machine model.

Our hardness results improve upon existing lower bounds by Deng et al. [17] in two ways:

- Our lower bounds apply to valuation functions, whereas the lower bounds of Deng et al. only apply to a much more general class of preference functions, where the value of an agent for a piece can depend on how the whole cake is divided.
- Whereas their lower bounds apply only to the query and Turing machine models, we show hardness in the communication model.

We note that the communication model was recently formalized and studied by Brânzei and Nisan [13].

**Open problems and future directions.** In Tables 1–3, we summarize the current state-of-the-art results for connected  $\varepsilon$ -envy-free cake-cutting, including our results, in three natural models of computation for this problem. In all of these tables, upper bounds appearing in a certain cell also continue to apply for any cell to the left or above. Similarly, lower bounds also apply to any cell to the right or below. This is due to the fact that “monotone” is a special case of “general”, as well as to the (surprisingly) non-trivial fact that the problem with  $n$  agents can be efficiently reduced to the problem with  $n + 1$  agents; see Appendix A.

The following questions are particularly interesting:

- What is the complexity of the problem for five agents with monotone valuations? Can the problem be solved efficiently, or can we show a lower bound? Can we at least show a lower bound for some larger number of players?
- What is the complexity of the problem for three agents with general valuations?
- Can any of the insights used in our algorithm for four players be used to tackle the problem of proving existence of EFX allocations for four agents in the indivisible goods setting?

---

<sup>1</sup>The conjecture of Brânzei and Nisan [14] was stated in a slightly different model (compared to the one in [17]) where one does not assume any upper bound on the Lipschitz-constant of the valuations, but instead allows the added power of cut queries from the Robertson–Webb model (as well as restricting attention to additive valuations). We show that a modification of our algorithm also applies to their model, thus disproving their conjecture; see Section 6.

Table 1. Query Complexity Bounds for  $\varepsilon$ -Envy-free Cake-cutting

valuations	$n = 2$	$n = 3$	$n = 4$	$n \geq 5$
<b>monotone</b>	$\Theta(\log(1/\varepsilon))$	$O(\log^2(1/\varepsilon))$	$O(\log^3(1/\varepsilon))$	?
<b>general</b>		?	$\Theta(\text{poly}(1/\varepsilon))$	$\Theta(\text{poly}(1/\varepsilon))$

Here “ $\Theta(\text{poly}(1/\varepsilon))$ ” denotes that there is a polynomial upper bound and a (possibly different) polynomial lower bound. For  $n = 2$  the upper bound is obtained by using binary search to simulate the cut-and-choose protocol, while the lower bound is a simple exercise. The upper bound for  $n = 3$  was shown by Deng et al. [17]. Alternatively, this bound can also be obtained by simulating the Barbanel–Brams [6] or Stromquist [39] moving-knife protocols using value queries. (We note that for additive valuations with the Robertson–Webb query model, Brânzei and Nisan [14] have proved a  $\Theta(\log(1/\varepsilon))$  bound for  $n = 3$ .) The remaining results in the table are proved in this article (Sections 5 and 8). All the lower bounds also hold for agents with identical valuations.

Table 2. Communication Complexity Bounds for  $\varepsilon$ -Envy-free Cake-cutting

valuations	$n = 2$	$n = 3$	$n = 4$	$n \geq 5$
<b>monotone</b>	$O(\log(1/\varepsilon))$	$O(\log(1/\varepsilon))$	$O(\log^2(1/\varepsilon))$	?
<b>general</b>		?	$\Theta(\text{poly}(1/\varepsilon))$	$\Theta(\text{poly}(1/\varepsilon))$

For  $n = 2$  the bound is obtained by the cut-and-choose protocol. For  $n = 3$  the bound follows by a simple generalization of the same bound shown for additive valuations by Brânzei and Nisan [14]. For  $n = 4$  the upper bound for monotone valuations follows from our algorithm together with some tools introduced in [14]. We provide proof sketches for these two bounds in Appendix B. The lower bounds are proved in this article (Section 8).

Table 3. Computational Complexity of  $\varepsilon$ -Envy-free Cake-cutting

valuations	$n = 2$	$n = 3$	$n = 4$	$n \geq 5$
<b>monotone</b>	P	P	P	?
<b>general</b>		?	PPAD	PPAD

Here “PPAD” denotes that the problem is PPAD-complete. The problem lies in PPAD for all  $n$ . As before, the results for  $n = 2$  and  $n = 3$  follow from cut-and-choose and the work of Deng et al. [17], respectively. The remaining results in the table are proved in this article (Sections 5 and 8), and the PPAD-hardness results also hold for agents with identical valuations.

**Further related work.** The envy-free cake-cutting model has also been extensively studied without requiring that the pieces allocated to the agents be connected. The celebrated works of Aziz and Mackenzie [4] have shown that an exact envy-free allocation can be found in the Robertson–Webb query model for any number of players, albeit with a prohibitive number of cuts. It is an open question whether this can be improved, as only a lower bound of  $\Omega(n^2)$  due to Procaccia [30] is currently known.

Efficient algorithms for envy-free cake cutting have been obtained for some relatively large constant approximation factors [1, 7, 22], or under stronger assumptions, e.g., [8, 12]. Our focus in this work is on (approximately) *envy-free* cake cutting, but other objectives have been considered such as equitable [31], (Nash) Social Welfare [1, 2], and extensions to group fairness [34–36]. In this work we are only concerned with the algorithmic question of finding an envy free allocation, but several works have also considered the important question of incentives, e.g., [3, 9, 16, 26–28, 42].

## 2 Technical Overview

We begin with a technical overview of the algorithm in Section 2.1. The most novel technical contribution of our work is in proving a communication complexity lower bound for non-monotone agents. In Section 2.2, we contrast with previous work and explain the first step in the reduction, a novel communication variant of the END-OF-LINE problem. In Section 2.3, we give an overview of our embedding of the discrete INTERSECTION END-OF-LINE into the inherently continuous EF-CAKE-CUTTING problem.

### 2.1 Overview of the Algorithm

The core idea for our algorithm is to define a special invariant, parameterized by  $\alpha \in [0, 1]$  that satisfies the following useful desiderata:

- For any  $\alpha \in [0, 1]$ , we can efficiently find a partition satisfying the invariant, if one exists.
- Starting from any partition satisfying the invariant, there is a continuous path in the space of partitions where the invariant holds and  $\alpha$  increases monotonically. The partition at the end of this path yields an EF allocation.
- The invariant is guaranteed to hold at Agent 1’s equipartition<sup>2</sup> (i.e., a partition where she values all pieces of the cake equally—this equipartition can also be found efficiently). It is guaranteed *not* to hold for  $\alpha = 1$ .

Together, these suggest a simple algorithm for finding  $\varepsilon$ -EF allocations: Set  $\underline{\alpha}$  to be the  $\alpha$  corresponding to Agent 1’s equipartition, and set  $\bar{\alpha} = 1$ . Then we can use binary search to find a (locally) maximal  $\alpha$  where the invariant holds, and output the corresponding partition.

The key is of course in identifying such a nice invariant. In Section 4, we identify such an invariant and use it to present a new proof of the existence of envy-free allocations for four monotone agents. The invariant is defined as the OR of the following two conditions:

**Condition A:** Agent 1 is indifferent between its three favorite pieces, and the remaining piece is (weakly) preferred by (at least) two of the three other agents.

**Condition B:** Agent 1 is indifferent between its two favorite pieces, and the two remaining pieces are each (weakly) preferred by (at least) two of the three other agents.

The value  $\alpha$  is the value that Agent 1 has for its favorite piece(s) in the division.

We then use these insights to obtain an efficient algorithm in Section 5. In Section 6, we also present a variant of the algorithm that works in the Robertson–Webb model (i.e., for additive valuations without the bounded Lipschitzness assumption).

### 2.2 Technical Highlight: Communication of END-OF-LINE

**Background: totality, END-OF-LINE, and lifting gadgets.** The first major obstacle for proving the hardness of cake cutting is that the problem is total, i.e., there always exists an (exactly) envy-free allocation. Since the proof of existence uses Sperner’s Lemma/Brouwer’s fixed point theorem, the natural candidate for understanding the complexity of actually finding the solution is the END-OF-LINE problem.

*Definition 1* (END-OF-LINE [29]).

**Input** A directed graph  $G = (V, E)$ , described as functions (or circuits in the computational model)  $S, P$  that, for each vertex, return its list of outgoing, incoming edges (respectively); a special vertex 0 with a single outgoing edge and no incoming edges.

---

<sup>2</sup>Unless the equipartition is already envy-free, in which case we are done.

**Output** One of the following:

- A vertex (different from 0) with in-degree  $\neq$  out-degree.
- An inconsistency: pair of vertices  $u, v$  such that  $S(u) = v$  but  $P(v) \neq u$  (or vice versa).
- A vertex with more than one outgoing or more than one incoming edges.<sup>3</sup>

It is known that END-OF-LINE is hard in the query complexity model. Furthermore, while END-OF-LINE is inherently a discrete graph problem, using known techniques, e.g., due to [17], we can embed it as a continuous cake-cutting instance (albeit for general *preferences*; our extension to non-monotone *valuations* is non-trivial). The instances resulting from this reduction require high query complexity even when all the agents' preferences are identical—i.e., they do not at all capture the difficulty from communication between agents with different utilities.

Fortunately, there is a very powerful machinery for *lifting* query complexity lower bounds to communication complexity. These lifting theorems (sometimes also called “simulation theorems”) replace each bit in the query problem with a small *lifting gadget* whose input is distributed between the parties of the communication problem. This must be done in a special way to ensure that the problem remains hard for communication complexity. (e.g., naively partitioning the edges of the END-OF-LINE instance results in a communication-easy problem!)

**The main dilemma.** On one hand, those special lifting gadgets are inherently discrete, so even if we have a query-hard cake cutting instance, if we try to directly lift it to communication complexity, the resulting information structure looks nothing like agents' valuations over intervals of cake. On the other hand, if we first lift the END-OF-LINE problem, the new communication problem loses the special structure of END-OF-LINE, and it is not clear how to embed it as a continuous, total problem.

**Previous approaches (and why they fail for cake cutting).** The same conundrum was previously encountered in works on communication complexity of Nash equilibria and Brouwer fixed points [5, 20, 24, 33]. These works used very different approaches for this issue, but ultimately they all consider Brouwer's fixed point on a hypercube of some dimension, and then go to higher dimensions<sup>4</sup> to embed the lifting gadgets. In fact, while the query variant of Brouwer is already hard in 2-D [25], none of these articles give non-trivial lower bounds on communication complexity for any constant dimension.

For cake cutting, we would really like to keep the dimension as small as possible: First, in the valuations model, the input is a function from the cake intervals; each interval is defined by only two cuts, i.e., it is inherently a 2-D object. Furthermore, even if we try to get away with higher dimension by simultaneously considering more cuts in the partition in the preferences model, the number of cuts (and hence the number of agents) should scale with the dimension. But our goal is to show hardness for as few agents as possible.

**Step I: A new total communication problem: INTERSECTION END-OF-LINE.** Instead of working with the END-OF-LINE instance and the lifting gadgets separately, we present a new communication variant of END-OF-LINE that naturally combines END-OF-LINE with the flagship hard problem of communication complexity: SETDISJOINTNESS (equivalently, “SETINTERSECTION”):

*Definition 2 (INTERSECTION END-OF-LINE).*

---

<sup>3</sup>Usually  $S, P$  are hardcoded to have exactly one neighbor; this representation will be more convenient for our purposes.

<sup>4</sup>[24] obtain tight bounds by reducing the “dimension” of their instance by encoding a single point in the Brouwer hypercube across the different actions in the support of a player's mixed strategy at equilibrium. It would be very interesting if there is an analogue of this approach for cake cutting.

**Input** Each party receives a superset of the edges; we say that an edge is *active* in the END-OF-LINE instance  $G = (V, E)$  if it is in the intersection of all the supersets. Edges that only appear in the supersets of some parties are called *inactive*.

**Output** A solution to the END-OF-LINE instance defined by the active edges.

We prove that INTERSECTION END-OF-LINE requires communication polynomial in the input size, even under strong assumptions on the inputs. In particular, by starting from a Number-on-Forehead (NoF) lifting theorem, we can get instances where every inactive edge only appears in one party's inputs. This is helpful for embedding as a cake cutting problem, because for any cake-partition most of the parties will correctly evaluate it and can guarantee a conflict in case it does not correspond to a solution to the original END-OF-LINE instance.<sup>5</sup>

LEMMA 2.1. INTERSECTION END-OF-LINE with  $k \geq 3$  parties requires  $\text{poly}(|G|)$  communication complexity, even in the special case where the instances satisfy the following promises:

- (0) Every node has at most one active incoming edge, and at most one active outgoing edge.
- (1) Every inactive edge is only included in at most one party's set of edges.
- (2) No vertex has both an  $i$ -inactive and a  $j$ -inactive incident edge, for some  $i \neq j$ . (An edge is said to be  $i$ -inactive, if it is inactive, but included in Party  $i$ 's superset.)
- (3) Every vertex is assigned to one party (the vertex-party assignment is publicly known): Only that party's superset may have more than one edge coming into that vertex. Similarly, only that party's superset may have more than one edge going out of that vertex.

We expect that INTERSECTION END-OF-LINE will find other applications. For example, it can be used to show that various (total) communication variants of Brouwer's fixed point problem are hard even in 2-D.

**A few words about the hardness of INTERSECTION END-OF-LINE.** We give a very brief overview of the proof of the lemma (see Section 7.2 for details). Our starting point is a NoF lifting of END-OF-LINE, i.e., the parties need to solve a small communication problem (the “lifting gadget”) to compute  $S(v)$  and  $P(v)$  for each  $v$ .

For every possible edge  $(u \rightarrow v)$ , and every possible vector of inputs to  $v$ 's and  $u$ 's lifting gadgets corresponding to edge  $(u \rightarrow v)$ , we construct a sequence of vertices that incrementally build this vector. At each vertex in the sequence, the edge to the next vertex is determined by an input on a single party's forehead. That party's superset includes all the edges that could correspond to this input, while all the other parties only keep the edge that corresponds to the true input. When we take the intersection of the parties supersets of edges, we're left with just the path from  $u$  to  $v$ .

### 2.3 Overview of Reduction to Communication Complexity of Cake Cutting

We now give an overview of Steps II and III of our hardness result for Cake Cutting in the communication model. This is probably the most technically involved part of our article.

**Step II: Embedding INTERSECTION END-OF-LINE in the plane.** Our next step is to embed INTERSECTION END-OF-LINE à-la Sperner-coloring in 2D. Our approach is inspired by the work of [15] that embed END-OF-LINE as a 2D Sperner coloring problem, but the final embedding is far more delicate in order to accommodate the double information restriction imposed by our setting: (i) first, because the input to INTERSECTION END-OF-LINE is distributed, parties disagree about which path goes through any point in the plane; (ii) in the next step, we want to turn our

<sup>5</sup>While capturing the intuition, this is not exactly true. In some points in our reduction 2 out of 4 parties do not have sufficient information. But in those special points there is additional structure that allows us to guarantee a conflict between the 2 remaining parties.

embedding into *valuations* of cake pieces, but each piece only “sees” one or two of the three cuts defining the partition.

Our embedding assigns a color for each grid point in the discretized unit square; colors of other points are defined as weighted averages of the corner of their cells. Each color consists of two binary labels  $(\pm 1, \pm 1)$ , and a solution to the Sperner problem consists of a grid cell where two of the four corners have opposite labels in both coordinates. In the communication problem, the embedding for each player is different because they have a different view of the graph, but we guarantee, roughly, that in at least one coordinate a majority of the agents see the same sign.

Modulo using the somewhat non-standard four colors (rather than three) for a 2D Sperner, at a high level our embedding is similar to previous work (e.g., [15, 25]): Embedded END-OF-LINE paths are colored by three stripes:  $(+1, +1), (-1, +1), (-1, -1)$ , with the background (where embedded paths do not pass) colored by the default  $(+1, -1)$ ; intuitively this ensures that Sperner solutions (i.e.,  $(+1, +1), (-1, -1)$  or  $(+1, -1), (-1, +1)$  in the same grid cell) can only happen at endpoints of embedded paths.

**A special intersection gadget.** Since we’re working in 2D, we also have to worry about intersection of embedded edges/paths; at a high level, we use the intersection gadget of [15]. However, the details are extremely subtle, especially where one party sees a crossing of embedded edges, while for the other three parties only one of those edges exist. We design custom variants of both the intersection gadget and simple horizontal paths (we don’t need to modify vertical paths) that guarantee a very specific desideratum. See Section 8.1 for the full details.

**Step III: How to evaluate a piece of cake.** The final step in our construction is to define the valuations of cake pieces. The value of every piece is always fairly close to its length, so we can assume wlog that in any  $\varepsilon$ -EF partition, the lengths are not too far from equal; in particular it is easy to tell whether a piece is the second or third piece based on its endpoints.

Given cuts  $(\ell, m, r)$  we want to ensure that  $\ell \approx 1 - r$ ; then we use  $\ell \approx 1 - r$  to encode the  $x$  coordinate of the unit square, and  $m$  to encode the  $y$  coordinate. The challenges are that (i) the values of the external pieces cannot depend on the  $y$  coordinate of the embedding (this is why our embedding treats horizontal and vertical paths differently); and (ii) it is difficult to enforce  $\ell \approx 1 - r$  without creating spurious envy-free solutions.

Since the first and fourth piece do not “see” the cut  $m$  that encodes the  $y$  coordinate, their values are simpler. The value of the fourth piece is always exactly its length. The value of the first piece is usually exactly its length (this helps ensure that  $\ell \approx 1 - r$ ), except when it receives a special boost (discussed below) on carefully chosen vertical strips.

The second and third piece “see” both coordinates of the embedding, so we can slightly adjust their values based on the labels of the corresponding grid cells; specifically, for a sufficiently small  $\gamma > 0$ , the value of the second piece is equal to the value of the first piece plus  $\gamma \cdot \text{first-label}$  and the value of the third piece is equal to the value of the fourth piece plus  $\gamma \cdot \text{second-label}$ . This guarantees, for example, that in any grid cell where all parties agree that the first label is positive (resp. negative), the second piece is over-demanded (resp. under-demanded) so the corresponding partition cannot be  $\varepsilon$ -EF.

When only three of the parties agree, a positive first label will still cause the second piece to be over-demanded, but with a negative first label the fourth party may find the second piece acceptable, so it would not be under-demanded. So the riskiest regions are ones where three parties share a negative first label and neutral second label (resp. neutral first and negative second). In this case we want to ensure that the fourth party does not want the second piece (resp. the third).

In some carefully chosen vertical strips, the first (and thus, also the second piece) receive a *boost* of  $\pm \beta$  (we set  $\beta = 8\gamma$ ). Specifically, in strips where agent  $i$  thinks that the remaining three agents

may have a neutral second label, she gives the first and second pieces a negative boost in case the remaining three agents have a negative first label. Similarly, in strips where agent  $i$  thinks that the remaining three agents may have a neutral first label, she gives the first and second pieces a positive boost in case the remaining three agents have a negative second label. Note that because of the careful construction of the crossing gadgets, agent  $i$  knows based only on the  $x$  coordinate where the other three parties may see a path she does not, and in what direction it goes. This allows her to correctly implement the boost on the first piece which doesn't "see" the  $y$  coordinate.

The full details of the construction can be found in Section 8.

### 3 Preliminaries

We consider a resource, called "the cake", which is modeled as the interval  $[0, 1]$ . There are  $n$  agents, and each agent has a valuation function defined over intervals of the cake. Formally, each agent  $i \in [n]$  has a valuation function  $v_i : [0, 1]^2 \rightarrow [0, 1]$ , where  $v_i(a, b)$  represents the value that agent  $i$  has for interval  $[a, b]$ . For this to be well-defined, we require that  $v_i(a, b) = 0$  whenever  $b \leq a$ . Furthermore, we always assume that valuations  $v_i$  are continuous.

Next, we define some additional properties of valuation functions that we will sometimes assume, depending on the setting. A valuation  $v$  is

- *monotone*:  $v(a, b) \leq v(a', b')$ , whenever  $[a, b] \subseteq [a', b']$ .
- *hungry*:  $v(a, b) < v(a', b')$ , whenever  $[a, b] \subsetneq [a', b']$ .
- $\delta$ -*strongly-hungry*, for some  $\delta > 0$ :  $v(a', b') \geq v(a, b) + \delta(b' - b) + \delta(a - a')$ , whenever  $[a, b] \subseteq [a', b']$ .

Note that hungriness corresponds to strict monotonicity. Furthermore, a valuation that is  $\delta$ -strongly-hungry for some  $\delta > 0$  is in particular hungry.

For our computational results (except in Section 6), we will assume Lipschitz-continuity of the valuations. A valuation  $v$  is Lipschitz-continuous with Lipschitz constant  $L$ , if, for all  $a, b, a', b' \in [0, 1]$ :

$$|v(a, b) - v(a', b')| \leq L(|a - a'| + |b - b'|).$$

A (connected) allocation of the cake to  $n$  agents is a division of the cake  $[0, 1]$  into  $n$  intervals  $A_1, \dots, A_n$  such that  $A_i$  is assigned to agent  $i$ , the  $A_i$ 's are all pairwise disjoint,<sup>6</sup> and  $\bigcup_i A_i = [0, 1]$ . In the setting with four agents, we will denote a division of the cake by its three cuts  $(\ell, m, r)$ , where  $0 \leq \ell \leq m \leq r \leq 1$ .

**Envy-freeness.** An allocation  $(A_1, \dots, A_n)$  is said to be *envy-free* if, for all agents  $i$ , we have  $v_i(A_i) \geq v_i(A_j)$  for all  $j$ . For  $\epsilon \in [0, 1]$ , we say that the allocation is  $\epsilon$ -*envy-free* if, for all agents  $i$ , we have  $v_i(A_i) \geq v_i(A_j) - \epsilon$  for all  $j$ . An envy-free allocation is guaranteed to exist in the setting we consider [39, 41, 44].

**Normalization.** For our computational results (except in Section 6), without loss of generality, we will assume that the valuation functions are 1-Lipschitz-continuous, i.e.,  $L = 1$ . If  $L > 1$ , we can replace  $v_i$  by  $v_i/L$ , and  $\epsilon$  by  $\epsilon/L$ .

**Query complexity.** In this black-box model, we can query the valuation functions of the agents. A (value) query consists of the endpoints of an interval  $[x, y]$ , and the agent responds with its value for that interval, i.e.,  $v_i(x, y)$ . The running time of an algorithm consists of the number of queries. The problem of computing an  $\epsilon$ -envy-free allocation can be solved using  $\text{poly}(1/\epsilon)$  queries by brute force [14]. We say that an algorithm is efficient if it uses  $\text{poly}(\log(1/\epsilon))$  queries.

<sup>6</sup>To be more precise, here we assume that all  $A_i$  are closed intervals and we say that  $A_i$  and  $A_j$  are disjoint if their intersection has zero measure.

The problem has traditionally been studied in an extended query model called the Robertson–Webb model [43]. In this model, in addition to the value queries, there is a second type of query called a *cut* query. On cut query  $(x, \alpha)$ , where  $x$  is a position on the cake and  $\alpha$  is a value, the agent responds by returning a position  $y$  on the cake such that  $v_i(x, y) = \alpha$  (or responds that there is no such  $y$ ). Although this model is usually studied with additive valuations, it can easily be extended to the setting of monotone valuations.<sup>7</sup>

In the monotone setting, with our assumption of Lipschitz-continuity of the valuations and in the context of looking for approximate fairness (such as  $\epsilon$ -envy-freeness), the two models are equivalent up to  $\text{poly}(\log(1/\epsilon))$  factors, since a cut query can be simulated by  $O(\log(1/\epsilon))$  value queries [14]. In particular, a lower bound for the (value) query complexity also implies a (qualitatively) similar lower bound for the Robertson–Webb query model.

Since the Robertson–Webb model is usually studied for additive valuations and *without* the bounded Lipschitz-continuity assumption, we also present a version of our algorithm that applies to this setting, namely, additive valuations that are non necessarily  $L$ -Lipschitz-continuous. See Section 6 for the details and the definition of the Robertson–Webb model.

**Communication complexity.** In this model, each agent corresponds to one party in a communication setting. Each agent/party only knows its own valuation function. The parties can communicate by sending messages and the complexity of such a communication protocol is measured in terms of the number of bits sent between parties. A protocol is efficient if it uses at most  $\text{poly}(\log(1/\epsilon))$  communication.

A query algorithm yields a communication protocol of (qualitatively) equivalent complexity [13]. As a result, a lower bound in the communication complexity setting yields a lower bound in the query complexity setting.

**Computational complexity.** In the “white box” model, we assume that the valuations are given to us in the input, and an efficient algorithm is one that runs in polynomial time in the size of the representation of the valuations and in  $\text{poly}(\log(1/\epsilon))$ . For example, the valuations can be given as well-behaved arithmetic circuits [18] or as Turing machines together with a polynomial upper bound on their running time. In this model, the problem is a total NP search problem, i.e., it lies in the class TFNP. Furthermore, it is known to lie in the subclass PPAD of TFNP [17].

## 4 A New Proof of Existence of Connected Envy-free Allocations for Four Monotone Agents

In this section, we present a new proof for the existence of connected envy-free allocations for four agents with monotone valuations. This new proof is the main insight that allows us to obtain efficient algorithms for the problem in subsequent sections.

**THEOREM 4.1.** *For four agents with monotone valuations, there always exists a connected envy-free allocation.*

Existing proofs [39, 41, 44] apply to more general valuations, but rely on strong topological tools such as Brouwer’s fixed point theorem, or equivalent formulations. Here we show that monotonicity allows for a simpler proof that bypasses these tools.

**Hungry valuations.** The first step of the proof is to restrict ourselves to hungry valuations. This is without loss of generality, since the existence of envy-free allocations for hungry agents is

---

<sup>7</sup>The natural extension of the Robertson–Webb model to monotone valuations should also allow for a cut query in the other direction, i.e., on cut query  $(\alpha, y)$ , the agent responds by returning a position  $x$  on the cake such that  $v_i(x, y) = \alpha$  (or responds that there is no such  $x$ ). For additive valuations such queries can be easily simulated using standard cut queries.

sufficient to obtain existence for the non-hungry case too. Indeed, given (possibly non-hungry) valuations  $v_i$ , for any  $\varepsilon \in (0, 1)$  we can replace them by hungry valuations  $v'_i$  defined as  $v'_i(a, b) := (1 - \varepsilon)v_i(a, b) + \varepsilon(b - a)$ . An envy-free allocation for the  $v'_i$  will then be approximately envy-free for the  $v_i$ , where the approximation can be made arbitrarily small by picking  $\varepsilon > 0$  sufficiently small. The following standard compactness argument then shows that a connected envy-free allocation must also exist for the original valuations  $v_i$ . Thus, in the remainder of this section we assume that the valuations are hungry.

**LEMMA 4.2.** *If for some valuations  $v_1, \dots, v_n$  there exist  $\varepsilon$ -envy-free allocations for all  $\varepsilon > 0$ , then there exists an (exact) envy-free allocation.*

**PROOF.** Consider a sequence  $(A^k)_k$  of  $1/k$ -envy-free divisions. Since the domain of all divisions is compact, there exists a converging subsequence of  $(A^k)_k$ . Furthermore, there exists such a converging subsequence such that the assignment of pieces to agents (i.e., Agent 1 gets piece 3, etc) that makes the division  $1/k$ -envy-free remains the same throughout the subsequence. Since the valuations are continuous, the division that is the limit of the constructed subsequence together with the fixed assignment of pieces to agents must be envy-free.  $\square$

**Equipartitions.** The next step is to prove that an envy-free allocation always exists when all four agents have the same valuation function  $v_i = v$ . In this case, an envy-free allocation corresponds to an *equipartition* according to  $v$ , i.e., a division into four pieces which all have the same value according to  $v$ . The monotonicity of the valuation function allows for a simple proof of existence of equipartitions.

**LEMMA 4.3.** *For any monotone hungry valuation function  $v$ , and any  $n \geq 1$ , there exists a (unique) equipartition of the cake into  $n$  equal parts according to  $v$ .*

**PROOF.** We only present the proof here for  $n = 4$ , but it straightforwardly generalizes to any  $n \geq 1$ . For any  $\alpha \in [0, 1]$ , we let  $\ell(\alpha)$  denote the unique cut position satisfying  $v(0, \ell(\alpha)) = \alpha$  if it exists, and set  $\ell(\alpha) = 1$  otherwise (i.e., when  $v(0, 1) < \alpha$ ). Since  $v$  is hungry and continuous, it is easy to see that  $\ell(\cdot)$  is well-defined and continuous. Similarly, we define  $m(\cdot)$  such that  $v(\ell(\alpha), m(\alpha)) = \alpha$  (or  $m(\alpha) = 1$  if this is not possible), and  $r(\cdot)$  such that  $v(m(\alpha), r(\alpha)) = \alpha$  (or  $r(\alpha) = 1$  if this is not possible). Finally, since  $v(r(0), 1) - 0 = v(0, 1) > 0$  and  $v(r(1), 1) - 1 = -1 < 0$ , by continuity there exists  $\alpha^* \in (0, 1)$  such that  $v(r(\alpha^*), 1) = \alpha^*$ . Then the division  $(\ell(\alpha^*), m(\alpha^*), r(\alpha^*))$  is an equipartition into four parts. Furthermore, it is easy to see that the hungriness of  $v$  also implies that the equipartition is unique.  $\square$

**Continuous path: Intuition.** We are now ready to prove the existence of envy-free allocations. At a high level, the proof proceeds as follows. Starting from the equipartition into four parts according to Agent 1's valuation, we show that we can move the cuts in a continuous manner such that the following two properties hold:

- (1) As long as we have not reached an envy-free allocation, there is a way to continue moving the cuts.
- (2) When we move the cuts, we make sure that the value of Agent 1 for its favorite piece always strictly increases.

It then follows that this continuous path in the space of divisions has to terminate, and thus an envy-free division must exist.

In more detail, we move the cuts continuously while maintaining the following invariant: the division is *critical*. We say that a given division into four pieces is *critical* if it satisfies Condition A or Condition B (or both):

**Condition A:** Agent 1 is indifferent between its three favorite pieces, and the remaining piece is (weakly) preferred by (at least) two of the three other agents.

**Condition B:** Agent 1 is indifferent between its two favorite pieces, and the two remaining pieces are each (weakly) preferred by (at least) two of the three other agents.

It is easy to see that Condition A necessarily holds at the starting point, i.e., for the equipartition into four parts according to Agent 1, unless this division is already envy-free. As long as Condition A holds, we shrink the remaining piece, while also making sure that Agent 1 remains indifferent between the other three pieces. When we reach a point where Condition A would no longer hold if we continued shrinking the piece, we can show that either the division is envy-free, or Condition B is satisfied. As long as Condition B holds, we increase the two pieces preferred by Agent 1, while making sure that Agent 1 remains indifferent between them, and also that one of the other agents remains indifferent between the remaining two pieces. When we reach a point where Condition B would no longer hold if we continued, we can show that either the division is envy-free, or Condition A is satisfied. This continuous movement of cuts thus satisfies point 1 above, namely, that the path can always be extended as long as we have not found an envy-free division. Furthermore, the value that Agent 1 has for its favorite pieces also always strictly increases, so point 2 is also satisfied.

**Continuous path: Formal argument.** We begin with the formal definitions of the two conditions. A division into four pieces  $(P_1, P_2, P_3, P_4)$  is *critical* if at least one of the following two conditions holds:

**Condition A:** There exists a piece  $k \in \{1, 2, 3, 4\}$  such that (i) for all other pieces  $t, t' \in \{1, 2, 3, 4\} \setminus \{k\}$ ,  $v_1(P_t) = v_1(P_{t'}) \geq v_1(P_k)$ , and (ii) there exist two (distinct) agents  $i, i' \in \{2, 3, 4\}$  such that  $v_i(P_k) \geq \max_t v_i(P_t)$  and  $v_{i'}(P_k) \geq \max_t v_{i'}(P_t)$ .

**Condition B:** There exist two (distinct) pieces  $k, k' \in \{1, 2, 3, 4\}$  such that (i) for the other two pieces  $t, t' \in \{1, 2, 3, 4\} \setminus \{k, k'\}$ ,  $v_1(P_t) = v_1(P_{t'}) \geq \max\{v_1(P_k), v_1(P_{k'})\}$ , and (ii) for each piece  $t \in \{k, k'\}$ , there exist two agents  $i, i' \in \{2, 3, 4\}$  such that  $v_i(P_t) \geq \max_{t'} v_i(P_{t'})$  and  $v_{i'}(P_t) \geq \max_{t'} v_{i'}(P_{t'})$ .

Note that Conditions A and B can possibly hold at the same time.

By Lemma 4.3 there exists an equipartition of the cake into four parts according to Agent 1. If this division is not envy-free, then it satisfies Condition A. The following lemma then allows us to conclude that an envy-free division exists and thus provides a new proof of Theorem 4.1. The lemma will also be used in the next sections when we develop algorithms for the problem.

LEMMA 4.4. *Let  $(\ell_0, m_0, r_0)$  be a critical division and let  $\alpha_0$  denote the value that Agent 1 has for its favorite pieces in that division. Then there exists  $\alpha^* \in [\alpha_0, 1)$  and a continuous path  $T: [\alpha_0, \alpha^*] \rightarrow \{(\ell, m, r) : 0 \leq \ell \leq m \leq r \leq 1\}$  such that:*

- (1) *The path begins at  $T(\alpha_0) = (\ell_0, m_0, r_0)$ .*
- (2) *For all  $\alpha \in [\alpha_0, \alpha^*]$ , the division  $T(\alpha)$  is critical and is such that Agent 1 has value  $\alpha$  for its favorite pieces.*
- (3) *The division  $T(\alpha^*)$  at the end of the path is envy-free.*

The following two claims will play a crucial role in the proof of Lemma 4.4.

CLAIM 1. *If a division  $(P_1, P_2, P_3, P_4)$  satisfies Condition B with pieces  $k, k'$ , but is not envy-free, then each of the Agents 2, 3, and 4 strictly prefers one of the pieces  $k$  or  $k'$  (or both) to any other piece, i.e.,  $\max\{v_i(P_k), v_i(P_{k'})\} > \max_{t \neq k, k'} v_i(P_t)$  for all  $i \in \{2, 3, 4\}$ .*

PROOF. According to Condition B, the (distinct) pieces  $k, k'$  satisfy (i) Agent 1 (weakly) prefers any of the two pieces in  $\{1, 2, 3, 4\} \setminus \{k, k'\}$ , and (ii) for each of the pieces  $t \in \{k, k'\}$ , there are at

least two distinct agents in  $\{2, 3, 4\}$  who (weakly) prefer piece  $t$ . To be more concrete, and without loss of generality, say that Agent 2 (weakly) prefers piece  $k$ , Agent 3 (weakly) prefers piece  $k'$ , and Agent 4 (weakly) prefers both piece  $k$  and  $k'$ . Now, we claim that no agent in  $\{2, 3, 4\}$  can (weakly) prefer any piece other than  $k$  or  $k'$ . Indeed, assume that in addition to piece  $k$ , Agent 2 also likes some piece  $t \notin \{k, k'\}$ . Then the division is envy-free: assign piece  $t$  to Agent 2, piece  $k$  to Agent 4, piece  $k'$  to Agent 3, and the remaining piece to Agent 1. Similarly, if we instead assume that Agent 3 or Agent 4 likes some piece  $t \notin \{k, k'\}$ , we again can assign a desired piece to every agent, and thus the division is envy-free. Since by assumption the division is not envy-free, it must be that each of the Agents 2, 3, and 4 strictly prefers one of the pieces  $k$  or  $k'$  (or both) to any other piece.  $\square$

**CLAIM 2.** *If a division  $(P_1, P_2, P_3, P_4)$  satisfies Condition A with piece  $k$ , but not Condition B, and is not envy-free, then there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  that strictly prefer piece  $k$  to any other piece, i.e.,  $v_i(P_k) > \max_{t \neq k} v_i(P_t)$  and  $v_{i'}(P_k) > \max_{t \neq k} v_{i'}(P_t)$ .*

**PROOF.** According to Condition A, the piece  $k$  satisfies (i) Agent 1 (weakly) prefers any of the three pieces in  $\{1, 2, 3, 4\} \setminus \{k\}$ , and (ii) there exist two (distinct) agents  $i, i' \in \{2, 3, 4\}$  who (weakly) prefer piece  $k$ . Now, because the division is not envy-free and does not satisfy Condition B, we claim that we can strengthen point (ii) to say that there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  who *strictly* prefer piece  $k$  to any of the other pieces. Indeed, assume that this is not the case. Then, Agents 2, 3, and 4 have to satisfy the following pattern. One agent, say Agent 2, (weakly) prefers piece  $k$ . Another agent, say Agent 3, (weakly) prefers both piece  $k$  and another piece  $t$ . The remaining agent, Agent 4, (weakly) prefers a piece  $t'$ , that is different from  $k$ . (These agents can of course also weakly prefer some additional pieces.) Now, if  $t \neq t'$ , then the division is envy-free: assign piece  $k$  to Agent 2, piece  $t$  to Agent 3, piece  $t'$  to Agent 4, and the remaining piece to Agent 1. If, on the other hand,  $t = t'$ , then the division satisfies Condition B with pieces  $k$  and  $t$ . Thus, we obtain a contradiction in both cases, and it must be that there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  who *strictly* prefer piece  $k$  to any of the other pieces.  $\square$

We are now ready to proceed with the proof of the lemma.

**PROOF OF LEMMA 4.4.** We begin by introducing some notation that will be used in this proof, but also later in the sections presenting the algorithms. We let  $\alpha_2^{\pm} \in (0, 1)$  be the value of the pieces in the equipartition into two parts according to Agent 1. Similarly, we let  $\alpha_3^{\pm} \in (0, \alpha_2^{\pm})$  and  $\alpha_4^{\pm} \in (0, \alpha_3^{\pm})$  be the value of the pieces in the equipartition into three or four parts according to Agent 1, respectively. Note that Condition A can only (possibly) hold at values  $\alpha \in [\alpha_4^{\pm}, \alpha_3^{\pm}]$ , and Condition B can only (possibly) hold at values  $\alpha \in [\alpha_4^{\pm}, \alpha_2^{\pm}]$ .

**Trails.** Next, we define “trails” that our continuous path will follow. A trail is itself a continuous path that satisfies some properties, and our final continuous path will be following one of these trails at any given time, but will sometimes choose to switch from one trail to another, when they intersect. The trails, which we define in detail in the next paragraph, are a succinct way of formalizing the intuition provided above for how the cuts should move in order to try to maintain the invariant, namely, to ensure that the division remains critical.

For any choice of two distinct pieces  $k, k' \in \{1, 2, 3, 4\}$  with  $k < k'$  (i.e., piece  $k$  lies on the left of piece  $k'$ ), we define the trail  $\gamma_{k,k'}$  to be the function that maps any  $\alpha \in [\alpha_4^{\pm}, \alpha_2^{\pm}]$  to the division  $(P_1, P_2, P_3, P_4)$  that is obtained as follows:

- (1) Require that  $v_1(P_t) = \alpha$  for all  $t \in \{1, 2, 3, 4\} \setminus \{k, k'\}$ . Given that requirement, the division is uniquely determined by the position  $x$  of the cut at the right end of piece  $P_k$ . This position  $x$  is then chosen as follows.

- (2) For each agent  $i \in \{2, 3, 4\}$ , let  $x_i$  be the (unique) position of that cut that ensures  $v_i(P_k) = v_i(P_{k'})$ . Let  $x_M$  be the median of the three positions  $x_2, x_3$ , and  $x_4$ .
- (3) If  $\alpha \leq \alpha_3^{\bar{\alpha}}$ , let  $x_L$  be the (unique) position of the cut that ensures  $v_1(P_{k'}) = \alpha$ , and  $x_R$  be the (unique) position that ensures  $v_1(P_k) = \alpha$ .
- (4) If  $\alpha > \alpha_3^{\bar{\alpha}}$ , then simply let  $x_L := -\infty$  and  $x_R := +\infty$ .
- (5) Finally, in both cases, set  $x$  to be the median of  $x_L, x_M$ , and  $x_R$ .

The division thus obtained is always well-defined, i.e., it exists and is unique. Furthermore, the trail  $\gamma_{k,k'}$  is continuous, namely, the division above depends continuously on  $\alpha$ . Indeed, it is easy to see that the positions  $x_2(\alpha), x_3(\alpha), x_4(\alpha)$ , and thus  $x_M(\alpha)$ , are continuous in  $\alpha$ . In addition,  $x_L(\alpha)$  and  $x_R(\alpha)$  are continuous over  $[\alpha_4^{\bar{\alpha}}, \alpha_2^{\bar{\alpha}}]$  and over  $(\alpha_3^{\bar{\alpha}}, \alpha_2^{\bar{\alpha}}]$ . Finally, the continuity of  $x(\alpha)$  follows from the fact that at  $\alpha = \alpha_3^{\bar{\alpha}}$ , we necessarily have  $x_M \in [x_L, x_R]$  and thus,  $x = x_M$ .

An important observation is that any division that is critical must lie on one of these trails. In more detail, we have the following equivalent formulations of the two conditions:

**Condition A:** Division  $(P_1, P_2, P_3, P_4)$  satisfies Condition A if there exist  $\alpha \in [\alpha_4^{\bar{\alpha}}, \alpha_2^{\bar{\alpha}}]$  and pieces  $k < k'$  such that  $(P_1, P_2, P_3, P_4) = \gamma_{k,k'}(\alpha)$  and additionally there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  such that either (a)  $x_M(\alpha) \leq x_L(\alpha)$ ,  $v_i(P_k) = \max_t v_i(P_t)$  and  $v_{i'}(P_k) = \max_t v_{i'}(P_t)$ , or (b)  $x_M(\alpha) \geq x_R(\alpha)$ ,  $v_i(P_{k'}) = \max_t v_i(P_t)$  and  $v_{i'}(P_{k'}) = \max_t v_{i'}(P_t)$ .

**Condition B:** Division  $(P_1, P_2, P_3, P_4)$  satisfies Condition B if there exist  $\alpha \in [\alpha_4^{\bar{\alpha}}, \alpha_2^{\bar{\alpha}}]$  and pieces  $k < k'$  such that  $(P_1, P_2, P_3, P_4) = \gamma_{k,k'}(\alpha)$  and additionally  $x_M(\alpha) \in [x_L(\alpha), x_R(\alpha)]$  and  $\max\{v_i(P_k), v_i(P_{k'})\} = \max_t v_i(P_t)$  for all agents  $i \in \{2, 3, 4\}$ .

Here the positions  $x_M(\alpha), x_L(\alpha), x_R(\alpha)$  refer to the corresponding positions on the trails  $\gamma_{k,k'}$  that are under consideration. Using these formulations of the two conditions, we obtain the following characterization of criticality.

**OBSERVATION 1.** *A division  $(P_1, P_2, P_3, P_4)$  is critical if and only if there exist  $\alpha \in [\alpha_4^{\bar{\alpha}}, \alpha_2^{\bar{\alpha}}]$  and pieces  $k < k'$  such that  $(P_1, P_2, P_3, P_4) = \gamma_{k,k'}(\alpha)$  and  $\max\{v_i(P_k), v_i(P_{k'})\} = \max_t v_i(P_t)$  for all agents  $i \in \{2, 3, 4\}$ .*

The “if” direction can be shown by considering the three cases: (i)  $x_M(\alpha) \in [x_L(\alpha), x_R(\alpha)]$ , (ii)  $x_M(\alpha) \leq x_L(\alpha)$ , and (iii)  $x_M(\alpha) \geq x_R(\alpha)$ . In case (i), it is immediate that Condition B holds. In case (ii),  $x_M(\alpha) \leq x_L(\alpha)$  implies that there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  such that  $v_i(P_k) \geq v_i(P_{k'})$  and  $v_{i'}(P_k) \geq v_{i'}(P_{k'})$ . As a result, Condition A holds. A very similar argument shows that case (iii) also satisfies Condition A. The proof of the “only if” direction is omitted; we will not use this direction below.

**Construction of the path.** The path  $T$  starts at the division  $T(\alpha_0) = (\ell_0, m_0, r_0)$  and follows a trail  $\gamma_{k,k'}$  as long as the invariant holds on the trail, i.e., as long as the division remains critical. When it reaches a point such that going any further would make the division not be critical anymore, we show below that either the division is envy-free, or the path can follow a different trail on which the division remains critical. In more detail, below we prove the following crucial property:

**Path Extension Property:** If a division is critical but not envy-free, then there exists a trail  $\gamma_{k,k'}$  passing through the division (at some value  $\alpha$ ) that we can follow, i.e., there exists  $\delta > 0$  such that for all  $\alpha' \in [\alpha, \alpha + \delta]$  the division  $\gamma_{k,k'}(\alpha')$  remains critical.

This means that as long as no envy-free division is encountered, the continuous path  $T$  can always be extended while remaining on critical divisions. Since no trail ends in a critical division, the path has to stop before that. Let  $\alpha^*$  denote the highest value attained by the path. In theory, the path could approach  $\alpha^*$  without ever attaining it. Namely, the divisions could be critical on the path at values  $[\alpha_0, \alpha^*)$ , but not at  $\alpha^*$ . However, this is in fact impossible, since the set of critical divisions

is closed. This is because it is a finite union of sets defined by non-strict inequalities on the values of various pieces, and the valuation functions are continuous. Thus, if the divisions on the path at values  $[\alpha_0, \alpha^*]$  are critical, then so is the division at  $\alpha^*$ . As a result, the end of the path is closed, and  $T(\alpha^*)$  must be envy-free.

**Proof of the Path Extension Property.** Let  $(P_1, P_2, P_3, P_4)$  be a division that is critical, but not envy-free. Consider first the case where the division satisfies Condition B with pieces  $k < k'$ . In particular, the division lies on the trail  $\gamma_{k,k'}$  at some value  $\alpha$ . By Claim 1, we have that Agents 2, 3, and 4 strictly prefer one of the pieces  $k$  or  $k'$  (or both) to any other piece. By continuity of the trail and the valuations, it follows that there exists  $\delta > 0$  such that this remains the case on the trail  $\gamma_{k,k'}(\alpha')$  for all  $\alpha' \in [\alpha, \alpha + \delta]$ . Thus, by Observation 1 these divisions are also critical.

Now consider the case where the division satisfies Condition A with piece  $k$ , but not Condition B. Since we assumed that the division is not envy-free, Claim 2 implies that there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  that strictly prefer piece  $k$  to any other piece. Assuming for now that  $k$  is not the rightmost piece, the division lies on a trail  $\gamma_{k,k'}$  at some value  $\alpha$ , where  $k' > k$  is an arbitrary piece lying to the right of piece  $k$ . By continuity of the trail and the valuations, there exists  $\delta > 0$  such that  $i$  and  $i'$  strictly prefer piece  $k$  in the division  $\gamma_{k,k'}(\alpha')$  for all  $\alpha' \in [\alpha, \alpha + \delta]$ . In particular, agents  $i, i'$  strictly prefer piece  $k$  over piece  $k'$ , which implies that  $x_M(\alpha') < x(\alpha')$  and thus  $x_M(\alpha') < x_L(\alpha')$  on the trail  $\gamma_{k,k'}(\alpha')$  for all  $\alpha' \in [\alpha, \alpha + \delta]$ . As a result,  $\gamma_{k,k'}(\alpha')$  satisfies Condition A for all  $\alpha' \in [\alpha, \alpha + \delta]$ . In the case where  $k$  is the rightmost piece, we pick an arbitrary piece  $k' < k$  and apply the same arguments to  $\gamma_{k',k}$ . In that case, we obtain  $x_M(\alpha') > x_R(\alpha')$  instead of  $x_M(\alpha') < x_L(\alpha')$ , and the same conclusion follows.

With the Path Extension Property now proved, the Lemma follows.  $\square$

*Remark 1.* In a previous version of this article, the proof of Lemma 4.4 used a different set of trails, which were closer to the intuition provided earlier in this section, but which also made the proof longer and more involved.<sup>8</sup> Nevertheless, these trails had the advantage of being very simple and so we will sometimes use them in subsequent sections. We include their definition below.

For any piece  $k \in \{1, 2, 3, 4\}$ , we define the trail  $\gamma_k^A$  that maps any  $\alpha \in [\alpha_4^-, \alpha_3^-]$  to the unique division  $(P_1, P_2, P_3, P_4)$  that satisfies  $v_1(P_t) = \alpha$  for all  $t \in \{1, 2, 3, 4\} \setminus \{k\}$ .

For any agent,  $i \in \{2, 3, 4\}$  and any two distinct pieces  $k, k' \in \{1, 2, 3, 4\}$ , we define the trail  $\gamma_{i,k,k'}^B$  that maps any  $\alpha \in [\alpha_4^-, \alpha_2^-]$  to the unique division  $(P_1, P_2, P_3, P_4)$  that satisfies  $v_i(P_k) = v_i(P_{k'})$ , and  $v_1(P_t) = \alpha$  for all  $t \in \{1, 2, 3, 4\} \setminus \{k, k'\}$ .

It is easy to check that the trails used in the proof are a refinement of these trails, namely they always follow one of these trails and will sometimes switch from following one trail to another.

## 5 An Efficient Algorithm for Four Monotone Agents

In this section, we prove the following result.

**THEOREM 5.1 (EFFICIENT ALGORITHM FOR FOUR MONOTONE AGENTS).** *For four agents with monotone 1-Lipschitz valuations, we can compute an  $\varepsilon$ -envy-free connected allocation using  $O(\log^3(1/\varepsilon))$  value queries.*

The crucial observation that allows us to obtain an efficient algorithm is that we can perform binary search on the continuous path whose existence is guaranteed by Lemma 4.4. It is straightforward to check that the algorithm also yields a polynomial-time algorithm in the standard Turing machine model.

---

<sup>8</sup>We thank an anonymous reviewer for suggesting the trails that are now used in the proof.

## 5.1 Preprocessing

In this section, we show that, without loss of generality, we can assume that:

- $\varepsilon$  is the inverse of an integer, i.e.,  $\varepsilon = 1/m$  for some integer  $m \geq 2$ .
- The valuations  $v_i$  are  $\varepsilon$ -strongly-hungry, meaning that if we extend an interval by some length  $t$ , then its value increases by at least  $t\varepsilon$ .
- The valuations  $v_i$  are piecewise linear on the  $\varepsilon$ -grid  $\{0, \varepsilon, 2\varepsilon, \dots, 1 - \varepsilon, 1\}$ , as defined below.

Thus, in the next section it will suffice to provide an algorithm that finds an  $\varepsilon$ -envy-free allocation for 1-Lipschitz valuations that are  $\varepsilon$ -strongly-hungry and piecewise linear on the  $\varepsilon$ -grid, whenever  $\varepsilon$  is the inverse of an integer.

*Definition 3.* Let  $m \in \mathbb{N}$  and  $x_i := i \cdot \delta$  for all  $i \in \{0, 1, \dots, m\}$ . A valuation function  $v$  is *piecewise linear on the  $1/m$ -grid* if it is continuous over its entire domain and (affine) linear in each of the triangles of the following triangulation of  $\{(a, b) \in [0, 1]^2 : a \leq b\}$ :

$$\begin{aligned} T_{i,j}^{\leq} &:= \{(a, b) : x_{i-1} \leq a \leq x_i, x_{j-1} \leq b \leq x_j, a - x_{i-1} \leq b - x_{j-1}\} \quad \forall i, j \in [m] \text{ with } i \leq j, \\ T_{i,j}^{\geq} &:= \{(a, b) : x_{i-1} \leq a \leq x_i, x_{j-1} \leq b \leq x_j, a - x_{i-1} \geq b - x_{j-1}\} \quad \forall i, j \in [m] \text{ with } i < j. \end{aligned}$$

The piecewise linear structure on the  $\varepsilon$ -grid allows us to answer various types of queries exactly. For example, we can simulate exact cut queries: given a position  $a$  on the cake, and a desired value  $\alpha$ , we can find the unique position  $b$  such that  $v(a, b) = \alpha$  (or output that it does not exist). To do this, we perform binary search on the  $\varepsilon$ -grid for  $O(\log(1/\varepsilon))$  steps to determine the interval on the  $\varepsilon$ -grid in which the cut  $b$  must lie. Then, using a constant number of queries we obtain full information about  $v(a, b')$  for any  $b'$  in that interval, and can thus, determine the exact value of  $b$ . Similarly, given two cuts  $a, c$ , we can determine the exact position of cut  $b \in [a, c]$  such that  $v(a, b) = v(c, b)$ . More generally, if we are looking for multiple cuts satisfying some property, it suffices to determine for each cut the interval on the  $\varepsilon$ -grid in which it must lie, and then with a constant number of queries we can determine the exact positions of the cuts. We will make extensive use of this in the algorithm, which is described in the next section.

The next lemma is the main tool used to show that the restrictions are without loss of generality. A simpler version of this construction was used by Brânzei and Nisan [13] for additive valuation functions.

**LEMMA 5.2.** *Let  $m \in \mathbb{N}$  and let  $v$  be a monotone 1-Lipschitz valuation function. Then there exists a monotone 1-Lipschitz valuation function  $\tilde{v}$  that satisfies:*

- (1)  $\tilde{v}$  is a  $3/m$ -approximation of  $v$ , i.e.,  $|v(a, b) - \tilde{v}(a, b)| \leq 3/m$  for all  $0 \leq a \leq b \leq 1$ .
- (2)  $\tilde{v}$  is  $1/m$ -strongly-hungry, i.e.,  $\tilde{v}(a', b') \geq \tilde{v}(a, b) + (b' - b)/m + (a - a')/m$  for all  $a, b, a', b' \in [0, 1]$  with  $a' \leq a \leq b \leq b'$ .
- (3) Any value query to  $\tilde{v}$  can be answered by making a constant number of value queries to  $v$ .
- (4)  $v$  is piecewise linear on the  $1/m$ -grid.

**PROOF.** First, define  $v'$  by letting  $v'(a, b) = v(a, b) \cdot (m - 1)/m + |b - a|/m \in [0, 1]$ . Note that  $v'$  is 1-Lipschitz-continuous and  $1/m$ -strongly-hungry. Furthermore,  $v'$  is a  $1/m$ -approximation of  $v$  and any value query to  $v'$  can be answered by making a single value query to  $v$ .

Next, define  $\tilde{v}$  to be the piecewise linear interpolation of  $v'$  on the  $1/m$ -grid. In other words, the valuation  $\tilde{v}$  is piecewise linear on the  $1/m$ -grid and agrees with  $v'$  at the grid points, i.e.,  $\tilde{v}(i/m, j/m) = v'(i/m, j/m)$  for all  $i, j \in \{0, 1, \dots, m\}$  with  $i \leq j$ .

More formally, for any  $a, b \in [0, 1]$  with  $a \leq b$ , let  $\underline{a}, \bar{a}$  be consecutive multiples of  $1/m$  such that  $\underline{a} \leq a \leq \bar{a}$ , and let  $\underline{b}, \bar{b}$  be consecutive multiples of  $1/m$  such that  $\underline{b} \leq b \leq \bar{b}$ . Then, when  $b - \underline{b} \geq a - \underline{a}$ ,

we let

$$\tilde{v}(a, b) = \frac{a - \underline{a}}{1/m} \cdot v'(\bar{a}, \bar{b}) + \frac{\bar{b} - b}{1/m} \cdot v'(\underline{a}, \underline{b}) + \frac{(b - \underline{b}) - (a - \underline{a})}{1/m} \cdot v'(\underline{a}, \bar{b})$$

and, when  $b - \underline{b} \leq a - \underline{a}$ , we let

$$\tilde{v}(a, b) = \frac{\bar{a} - a}{1/m} \cdot v'(\underline{a}, \underline{b}) + \frac{b - \underline{b}}{1/m} \cdot v'(\bar{a}, \bar{b}) + \frac{(a - \underline{a}) - (b - \underline{b})}{1/m} \cdot v'(\bar{a}, \underline{b}).$$

Clearly, a value query to  $\tilde{v}$  can be answered by making three value queries to  $v'$ , since it suffices to query the three vertices of the triangle containing the queried point  $(a, b)$ . Furthermore, since  $v'$  is 1-Lipschitz-continuous, so is  $\tilde{v}$ . In particular, it follows that  $\tilde{v}$  is a  $2/m$ -approximation of  $v'$ , and thus a  $3/m$ -approximation of  $v$ .

It remains to argue that  $\tilde{v}$  is  $1/m$ -strongly-hungry. In other words, we want to show that for any  $0 \leq a' \leq a \leq b \leq b' \leq 1$ , we have  $\tilde{v}(a', b') \geq \tilde{v}(a, b) + (b' - b)/m + (a - a')/m$ . It suffices to prove that this is the case whenever  $(a, b)$  and  $(a', b')$  lie in the same triangle of the triangulation. Indeed, if the condition holds within each triangle, then it also holds globally. This is easy to see by considering the straight path connecting  $(a, b)$  to  $(a', b')$  and using the fact that the statement holds for each portion of the path that lies within a triangle.

If  $(a, b)$  and  $(a', b')$  lie in the same triangle, then there exist consecutive multiples  $\underline{a}, \bar{a}$  of  $1/m$  such that  $a, a' \in [\underline{a}, \bar{a}]$ . Similarly, there exist consecutive multiples  $\underline{b}, \bar{b}$  of  $1/m$  such that  $b, b' \in [\underline{b}, \bar{b}]$ . Consider first the case where  $b - \underline{b} \geq a - \underline{a}$ . Since  $(a, b)$  and  $(a', b')$  lie in the same triangle, we must also have  $b' - \underline{b} \geq a' - \underline{a}$ . We can thus write

$$\begin{aligned} \tilde{v}(a', b') - \tilde{v}(a, b) &= \frac{a' - a}{1/m} \cdot v'(\bar{a}, \bar{b}) + \frac{b - b'}{1/m} \cdot v'(\underline{a}, \underline{b}) + \frac{(b' - b) - (a' - a)}{1/m} \cdot v'(\underline{a}, \bar{b}) \\ &= (b' - b) \cdot m \cdot (v'(\underline{a}, \bar{b}) - v'(\underline{a}, \underline{b})) + (a - a') \cdot m \cdot (v'(\underline{a}, \bar{b}) - v'(\bar{a}, \bar{b})) \\ &\geq (b' - b)/m + (a - a')/m \end{aligned}$$

where we used the fact that  $v'$  is  $1/m$ -strongly-hungry, and thus,  $v'(\underline{a}, \bar{b}) - v'(\underline{a}, \underline{b}) \geq (\bar{b} - \underline{b})/m = 1/m^2$  and  $v'(\underline{a}, \bar{b}) - v'(\bar{a}, \bar{b}) \geq (\bar{a} - \underline{a})/m = 1/m^2$ . The case where  $b - \underline{b} \leq a - \underline{a}$  (and thus  $b' - \underline{b} \leq a' - \underline{a}$ ) is handled analogously.  $\square$

Now assume that we have an algorithm that solves the problem using  $O(\log^3(1/\varepsilon))$  queries under the restrictions mentioned above. We show how to obtain an algorithm for the problem without these restrictions. Let  $\varepsilon \in (0, 1)$  be given and let  $m \in \mathbb{N}$  be the smallest integer such that  $\varepsilon' := 1/m$  satisfies  $\varepsilon' \leq \varepsilon/7$ . Note that  $\varepsilon' \geq \varepsilon/8$ . By Lemma 5.2, we can run the algorithm on the modified valuations  $\tilde{v}_i$  and obtain an allocation that is  $\varepsilon'$ -envy-free with respect to the modified valuations. Since  $\tilde{v}_i$  is a  $3\varepsilon'$ -approximation of  $v_i$ , the allocation is  $7\varepsilon'$ -envy-free with respect to the original valuations, i.e.,  $\varepsilon$ -envy-free. The algorithm made  $O(\log^3(1/\varepsilon'))$  queries to the modified valuations, which corresponds to  $O(\log^3(8/\varepsilon))$  queries to the original valuations by Lemma 5.2.

## 5.2 The Algorithm

Consider four agents with valuation functions  $v_1, v_2, v_3, v_4$  that are monotone and 1-Lipschitz continuous. In this section, we present an algorithm that computes an  $\varepsilon$ -envy-free allocation using  $O(\log^3(1/\varepsilon))$  value queries. Using the preprocessing construction presented in the previous section, we also assume that the valuations are  $\varepsilon$ -strongly-hungry and piecewise linear on the  $\varepsilon$ -grid.

**Critical divisions.** We briefly recall the following definition. A division of the cake into four pieces is *critical* if at least one of the following two conditions holds:

**Condition A:** Agent 1 is indifferent between its three favorite pieces, and the remaining piece is (weakly) preferred by (at least) two of the three other agents.

**Condition B:** Agent 1 is indifferent between its two favorite pieces, and the two remaining pieces are each (weakly) preferred by (at least) two of the three other agents.

We say that “Condition A (resp. B) holds at value  $\alpha$ ” if there exists a division of the cake for which Condition A (resp. B) holds, and in which Agent 1 has value  $\alpha$  for its favorite pieces. Similarly, we say that “there exists a critical division at value  $\alpha$ ”, if there exists a critical division where Agent 1 has value  $\alpha$  for its favorite pieces. For a more formal definition of the two conditions, see the previous section.

### The Algorithm.

- (1) Compute the (unique) equipartition of the cake into four equal parts according to Agent 1. If this equipartition yields an envy-free division, then return it. Otherwise, let  $\underline{\alpha}_4^{\pm} \in (0, 1)$  be the value that Agent 1 has for each of the equal parts. Set  $\underline{\alpha} := \underline{\alpha}_4^{\pm}$  and  $\bar{\alpha} := v_1(0, 1) \leq 1$ .
- (2) Repeat until  $|\bar{\alpha} - \underline{\alpha}| \leq \varepsilon^4/12$ :
  - (a) Let  $\alpha := (\underline{\alpha} + \bar{\alpha})/2$ .
  - (b) If there exists a critical division at value  $\alpha$ , then set  $\underline{\alpha} := \alpha$ . Otherwise, set  $\bar{\alpha} := \alpha$ .
- (3) Return a critical division at value  $\underline{\alpha}$ .

We begin by explaining how each step can be implemented efficiently using value queries. Then, we argue about the correctness of the algorithm.

### 5.3 Implementation Using Value Queries

LEMMA 5.3. *The algorithm can be implemented using  $O(\log^3(1/\varepsilon))$  value queries.*

PROOF. The loop in Step 2 runs for  $\log(12/\varepsilon^4) = O(\log(1/\varepsilon))$  iterations. Thus, in order to prove the lemma, it suffices to prove that:

- (I) We can find the (unique) equipartition of the cake into four equal parts according to Agent 1 using  $O(\log^2(1/\varepsilon))$  value queries.
- (II) Given  $\alpha \in [0, 1]$ , we can check whether there exists a critical division at value  $\alpha$  using  $O(\log^2(1/\varepsilon))$  value queries, and, if so, output such a critical division.

Our proof crucially uses the fact that cuts satisfying various properties are unique, because the valuation functions are hungry, and the fact that they can be found by binary search due to the monotonicity of the valuations and the preprocessing.

**Proof of I.** For any position of the leftmost cut  $\ell$ , we can determine using  $O(\log(1/\varepsilon))$  value queries the exact position of the cut  $m$ , and then  $r$ , such that the three leftmost pieces have identical value for Agent 1. With one additional value query we can then check if the rightmost piece has a higher or a lower value for Agent 1 than the other three pieces. By the monotonicity of the valuation function, this comparison is monotone with respect to the position of cut  $\ell$ . As a result, after at most  $\log(1/\varepsilon)$  steps of binary search on the  $\varepsilon$ -grid (where each step requires  $O(\log(1/\varepsilon))$  value queries), we can determine in which interval of the  $\varepsilon$ -grid the cut  $\ell$  must lie in the equipartition. A symmetric approach also allows us to determine the interval of the  $\varepsilon$ -grid in which the cut  $r$  must lie in the equipartition.

For any position  $m$  of the middle cut, we can determine using  $O(\log(1/\varepsilon))$  value queries the exact position of the cut  $\ell$  such that the two leftmost pieces have identical value for Agent 1. Similarly, we can also find the exact position of the cut  $r$  such that the two rightmost pieces have identical value of Agent 1. We can then check if the rightmost piece has a higher or lower value for Agent 1 than the leftmost piece. By the monotonicity of the valuation function, this comparison is monotone

with respect to the position of cut  $m$ . Thus, we can perform binary search to locate the interval on the  $\varepsilon$ -grid in which the cut  $m$  must lie in the equipartition.

Once we have determined the interval on the  $\varepsilon$ -grid in which each cut must lie, we can determine the exact positions of the cuts using a constant number of value queries.

**Proof of II.** Given a value  $\alpha \in [0, 1]$ , we can check whether Condition A or Condition B holds at value  $\alpha$  as follows.

**Condition A.** For each of the four possible choices for the identity of the piece  $k$  that is not (necessarily) favored by Agent 1, we can determine the exact (and unique) positions of the cuts that ensure that Agent 1 has value exactly  $\alpha$  for all pieces except (possibly)  $k$ . For example, if piece  $k$  is the rightmost piece, then we first determine the position of  $\ell$  such that  $v_1(0, \ell) = \alpha$ , then the position of  $m$  such that  $v_1(\ell, m) = \alpha$ , and finally, the position of  $r$  such that  $v_1(m, r) = \alpha$ . Depending on the identity of piece  $k$ , we might have to compute the cuts in some other order, but the cuts will always be unique and can be found using  $O(\log(1/\varepsilon))$  value queries.

Once we have determined the positions of the cuts, it then suffices to check that Agent 1 has value at most  $\alpha$  for piece  $k$ , and that at least two of the other agents weakly prefer piece  $k$ . If this check succeeds for at least one choice of piece  $k$ , then Condition A holds at value  $\alpha$  and we can output a division that satisfies it. If the check fails for all four possible choices of piece  $k$ , then Condition A does not hold at value  $\alpha$ .

**Condition B.** We proceed similarly to the procedure for Condition A above. We consider all possible choices for the identity of the two pieces  $k, k'$  not (necessarily) favored by Agent 1, and all possible choices for the identity of the agent  $i \in \{2, 3, 4\}$  who is indifferent between pieces  $k$  and  $k'$ . Indeed, observe that if Condition B holds, then one of the three players 2,3,4 must be indifferent between pieces  $k$  and  $k'$ .

Once we have fixed the identities of pieces  $k, k'$  and agent  $i$ , the positions of the cuts are uniquely determined and, as we show below, can be found using at most  $O(\log^2(1/\varepsilon))$  value queries. We can then check if the resulting division satisfies Condition B or not.

The unique positions of the cuts can be found as follows. There are three cases:

- The pieces  $k$  and  $k'$  are adjacent. Consider the representative example where pieces  $k$  and  $k'$  are the two middle pieces. The other instantiations of this case are handled analogously. Using  $O(\log(1/\varepsilon))$  value queries, we can determine the positions of cuts  $\ell$  and  $r$  such that  $v_1(0, \ell) = \alpha$  and  $v_1(r, 1) = \alpha$ . Then, using  $O(\log(1/\varepsilon))$  value queries, we can find the cut  $m$  that ensures that  $v_i(\ell, m) = v_i(m, r)$ .
- There is one piece between pieces  $k$  and  $k'$ . Consider the representative example where piece  $k$  is the leftmost piece  $[0, \ell]$  and piece  $k'$  is the third piece from the left  $[m, r]$ . Using  $O(\log(1/\varepsilon))$  value queries we first determine the exact position of cut  $r$  such that  $v_1(r, 1) = \alpha$ . For each possible position of cut  $\ell$ , let  $m(\ell)$  denote the unique position of the middle cut that satisfies  $v_1(\ell, m(\ell)) = \alpha$ . Note that, given  $\ell$ , we can find  $m(\ell)$  using  $O(\log(1/\varepsilon))$  value queries. The crucial observation is that as we move  $\ell$  to the right,  $m(\ell)$  must also move to the right, due to the monotonicity of Agent 1. As a result, when we move  $\ell$  to the right, piece  $[0, \ell]$  becomes larger (i.e., a superset of its previous state), and piece  $[\ell, m(\ell)]$  becomes smaller (i.e., a subset of its previous state). But this means that we can use binary search on the  $\varepsilon$ -grid to find the interval on the  $\varepsilon$ -grid which contains the cut  $\ell$  that satisfies  $v_i(\ell, m(\ell)) = \alpha$ . This uses  $O(\log^2(1/\varepsilon))$  queries, since we perform binary search for  $O(\log(1/\varepsilon))$  steps and each step requires the computation of  $m(\ell)$  given  $\ell$ , which takes  $O(\log(1/\varepsilon))$  value queries. By an analogous argument, namely, by letting  $\ell(m)$  be a function of  $m$  such that  $v_1(\ell(m), m) = \alpha$ , we can also determine the interval on the  $\varepsilon$ -grid which must contain cut  $m$ , again using

$O(\log^2(1/\varepsilon))$  value queries. Finally, we now have all the information needed to find the exact positions of cuts  $\ell$  and  $m$ .

- Piece  $k$  is the leftmost piece  $[0, \ell]$  and piece  $k'$  is the rightmost piece  $[r, 1]$ . This case is handled analogously to the previous one. Given cut  $\ell$ , using  $O(\log(1/\varepsilon))$  value queries, we can determine cuts  $m(\ell)$  and  $r(\ell)$  such that  $v_1(\ell, m(\ell)) = \alpha$  and  $v_1(m(\ell), r(\ell)) = \alpha$ . As a result, by monotonicity we can use  $O(\log^2(1/\varepsilon))$  value queries to determine the interval on the  $\varepsilon$ -grid in which cut  $\ell$  must lie to satisfy  $v_1(0, \ell) = v_1(r(\ell), 1)$ . Similarly, we can determine the interval in which  $r$  must lie. Finally, given a position for cut  $m$ , we can find cuts  $\ell(m)$  and  $r(m)$  such that  $v_1(\ell(m), m) = \alpha$  and  $v_1(m, r(m)) = \alpha$  using  $O(\log(1/\varepsilon))$  value queries. Again, using  $O(\log^2(1/\varepsilon))$  value queries, we can perform binary search to find the interval on the  $\varepsilon$ -grid that must contain cut  $m$ . Then, we have all the information that is needed to find the exact positions of the cuts.

This completes the proof of the lemma.  $\square$

#### 5.4 Proof of Correctness

In this section, we argue about the correctness of the algorithm. First of all, note that if the equipartition computed in Step 1 of the algorithm is an envy-free division, then the algorithm's output is correct. If it is not an envy-free division, then Condition A or B must hold, and thus it is a critical division at value  $\alpha_4^\pm$ . Furthermore, it is easy to see that there is no critical division at value  $v_1(0, 1)$ , since the valuations are hungry. As a result, the binary search procedure performed in Step 2 must return  $\underline{\alpha}$  and  $\bar{\alpha}$  such that: (i) there exists a critical division at value  $\underline{\alpha}$ , and (ii) there is no critical division at value  $\bar{\alpha}$ . Furthermore, we have  $\underline{\alpha} < \bar{\alpha}$  and  $|\bar{\alpha} - \underline{\alpha}| \leq \varepsilon^4/12$ .

The algorithm returns a critical division  $(\ell_0, m_0, r_0)$  where Agent 1 has value  $\underline{\alpha}$  for its favorite pieces. Since this division is critical, we can apply Lemma 4.4 to obtain a continuous path  $T$  that starts at the division and ends at an envy-free division  $T(\alpha^*)$ . Using the fact that all divisions on the path are critical, and that there is no critical division at value  $\bar{\alpha}$ , it must be that  $\alpha^* < \bar{\alpha}$ . As a result,  $|\alpha^* - \underline{\alpha}| \leq \varepsilon^4/12$ . Finally, as we show below, the path  $T$  is  $6/\varepsilon^3$ -Lipschitz-continuous, and thus the division  $(\ell_0, m_0, r_0)$  is  $\varepsilon/2$ -close to the division  $T(\alpha^*)$ , which is envy-free. As a result, given that the valuations are 1-Lipschitz-continuous, the division  $(\ell_0, m_0, r_0)$  must be  $\varepsilon$ -envy-free.

The Lipschitz-continuity of the continuous path  $T$  follows from the following lemma. Recall that the path follows trails of type  $\gamma_k^A$  and  $\gamma_{i,k,k'}^B$  (see Remark 1).

**LEMMA 5.4.** *The trails  $\gamma_k^A$  and  $\gamma_{i,k,k'}^B$  are Lipschitz-continuous (w.r.t. the  $\ell_1$ -norm) with Lipschitz constant  $6/\varepsilon^3$ .*

**PROOF.** We consider the two types of trails separately. In each case, we show that for any  $\alpha$  and infinitesimal  $d\alpha$ , the cuts in division  $\gamma(\alpha + d\alpha)$  are  $6d\alpha/\varepsilon^3$ -close to the cuts in division  $\gamma(\alpha)$ . Since the valuations are piecewise-linear, this is sufficient to deduce the statement of the lemma.

**Trail of type A.** Consider the division  $(\ell, m, r) := \gamma_k^A(\alpha)$ , i.e., Agent 1 has value  $\alpha$  for all pieces except piece  $k$ . In going from  $\gamma_k^A(\alpha)$  to  $\gamma_k^A(\alpha + d\alpha)$  the cuts move continuously as follows.

- If piece  $k$  is the rightmost (i.e.,  $[r, 1]$ ), the leftmost cut  $\ell$  moves to the right by  $d\ell$  such that  $v_1(0, \ell + d\ell) = \alpha + d\alpha$ . Simultaneously, the other two cuts move to the right, namely  $m$  by  $dm$  and  $r$  by  $dr$ , such that  $v_1(\ell + d\ell, m + dm) = \alpha + d\alpha$  and  $v_1(m + dm, r + dr) = \alpha + d\alpha$ . Since  $v_1$  is  $\varepsilon$ -strongly-hungry, we have that  $d\alpha = v_1(0, \ell + d\ell) - v_1(0, \ell) \geq \varepsilon \cdot d\ell$ , which implies  $d\ell \leq d\alpha/\varepsilon$ . Then, since  $v_1$  is also 1-Lipschitz-continuous, we have  $dm = v_1(\ell + d\ell, m + dm) - v_1(\ell, m) \geq \varepsilon \cdot dm - 1 \cdot d\ell$ , and thus  $dm \leq d\alpha/\varepsilon + d\ell/\varepsilon \leq 2d\alpha/\varepsilon^2$ . Similarly, we also obtain that  $dr \leq 3d\alpha/\varepsilon^3$ . Thus, the cuts collectively move by at most  $6d\alpha/\varepsilon^3$ . The case where piece  $k$  is  $[0, \ell]$  is symmetric.

- If piece  $k$  is the second from the right (i.e.,  $[m, r]$ ), the cuts  $\ell$  and  $m$  move as before such that  $v_1(0, \ell + d\ell) = \alpha + d\alpha$  and  $v_1(\ell + d\ell, m + dm) = \alpha + d\alpha$ . Simultaneously, cut  $r$  moves to the left such that  $v_1(r - dr, 1) = \alpha + d\alpha$ . Using the same arguments as above, the total movement of the cuts can be bounded by  $4d\alpha/\varepsilon^2$ . The case where piece  $k$  is  $[\ell, m]$  is symmetric.

**Trail of type B.** Consider the division  $(\ell, m, r) := \gamma_{i,k,k'}^B(\alpha)$ , i.e., Agent 1 has value  $\alpha$  for the pieces  $p$  and  $q$  ( $\{p, q\} = \{1, 2, 3, 4\} \setminus \{k, k'\}$ ), and Agent  $i$  is indifferent between pieces  $k$  and  $k'$ . In going from  $\gamma_{i,k,k'}^B(\alpha)$  to  $\gamma_{i,k,k'}^B(\alpha + d\alpha)$  the cuts move continuously as follows.

- If pieces  $p$  and  $q$  are the two leftmost pieces ( $[0, \ell]$  and  $[\ell, m]$ ),  $\ell$  and  $m$  move to the right, such that  $v_1(0, \ell + d\ell) = \alpha + d\alpha$  and  $v_1(\ell + d\ell, m + dm) = \alpha + d\alpha$ . Simultaneously,  $r$  moves to the right such that  $v_i(m + dm, r + dr) = v_i(r + dr, 1)$ . Note that  $r$  indeed needs to be shifted to the right (and not to the left) because of the monotonicity of  $v_i$ . As above, we have  $d\ell \leq d\alpha/\varepsilon$  and  $dm \leq 2d\alpha/\varepsilon^2$ . Furthermore, since the third piece's value decreased by at most  $1 \cdot dm$  for Agent  $i$  (before moving cut  $r$ ), it follows that cut  $r$  must move by at most  $dm/\varepsilon$ , i.e.,  $dr \leq dm/\varepsilon \leq 2d\alpha/\varepsilon^3$ . Thus, the cuts collectively move by at most  $5d\alpha/\varepsilon^3$ . The case where pieces  $p$  and  $q$  are the two rightmost pieces ( $[m, r]$  and  $[r, 1]$ ) is symmetric.
- If pieces  $p$  and  $q$  are  $[0, \ell]$  and  $[r, 1]$ , cut  $\ell$  moves to the right, and cut  $r$  to the left, while maintaining Agent 1's indifference, i.e., such that  $v_1(0, \ell + d\ell) = \alpha + d\alpha$  and  $v_1(r - dr, 1) = \alpha + d\alpha$ . Cut  $m$  moves to the unique position that makes Agent  $i$  indifferent between the two middle pieces, i.e., such that  $v_i(\ell + d\ell, m \pm dm) = v_i(m \pm dm, r + dr)$ . Note that, depending on the situation,  $m$  might have to move to the left or to the right to satisfy the indifference condition. For this reason, we denote the shifted cut by  $m \pm dm$ . Using similar arguments to above, we obtain that  $d\ell \leq d\alpha/\varepsilon$ ,  $dr \leq d\alpha/\varepsilon$ , and  $|dm| \leq d\alpha/\varepsilon^2$ , which yields that the total movement of cuts is at most  $4d\alpha/\varepsilon^2$ .
- If pieces  $p$  and  $q$  are  $[0, \ell]$  and  $[m, r]$ , cut  $\ell$  moves to the right such that  $v_1(0, \ell + d\ell) = \alpha + d\alpha$ . Then, cuts  $m$  and  $r$  simultaneously move such that  $v_1(m \pm dm, r + dr) = \alpha + d\alpha$  and  $v_i(\ell + d\ell, m \pm dm) = v_i(m \pm dm, r + dr)$ . Note that, depending on the situation,  $m$  might need to move left or right, whereas  $r$  will have to move to the right by monotonicity of the valuations. Using similar arguments to above, we can show that  $d\ell \leq d\alpha/\varepsilon$ ,  $dm \geq -d\alpha/\varepsilon$ ,  $dm \leq d\alpha/\varepsilon^2$ , and  $dr \leq 2d\alpha/\varepsilon^3$ . The total movement of the cuts is thus at most  $4d\alpha/\varepsilon^3$ . The case where pieces  $p$  and  $q$  are  $[\ell, m]$  and  $[r, 1]$  is symmetric.
- If pieces  $p$  and  $q$  are  $[\ell, m]$  and  $[m, r]$ , then  $\ell$  moves to the left and  $r$  moves to the right such that Agent  $i$  remains indifferent between the leftmost and rightmost piece, i.e., such that  $v_i(0, \ell - d\ell) = v_i(r + dr, 1)$ . This specifies  $dr$  as a function of  $d\ell$ . Next, the middle cut  $m$  is shifted such that Agent 1 remains indifferent between the two middle pieces, i.e., such that  $v_1(\ell - d\ell, m \pm dm) = v_1(m \pm dm, r + dr)$ . This also specifies  $dm$  as a function of  $d\ell$ . Finally, note that the value of Agent 1 for the two middle pieces strictly increases as  $\ell$  is moved left (and the other cuts follow according to the above). We then also require that  $v_1(\ell - d\ell, m \pm dm) = \alpha + d\alpha$ , and this specifies the shifts of all cuts as functions of  $d\alpha$ . Using similar arguments to above, we can bound  $|dm| \leq d\alpha/\varepsilon$ , and then  $d\ell \leq 2d\alpha/\varepsilon^2$  and  $dr \leq 2d\alpha/\varepsilon^2$ . Thus, the total movement of the cuts is at most  $5d\alpha/\varepsilon^2$ .

This completes the proof of the lemma. □

## 6 An Efficient Algorithm in the Robertson–Webb Model

In this section, we show how the algorithm can also be implemented in the Robertson–Webb cake-cutting model.

**Robertson–Webb query model.** In the Robertson–Webb cake-cutting model [43], the valuations are assumed to be additive, but not necessarily Lipschitz-continuous (and even if they are  $L$ -Lipschitz-continuous, we are not given any bound on  $L$ ). Formally, for each valuation function  $v_i : [0, 1]^2 \rightarrow [0, 1]$  there exists a non-atomic integrable density function  $f_i : [0, 1] \rightarrow [0, +\infty)$  such that for all  $0 \leq a \leq b \leq 1$  we have

$$v_i(a, b) = \int_a^b f_i(x) dx.$$

Note that  $v_i$  is indeed continuous because  $f_i$  is non-atomic. It is also monotone. Furthermore, we assume that  $v_i(0, 1) = 1$ . It is easy to see that the lack of a bound on the Lipschitzness of  $v_i$  (or equivalently the lack of a bound on the density  $f_i$ ) implies that we cannot hope to solve the problem (even for two agents) using only value queries. For this reason, the Robertson–Webb model also introduces a second type of query: *cut* queries. On cut query  $(x, \alpha)$ , where  $x$  is a position on the cake and  $\alpha$  is a value, the agent responds by returning a position  $y$  on the cake such that  $v_i(x, y) = \alpha$  (or responds that there is no such  $y$ ). Due to the additivity, it is sufficient to only allow cut queries with  $x = 0$ . It is also easy to see that we can simulate reverse cut queries, i.e., given  $(y, \alpha)$ , return  $x$  such that  $v_i(x, y) = \alpha$  (or respond that there is no such  $x$ ).

In this section, we prove the following result.

**THEOREM 6.1.** *For four agents with additive valuations, we can compute an  $\epsilon$ -envy-free connected allocation using  $O(\log^2(1/\epsilon))$  value and cut queries in the Robertson–Webb model.*

## 6.1 Preprocessing

We begin with some useful “preprocessing” which will allow us to assume that the valuations have some additional structure.

**LEMMA 6.2.** *Let  $m \in \mathbb{N}$  and let  $v$  be an additive continuous valuation function. Then there exists an additive continuous valuation function  $\tilde{v}$  that satisfies:*

- (1)  $\tilde{v}$  is a  $2/m$ -approximation of  $v$ , i.e.,  $|v(a, b) - \tilde{v}(a, b)| \leq 2/m$  for all  $0 \leq a \leq b \leq 1$ .
- (2)  $\tilde{v}$  is  $1/m$ -strongly-hungry, i.e.,  $\tilde{v}(a, b) \geq (b - a)/m$  for all  $0 \leq a \leq b \leq 1$ .
- (3) Any value or cut query to  $\tilde{v}$  can be answered by making a constant number of value and cut queries to  $v$ .
- (4)  $\tilde{v}$  is linear between its  $m$ -quantiles, i.e., letting  $0 = x_0 < x_1 < \dots < x_m = 1$  be the (unique) positions such that for all  $j \in [m]$

$$\tilde{v}(x_{j-1}, x_j) = 1/m$$

we have that for all  $j \in [m]$  and all  $t \in [0, 1]$

$$\tilde{v}(x_{j-1}, x_{j-1} + t(x_j - x_{j-1})) = t/m.$$

**PROOF.** The  $m$ -quantiles of  $v$ , i.e., positions  $0 = x_0 < x_1 < \dots < x_m = 1$  such that for all  $j \in [m]$

$$v(x_{j-1}, x_j) = 1/m$$

are in general not unique, because  $v$  might not be hungry. For any choice  $x_0, x_1, \dots, x_m$  of  $m$ -quantiles of  $v$ , we can define a corresponding valuation function  $\tilde{v}$  as follows.

The valuation function  $\tilde{v}$  is constructed by evening out the mass between the  $m$ -quantiles  $x_0, x_1, \dots, x_m$  of  $v$ . Formally,  $\tilde{v}$  is given by its density function  $\tilde{f}$ , which is defined as

$$\tilde{f}(x) = \frac{1}{m} \sum_{j=1}^m \chi_{[x_{j-1}, x_j]}(x) \cdot \frac{1}{(x_j - x_{j-1})}$$

for all  $x \in [0, 1]$ . Here  $\chi_I : [0, 1] \rightarrow \{0, 1\}$  denotes the characteristic function of interval  $I$ . Clearly,  $\tilde{v}$  is additive and continuous. Furthermore, it immediately follows from the construction that  $\tilde{v}(0, 1) = 1$ , that  $x_0, x_1, \dots, x_m$  are the (unique)  $m$ -quantiles of  $\tilde{v}$ , and that  $\tilde{v}$  is linear between its  $m$ -quantiles. Finally, it is easy to check that  $\tilde{v}$  is  $1/m$ -strongly-hungry and that it is a  $2/m$ -approximation of  $v$ .

It remains to argue that we can simulate queries to  $\tilde{v}$  by using queries to  $v$ . To do this, we will fix the choice of  $m$ -quantiles  $x_0, x_1, \dots, x_m$  of  $v$  as follows:  $x_j$  is defined as the cut position returned when we perform a cut query<sup>9</sup>  $(0, j/m)$  to  $v$ . If the simulation never queries  $(0, j/m)$  for some  $j$ , then all simulated queries to  $\tilde{v}$  will be consistent with any of the possible choices for  $x_j$ .

First, we observe that for any  $j \in [m] \cup \{0\}$  we can obtain  $x_j$  by using at most one cut query to  $v$ . Indeed, if  $j = 0$  or  $j = m$ , then we just use  $x_0 = 0$  or  $x_m = 1$ , respectively. Otherwise, when  $j \in [m - 1]$ , we can use one cut query  $(0, j/m)$  to  $v$  to obtain  $x_j$ . We can now simulate queries to  $\tilde{v}$  as follows:

- cut query  $(0, \alpha)$ : We first find  $j \in [m]$  such that  $(j - 1)/m \leq \alpha \leq j/m$ . Then we compute  $x_{j-1}$  and  $x_j$  as described above and output  $x_{j-1} + (x_j - x_{j-1})(m\alpha - (j - 1))$ .
- value query  $(0, b)$ : We first perform one value query to  $v$  to obtain  $v(0, b) =: \alpha$ . If  $\alpha = 0$ , then we compute  $x_1$  as described above and output  $b/(mx_1)$ . If  $\alpha = 1$ , then we compute  $x_{m-1}$  as described above and output  $1 - (1 - b)/(m(1 - x_{m-1}))$ . If  $\alpha \in (0, 1)$ , then we find  $j \in [m - 1]$  such that  $(j - 1)/m < \alpha < (j + 1)/m$  and compute  $x_j$ . If  $b \geq x_j$ , then we compute  $x_{j+1}$  and output  $j/m + (b - x_j)/(m(x_{j+1} - x_j))$ . If  $b < x_j$ , then we compute  $x_{j-1}$  and output  $j/m - (x_j - b)/(m(x_j - x_{j-1}))$ .

Note that it is sufficient to only consider queries with endpoint 0, since the valuations are additive.  $\square$

In the rest of this section, we will assume that the valuations  $v_i$  have been obtained from the original valuations by applying Lemma 6.2 with  $m := \lceil 4/\epsilon \rceil$ , i.e., we have replaced  $v_i$  by  $\tilde{v}_i$ . Thus, finding an envy-free allocation with respect to these valuations will yield an  $\epsilon$ -envy-free allocation with respect to the original valuations.

We let  $x_0^i, \dots, x_m^i$  denote the (unique)  $m$ -quantiles of valuation  $v_i$ . The following will be used repeatedly in our algorithm.

**LEMMA 6.3.** *Let  $c : [t_1, t_2] \rightarrow [0, 1]$  be continuous, strictly monotone, and let  $c(t_1)$  and  $c(t_2)$  be given. Furthermore, assume that given any  $a \in \{c(t) : t \in [t_1, t_2]\}$  we can find the unique  $t$  such that  $c(t) = a$  using at most  $Q(m)$  queries to the agents' valuations. Finally, let  $s : [t_1, t_2] \rightarrow \mathbb{R}$  be an arbitrary function that can be evaluated using at most  $Q(m)$  queries and that satisfies  $s(t_1) \leq 0$  and  $s(t_2) \geq 0$ . Then, using at most  $O(Q(m) \log m)$  queries, we can compute  $t_1^*, t_2^* \in [t_1, t_2]$  with  $t_1^* < t_2^*$ ,  $s(t_1^*) \leq 0$ ,  $s(t_2^*) \geq 0$ , and such that for each agent  $i \in \{1, 2, 3, 4\}$ , there exist some (possibly different)  $j \in [m]$  such that  $c(t_1^*), c(t_2^*) \in [x_{j-1}^i, x_j^i]$ .*

**PROOF.** We consider the case where  $c$  is strictly increasing. The case where it is strictly decreasing can be handled analogously. We initialize  $t_1^* := t_1$  and  $t_2^* := t_2$ . Using two value queries to Agent 1 we can determine  $j_1 \leq j_2$  such that  $x_{j_1-1}^1 \leq c(t_1^*) \leq x_{j_1}^1$  and  $x_{j_2-1}^1 \leq c(t_2^*) \leq x_{j_2}^1$ . If  $j_1 = j_2$  then we are done with Agent 1. Otherwise, we perform binary search as follows:

As long as  $j_1 < j_2$ :

<sup>9</sup>A somewhat subtle detail is that performing the same cut query  $(0, j/m)$  to  $v$  multiple times could potentially output different results (because the quantiles might not be unique). However, this is not an issue: it suffices for the simulation to memorize the answer the first time it performs the query and then to just reuse this answer for subsequent queries. This will ensure that the  $x_j$ 's are fixed throughout the simulation.

- (1) Let  $j := \lfloor (j_1 + j_2)/2 \rfloor$ .
- (2) Compute  $t$  such that  $c(t) = x_j^1$ .
- (3) If  $s(t) \geq 0$ , then set  $t_2^* := t$  and  $j_2 := j$ . Otherwise, set  $t_1^* := t$  and  $j_1 := j + 1$ .

It is easy to see that this runs for at most  $\log m$  iterations, and each iteration requires  $O(Q(m))$  queries. When it terminates, we have that  $[c(t_1^*), c(t_2^*)] \in [x_{j-1}^1, x_j^1]$ . Repeating this in sequence for the other three agents yields the desired outcome. In particular, note that each binary search procedure can only make  $[c(t_1^*), c(t_2^*)]$  smaller (w.r.t. inclusion order), so the property will hold simultaneously for all agents at the end.  $\square$

## 6.2 Algorithm

We recall some notation that was introduced in the proof of Lemma 4.4 and in Remark 1. We let  $\alpha_2^\pm, \alpha_3^\pm, \alpha_4^\pm$  denote the value of the pieces in the equipartition according to Agent 1 into two, three, and four parts, respectively. Since we consider the additive setting here, we know that  $\alpha_2^\pm = 1/2$ ,  $\alpha_3^\pm = 1/3$ , and  $\alpha_4^\pm = 1/4$ . As mentioned in Remark 1, the continuous path guaranteed by Lemma 4.4 always uses the following two types of trails.

For any piece  $k \in \{1, 2, 3, 4\}$ , we define the trail  $\gamma_k^A$  that maps any  $\alpha \in [\alpha_4^\pm, \alpha_3^\pm]$  to the unique division  $(P_1, P_2, P_3, P_4)$  that satisfies  $v_1(P_t) = \alpha$  for all  $t \in \{1, 2, 3, 4\} \setminus \{k\}$ .

For any agent  $i \in \{2, 3, 4\}$  and any two distinct pieces  $k, k' \in \{1, 2, 3, 4\}$ , we define the trail  $\gamma_{i,k,k'}^B$  that maps any  $\alpha \in [\alpha_4^\pm, \alpha_2^\pm]$  to the unique division  $(P_1, P_2, P_3, P_4)$  that satisfies  $v_i(P_k) = v_i(P_{k'})$ , and  $v_i(P_t) = \alpha$  for all  $t \in \{1, 2, 3, 4\} \setminus \{k, k'\}$ .

Note that any critical division must lie on one of these trails. If it satisfies Condition A then it must lie on a trail of type  $\gamma_k^A$  for some  $k$ . If it satisfies Condition B then it must lie on a trail of type  $\gamma_{i,k,k'}^B$  for some  $i, k, k'$ . We can equivalently restate the definitions of the conditions as follows.

**Condition A:** We say that division  $(P_1, P_2, P_3, P_4)$  satisfies Condition A if there exist  $\alpha \in [\alpha_4^\pm, \alpha_3^\pm]$  and  $k \in \{1, 2, 3, 4\}$  such that  $(P_1, P_2, P_3, P_4) = \gamma_k^A(\alpha)$  and additionally there exist two distinct agents  $i, i' \in \{2, 3, 4\}$  such that  $v_i(P_k) = \max_t v_i(P_t)$  and  $v_{i'}(P_k) = \max_t v_{i'}(P_t)$ . (Note that  $v_1(P_k) \leq \alpha$  automatically holds since  $\alpha \geq \alpha_4^\pm$ .)

**Condition B:** We say that division  $(P_1, P_2, P_3, P_4)$  satisfies Condition B if there exist  $\alpha \in [\alpha_4^\pm, \alpha_2^\pm]$ , an agent  $i \in \{2, 3, 4\}$ , and two distinct pieces  $k, k' \in \{1, 2, 3, 4\}$  such that  $(P_1, P_2, P_3, P_4) = \gamma_{i,k,k'}^B(\alpha)$  and additionally:

- (i)  $v_1(P_k) \leq \alpha$  and  $v_1(P_{k'}) \leq \alpha$ , and
- (ii)  $(v_i(P_{k'}) =) v_i(P_k) = \max_t v_i(P_t)$ , and
- (iii) there exists  $i' \in \{2, 3, 4\} \setminus \{i\}$  such that  $v_{i'}(P_k) = \max_t v_{i'}(P_t)$ , and
- (iv) there exists  $i' \in \{2, 3, 4\} \setminus \{i\}$  with  $v_{i'}(P_{k'}) = \max_t v_{i'}(P_t)$ .

We say that there exists a critical division at value  $\alpha$ , if there exists a critical division where Agent 1 has value  $\alpha$  for its favorite pieces.

Let  $\gamma$  be a trail, and  $[\underline{\alpha}, \bar{\alpha}]$  some interval of values. Let  $(\ell(\alpha), m(\alpha), r(\alpha)) := \gamma(\alpha)$  whenever this is well-defined. We say that for all  $\alpha \in [\underline{\alpha}, \bar{\alpha}]$ , the cuts in  $\gamma(\alpha)$  remain in the same  $m$ -quantile for all agents if for each cut  $c \in \{\ell, m, r\}$ , and each agent  $i \in \{1, 2, 3, 4\}$  there exists  $j \in [m]$  such that for all  $\alpha \in [\underline{\alpha}, \bar{\alpha}]$  we have  $x_{j-1}^i \leq c(\alpha) \leq x_j^i$  (whenever  $c(\alpha)$  is well-defined).

We are now ready to state the algorithm.

- (1) Compute the (unique) equipartition of the cake into four equal parts according to Agent 1. If this equipartition yields an envy-free division, then stop and return this division.
- (2) Let  $\underline{\alpha} = 1/4$  and  $\bar{\alpha} = 1/2$ .
- (3) Using binary search, find  $\underline{\alpha}$  and  $\bar{\alpha}$  such that
  - There exists a critical division at value  $\underline{\alpha}$ , but not at value  $\bar{\alpha}$ .

- For any trail  $\gamma$  and for all  $\alpha \in [\underline{\alpha}, \bar{\alpha}]$ , the cuts in  $\gamma(\alpha)$  remain in the same  $m$ -quantile for all agents.

(4) Using the information gathered, output an envy-free allocation.

Before explaining how the steps of the algorithm are implemented, we briefly argue about the correctness. First of all, note that if the equipartition computed in Step 1 of the algorithm is an envy-free division, then the algorithm's output is correct. If it is not an envy-free division, then Condition A or B must hold at value  $\alpha_4^{\pm} = 1/4$ . Furthermore, it is easy to see that neither Condition A nor Condition B can hold at value  $\alpha_2^{\pm} = 1/2$ . After the binary search procedure, Lemma 4.4 guarantees that there is an envy-free division on one of the trails in the interval  $[\underline{\alpha}, \bar{\alpha}]$ . Since all cuts remain in the same  $m$ -quantile for all agents in all trails, we have full information about all the trails in the interval  $[\underline{\alpha}, \bar{\alpha}]$ . We can thus locate the envy-free division and return it.

### 6.3 Implementation Using Value and Cut Queries

In this section, we argue that all steps of the algorithm can be implemented efficiently using value and cut queries. First of all, computing the equipartition in the first step of the algorithm is straightforward using a constant number of cut queries. Next, we show that we can check whether there exists a critical division at some value  $\alpha$ .

**LEMMA 6.4.** *Given  $\alpha \in [0, 1]$ , we can check whether there exists a critical division at value  $\alpha$  using  $O(\log(1/\varepsilon))$  value and cut queries.*

**PROOF.** We show that we can check whether Condition A or B holds at a given value  $\alpha$ . For Condition A it suffices to show that for any  $k \in \{1, 2, 3, 4\}$  we can compute the division  $y_k^A(\alpha)$ , i.e., the division where Agent 1 has value  $\alpha$  for all pieces except (possibly) piece  $k$ . Indeed, given that division, we can then easily verify the condition by querying the value of each piece for each agent (and this only uses a constant number of value queries). The division  $y_k^A(\alpha)$  can be computed using three cut queries. For example, if piece  $k = 3$ , then we first use one cut query to determine the position  $\ell$  such that  $v_1(0, \ell) = \alpha$ , then another cut query to determine  $m$  such that  $v_1(\ell, m) = \alpha$ , and one final cut query to find  $r$  such that  $v_1(r, 1) = \alpha$ .

Similarly, for Condition B it also suffices to show that for any agent  $i \in \{2, 3, 4\}$  and any distinct pieces  $k, k' \in \{1, 2, 3, 4\}$  we can compute the division  $y_{i,k,k'}^B(\alpha)$ , i.e., the division where Agent  $i$  has identical value for pieces  $k$  and  $k'$ , and where Agent 1 has value  $\alpha$  for each of the remaining two pieces. We distinguish between the following three cases:

**Case 1: The pieces  $k$  and  $k'$  are adjacent.** We consider the representative example where  $k$  and  $k'$  are the two middle pieces. The other cases are handled analogously. Using two cut queries we can determine  $\ell$  and  $r$  such that  $v_1(0, \ell) = \alpha$  and  $v_1(r, 1) = \alpha$ . Then, we use one value query to obtain  $\beta := v_i(\ell, r)$ , and one cut query to obtain  $m$  that satisfies  $v_i(\ell, m) = \beta/2$ . The division  $(\ell, m, r)$  corresponds to  $y_{i,k,k'}^B(\alpha)$ .

**Case 2: There is one piece between pieces  $k$  and  $k'$ .** Consider the representative example where piece  $k$  is the piece  $[\ell, m]$ , i.e., the second piece from the left, and piece  $k'$  is the piece  $[r, 1]$ , i.e., the rightmost piece. Using one cut query we can determine  $\ell$  such that  $v_1(0, \ell) = \alpha$ , and then using one value query we obtain  $\beta := v_i(\ell, 1)$ . For any  $t \in [0, \beta/2]$ , let  $m(t)$  and  $r(t)$  denote the unique cuts that satisfy  $v_i(\ell, m(t)) = t$  and  $v_i(r(t), 1) = t$ . Our goal is to find the unique  $t^*$  that satisfies  $v_1(m(t^*), r(t^*)) = \alpha$ . For this, we can use the binary search approach of Lemma 6.3. Indeed,  $t \mapsto m(t)$  is continuous and strictly increasing, and both  $m(\cdot)$  and its inverse  $m^{-1}(\cdot)$  can be computed using a single query. Similarly,  $t \mapsto r(t)$  is continuous and strictly decreasing, and both  $r(\cdot)$  and its inverse  $r^{-1}(\cdot)$  can be computed using one query. Let  $s : t \mapsto \alpha - v_1(m(t), r(t))$ , which

can be evaluated using three queries, and note that  $s(0) = \alpha - v_1(\ell, 1) \leq 0$  and  $s(\beta/2) = \alpha \geq 0$ . Applying Lemma 6.3 twice, first on  $m(\cdot)$  and then subsequently on  $r(\cdot)$ , we use  $O(\log(1/\varepsilon))$  queries to obtain  $[t_1^*, t_2^*] \subseteq [0, \beta/2]$  such that  $s(t_1^*) \leq 0$ ,  $s(t_2^*) \geq 0$ , and  $[m(t_1^*), m(t_2^*)]$  and  $[r(t_2^*), r(t_1^*)]$  lie within  $\mathbf{m}$ -quantiles for all agents. As a result, with an additional constant number of queries we now have full information about the functions  $m$ ,  $r$ , and thus  $s$  on  $[t_1^*, t_2^*]$ , and we can find  $t^*$  such that  $s(t^*) = 0$ , i.e.,  $v_1(m(t^*), r(t^*)) = \alpha$ . The division  $(\ell, m(t^*), r(t^*))$  then corresponds to  $\gamma_{i,k,k'}^B(\alpha)$ .

**Case 3: Piece  $k$  is the leftmost piece and piece  $k'$  is the rightmost piece.** This case is handled using the same approach as in the previous case. For any  $t \in [0, 1/2]$ , we let  $\ell(t)$  and  $r(t)$  denote the unique cuts that satisfy  $v_i(0, \ell(t)) = t$  and  $v_i(r(t), 1) = t$ . Similar arguments show that we can find  $t^*$  such that  $v_1(\ell(t^*), r(t^*)) = 2\alpha$  using  $O(\log(1/\varepsilon))$  queries. The desired division is then  $(\ell(t^*), m, r(t^*))$  where  $m$  is picked so as to satisfy  $v_1(\ell(t^*), m) = \alpha$ .

This completes the proof of the lemma.  $\square$

Finally, we have to explain how the third step of the algorithm is implemented, namely, the binary search. Starting with  $[\underline{\alpha}, \bar{\alpha}] = [1/4, 1/2]$ , we first consider the trail  $\gamma_1^A$  and using binary search we shrink the search interval  $[\underline{\alpha}, \bar{\alpha}]$  until the cuts in  $\gamma_1^A(\alpha)$  remain in the same  $\mathbf{m}$ -quantile for all agents (we show below how exactly to do this). Next, we consider the trail  $\gamma_2^A$  and further shrink the search interval  $[\underline{\alpha}, \bar{\alpha}]$  until the desired property holds for  $\gamma_2^A$  as well. Importantly, note that the property that we ensured for  $\gamma_1^A$  continues to hold, because we have only made the search interval smaller. We continue like this until all four trails of type A and all 18 trails of type B have been handled. The final search interval thus obtained now satisfies the desired property for all trails simultaneously.

In order to complete the proof it remains to show that for any given trail we can efficiently shrink the interval until the desired property holds. This is proved in the following two lemmas.

**LEMMA 6.5.** *Let  $\underline{\alpha} < \bar{\alpha}$  be such that there is a critical division at  $\underline{\alpha}$  but not at  $\bar{\alpha}$ . Let  $k \in \{1, 2, 3, 4\}$ . Then using  $O(\log^2(1/\varepsilon))$  queries we can find  $\underline{\alpha}^* < \bar{\alpha}^*$  such that  $[\underline{\alpha}^*, \bar{\alpha}^*] \subseteq [\underline{\alpha}, \bar{\alpha}]$ , there is a critical division at  $\underline{\alpha}^*$  but not at  $\bar{\alpha}^*$ , and such that for all  $\alpha \in [\underline{\alpha}^*, \bar{\alpha}^*]$  the cuts  $(\ell(\alpha), m(\alpha), r(\alpha)) = \gamma_k^A(\alpha)$  remain in the same  $\mathbf{m}$ -quantile for all agents.*

**PROOF.** For concreteness let us consider the case where  $k = 3$ . The other cases are handled analogously. For  $\alpha \in [\underline{\alpha}, \bar{\alpha}]$  let  $(\ell(\alpha), m(\alpha), r(\alpha)) := \gamma_k^A(\alpha)$ . Recall that these are the unique cuts satisfying  $v_1(0, \ell(\alpha)) = v_1(\ell(\alpha), m(\alpha)) = v_1(r(\alpha), 1) = \alpha$ . It is easy to check that  $\ell(\cdot)$ ,  $m(\cdot)$ , and  $r(\cdot)$  are continuous, strictly monotone, and that their inverses can be computed using a single query. We define the function  $s : [\underline{\alpha}, \bar{\alpha}] \rightarrow \{-1, +1\}$  as follows:

$$s(\alpha) = \begin{cases} -1 & \text{if there exists a critical division at value } \alpha \\ +1 & \text{if there is no critical division at value } \alpha \end{cases} \quad (1)$$

Note that  $s(\underline{\alpha}) = -1$  and  $s(\bar{\alpha}) = +1$ , and that by Lemma 6.4, we can evaluate  $s$  using  $O(\log(1/\varepsilon))$  queries.

It follows that we can apply the binary search approach of Lemma 6.3 to find  $\underline{\alpha}^*$  and  $\bar{\alpha}^*$  that satisfy the desired properties using  $O(\log^2(1/\varepsilon))$  queries. Namely, we first apply Lemma 6.3 to ensure that  $\ell(\cdot)$  remains in the same  $\mathbf{m}$ -quantile for all agents, then we apply the lemma again to further restrict the interval of  $\alpha$  so that the same holds for  $m(\cdot)$ , and then a third time for  $r(\cdot)$ .  $\square$

**LEMMA 6.6.** *Let  $\underline{\alpha} < \bar{\alpha}$  be such that there is a critical division at  $\underline{\alpha}$  but not at  $\bar{\alpha}$ . Let  $i \in \{2, 3, 4\}$  and  $k, k' \in \{1, 2, 3, 4\}$  with  $k \neq k'$ . Then using  $O(\log^2(1/\varepsilon))$  queries we can find  $\underline{\alpha}^* < \bar{\alpha}^*$  such that*

$[\underline{\alpha}^*, \bar{\alpha}^*] \subseteq [\underline{\alpha}, \bar{\alpha}]$ , there is a critical division at  $\underline{\alpha}^*$  but not at  $\bar{\alpha}^*$ , and such that for all  $\alpha \in [\underline{\alpha}^*, \bar{\alpha}^*]$  the cuts  $(\ell(\alpha), m(\alpha), r(\alpha)) = \gamma_{i,k,k'}^B(\alpha)$  remain in the same  $m$ -quantile for all agents.

PROOF. Our approach will be similar to the proof of Lemma 6.5. Namely, we will seek to apply the binary search procedure of Lemma 6.3 repeatedly to ensure that all cuts remain in the same  $m$ -quantile for all agents. In particular, we will use the same function  $s$  from (1). However, some cuts will no longer be monotone functions of  $\alpha$ , and as a result we will have to argue more carefully. For  $\alpha \in [\underline{\alpha}, \bar{\alpha}]$ , let  $(\ell(\alpha), m(\alpha), r(\alpha)) := \gamma_{i,k,k'}^B(\alpha)$ . We consider three different cases.

**Case 1: The pieces  $k$  and  $k'$  are adjacent.** We focus on the most challenging subcase, namely, when  $k$  and  $k'$  are the two middle pieces. Recall that the cuts satisfy  $v_1(0, \ell(\alpha)) = v_1(r(\alpha), 1) = \alpha$  and  $v_i(\ell(\alpha), m(\alpha)) = v_i(m(\alpha), r(\alpha))$  in this case. As a result, it is easy to see that  $\ell(\cdot)$  is strictly increasing and  $r(\cdot)$  is strictly decreasing, but  $m(\cdot)$  is not necessarily monotone. Nevertheless, we can apply the binary search approach from Lemma 6.3 first to  $\ell(\cdot)$ , and then to  $r(\cdot)$  to ensure that those cuts remain in the same  $m$ -quantile for all agents. Recall that this requires  $O(\log^2(1/\varepsilon))$  queries, because evaluating  $s$  requires  $O(\log(1/\varepsilon))$  queries. In particular, we now have full information about  $\ell(\cdot)$  and  $r(\cdot)$  as functions of  $\alpha$ .

The crucial observation is that  $m(\cdot)$  is now a monotone function in  $\alpha$ . To be more precise,  $m(\cdot)$  is either constant or strictly monotone. The reason for this is that as  $\alpha$  increases, the cuts  $\ell(\alpha)$  and  $r(\alpha)$  traverse the value of Agent  $i$  at some constant rate (because these two cuts do not cross any  $m$ -quantile for any agent). More formally, we have that  $v_i(\ell(\alpha), \ell(\alpha + d\alpha)) = c \cdot d\alpha$ , and  $v_i(r(\alpha + d\alpha), r(\alpha)) = c' \cdot d\alpha$ , where  $c$  and  $c'$  do not depend on  $\alpha$ . If the two rates are the same, i.e.,  $c = c'$ , then  $m(\alpha)$  will not move. Otherwise, if the two rates are different, then  $m(\alpha)$  will move left or right depending on which rate is larger.

Now, if  $m(\cdot)$  is constant, then it trivially remains in the same  $m$ -quantile for all agents. If it is strictly monotone, then its inverse is well-defined and we can evaluate it using a constant number of queries. Indeed, for any fixed  $m^*$ , using a constant number of queries we can obtain full information about the values of all pieces to all agents, and thus determine the value of  $\alpha$  such that  $m^* = m(\alpha)$ . As a result we can now apply Lemma 6.3 to ensure that  $m(\cdot)$  also remains in the same  $m$ -quantile for all agents.

**Case 2: There is one piece between pieces  $k$  and  $k'$ .** We consider the representative example where piece  $k$  is the piece  $[\ell, m]$ , i.e., the second piece from the left, and piece  $k'$  is the piece  $[r, 1]$ , i.e., the rightmost piece. Recall that in this case the cuts satisfy  $v_1(0, \ell(\alpha)) = v_1(m(\alpha), r(\alpha)) = \alpha$  and  $v_i(\ell(\alpha), m(\alpha)) = v_i(r(\alpha), 1)$ . Clearly,  $\ell(\cdot)$  is strictly increasing in  $\alpha$  and we can thus use the binary search approach of Lemma 6.3 to restrict the interval of  $\alpha$  so that  $\ell(\cdot)$  remains in the same  $m$ -quantile for all agents.

Next, we observe that  $r(\cdot)$  is also strictly increasing in  $\alpha$ . Indeed, if there exist  $\alpha_1 < \alpha_2$  such that  $r(\alpha_1) \geq r(\alpha_2)$ , then it follows that  $m(\alpha_1) \geq m(\alpha_2)$ , since  $m(\alpha_1) < m(\alpha_2)$  would imply the contradiction that  $v_1(m(\alpha_1), r(\alpha_1)) > v_1(m(\alpha_2), r(\alpha_2)) = \alpha$  because  $v_1$  is hungry. But since  $\ell(\alpha_1) < \ell(\alpha_2)$ , we then obtain that  $v_i(\ell(\alpha_1), m(\alpha_1)) > v_i(\ell(\alpha_2), m(\alpha_2)) = v_i(r(\alpha_2), 1) \geq v_i(r(\alpha_1), 1)$ , again a contradiction.

Furthermore, we can compute the inverse of  $r(\cdot)$  using  $O(\log(1/\varepsilon))$  queries. Indeed, given any  $r^*$ , we can query  $\beta := v_i(r^*, 1)$ , and then use  $O(\log(1/\varepsilon))$  queries to find the cuts  $\ell^*, m^*$  such that  $v_i(\ell^*, m^*) = \beta$  and  $v_1(0, \ell^*) = v_1(m^*, r^*)$  (use the same approach as in the proof of Lemma 6.4, but swap the roles of Agents 1 and  $i$ , and replace  $\alpha$  by  $\beta$ ). Letting  $\alpha := v_1(0, \ell^*)$  we then must have  $r^* = r(\alpha)$ . As a result, we can use  $O(\log^2(1/\varepsilon))$  queries to perform the binary search approach of Lemma 6.3 to restrict the search interval of  $\alpha$  so that  $r(\cdot)$  remains in the same  $m$ -quantile for all agents.

In general,  $m(\cdot)$  is not a monotone function of  $\alpha$ . However, now that  $\ell(\cdot)$  and  $r(\cdot)$  remain in the same  $\mathbf{m}$ -quantile for all agents,  $m(\cdot)$  must be a monotone function. Namely, it will be either constant or strictly monotone. To see why this is the case, let us assume that  $m(\cdot)$  is not constant and show that  $m(\cdot)$  is then necessarily strictly monotone. Let  $m^*$  be such that there exists  $\alpha^*$  in the current search interval with  $m(\alpha^*) = m^*$ . We will show that  $\alpha^*$  is unique (in the current search interval) and thus  $m(\cdot)$  has to be strictly monotone. Since we have fixed  $m^*$ , using a constant number of queries we now have full information about the position of cut  $r'(\alpha)$  as a function of  $\alpha$ , where  $r'(\cdot)$  is defined such that  $v_i(m^*, r'(\alpha)) = \alpha$ .<sup>10</sup> Furthermore, since the cuts  $\ell(\cdot)$  and  $r'(\cdot)$  do not cross any  $\mathbf{m}$ -quantiles, we know that  $v_i(\ell(\alpha), \ell(\alpha + d\alpha)) = c \cdot v_i(\ell(\alpha), \ell(\alpha + d\alpha))$  and  $v_i(r'(\alpha), r'(\alpha + d\alpha)) = c' \cdot v_i(r'(\alpha), r'(\alpha + d\alpha))$ . If  $c \neq c'$ , then there exists at most one  $\alpha^*$  with  $v_i(\ell(\alpha^*), m^*) = v_i(r'(\alpha^*), 1)$ , and thus at most one  $\alpha^*$  with  $m^* = m(\alpha^*)$ . If  $c = c'$ , then since there exists  $\alpha^*$  with  $m^* = m(\alpha^*)$ , it must be that all  $\alpha$  in the current interval satisfy  $m^* = m(\alpha)$ , i.e.,  $m(\cdot)$  is constant, a contradiction.

If  $m(\cdot)$  is constant, then it trivially remains in the same  $\mathbf{m}$ -quantile for all agents. If it is strictly monotone, then its inverse is well-defined and it can be evaluated using a constant number of queries (by using the approach for inverting in the previous paragraph). It follows that we can use the binary search approach of Lemma 6.3 to ensure that  $m(\cdot)$  also remains in the same  $\mathbf{m}$ -quantile for all agents.

**Case 3: Piece  $k$  is the leftmost piece and piece  $k'$  is the rightmost piece.** Recall that in this case the cuts satisfy  $v_i(0, \ell(\alpha)) = v_i(r(\alpha), 1)$  and  $v_i(\ell(\alpha), m(\alpha)) = v_i(m(\alpha), r(\alpha)) = \alpha$ . It is easy to see that  $\ell(\cdot)$  is strictly decreasing and  $r(\cdot)$  is strictly increasing. Furthermore, they can both be inverted using a constant number of queries. As a result, we can use Lemma 6.3 to ensure that  $\ell(\cdot)$  and  $r(\cdot)$  remain in the same  $\mathbf{m}$ -quantile for all agents.

In general,  $m(\cdot)$  is not a monotone function, but now that  $\ell(\cdot)$  and  $r(\cdot)$  remain in the same  $\mathbf{m}$ -quantile for all agents,  $m(\cdot)$  will be constant or strictly monotone. Furthermore, when it is strictly monotone, it can also be inverted using a constant number of queries, because we can obtain full information about the values of all pieces for all agents when the middle cut is fixed (again using the fact that  $\ell(\cdot)$  and  $r(\cdot)$  remain in the same  $\mathbf{m}$ -quantile for all agents). As a result, by applying Lemma 6.3, we can again ensure that  $m(\cdot)$  remains in the same  $\mathbf{m}$ -quantile for all agents.

This completes the proof of the lemma.  $\square$

## 7 The Intersection End-of-Line Problem

We begin with the definition of the problem.

*Definition 4 (INTERSECTION END-OF-LINE).*

**Input** Each party receives a superset of the edges; we say that an edge is *active* in the END-OF-LINE instance  $G = (V, E)$  if it is in the intersection of all the supersets. Edges that only appear in the supersets of some parties are called *inactive*.

**Output** A solution to the END-OF-LINE instance defined by the active edges.

In this section, we prove the following result.

**LEMMA 7.1.** INTERSECTION END-OF-LINE with  $k \geq 3$  parties requires  $\text{poly}(|G|)$  communication complexity, even in the special case where the instances satisfy the following promises:

- (0) Every node has at most one active incoming edge, and at most one active outgoing edge.
- (1) Every inactive edge is only included in at most one party's set of edges.

<sup>10</sup>Here we also restrict  $\alpha$  to be such that  $r'(\alpha)$  lies in the same  $\mathbf{m}$ -quantile as  $r(\cdot)$  for all agents. Indeed, if  $r'(\alpha)$  does not satisfy this, then it immediately follows that  $\alpha$  is such that  $m^* \neq m(\alpha)$ .

- (2) No vertex has both an  $i$ -inactive and a  $j$ -inactive incident edge, for some  $i \neq j$ . (An edge is said to be  $i$ -inactive, if it is inactive, but included in Party  $i$ 's superset.)
- (3) Every vertex is assigned to one party (the vertex-party assignment is publicly known): Only that party's superset may have more than one edge coming into that vertex. Similarly, only that party's superset may have more than one edge going out of that vertex.

## 7.1 Number-on-Forehead End-of-Line

The starting point for the proof of Lemma 7.1 is the NoF communication complexity of lifted END-OF-LINE.

*Definition 5 (Number-on-Forehead (NoF) End-of-Line).*

We consider a (common knowledge) host graph  $H$  and  $k$ -party lifting gadgets  $g : \Sigma^k \rightarrow \{0, 1\}$ .

**Inputs:**  $k$  parties receive NoF inputs to gadgets  $g$  for each edge of  $H$ . The inputs induce a subgraph  $G'$  of the host graph  $H$ , i.e., the edges with  $g(\cdot) = 1$ .

**Output:** The goal is to find a solution of the END-OF-LINE problem on  $G'$ .

The following lemma follows as a corollary of results from [23, 24, 37].

**LEMMA 7.2 (COMPLEXITY OF NOF END-OF-LINE).** *For every constant number of parties  $k \geq 2$ , there exist gadgets  $g_k : \Sigma^k \rightarrow \{0, 1\}$  for  $|\Sigma| = 2^{L^{O(1)}}$ , and a constant-degree host graph  $H$  over  $n$  vertices such that NoF END-OF-LINE requires bounded-error randomized communication of  $\tilde{\Omega}(\sqrt{n})$ .*

**PROOF.** By [23], there exist gadgets as in the theorem statement such that the communication complexity of NoF END-OF-LINE is lower bounded by the  $k$ -party NoF communication complexity of unique set disjointness of size  $cbs(EoL)$ , where  $cbs(\cdot)$  denotes the critical block sensitivity (we will not define unique set disjointness of critical block sensitivity here—refer, e.g., to [23]).

By [24], the critical block sensitivity of END-OF-LINE is  $cbs(EoL) = \tilde{\Omega}(n)$ . Finally, by [37], the  $k$ -party NoF communication complexity of the unique set disjointness of size  $cbs(EoL)$  is at least  $\Omega(\frac{\sqrt{cbs(EoL)}}{2^k}) = \tilde{\Omega}(\sqrt{n})$ .  $\square$

## 7.2 Embedding the Lifting Gadgets in an End-of-Line Graph

**PROOF OF LEMMA 7.1.** We reduce NoF END-OF-LINE to INTERSECTION END-OF-LINE. We present the reduction for  $k = 4$ , since this is the version we will later use, but it easily generalizes to any  $k \geq 3$ .

**7.2.1 Constructing the Vertices.** Given host graph  $H = (V_H, E_H)$  for the NoF END-OF-LINE problem, we construct the vertex set  $V$  of the INTERSECTION END-OF-LINE as follows.

For each vertex  $v \in V_H$ , we construct the following vertices:

- (1) We construct a vertex  $v^1 \in V$ , and assign it to Party 1. It will later be convenient to refer to  $v^1$  using the labels  $v^{\text{out},1}$  and  $v^{\text{in},1}$  (both refer to the same vertex).
- (2)(a) For each possible vector of inputs  $\sigma^{\text{out},1} \in \Sigma^{\text{out-deg } v}$  on Party 1's forehead in the lifting gadgets corresponding to edges going out of  $v$ , we construct a vertex  $v_{\sigma^{\text{out},1}}^{\text{out},2} \in V$ , and assign it to Party 2.
- (b) Similarly, for each possible vector of inputs  $\sigma^{\text{in},1} \in \Sigma^{\text{in-deg } v}$  on Party 1's forehead in the lifting gadgets corresponding to edges coming into  $v$ , we construct a vertex  $v_{\sigma^{\text{in},1}}^{\text{in},2} \in V$ , and assign it to Party 2.
- (3)(a) For each possible vector of outgoing gadgets inputs  $(\sigma^{\text{out},1}, \sigma^{\text{out},2}) \in \Sigma^{\text{out-deg } v} \times \Sigma^{\text{out-deg } v}$  on Party 1's and Party 2's foreheads, we construct a vertex  $v_{\sigma^{\text{out},1}, \sigma^{\text{out},2}}^{\text{out},3} \in V$ , and assign it to Party 3.

- (b) Similarly, for each possible vector of incoming gadgets inputs  $(\sigma^{\text{in},1}, \sigma^{\text{in},2}) \in \Sigma^{\text{out-deg } v} \times \Sigma^{\text{out-deg } v}$  on Party 1's and Party 2's foreheads, we construct a vertex  $v_{\sigma^{\text{in},1}, \sigma^{\text{in},2}}^{\text{in},3} \in V$ , and assign it to Party 3.
- (4) Analogously, we construct vertices  $v_{\sigma^{\text{out},1}, \sigma^{\text{out},2}, \sigma^{\text{out},3}}^{\text{out},4}, v_{\sigma^{\text{in},1}, \sigma^{\text{in},2}, \sigma^{\text{in},3}}^{\text{in},4} \in V$  and assign them to Party 4.

We're now ready to construct vertices for the full vectors  $\sigma^{\text{out}} = (\sigma^{\text{out},1}, \sigma^{\text{out},2}, \sigma^{\text{out},3}, \sigma^{\text{out},4})$  (and similarly,  $\sigma^{\text{in}} = (\sigma^{\text{in},1}, \sigma^{\text{in},2}, \sigma^{\text{in},3}, \sigma^{\text{in},4})$ ) that correspond to all the inputs for the edges potentially going out of vertex  $v$ . The vector  $\sigma^{\text{out}}$  determines all outgoing neighbors of  $v$  in the original graph. We let  $w$  denote an arbitrary such outgoing neighbor. If there are no outgoing neighbors, we set  $w := v$ . Similarly, we let  $u$  denote an arbitrary incoming neighbor, and set  $u := v$  if there is no such neighbor. We use  $\rho$  and  $\tau$  to denote the parties' inputs corresponding to vertices  $u$  and  $w$  (resp.).

For each  $v \in V$  and corresponding lifting gadget inputs  $\sigma^{\text{out}}$  and  $\sigma^{\text{in}}$ , yielding neighborhood  $u \rightarrow v \rightarrow w$ , we construct the following vertices:<sup>11</sup>

- (5) Two vertices  $(v_{\sigma^{\text{out}}}^{\text{out},5}, w^{\text{in}})$  and  $(v_{\sigma^{\text{in}}}^{\text{in},5}, u^{\text{out}})$ , assigned to Party 1.
- (6) Another layer of vertices, depending on Party 1's inputs and assigned to Party 2:
- (a) For each input  $\tau^{\text{in},1}$  on Party 1's forehead for  $w$ 's incoming edges, construct vertex  $(v_{\sigma^{\text{out}}}^{\text{out},6}, w_{\tau^{\text{in},1}}^{\text{in}})$ .
  - (b) For each input  $\rho^{\text{out},1}$  on Party 1's forehead for  $u$ 's outgoing edges, construct vertex  $(v_{\sigma^{\text{in}}}^{\text{in},6}, u_{\rho^{\text{out},1}}^{\text{out}})$ .
- (7) Vertices  $(v_{\sigma^{\text{out}}}^{\text{out},7}, w_{\tau^{\text{in},1}, \tau^{\text{in},2}}^{\text{in}}), (v_{\sigma^{\text{in}}}^{\text{in},7}, u_{\rho^{\text{out},1}, \rho^{\text{out},2}}^{\text{out}})$  assigned to Party 3.
- (8) Vertices  $(v_{\sigma^{\text{out}}}^{\text{out},8}, w_{\tau^{\text{in},1}, \tau^{\text{in},2}, \tau^{\text{in},3}}^{\text{in}}), (v_{\sigma^{\text{in}}}^{\text{in},8}, u_{\rho^{\text{out},1}, \rho^{\text{out},2}, \rho^{\text{out},3}}^{\text{out}})$  assigned to Party 4.

Finally, we're ready to construct the last layer of vertices, which is not assigned to any party.<sup>12</sup>

- (9) We construct two vertices, each with two symmetric labels:

$$\begin{aligned} (v_{\sigma^{\text{out}}}^{\text{out},9}, w_{\tau^{\text{in}}}^{\text{in}}) &= (w_{\tau^{\text{in}}}^{\text{in},9}, v_{\sigma^{\text{out}}}^{\text{out}}) \\ (v_{\sigma^{\text{in}}}^{\text{in},9}, u_{\rho^{\text{out}}}^{\text{out}}) &= (u_{\rho^{\text{out}}, 9}^{\text{out}}, v_{\sigma^{\text{in}}}^{\text{in}}). \end{aligned}$$

**7.2.2 Constructing the Edges.** We describe the edges between the "out" vertices; the edge construction for "in" vertices is analogous.

- For  $i \in \{1, 2, 3, 4\}$ , Party  $i$ 's superset includes the edge from  $v^{\text{out},i}$ -vertex to a  $v^{\text{out},i+1}$ -vertex whenever:
  - $v^{\text{out},i}$ 's subscript is a prefix of  $v^{\text{out},i+1}$ 's subscript; and
  - all the lifting gadget inputs in the subscripts (in particular  $v^{\text{out},i+1}$ 's subscript) are consistent with Party  $i$ 's information, i.e., all the inputs not on  $i$ 's forehead are correct.
- Similarly, for  $i \in \{5, 6, 7, 8\}$ , Party  $(i-4)$ 's superset includes the edge from  $(v^{\text{out},i}, w^{\text{out}})$ -vertex to a  $(v^{\text{out},i+1}, w^{\text{out}})$ -vertex whenever the former subscript is a prefix of the latter, and they are consistent with Party  $i$ 's input.

Notice that an edge  $(v \rightarrow w)$  in the NoF END-OF-LINE instance is embedded as the following path in the intersection of the parties' supersets:

$$(v^1 \rightarrow \dots \rightarrow (v_{\sigma^{\text{out}}}^{\text{out},5}, w) \rightarrow \dots \rightarrow (v_{\sigma^{\text{out}}}^{\text{out},9}, w_{\tau^{\text{in}}}^{\text{in}}) = (w_{\tau^{\text{in}}}^{\text{in},9}, v_{\sigma^{\text{out}}}^{\text{out}}) \rightarrow \dots \rightarrow w^1).$$

<sup>11</sup>We remark that Layers 5–8 are useful for guaranteeing the desideratum about vertices assigned to parties; for hardness of INTERSECTION END-OF-LINE in the general case they are unnecessary.

<sup>12</sup>These vertices can be assigned to an arbitrary party for the purpose of ensuring desideratum 3.

**7.2.3 NoF END-OF-LINE to INTERSECTION END-OF-LINE: Analysis.** First, notice that by construction every vertex has at most one active incoming edge, and at most one active outgoing edge. This is due to the fact that whenever there are multiple possible successors in the tree-like graph that we construct, at least three parties have all the information needed to exclude all but one of those outgoing edges. Furthermore, the only vertices that can have an odd degree are the vertices on the last layer, namely, the vertices of the form  $(v_{\sigma^{\text{out}}}^{\text{out},9}, w_{\tau^{\text{in}}}^{\text{in}})$  and  $(v_{\sigma^{\text{in}}}^{\text{in},9}, u_{\rho^{\text{out}}}^{\text{out}})$ . This is because our construction ensures that there is a unique active path from  $v^1$  to one of its leaves  $(v_{\sigma^{\text{out}}}^{\text{out},9}, w_{\tau^{\text{in}}}^{\text{in}})$ , and a unique active path from one of the leaves  $(v_{\sigma^{\text{in}}}^{\text{in},9}, u_{\rho^{\text{out}}}^{\text{out}})$  to  $v^1$ . Every other edge in any of the two trees around  $v^1$  will be inactive, because at least one of the endpoints of such an edge must have a subscript that is inconsistent with the correct input information, and thus at least one party (in fact, all but one) will see this and not include the edge. Now, if a vertex of the form  $(v_{\sigma^{\text{out}}}^{\text{out},9}, w_{\tau^{\text{in}}}^{\text{in}})$  has an odd degree, then it must have out-degree 0, since it has in-degree 1. But if it has out-degree 0, that means that it is not included in the tree constructed around  $w^1$ . This can only happen if: (i)  $w$  has multiple predecessors in the original graph (and picked a different one in the tree branch corresponding to those inputs), or (ii)  $w = v$  (i.e.,  $v$  has no outgoing edge) and  $v$  has at least one incoming edge. In both cases we obtain a solution to NoF END-OF-LINE.

We now verify the remaining particular desiderata from the statement of Lemma 7.1.

- (1) Every inactive edge is only included in at most one party's set of edges.

Every inactive edge has at least one endpoint with at least one label that is different from the corresponding lifting gadget input on some party's forehead. Hence it will not be included in any of the other parties' supersets (all other parties know this label is wrong).

- (2) No vertex has both an  $i$ -inactive and a  $j$ -inactive incident edge.

If an  $i$ -inactive edge is between two vertices that both have (purported) information from Party  $i$ 's input in their subscript, then that information is the same on both vertices and does not correspond to what lies on Party  $i$ 's forehead. As a result, no other Party will have any incident edges on these two vertices. If only one of the two endpoints contains (purported) information from Party  $i$ 's input, then that information is incorrect (so that endpoint does not have any  $j$ -inactive edge) and all other inactive edges incident on the other endpoint must have an endpoint with incorrect  $i$ -information as well.

- (3) Only the assigned party's superset may have  $>1$  outgoing or  $>1$  incoming edges on a vertex.

Party  $i$ 's superset can have more than one edge coming into the same vertex if and only if it does not know the next label because it is written on Party  $i$ 's forehead. By construction, this only happens for vertices assigned to Party  $i$ . An analogous argument holds for outgoing edges.

This completes the proof. □

## 8 Communication Lower Bound for Envy-free Cake-cutting with Four Non-monotone Agents

In this section, we prove the following result.

**THEOREM 8.1.** *For four agents with non-monotone valuations, finding an  $\epsilon$ -envy-free connected allocation requires  $\text{poly}(1/\epsilon)$  communication.*

The rest of this section is devoted to the proof of this theorem. We reduce from the 4-party INTERSECTION END-OF-LINE problem. In Section 8.1, we show how each party can embed its superset of edges into a two-dimensional variant of the SPERNER problem. In Section 8.2, we then use each party's embedding to construct a corresponding valuation function for a cake cutting instance.

Finally, in Section 8.3, we analyze this reduction; specifically we show that any  $\varepsilon$ -envy free partition of the cake maps to a region in the Sperner embedding that embeds a solution of the original INTERSECTION END-OF-LINE problem.

Our reduction can also be used to obtain the following two results. We provide more details about this in Section 8.4.

**THEOREM 8.2.** *For four agents with identical non-monotone valuations, finding an  $\varepsilon$ -envy-free connected allocation requires  $\text{poly}(1/\varepsilon)$  queries.*

**THEOREM 8.3.** *For four agents with identical non-monotone valuations, finding an  $\varepsilon$ -envy-free connected allocation is PPAD-complete.*

## 8.1 Special Sperner Embedding of End-of-Line

In this section, we present the first step of the reduction, which is a special embedding of an END-OF-LINE instance into a two-dimensional Sperner-type problem. While it is well-known how to do this in the query or white-box model [15], we have to very carefully construct a special embedding here since we are reducing from the communication problem INTERSECTION END-OF-LINE. In more detail, each of the four parties is going to separately embed the edges of INTERSECTION END-OF-LINE that it sees (i.e., that are included in its superset of the edges) in a very specific way.<sup>13</sup> This will ensure that an important set of properties (proved in the claims below) will hold. In the next section, the embedding of party  $i$  will then be used to construct the valuation function of agent  $i$ . The properties proved here will be crucial for the latter analysis of the reduction. In this section, we use “agent” and “party” interchangeably.

Consider an INTERSECTION END-OF-LINE instance with four parties/agents that satisfies the promises of Lemma 7.1. Let  $n$  denote the number of nodes of the instance. Let vertex 1 be the trivial source.<sup>14</sup>

**Grid and labeling.** Consider the two-dimensional grid  $S = \{0, 1, 2, \dots, N\}^2$  where  $N := 300n^4$ . For each agent  $i \in \{1, 2, 3, 4\}$ , we construct a labeling  $\lambda_i : S \rightarrow \{+1, -1\}^2$ . In other words, every agent assigns one label to each grid point, and the possible labels are  $(+1, +1)$ ,  $(+1, -1)$ ,  $(-1, +1)$ ,  $(-1, -1)$ . We say that  $(+1, +1)$  and  $(-1, -1)$  are opposite labels, and similarly  $(+1, -1)$  and  $(-1, +1)$  are opposite labels too. A square of the grid is a set of points  $\{j, j+1\} \times \{k, k+1\}$  for some  $j, k \in \{0, 1, 2, \dots, N-1\}$ . We say that an agent sees a Sperner solution in a square of the grid, if the four corners contain two opposite labels, i.e., if both labels  $(+1, +1)$  and  $(-1, -1)$  appear, or if both labels  $(+1, -1)$  and  $(-1, +1)$  appear. In particular, if a square is not a Sperner solution, then the labels appearing at its four corners must be all be the same or belong to one of the following four sets:  $\{(+1, +1), (+1, -1)\}$ ,  $\{(+1, +1), (-1, +1)\}$ ,  $\{(-1, -1), (-1, +1)\}$ , or  $\{(-1, -1), (+1, -1)\}$ .

**Vertex regions and lanes.** For each of the  $n$  vertices of End-of-Line, we define a corresponding region on the diagonal of grid  $S$ . Namely, for vertex  $j \in [n]$ , its vertex region is

$$V(j) := \{300n^3(j-1) + 1, \dots, 300n^3j\}^2.$$

The vertex region has a certain number of lanes incident to it. They are shown in Figure 1 and defined formally below.

We subdivide the set of grid points

$$\{1, \dots, 300n^3(j-1) + 1\} \times \{300n^3(j-1) + 100n^3 + 1, \dots, 300n^3(j-1) + 200n^3\}$$

<sup>13</sup>An important point is that agents can see multiple incoming or outgoing edges on a vertex. Existing embeddings do not allow for this, because it cannot occur in a standard End-of-Line instance. Our embedding is carefully constructed in order to be able to handle this.

<sup>14</sup>We sometimes use 0 to denote the trivial source in other parts of the article.

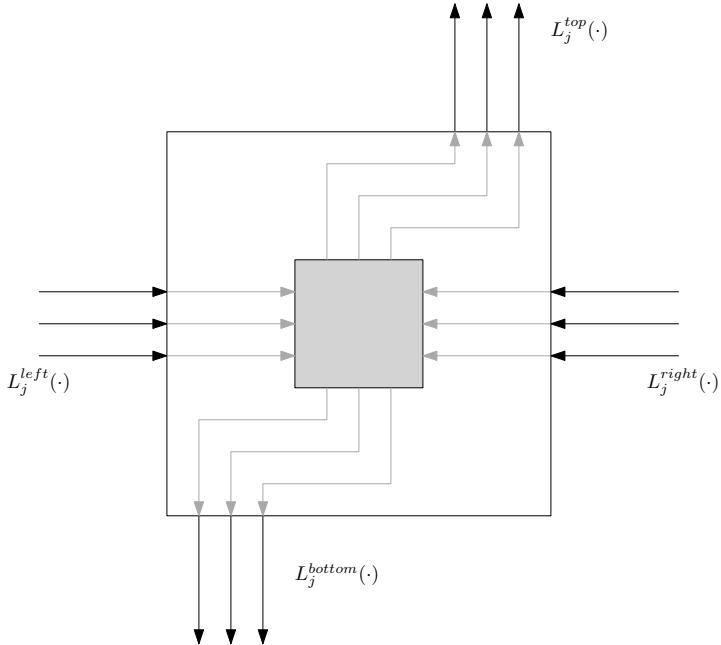


Fig. 1. Illustration of the vertex region  $V(j)$ . The black edges represent lanes (where we have only represented three of each type here). The gray square in the center of the vertex region represents the connector region. The gray edges inside the vertex region show how (potential) paths can be routed to the connector region without intersecting.

into  $n$  horizontal lanes of width  $100n^2$  each. For any  $k < j$ , we let  $L_j^{left}(k)$  denote the  $k$ th such lane. Formally,  $L_j^{left}(k) := \{1, \dots, 300n^3(j-1)+1\} \times \{300n^3(j-1)+100n^3+100n^2(k-1)+1, \dots, 300n^3(j-1)+100n^3+100n^2k\}$ .

We subdivide the set of grid points

$$\{300n^3j, \dots, N\} \times \{300n^3(j-1)+100n^3+1, \dots, 300n^3(j-1)+200n^3\}$$

into  $n$  horizontal lanes of width  $100n^2$  each. For any  $k > j$ , we let  $L_j^{right}(k)$  denote the  $k$ th such lane. Formally,  $L_j^{right}(k) := \{300n^3j, \dots, N\} \times \{300n^3(j-1)+100n^3+100n^2(k-1)+1, \dots, 300n^3(j-1)+100n^3+100n^2k\}$ .

We subdivide the set of grid points

$$\{300n^3(j-1)+200n^3+1, \dots, 300n^3j\} \times \{300n^3j, \dots, N\}$$

into  $n$  vertical lanes of width  $100n^2$  each. For any  $k > j$ , we let  $L_j^{top}(k)$  denote the  $k$ th such lane. Formally, we have  $L_j^{top}(k) := \{300n^3(j-1)+200n^3+100n^2(k-1)+1, \dots, 300n^3(j-1)+200n^3+100n^2k\} \times \{300n^3j, \dots, N\}$ .

Similarly, we also subdivide the set of grid points

$$\{300n^3(j-1)+1, \dots, 300n^3(j-1)+100n^3\} \times \{1, \dots, 300n^3(j-1)+1\}$$

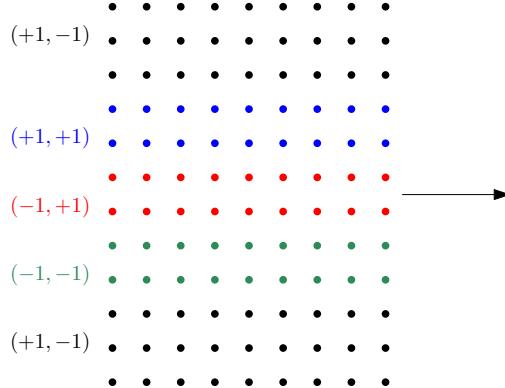


Fig. 2. Implementation of a path using the Sperner labels. The arrow indicates that the path is moving from left to right. The color of each grid point indicates its label. The label corresponding to each color is shown on the left.

into  $n$  vertical lanes of width  $100n^2$  each. For any  $k < j$ , we let  $L_j^{bottom}(k)$  denote the  $k$ th such lane. Formally, we have  $L_j^{bottom}(k) := \{300n^3(j-1) + 100n^2(k-1) + 1, \dots, 300n^3(j-1) + 100n^2k\} \times \{1, \dots, 300n^3(j-1) + 1\}$ .

**Embedding of edges.** If agent  $i$  sees an edge from vertex  $j$  to vertex  $k$  in the INTERSECTION END-OF-LINE instance, then it will implement a path from vertex region  $V(j)$  to vertex region  $V(k)$  in its labeling  $\lambda_i$ . In more detail:

- If  $j < k$ , then the path leaves  $V(j)$  using vertical lane  $L_j^{top}(k)$  and moves up until it reaches horizontal lane  $L_k^{left}(j)$ . The path then turns towards the right and follows the horizontal lane  $L_k^{left}(j)$  until it reaches  $V(k)$ .
- If  $j > k$ , then the path leaves  $V(j)$  using vertical lane  $L_j^{bottom}(k)$  and moves down until it reaches horizontal lane  $L_k^{right}(j)$ . The path then turns towards the left and follows the horizontal lane  $L_k^{right}(j)$  until it reaches  $V(k)$ .

**Environment label and paths.** The environment label is  $(+1, -1)$ . Unless otherwise specified, the label of a grid point is the environment label  $(+1, -1)$ . A path consists of a central part with label  $(-1, +1)$  that is surrounded by one protective layer on each side. The protective layer on the left (when moving in the forward direction) has label  $(+1, +1)$ . The protective layer on the right has label  $(-1, -1)$ . Note that a path does not introduce a Sperner solution, because opposite labels are isolated from each other. The path can also turn without introducing Sperner solutions. Solutions do however appear at the beginning or end of a path, because the central part with label  $(-1, +1)$  is in contact with the environment label  $(+1, -1)$ . For our purposes it will be convenient to use paths where the central part and protective layers have width two. See Figure 2 for an illustration of the implementation of a path.

**Boundary.** On the boundary of the grid  $S$ , we fix the labels so that they satisfy a Sperner-type boundary condition. In more detail

- Points close to the top boundary of the grid, i.e., in  $\{0, \dots, N\} \times \{N-2, N-1, N\}$ , have label  $(+1, -1)$ .

- Points close to the bottom boundary of the grid, i.e., in  $\{0, \dots, N\} \times \{0, 1, 2\}$ , have label  $(-1, +1)$ .
- Points close to the left boundary of the grid, i.e., in  $\{0, 1, 2\} \times \{0, \dots, N\}$ , have label  $(+1, +1)$ .
- Points close to the right boundary of the grid, i.e., in  $\{N-2, N-1, N\} \times \{0, \dots, N\}$ , have label  $(-1, -1)$ .

For points close to the corners of the grid, where the boundary conditions above specify two different labels, we pick one of those two labels arbitrarily but consistently across agents.

Using a simple and standard construction [15, 21], it is possible to use these boundary conditions to create a single starting path in the vertex region  $V(1)$ . Since all agents agree that the trivial source 1 has no incoming edges and exactly one outgoing edge, this starting path is used to implement this outgoing edge. Note that the agents also agree on the successor of vertex 1. Thus, all four agents also agree on the labeling in region  $V(1)$ .

**Crossing regions.** Outside of vertex regions, the only regions where two paths can cross are the regions where a horizontal lane crosses a vertical lane. Previous work [15] has shown how such crossings can be handled locally in order to ensure that no Sperner solutions occur there. We will also make use of such crossing gadgets, although we will have to design them very carefully in our setting. But before we even get to the crossing gadgets, our arguments later in the reduction will require us to ensure that all crossing gadgets occur on different  $x$ -coordinates on the grid  $S$ .

Recall that we have specified that edges are implemented by paths that follow specific lanes, depending on which edge is implemented. We have however left some freedom in how exactly the paths travel inside their respective lanes. Furthermore, we have constructed the lanes so that they each have width  $100n^2$ . Consider a path moving upwards in the vertical lane  $L_j^{top}(k)$ . At any given point, the path intersects at most one horizontal lane, and overall it is going to intersect at most  $n^2$  different horizontal lanes (because every vertex yields at most  $n$  horizontal lanes). Now, we subdivide lane  $L_j^{top}(k)$  into  $n^2$  sublanes of width 100 each. We can enforce that the path going up in  $L_j^{top}(k)$  always uses the sublane corresponding to the horizontal lane it is currently crossing. Namely, if the path is currently crossing the  $p$ th horizontal lane of vertex  $q$ , then our path will use the  $((q-1)n + p)$ th sublane of  $L_j^{top}(k)$ . Figure 3 shows how this can be implemented. For paths traveling downwards on a vertical lane  $L_j^{bottom}(k)$  we also define sublanes and proceed in the same way. For paths traveling on horizontal lanes, we do not need to do any of this. Instead, we just let the path travel in a straight line in the center of the lane.

The intersection of a horizontal lane and the corresponding sublane of a vertical lane is a big rectangle of size  $100 \times 100n^2$ . Note that the (potential) paths traveling in the horizontal lane and in the vertical sublane would meet in the center of that rectangle. We delimit a big square of dimensions  $50 \times 50$  around the center of the rectangle and call it the *crossing region* between the horizontal lane and the vertical lane. The crossing gadgets will be constructed inside the corresponding crossing region, when needed. The construction detailed above ensures that all crossing regions have separate  $x$ -coordinates. More formally, any two grid points that lie in two different crossing regions must have different  $x$ -coordinates.

**CLAIM 3 (EMBEDDING PROPERTIES OUTSIDE SPECIAL REGIONS).** *The special embedding ensures that outside any crossing or vertex region, every square satisfies one of the following properties:*

- (1) *All four agents agree on the labels of the four corners, and there is no Sperner solution.*
- (2) *At least three agents see the environment label  $(+1, -1)$  at all four corners.*

**PROOF.** Outside of crossing and vertex regions every potential path has its own lane, and these lanes do not intersect by construction (except in crossing regions, of course). Furthermore, if one

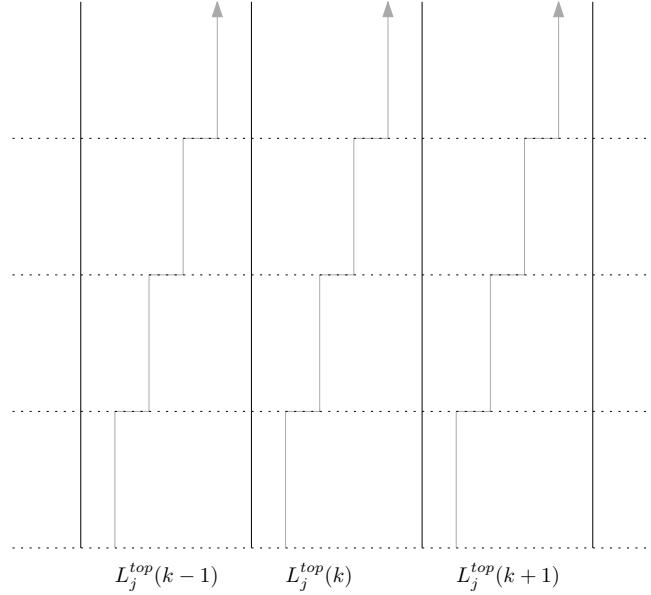


Fig. 3. Illustration showing how we can ensure that paths use the sublane corresponding to the horizontal lane currently being crossed. The vertical black lines delimit the vertical lanes, and the horizontal dotted lines delimit the horizontal lanes. The gray arrow shows the trajectory that a (potential) path in the corresponding vertical lane would follow.

agent sees a path in a lane, then by property 1 of Lemma 7.1, either all agents see the path, or no other agent sees the path. Finally, all agents agree on the labeling close to the boundary, and, in particular, in the creation of the starting path for  $V(1)$ . As a result, every square falls into one of the two categories above.  $\square$

It remains to handle vertex regions and crossing regions, and, in particular, to explain how the labeling is defined there.

**Labeling inside vertex regions.** Let  $j > 1$  be any vertex. Let the connector region be defined as the central part of  $V(j)$ , namely,  $\{300n^3(j-1) + 100n^3 + 1, \dots, 300n^3(j-1) + 200n^3\}^2$ . Note that by construction the connector region does not share any  $x$ -coordinates with any of the vertical lanes  $L_j^{top}(k)$  or  $L_j^{bottom}(k)$ .

For any agent  $i$ , the labeling inside the vertex region  $V(j)$  is constructed as follows. First, we route each incoming and outgoing path from the boundary of  $V(j)$  to its corresponding position on the boundary of the connector region. Figure 1 shows a principled way of achieving this while ensuring that no paths intersect.

It remains to specify the labeling inside the connector region. If the agent sees more than one incoming path, or more than one outgoing path, then all points in the connector region are labeled  $(-1, -1)$ . If the agent sees one incoming path and no outgoing path, or one outgoing path and no incoming path, then all points in the connector region are labeled with the environment label  $(+1, -1)$ . Finally, if the agent sees exactly one incoming path and exactly one outgoing path, then the two paths are connected inside the connector region. We can make sure that this connection is performed in a consistent way, namely such that different agents who see the same incoming and outgoing path perform the connection in the exact same way.

A vertex region  $V(j)$  is said to be an *end-of-line vertex region*, if the corresponding vertex  $j$  is an end-of-line solution of the INTERSECTION END-OF-LINE instance. We say that  $V(j)$  is a *non-end-of-line vertex region*, if it is not an end-of-line vertex region. Ultimately, our construction will ensure that any envy-free division yields a solution in an end-of-line vertex region.

**CLAIM 4 (EMBEDDING PROPERTIES IN NON-END-OF-LINE VERTEX REGION).** *The special embedding ensures that in any non-end-of-line vertex region, one of the following cases occurs:*

- (1) *All four agents see the same single path passing through (and no other paths).*
- (2) *At least three agents see no path.*
- (3) *Three agents see the same single path passing through, and the remaining agent sees that path along with possibly others, but has label  $(-1, -1)$  everywhere in the connector region.*

**PROOF.** Since the vertex region  $V(j)$  is a non-end-of-line vertex region, it follows that in the original graph, and in terms of the active edges,  $j$  is either an isolated vertex or has exactly one incoming and exactly one outgoing edge. If  $j$  is isolated, then by property 2 in Lemma 7.1, at most one agent sees incident edges on  $j$  (which are all inactive), and the other three agents see no incident edges on  $j$ . As a result, at least three agents see no path at all in  $V(j)$ .

If  $j$  has exactly one incoming and exactly one outgoing edge, then all four agents must see these two edges (by definition of what it means for an edge to be active). If they do not see any further incident edges on  $j$ , then all four agents agree and see that single path passing through  $V(j)$ . If, on the other hand, some agents see additional (inactive) edges, then by property 3 in Lemma 7.1, only a single agent, say agent  $i$ , can see additional edges, while the other three agents only see the active incoming and outgoing edge. As a result, the three agents see the same single path passing through  $V(j)$ , and agent  $i$  sees that path along with others. Furthermore, by construction of the labeling in  $V(j)$ , the connector region is labeled  $(-1, -1)$  for agent  $i$ , as claimed.  $\square$

**CLAIM 5.** *In any non-end-of-line vertex region, every square satisfies one of the following properties:*

- (1) *All four agents agree on the labels at the four corners, and there is no Sperner solution.*
- (2) *At least three agents agree and they see the same identical label at all four corners.*
- (3) *At least three agents see labels only in  $\{(-1, +1), (+1, +1)\}$ , or only in  $\{(+1, -1), (+1, +1)\}$ , and the square touches the connector region.*
- (4) *All four agents see labels only in  $\{(+1, -1), (-1, -1)\}$ , or only in  $\{(-1, +1), (-1, -1)\}$ .*

**PROOF.** By Claim 4 there are three possible cases. We show that in each of those three cases, every square in the vertex region falls into (at least) one of the four categories mentioned in the statement.

If all agents see the same single path passing through the vertex region and nothing else, then the construction ensures that all four agents agree on the labels of all points in the vertex region. In particular, in every square, all four agents agree on the labels at the four corners, and these labels do not correspond to a Sperner solution, by the construction of the paths. Thus, every square belongs to category 1.

If at least three agents see no path, then they label the whole vertex region with the environment label  $(+1, -1)$ . As a result, every square belongs to category 2.

The final case is when three agents see the same single path passing through, and the remaining agent, say agent  $i$ , sees that path along with possibly others, but has label  $(-1, -1)$  everywhere in the connector region. First of all, note that any square that is not a part of the single path seen by the three agents, must be fully labeled with the environment label  $(+1, -1)$  by the three agents, and thus falls into category 2. Next, for any square lying on the single path but not touching the connector region, agent  $i$  agrees with the labels of the other three agents (since agent  $i$  also sees that path). Thus, the square falls into category 1. Furthermore, any square on the single path where

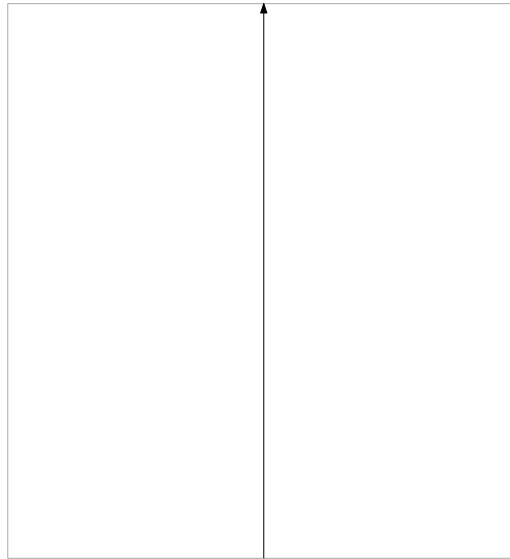


Fig. 4. Implementation of the crossing gadget when the agent only sees a vertical path.

the three agents have identical label (i.e., the same label at all four corners) falls into category 2. It remains to handle squares that lie on the single path and touch the connector region, and where the three agents see two different labels. By construction of the paths, the only possibilities for these two labels are: (i)  $\{(-1, +1), (+1, +1)\}$ , (ii)  $\{(+1, -1), (+1, +1)\}$ , (iii)  $\{(+1, -1), (-1, -1)\}$ , and (iv)  $\{(-1, +1), (-1, -1)\}$ . Cases (i) and (ii) immediately fall into category 3.

It remains to argue about cases (iii) and (iv). We consider two subcases: a square that lies on the single path and touches the connector region, and where the three agents see labels (iii) or (iv) must necessarily occur in (a) well inside the connector region, or (b) on the boundary of the connector region. In case (a), agent  $i$  has label  $(-1, -1)$  and the square falls into category 4, because label  $(-1, -1)$  is a subset of both (iii) and (iv). Finally, in case (b) the square lies on the boundary of the connector region, meaning that two corners lie inside the connector region, and two corners lie outside the connector region. Since agent  $i$  agrees with the other three agents on the two corners outside the connector region, and sees label  $(-1, -1)$  in the two corners inside the connector region, we again have that the square falls into category 4, because label  $(-1, -1)$  is a subset of both (iii) and (iv).  $\square$

**Labeling inside crossing regions.** Any crossing region is of one of the two following types: (a) a (potential) crossing between a path going up (increasing in the  $y$ -coordinate) and a path going from left to right (increasing in the  $x$ -coordinate), or (b) a (potential) crossing between a path going down (decreasing in the  $y$ -coordinate) and a path going from right to left (decreasing in the  $x$ -coordinate).

Let us first consider a crossing region of type (a). We now explain how the labeling is locally modified inside the crossing region. If the agent sees no paths in the crossing region, then the crossing is fully labeled by the environment label  $(+1, -1)$ . If the agent sees only the vertical path, then we just implement the vertical path normally (Figure 4). If the agent only sees the horizontal path, then we implement the horizontal path, but with a small modification, see Figure 5. If the agent sees both a horizontal and vertical path, then we implement a crossing gadget which locally reroutes the paths to avoid the intersection. In our setting, we have to construct this gadget very carefully, see Figure 6.

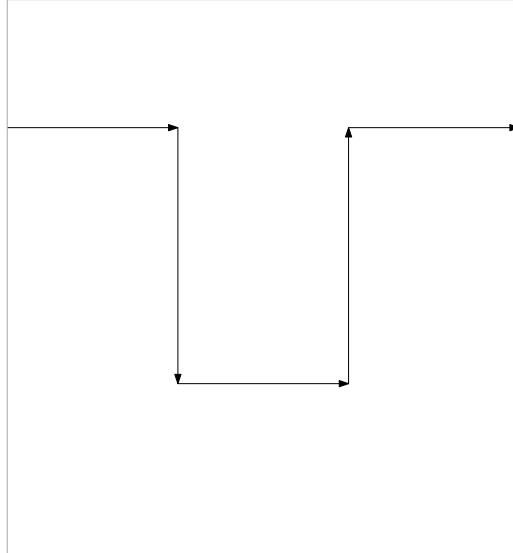


Fig. 5. Implementation of the crossing gadget when the agent only sees a horizontal path.

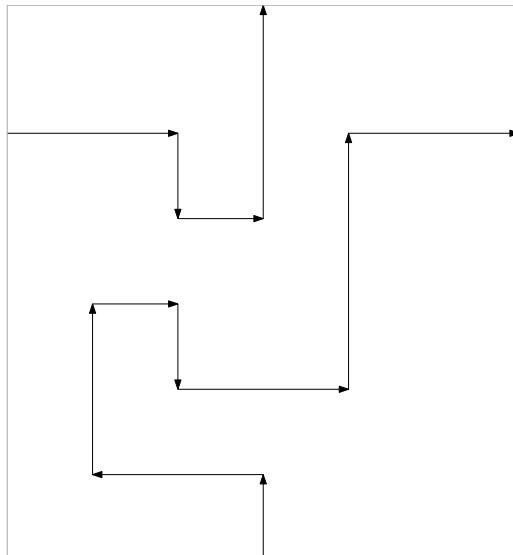


Fig. 6. Implementation of the crossing gadget when the agent sees a horizontal path crossing a vertical path.

In the latter parts of this reduction, we will have to argue very carefully about this crossing gadget. It is therefore convenient to define some notation that will be useful later. Consider the path that a single vertical path follows inside the crossing region. We can think of the path as being decomposed into separate columns, such that in each column all of the squares are identical. We let  $c_1^+$  denote the column that contains the squares that contain both label  $(+1, -1)$  and label  $(-1, -1)$ . Note that this indeed identifies a unique column of squares on that vertical path. Similarly, we let  $c_1^-$  denote the column that contains the squares that contain both label  $(-1, +1)$  and label  $(-1, -1)$ .

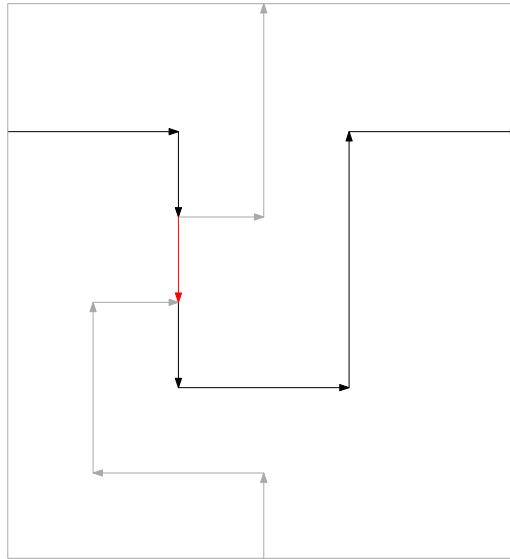


Fig. 7. One agent sees a crossing, and the other three agents see a horizontal path. The black edges are the ones where all four agents agree that there is a path. The gray edges correspond to portions of paths only seen by the agent who sees a crossing. The red edge is the only portion of path that is seen by the three agents, but not by the agent who sees a crossing.

Note that this column is also unique. In the same manner, we also consider the (bent) horizontal path and note that it has a portion of path that moves down vertically. For this vertical portion, we define columns  $c_2^+$  and  $c_2^-$  analogously. Namely,  $c_2^+$  is the column of squares with labels  $(+1, -1)$  and  $(-1, -1)$ , and  $c_2^-$  is the column of squares with labels  $(-1, +1)$  and  $(-1, -1)$ . The red edge in Figure 7 and in Figure 8 is the portion of path of interest here.

For crossing regions of type (b), we use the same construction for the crossing, except that all paths now run in the opposite direction, compared to type (a). We also define columns  $c_1^+, c_1^-, c_2^+, c_2^-$  analogously, where we use the upward vertical portion of the horizontal (bent) path, instead of the downward portion. In particular, we make sure that  $c_1^+$  and  $c_2^+$  still correspond to columns with squares with labels  $(+1, -1)$  and  $(-1, -1)$ , and  $c_1^-$  and  $c_2^-$  still correspond to columns with squares with labels  $(-1, +1)$  and  $(-1, -1)$ .

**CLAIM 6 (EMBEDDING PROPERTIES IN CROSSING REGION).** *The special embedding ensures that in any crossing region, one of the following cases occurs:*

- (1) All four agents agree and see no Sperner solution.
- (2) At least three agents see no path.
- (3) Two agents see no path, one agent sees a horizontal path, and one other agent sees a vertical path.
- (4) One agent sees a crossing, and the other three agents see the same single path (horizontal or vertical).

Note that some cases are not mutually exclusive (e.g., case 1 and case 2).

**PROOF.** Note that in a crossing region every agent sees either (i) no path, (ii) a single horizontal path, (iii) a single vertical path, or (iv) both a horizontal and a vertical path, in which case the agent sees a crossing. Furthermore, by property 1 of Lemma 7.1, if at least two agents see a specific path, then all four agents see that path.

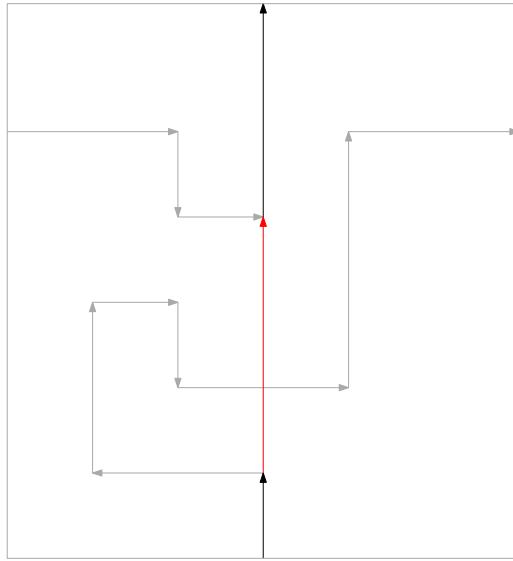


Fig. 8. One agent sees a crossing, and the other three agents see a vertical path. The black edges are the ones where all four agents agree that there is a path. The gray edges correspond to portions of paths only seen by the agent who sees a crossing. The red edge is the only portion of path that is seen by the three agents, but not by the agent who sees a crossing.

Consider first the case where at least one agent, say agent  $i$ , sees a crossing. Then the other three agents must have an identical view, i.e., they all see either (i) no path, (ii) the same single path (horizontal or vertical), or (iii) a crossing. Indeed, since agent  $i$  sees both paths, if a second agent also sees a path, then all other agents must also see it. If case (i) holds, then we are in case 2 of the claim. Case (ii) yields case 4 of the claim, and case (iii) yields case 1 of the claim.

Now consider the case where no agent sees a crossing. If all agents see no path, then clearly we are in case 1 of the claim. If an agent, say agent  $i$ , sees a single path (horizontal or vertical), then either (i) the other three agents see no path, (ii) the other three agents also see that single path, or (iii) one other agent sees the other path (vertical or horizontal, respectively) and the two remaining agents see no path. Note that at most one agent can see the other path, since if two agents see it, then all four agents must see it (which is not possible, since agent  $i$  does not see a crossing). Case (i) yields case 2 of the claim, case (ii) yields case 1 of the claim, and, finally, case (iii) yields case 3 of the claim.  $\square$

**CLAIM 7.** *In a crossing region where one agent, say agent  $i$ , sees a crossing, and the other three agents see the same single path (horizontal or vertical), every square satisfies (at least) one of the following properties:*

- (1) *All four agents agree on the labels at the four corners, and there is no Sperner solution.*
- (2) *At least three agents agree and they see the same identical label at all four corners.*
- (3) *The square lies in column  $c_1^+$  or  $c_2^+$ , and the three agents (other than  $i$ ) only see labels from  $\{(+1, -1), (-1, -1)\}$  in the square.*
- (4) *The square lies in column  $c_1^-$  or  $c_2^-$ , and the three agents (other than  $i$ ) only see labels from  $\{(-1, +1), (-1, -1)\}$  in the square.*
- (5) *The square is not directly adjacent to any of the columns  $c_1^+, c_1^-, c_2^+, c_2^-$ , and three agents see labels only from  $\{(-1, +1), (+1, +1)\}$  or only from  $\{(+1, -1), (+1, +1)\}$  in the square.*

**PROOF.** Since we have three agents seeing the same single path, it follows that any square outside that single path is fully labeled with the environment label  $(+1, -1)$  by these three agents. As a result these squares fall into category 2. Next, any square on the single path where the three agents have identical label (i.e., the same label at all four corners) also falls into category 2. It remains to handle squares on the single path where the three agents see two different labels. By construction of the paths, the only possibilities for these two labels are: (i)  $\{(-1, +1), (+1, +1)\}$ , (ii)  $\{(+1, -1), (+1, +1)\}$ , (iii)  $\{(+1, -1), (-1, -1)\}$ , and (iv)  $\{(-1, +1), (-1, -1)\}$ .

The crucial observation is that the intricate construction of the crossing gadget ensures that for any such square on the single path, agent  $i$  agrees with the other three agents on the labeling (in which case the square is in category 1), unless the square lies on:

- the single vertical path, or
- the downward vertical portion of the horizontal (bent) path in a crossing of type (a)
- the upward vertical portion of the horizontal (bent) path in a crossing of type (b)

See Figures 7 and 8 for an illustration. Now, if the square is of type (i) or (ii), then by construction we have that the square is not adjacent<sup>15</sup> to any of the columns  $c_1^+, c_1^-, c_2^+, c_2^-$ , and thus falls into category 5. If the square is of type (iii), then by construction it lies in column  $c_1^+$  or  $c_2^+$ , and thus falls into category 3. Finally, if the square is of type (iv), then by construction it lies in column  $c_1^-$  or  $c_2^-$ , and thus, falls into category 4.  $\square$

## 8.2 Construction of Valuations

We now describe how the instance of envy-free cake-cutting is constructed. For each party  $i$  of the INTERSECTION END-OF-LINE problem, we construct a corresponding agent  $i$ . The valuation function of agent  $i$  over the cake will be constructed using only the information of party  $i$ .

We let  $\delta := \frac{1}{10N} \leq 0.01$ , where  $N$  is the length of the grid  $S$  in the previous section, and let  $D$  denote the set  $\{0, 1/10N, \dots, 1\}$ , i.e., the  $\delta$ -discretization of  $[0, 1]$ . For agent  $i \in \{1, 2, 3, 4\}$ , its valuation function  $v_i$  is constructed as follows. We first specify the values  $v_i(a, b)$  for all  $a, b$  in  $D$ . Then we use the piecewise linear interpolation described in Section 5 to obtain a continuous valuation function.

At a high level, the valuation of an agent for an interval of the cake will depend on the Sperner labeling described in the previous section, when the endpoints of the interval are close to 0.25 and 0.5, or to 0.5 and 0.75. Otherwise, the valuation will essentially be the length of the interval.

We think of the Sperner grid from the previous section as being embedded in the grid  $(D \cap [0.2, 0.3]) \times (D \cap [0.45, 0.55])$ . Note that this grid has size exactly  $(N+1) \times (N+1)$ , so there is indeed a straightforward bijection with the  $\{0, \dots, N\} \times \{0, \dots, N\}$  Sperner grid. The labeling  $\lambda_i$  from the previous section is represented here by functions  $\text{first-label}_i$  and  $\text{second-label}_i$  that map a point in  $(D \cap [0.2, 0.3]) \times (D \cap [0.45, 0.55])$  to the corresponding (single-coordinate) label in  $\{+1, -1\}$ . Namely, if  $\lambda_i$  yields label  $(+1, -1)$ , then  $\text{first-label}_i$  outputs  $+1$ , and  $\text{second-label}_i$  outputs  $-1$ .

We let  $\beta := \delta/8$ , and  $\gamma := \beta/8 = \delta/64$ . For  $a, b \in D \subset [0, 1]$ , we let  $v_i(a, b) := \text{length}([a, b]) = b - a$  for all  $a, b \in D \subset [0, 1]$ , except in the following cases:

- When  $b \in D \cap [0.2, 0.3]$ , we let

$$v_i(0, b) := \text{length}([0, b]) + \text{boost}_i(b).$$

- When  $a \in D \cap [0.2, 0.3]$  and  $b \in D \cap [0.45, 0.55]$ , we let

$$v_i(a, b) := v_i(0, a) + \gamma \cdot \text{first-label}_i(a, b).$$

<sup>15</sup>Here we use the fact that the paths have internal width two.

- When  $a \in D \cap [0.45, 0.55]$  and  $b \in D \cap [0.7, 0.8]$ , we let

$$v_i(a, b) := v_i(b, 1) + \gamma \cdot \text{second-label}_i(1 - b, a).$$

In particular, note that we always have  $v_i(a, 1) = \text{length}([a, 1]) = 1 - a$  for all  $a \in D$ .

**The boost function.** The function  $\text{boost}_i : D \cap [0.2, 0.3] \rightarrow \{-\beta, 0, \beta\}$  is constructed as follows. The value  $\text{boost}_i(b)$  is

- $+\beta$  if  $b$ , interpreted as an  $x$ -coordinate in the original Sperner grid  $S$ , lies in a column  $c_1^+$  or  $c_2^+$  of a crossing region in which agent  $i$  sees a crossing.
- $-\beta$  if  $b$ , interpreted as an  $x$ -coordinate in the original Sperner grid  $S$ , lies in a column  $c_1^-$  or  $c_2^-$  of a crossing region in which agent  $i$  sees a crossing.
- 0 otherwise.

Note that  $\text{boost}_i$  is well-defined, since the construction in the previous section ensures that different crossing regions never overlap in terms of the  $x$ -coordinate.

**Intuition.** At a given division  $(\ell, m, r)$  of the cake, with  $\ell \in [0.2, 0.3]$ ,  $m \in [0.45, 0.55]$ ,  $r \in [0.7, 0.8]$ , the Sperner labels influence the agent's preferences as follows. The first label (i.e.,  $-1$  in label  $(-1, +1)$ ) determines which piece is preferred between the first two pieces (i.e., the two leftmost pieces). If the first label is  $-1$ , then the first piece is preferred to the second piece. If the first label is  $+1$ , then the second piece is preferred to the first piece. Similarly, the second label determines which piece is preferred between the two last pieces (i.e., the two rightmost pieces). If the second label is  $+1$ , then the third piece is preferred to the fourth piece.

**Further observations.** Since the valuation function  $v_i$  is constructed by piecewise linear interpolation, we can write for all  $\ell \in [0.2, 0.3]$  and  $m \in [0.45, 0.55]$

$$v_i(\ell, m) = v_i(0, \ell) + \gamma \cdot \text{first-label}_i(\ell, m)$$

where we abuse notation to let  $\text{first-label}_i$  also denote the piecewise linear interpolation of  $\text{first-label}_i$ . Similarly, we also have that for all  $m \in [0.45, 0.55]$  and  $r \in [0.7, 0.8]$

$$v_i(m, r) = v_i(r, 1) + \gamma \cdot \text{second-label}_i(1 - r, m)$$

where again  $\text{second-label}_i$  also denotes the piecewise linear interpolation of  $\text{second-label}_i$ . Note that both  $\text{first-label}_i$  and  $\text{second-label}_i$  are Lipschitz-continuous with Lipschitz constant  $2/\delta$ .

We can also write for all  $\ell \in [0, 1]$

$$v_i(0, \ell) = \ell + \text{boost}_i(\ell)$$

where we have abused notation to let  $\text{boost}_i$  denote the piecewise linear interpolation of  $\text{boost}_i$ , and where  $\text{boost}_i(\ell) = 0$  for  $\ell \notin [0.2, 0.3]$ .

### 8.3 Analysis

Let  $\varepsilon := \gamma/8$ . Note that  $\delta > \beta > \gamma > \varepsilon > 0$ . Furthermore, note that  $\varepsilon = \Theta(1/n^4)$  and thus,  $\text{poly}(n) = \text{poly}(1/\varepsilon)$ . In this section, we show that any  $\varepsilon$ -envy-free allocation of the cake among the four agents with valuations  $v_1, v_2, v_3, v_4$  yields a solution to the INTERSECTION END-OF-LINE problem. Let  $(\ell, m, r)$  be an  $\varepsilon$ -envy-free division of the cake. We begin with the following claim which states that  $\ell \approx 1 - r$ . We use the notation  $x = y \pm z$  as a shorthand for  $x \in [y - z, y + z]$ .

**CLAIM 8.** *It necessarily holds that  $\ell = 1 - r \pm 2\beta$ . Furthermore, if  $\text{boost}_i(\ell) = 0$  for all agents  $i \in \{1, 2, 3, 4\}$ , then we have  $\ell = 1 - r \pm \varepsilon$ .*

**PROOF.** Note that by construction, we have that for all agents  $i \in \{1, 2, 3, 4\}$ ,  $v_i(r, 1) = 1 - r$ , and  $v_i(0, \ell) = \ell \pm \beta$ . Since  $(\ell, m, r)$  is an  $\varepsilon$ -envy-free division, there exists an agent  $i$  that is allocated piece  $[0, \ell]$  and thus, satisfies  $v_i(0, \ell) \geq v_i(r, 1) - \varepsilon$ , which yields  $\ell + \beta \geq 1 - r - \varepsilon$ , i.e.,  $\ell \geq 1 - r - \varepsilon - \beta$ . Similarly, there exists an agent  $i$  that is allocated piece  $[r, 1]$  and thus satisfies  $v_i(r, 1) \geq v_i(0, \ell) - \varepsilon$ , which yields  $1 - r \geq \ell - \beta - \varepsilon$ , i.e.,  $\ell \leq 1 - r + \beta + \varepsilon$ . As a result, since  $\varepsilon \leq \beta$ , we indeed obtain that  $\ell = 1 - r \pm 2\beta$ .

If we additionally have that  $\text{boost}_i(\ell) = 0$  for all agents  $i \in \{1, 2, 3, 4\}$ , then  $v_i(0, \ell) = \ell$ , and the same arguments as above yield that  $\ell = 1 - r \pm \varepsilon$ .  $\square$

The analysis now proceeds as follows. In the next section, we prove that  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free if it lies outside the embedding region. Next, in Section 8.3.2, we prepare a toolbox for arguing that there are no unwanted solutions inside the embedding region, which we then proceed to prove in Section 8.3.3. Together, these three sections show that any  $\varepsilon$ -envy-free allocation must yield a solution to the INTERSECTION END-OF-LINE instance.

**8.3.1 Analysis: No Solutions outside the Embedding Region.** In this section, we show that if  $(\ell, m, r)$  is an  $\varepsilon$ -envy-free division of the cake, then it cannot lie outside the embedding region.

**CLAIM 9.** *It necessarily holds that  $\ell \in [0.2, 0.3]$ ,  $m \in [0.45, 0.55]$ , and  $r \in [0.7, 0.8]$ .*

**PROOF.** We consider the following four cases, and show that in each case the division cannot be  $\varepsilon$ -envy-free

- Regime 1:  $\ell \leq 0.2 + \delta$  and  $m \geq 0.5$ , or  $\ell \leq 0.25$  and  $m \geq 0.55$
- Regime 2:  $\ell \leq 0.2 + \delta$  and  $m \leq 0.5$ , or  $\ell \leq 0.25$  and  $m \leq 0.45$
- Regime 3:  $\ell \geq 0.3 - \delta$  and  $m \geq 0.5$ , or  $\ell \geq 0.25$  and  $m \geq 0.55$
- Regime 4:  $\ell \geq 0.3 - \delta$  and  $m \leq 0.5$ , or  $\ell \geq 0.25$  and  $m \leq 0.45$

It thus follows that we must have  $\ell \in [0.2 + \delta, 0.3 - \delta]$  and  $m \in [0.45, 0.55]$ . By Claim 8, we also necessarily have  $1 - r = \ell \pm 2\beta$ , and since  $2\beta \leq \delta$ , it follows that we also have  $r \in [0.7, 0.8]$ , as desired. In the remainder of this proof we now show that there is no  $\varepsilon$ -envy-free division in each of the four regimes.

**Regime 1.** For any pair  $(\ell, m)$  in Regime 1 that also lies on the  $D \times D$  grid, we show below that  $v_i(\ell, m) > v_i(0, \ell) + \varepsilon$  for all agents  $i \in \{1, 2, 3, 4\}$ . Since the valuations are constructed by piecewise linear interpolation, it then follows that  $v_i(\ell, m) > v_i(0, \ell) + \varepsilon$  also holds for points off the grid, i.e, for all  $(\ell, m)$  in Regime 1. As a result, the division cannot be  $\varepsilon$ -envy-free. It remains to prove that we indeed have  $v_i(\ell, m) > v_i(0, \ell) + \varepsilon$  for all grid points in Regime 1. If  $(\ell, m)$  lies on a grid point in Regime 1 inside the embedding region, then by construction of the labels on the boundary of the embedding region we must have  $\text{first-label}_i(\ell, m) = +1$ , because the top and left boundary of the Sperner instance have first label +1. As a result

$$v_i(\ell, m) = v_i(0, \ell) + \gamma \cdot \text{first-label}_i(\ell, m) = v_i(0, \ell) + \gamma > v_i(0, \ell) + \varepsilon$$

since  $\gamma > \varepsilon$ . On the other hand, if  $(\ell, m)$  lies on a grid point in Regime 1 outside the embedding region, then we consider two cases. If  $\ell \leq 0.2 + \delta$  and  $m \geq 0.5$ , then

$$v_i(\ell, m) = m - \ell \geq 0.5 - 0.2 - \delta > 0.25 > 0.2 + \delta + \beta + \varepsilon \geq \ell + \beta + \varepsilon \geq v_i(0, \ell) + \varepsilon$$

where we used  $v_i(0, \ell) \leq \ell + \beta$ , and  $\delta + \beta + \varepsilon < 0.05$ . If  $\ell \leq 0.25$  and  $m \geq 0.55$ , then

$$v_i(\ell, m) = m - \ell \geq 0.55 - 0.25 = 0.3 > 0.25 + \beta + \varepsilon \geq \ell + \beta + \varepsilon \geq v_i(0, \ell) + \varepsilon$$

where we used  $v_i(0, \ell) \leq \ell + \beta$ , and  $\beta + \varepsilon < 0.05$ .

**Regime 4.** We can argue that there is no solution in Regime 4 by using very similar arguments. For all grid points  $(\ell, m)$  inside the embedding region, we have  $\text{first-label}_i(\ell, m) = -1$  for all agents

$i \in \{1, 2, 3, 4\}$  by construction of the labels on the boundary (because the bottom and right boundary of the Sperner instance have first label  $-1$ ). As a result,

$$v_i(\ell, m) = v_i(0, \ell) + \gamma \cdot \text{first-label}_i(\ell, m) = v_i(0, \ell) - \gamma < v_i(0, \ell) - \varepsilon$$

since  $\gamma > \varepsilon$ . On the other hand, if  $(\ell, m)$  is a grid point in Regime 4 outside the embedding region, we again consider two cases. If  $\ell \geq 0.3 - \delta$  and  $m \leq 0.5$ , then

$$v_i(\ell, m) = m - \ell \leq 0.5 - 0.3 + \delta < 0.25 < 0.3 - \delta - \beta - \varepsilon \leq \ell - \beta - \varepsilon \leq v_i(0, \ell) - \varepsilon$$

where we used  $v_i(0, \ell) \geq \ell - \beta$ , and  $\delta + \beta + \varepsilon < 0.05$ . If  $\ell \geq 0.25$  and  $m \leq 0.45$ , then

$$v_i(\ell, m) = m - \ell \leq 0.45 - 0.25 = 0.2 < 0.25 - \beta - \varepsilon \leq \ell - \beta - \varepsilon \leq v_i(0, \ell) - \varepsilon$$

where we used  $v_i(0, \ell) \geq \ell - \beta$ , and  $\beta + \varepsilon < 0.05$ .

**Regime 2.** Next, we consider Regime 2. By Claim 8,  $\ell \leq 0.2 + \delta$  implies  $r \geq 1 - \ell - 2\beta \geq 0.8 - \delta - 2\beta \geq 0.8 - 2\delta$ , and similarly  $\ell \leq 0.25$  implies  $r \geq 0.75 - 2\beta \geq 0.75 - \delta$ , since  $\delta \geq 2\beta$ . Thus, it suffices to show that for any pair  $(m, r)$  satisfying

– Regime 2':  $r \geq 0.8 - 2\delta$  and  $m \leq 0.5$ , or  $r \geq 0.75 - \delta$  and  $m \leq 0.45$

the division cannot be  $\varepsilon$ -envy-free. As before, it will suffice to argue that  $v_i(m, r) > v_i(r, 1) + \varepsilon$  holds for all agents  $i \in \{1, 2, 3, 4\}$  and for all grid points  $(m, r) \in D \times D$  in Regime 2', since the valuations are constructed by piecewise linear interpolation. This will be enough to show that no agent is happy with piece  $[r, 1]$ , and thus the division cannot be  $\varepsilon$ -envy-free. Consider first any grid point  $(m, r)$  in Regime 2' such that  $(1 - r, m)$  lies inside the embedding region (i.e.,  $1 - r \in [0.2, 0.3]$  and  $m \in [0.45, 0.55]$ ). By construction of the labels on the boundary we have that  $\text{second-label}_i(1 - r, m) = +1$  (because the bottom and left boundary of the Sperner instance have second label  $+1$ ), and thus

$$v_i(m, r) = v_i(r, 1) + \gamma \cdot \text{second-label}_i(1 - r, m) = v_i(r, 1) + \gamma > v_i(r, 1) + \varepsilon$$

since  $\gamma > \varepsilon$ . On the other hand, if  $(m, r)$  is a grid point in Regime 2' such that  $(1 - r, m)$  lies outside the embedding region, then we have two cases. If  $r \geq 0.8 - 2\delta$  and  $m \leq 0.5$ , then

$$v_i(m, r) = r - m \geq 0.8 - 2\delta - 0.5 > 0.25 > 0.2 + 2\delta + \varepsilon \geq 1 - r + \varepsilon = v_i(r, 1) + \varepsilon$$

since  $2\delta + \varepsilon < 0.05$ . If  $r \geq 0.75 - \delta$  and  $m \leq 0.45$ , then

$$v_i(m, r) = r - m \geq 0.75 - \delta - 0.45 = 0.3 - \delta > 0.25 + \delta + \varepsilon \geq 1 - r + \varepsilon = v_i(r, 1) + \varepsilon$$

since  $2\delta + \varepsilon < 0.05$ .

**Regime 3.** We can argue that there is no solution in Regime 3 using very similar arguments. By Claim 8, we have that if  $(\ell, m)$  is in Regime 3, then  $(m, r)$  must satisfy

– Regime 3':  $r \leq 0.7 + 2\delta$  and  $m \geq 0.5$ , or  $r \leq 0.75 + \delta$  and  $m \geq 0.55$

For any grid point  $(m, r)$  in Regime 3' such that  $(1 - r, m)$  lies inside the embedding region, we have by construction of the labels on the boundary that  $\text{second-label}_i(1 - r, m) = -1$  for all agents  $i \in \{1, 2, 3, 4\}$  (because the top and right boundary of the Sperner instance have second label  $-1$ ), and thus

$$v_i(m, r) = v_i(r, 1) + \gamma \cdot \text{second-label}_i(1 - r, m) = v_i(r, 1) - \gamma < v_i(r, 1) - \varepsilon$$

since  $\gamma > \varepsilon$ . On the other hand, for any grid point  $(m, r)$  in Regime 3' such that  $(1 - r, m)$  lies outside the embedding region, we have two cases. If  $r \leq 0.7 + 2\delta$  and  $m \geq 0.5$ , then

$$v_i(m, r) = r - m \leq 0.7 + 2\delta - 0.5 < 0.25 < 0.3 - 2\delta - \varepsilon \leq 1 - r - \varepsilon = v_i(r, 1) - \varepsilon$$

since  $2\delta + \varepsilon < 0.05$ . If  $r \leq 0.75 + \delta$  and  $m \geq 0.55$ , then

$$v_i(m, r) = r - m \leq 0.75 + \delta - 0.55 = 0.2 + \delta < 0.25 - \delta - \varepsilon \leq 1 - r - \varepsilon = v_i(r, 1) - \varepsilon$$

since  $2\delta + \varepsilon < 0.05$ .  $\square$

**8.3.2 Analysis: Toolbox.** By the previous section, we can now assume that  $(\ell, m, r)$  is an  $\varepsilon$ -envy-free division with  $\ell \in [0.2, 0.3]$ ,  $m \in [0.45, 0.55]$ , and  $r \in [0.7, 0.8]$ . In particular,  $(\ell, m)$  is a point in the Sperner embedding and it lies inside a square of the Sperner discretization (or in multiple squares if it lies on the boundary between different squares). The following claims provide a toolbox for arguing that, depending on the labels at the corners of the square, the division cannot be  $\varepsilon$ -envy-free in various cases.

We begin with the following claim which will be heavily used in this section.

**CLAIM 10.** *It holds that*

$$|\text{second-label}_i(1 - r, m) - \text{second-label}_i(\ell, m)| \leq 1/2.$$

**PROOF.** Given that  $\text{second-label}_i$  is Lipschitz-continuous with Lipschitz constant  $2/\delta$ , by Claim 8 we have that

$$|\text{second-label}_i(1 - r, m) - \text{second-label}_i(\ell, m)| \leq \frac{2}{\delta} |1 - r - \ell| \leq \frac{4\beta}{\delta} \leq 1/2$$

since  $\beta \leq \delta/8$ .  $\square$

**CLAIM 11.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that all four agents agree on the labels of the four corners of the square, and these labels do not yield a Sperner solution, then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

This claim is a special case of the following stronger claim.

**CLAIM 12.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that the first or second label remains constant across all four agents and at all four corners (i.e., all the labels lie in one of the four following sets:  $\{(+1, +1), (+1, -1)\}$ ,  $\{(+1, +1), (-1, +1)\}$ ,  $\{(-1, -1), (-1, +1)\}$ , or  $\{(-1, -1), (+1, -1)\}$ ), then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

**PROOF.** Let us consider the case where for all agents the labels lie in  $\{(+1, +1), (+1, -1)\}$  at all four corners of the square containing  $(\ell, m)$ . Then, it must be that  $\text{first-label}_i(\ell, m) = +1$  for all agents  $i$ , and, as a result,  $v_i(\ell, m) = v_i(0, \ell) + \gamma > v_i(0, \ell) + \varepsilon$ , since  $\gamma > \varepsilon$ . But this means that no agent is happy with the leftmost piece and thus, the division cannot be  $\varepsilon$ -envy-free. The case  $\{(-1, -1), (-1, +1)\}$  is handled very similarly.

Next, we consider the case where for all agents the labels lie in  $\{(+1, +1), (-1, +1)\}$  at all four corners of the square containing  $(\ell, m)$ . Then, we necessarily have  $\text{second-label}_i(\ell, m) = +1$  for all agents  $i$ . By Claim 10, we have  $\text{second-label}_i(1 - r, m) \geq 1/2$ , and thus,  $v_i(m, r) = v_i(r, 1) + \gamma/2 > v_i(r, 1) + \varepsilon$ , since  $\gamma > 2\varepsilon$ . This means that no agent is happy with the rightmost piece and thus the division cannot be  $\varepsilon$ -envy-free. The case  $\{(-1, -1), (+1, -1)\}$  is handled very similarly.  $\square$

**CLAIM 13.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that for at least three agents all four corners of the square have the same identical label, then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

**PROOF.** It is easy to see that, in this case, there are two pieces that the three agents do not like. For example, if agents 1, 2, and 3 see identical label  $(+1, -1)$  at all four corners, then for all  $i \in \{1, 2, 3\}$  we have  $\text{first-label}_i(\ell, m) = +1$ , and  $\text{second-label}_i(1 - r, m) \leq -1/2$  (since  $\text{second-label}_i(\ell, m) = -1$  and by using Claim 10). This implies that for all three agents  $i \in \{1, 2, 3\}$ ,  $v_i(\ell, m) = v_i(0, \ell) + \gamma > v_i(0, \ell) + \varepsilon$ , and  $v_i(m, r) \leq v_i(r, 1) - \gamma/2 < v_i(r, 1) - \varepsilon$ , since  $\gamma > 2\varepsilon$ . In other words, none of the three agents is happy with piece  $[0, \ell]$  or piece  $[m, r]$ , and thus the division cannot be  $\varepsilon$ -envy-free. The arguments are very similar for the other three possible labels.  $\square$

**CLAIM 14.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that for at least two agents all four corners of the square have the environment label  $(+1, -1)$ , and if the boost function is zero at all four corners for all four agents, then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

**PROOF.** Without loss of generality, let agents 1 and 2 see the environment label  $(+1, -1)$  at all four corners. In that case, we must have that for  $i \in \{1, 2\}$ ,  $\text{first-label}_i(\ell, m) = +1$  and  $\text{second-label}_i(1 - r, m) \leq -1/2$  (since  $\text{second-label}_i(\ell, m) = -1$  and by using Claim 10). This implies that  $v_i(\ell, m) = v_i(0, \ell) + \gamma > v_i(0, \ell) + \varepsilon$ , and  $v_i(m, r) \leq v_i(r, 1) - \gamma/2 < v_i(r, 1) - \varepsilon$ , since  $\gamma > 2\varepsilon$ . In particular, the two agents strictly prefer piece  $[\ell, m]$  over piece  $[0, \ell]$ , and also strictly prefer piece  $[r, 1]$  over piece  $[m, r]$ . As a result, one of these two agents, say agent 1, must be allocated piece  $[r, 1]$ , which implies that, for the division to be  $\varepsilon$ -envy-free, we must have

$$1 - r = v_1(r, 1) \geq v_1(\ell, m) - \varepsilon = v_1(0, \ell) + \gamma - \varepsilon.$$

Now, given that agent 1 has boost function zero everywhere in the square, it follows that  $v_1(0, \ell) = \ell$ , and thus  $1 - r \geq \ell + \gamma - \varepsilon > \ell + \varepsilon$ , since  $\gamma > 2\varepsilon$ . But since all four agents have boost function equal to zero in the square, Claim 8 yields that  $\ell = 1 - r \pm \varepsilon$ , a contradiction.  $\square$

**CLAIM 15.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that three agents only see labels from  $\{(-1, +1), (+1, +1)\}$  or only from  $\{(+1, -1), (+1, +1)\}$ , and if the boost function is zero at all four corners for all four agents, then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

**PROOF.** Without loss of generality, let agents 1, 2, and 3 only see labels from  $\{(-1, +1), (+1, +1)\}$  at all four corners. Then, it must be that for all agents  $i \in \{1, 2, 3\}$ ,  $\text{second-label}_i(\ell, m) = +1$ , and thus  $\text{second-label}_i(1 - r, m) \geq 1/2$  by Claim 10. It follows that  $v_i(m, r) \geq v_i(r, 1) + \gamma/2 > v_i(r, 1) + \varepsilon$ , and thus all three agents strictly prefer piece  $[m, r]$  to piece  $(r, 1)$ . As a result, for the division to be  $\varepsilon$ -envy-free, one of the three agents, say agent 1, has to be allocated piece  $[0, \ell]$ , and satisfy

$$v_1(0, \ell) \geq v_1(m, r) - \varepsilon \geq v_1(r, 1) + \gamma/2 - \varepsilon = 1 - r + \gamma/2 - \varepsilon.$$

Given that agent 1 has boost function zero everywhere in the square, we have  $v_1(0, \ell) = \ell$ , and thus  $\ell \geq 1 - r + \gamma/2 - \varepsilon > 1 - r + \varepsilon$ , since  $\gamma > 4\varepsilon$ . Since all four agents have boost function equal zero in the square, Claim 8 yields  $\ell = 1 - r \pm \varepsilon$ , a contradiction. The case where the three agents only see labels from  $\{(+1, -1), (+1, +1)\}$  at all four corners is handled using the same arguments.  $\square$

**CLAIM 16.** *If  $(\ell, m)$  lies in a square of the Sperner embedding such that one of the following two cases holds*

- (1) *The boost function is  $+\beta$  at all four corners for some agent, and the other three agents have boost zero at all four corners and only labels in  $\{(+1, -1), (-1, -1)\}$ .*
- (2) *The boost function is  $-\beta$  at all four corners for some agent, and the other three agents have boost zero at all four corners and only labels in  $\{(-1, +1), (-1, -1)\}$ .*

*then  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.*

**PROOF.** Without loss of generality, let agents 1, 2, and 3 have boost zero in the square, and only see labels in  $\{(+1, -1), (-1, -1)\}$ , while agent 4 has boost  $+\beta$  in the square. For  $i \in \{1, 2, 3\}$ , we have  $\text{second-label}_i(\ell, m) = -1$ , and thus  $\text{second-label}_i(1 - r, m) \leq -1/2$  by Claim 10. This means that  $v_i(m, r) \leq v_i(r, 1) - \gamma/2 < v_i(r, 1) - \varepsilon$ , since  $\gamma > 2\varepsilon$ . As a result, all three agents strictly prefer piece  $[r, 1]$  to piece  $[m, r]$ . In particular, for the division to be  $\varepsilon$ -envy-free, agent 4 must necessarily be allocated piece  $[m, r]$ , and satisfy  $v_4(m, r) \geq v_4(0, \ell) - \varepsilon$ .

Furthermore, one of the three agents, say agent 1, must be allocated piece  $[0, \ell]$  and thus satisfy  $v_1(0, \ell) \geq v_1(r, 1) - \varepsilon = 1 - r - \varepsilon$ . Since agent 1 has boost zero in the square, we have  $v_1(0, \ell) = \ell$ , and thus deduce that  $\ell \geq 1 - r - \varepsilon$ .

Note that by construction of the valuation function, we have  $v_4(m, r) \leq v_4(r, 1) + \gamma = 1 - r + \gamma$ . On the other hand, given that agent 4 has boost  $+\beta$  in the square, we also have  $v_4(0, \ell) = \ell + \beta$ . Together, we obtain that

$$v_4(m, r) \leq 1 - r + \gamma \leq \ell + \varepsilon + \gamma = v_4(0, \ell) - \beta + \varepsilon + \gamma < v_4(0, \ell) - \varepsilon$$

where we used  $\beta > \gamma + 2\varepsilon$ . This means that agent 4 is not happy with its allocated piece, a contradiction.

For the case where the three agents have labels in  $\{(-1, +1), (-1, -1)\}$  only, using similar arguments we can deduce that all three agents strictly prefer piece  $[0, \ell]$  to piece  $[\ell, m]$ . In particular, agent 4 must be allocated piece  $[\ell, m]$  and satisfy  $v_4(\ell, m) \geq v_4(1, r) - \varepsilon$ . Since one of the three agents must be allocated piece  $[r, 1]$ , and that agent has boost zero in the square, we can deduce that  $1 - r \geq \ell - \varepsilon$ . Using the fact that agent 4 has boost  $-\beta$  in the square, we can now write

$$v_4(\ell, m) \leq v_4(0, \ell) + \gamma = \ell - \beta + \gamma \leq 1 - r + \varepsilon - \beta + \gamma = v_4(r, 1) + \varepsilon - \beta + \gamma < v_4(r, 1) - \varepsilon$$

where we used  $\beta > \gamma + 2\varepsilon$ . This means that agent 4 is not happy with its allocated piece, a contradiction.  $\square$

**8.3.3 Analysis: No Unwanted Solutions inside the Embedding Region.** In this last section of the proof, we use the toolbox created in the previous section, together with various properties of the construction of the embedding, to argue that no unwanted solutions occur inside the embedding region.

**No solutions outside crossing and vertex regions.** If  $(\ell, m)$  lies in a square outside any crossing or vertex region, then, by Claim 3, one of the following two cases occurs:

- (1) All four agents agree on the labels of the four corners of the square, and these labels do not correspond to a Sperner solution. By Claim 11,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (2) At least three agents see the environment label at all four corners of the square. By Claim 13,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.

Thus, we deduce that if  $(\ell, m, r)$  is  $\varepsilon$ -envy-free,  $(\ell, m)$  must lie in a crossing region or in a vertex region.

**No solutions in crossing regions.** If  $(\ell, m)$  lies in a square inside a crossing region, then, by Claim 6, one of the following four cases occurs:

- (1) All four agents agree on the labels of the four corners of the square, and these labels do not correspond to a Sperner solution. By Claim 11,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (2) At least three agents see the environment label at all four corners of the square. By Claim 13,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (3) Two agents see no path in the crossing region, and the other two agents see different single paths. In particular, by construction, the boost function is zero for all four agents in the crossing region, since no agent sees a crossing. As a result, we have two agents who see the environment label at all four corners of the square, and the boost function is zero at all four corners for all four agents. By Claim 14,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (4) One agent, say agent  $i$ , sees a crossing, and the other three agents see the same single path (horizontal or vertical). In that case, by Claim 7, one of the following cases occurs:
  - (a) All four agents agree on the labels at the four corners, and there is no Sperner solution. By Claim 11,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
  - (b) At least three agents agree and they see the same identical label at all four corners. By Claim 13,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
  - (c) The square lies in column  $c_1^+$  or  $c_2^+$ , and the three agents (other than  $i$ ) only see labels from  $\{(+1, -1), (-1, -1)\}$  in the square. By construction, agent  $i$  has boost  $+\beta$  at all four corners,

while the other agents have boost 0 at all corners. Thus, by Claim 16,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.

- (d) The square lies in column  $c_1^-$  or  $c_2^-$ , and the three agents (other than  $i$ ) only see labels from  $\{(-1, +1), (-1, -1)\}$  in the square. By construction, agent  $i$  has boost  $-\beta$  at all four corners, while the other agents have boost 0 at all corners. Thus, by Claim 16,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (e) The square is not incident on any of the columns  $c_1^+, c_1^-, c_2^+, c_2^-$ , and three agents see labels only from  $\{(-1, +1), (+1, +1)\}$  or only from  $\{(+1, -1), (+1, +1)\}$  in the square. By construction, all four agents have boost 0 at all corners. Thus, by Claim 15,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.

**No solutions in non-end-of-line vertex regions.** If  $(\ell, m)$  lies in a square inside a vertex region, then by Claim 5, one of the following cases occurs:

- (1) All four agents agree on the labels at the four corners, and there is no Sperner solution. By Claim 11,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (2) At least three agents agree and they see the same identical label at all four corners. By Claim 13,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (3) At least three agents see labels only in  $\{(-1, +1), (+1, +1)\}$ , or only in  $\{(+1, -1), (+1, +1)\}$ , and the square touches the connector region. Then, all four agents have boost 0 at all corners, since by construction the connector region does not overlap with any crossing region in terms of the  $x$ -coordinates. Thus, by Claim 15,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.
- (4) All four agents see labels only in  $\{(+1, -1), (-1, -1)\}$ , or only in  $\{(-1, +1), (-1, -1)\}$ . By Claim 12,  $(\ell, m, r)$  cannot be  $\varepsilon$ -envy-free.

As a result, if  $(\ell, m, r)$  is  $\varepsilon$ -envy-free, then  $(\ell, m)$  must lie in an end-of-line vertex region. This completes the proof.

## 8.4 Obtaining PPAD-hardness and a Query Lower Bound for Identical Agents

The same construction can be used to show PPAD-hardness and a query lower bound for the problem with identical agents. To do this, we reduce from the standard END-OF-LINE problem which is known to be PPAD-complete and to require  $\text{poly}(n)$  queries. Every agent sees the same END-OF-LINE instance and thus, has the same valuation function. Note that most of the arguments become much simpler, because all the agents agree on the labeling everywhere. Furthermore, there is no need to ever overwrite the connector region with label  $(-1, -1)$ , since all nodes have in-degree and out-degree at most one. It is easy to check that the construction can be implemented in a query-efficient way and in polynomial time, and thus, yields a valid reduction in the query complexity and white-box settings.

## Acknowledgments

We would like to thank the reviewers for useful comments and suggestions, and in particular one reviewer for suggesting a simplification of the proof in Section 4. We are also grateful to Simina Brânzei for helpful discussions.

## References

- [1] Eshwar Ram Arunachaleswaran, Siddharth Barman, Rachitesh Kumar, and Nidhi Rathi. 2019. Fair and efficient cake division with connected pieces. In *Proceedings of the 15th International Conference on Web and Internet Economics (WINE)*. 57–70. DOI : [https://doi.org/10.1007/978-3-030-35389-6\\_5](https://doi.org/10.1007/978-3-030-35389-6_5)
- [2] Yonatan Aumann, Yair Dombb, and Avinatan Hassidim. 2013. Computing socially-efficient cake divisions. In *Proceedings of the 12th International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 343–350. DOI : <https://dl.acm.org/doi/10.5555/2484920.2484976>

- [3] Yonatan Aumann, Yair Dombb, and Avinatan Hassidim. 2014. Auctioning a cake: Truthful auctions of heterogeneous divisible goods. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 1045–1052. DOI : <https://dl.acm.org/doi/10.5555/2615731.2617412>
- [4] Haris Aziz and Simon Mackenzie. 2020. A bounded and envy-free cake cutting algorithm. *Communications of the ACM* 63, 4 (2020), 119–126. DOI : <https://doi.org/10.1145/3382129>
- [5] Yakov Babichenko and Aviad Rubinstein. 2020. Communication complexity of nash equilibrium in potential games (extended abstract). In *Proceedings of the 61st IEEE Symposium on Foundations of Computer Science (FOCS)*. 1439–1445. DOI : <https://doi.org/10.1109/FOCS46700.2020.00137>
- [6] Julius B. Barbanel and Steven J. Brams. 2004. Cake division with minimal cuts: Envy-free procedures for three persons, four persons, and beyond. *Mathematical Social Sciences* 48, 3 (2004), 251–269. DOI : <https://doi.org/10.1016/j.mathsocsci.2004.03.006>
- [7] Siddharth Barman and Pooja Kulkarni. 2023. Approximation algorithms for envy-free cake division with connected pieces. In *Proceedings of the 50th International Colloquium on Automata, Languages, and Programming (ICALP)*. 16:1–16:19. DOI : <https://doi.org/10.4230/LIPIcs.ICALP.2023.16>
- [8] Siddharth Barman and Nidhi Rathi. 2022. Fair cake division under monotone likelihood ratios. *Mathematics of Operations Research* 47, 3 (2022), 1875–1903. DOI : <https://doi.org/10.1287/moor.2021.1192>
- [9] Xiaohui Bei, Ning Chen, Guangda Huzhang, Biaoshuai Tao, and Jiajun Wu. 2017. Cake cutting: Envy and truth. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*. 3625–3631. DOI : <https://doi.org/10.24963/ijcai.2017/507>
- [10] Steven J. Brams and Alan D. Taylor. 1996. *Fair Division: From Cake-cutting to Dispute Resolution*. Cambridge University Press.
- [11] Steven J. Brams, Alan D. Taylor, and William S. Zwicker. 1997. A moving-knife solution to the four-person envy-free cake-division problem. *Proceedings of the American Mathematical Society* 125, 2 (1997), 547–554. DOI : <https://doi.org/10.1090/S0002-9939-97-03614-9>
- [12] Simina Brânzei. 2015. A note on envy-free cake cutting with polynomial valuations. *Information Processing Letters* 115, 2 (2015), 93–95. DOI : <https://doi.org/10.1016/j.ipl.2014.07.005>
- [13] Simina Brânzei and Noam Nisan. 2019. Communication complexity of cake cutting. In *Proceedings of the 2019 ACM Conference on Economics and Computation (EC)*. 525. DOI : <https://doi.org/10.1145/3328526.3329644>
- [14] Simina Brânzei and Noam Nisan. 2022. The query complexity of cake cutting. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS)*. 37905–37919. Retrieved from [https://proceedings.neurips.cc/paper\\_files/paper/2022/hash/f7a7bb369e48f10e85fce85b67d8c516-Abstract-Conference.html](https://proceedings.neurips.cc/paper_files/paper/2022/hash/f7a7bb369e48f10e85fce85b67d8c516-Abstract-Conference.html)
- [15] Xi Chen and Xiaotie Deng. 2009. On the complexity of 2D discrete fixed point problem. *Theoretical Computer Science* 410, 44 (2009), 4448–4456. DOI : <https://doi.org/10.1016/j.tcs.2009.07.052>
- [16] Yiling Chen, John K. Lai, David C. Parkes, and Ariel D. Procaccia. 2013. Truth, justice, and cake cutting. *Games and Economic Behavior* 77, 1 (2013), 284–297. DOI : <https://doi.org/10.1016/j.geb.2012.10.009>
- [17] Xiaotie Deng, Qi Qi, and Amin Saberi. 2012. Algorithmic solutions for envy-free cake cutting. *Operational Research* 60, 6 (2012), 1461–1476. DOI : <https://doi.org/10.1287/opre.1120.1116>
- [18] John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. 2022. The complexity of gradient descent: CLS = PPAD  $\cap$  PLS. *Journal of the ACM* 70, 1 (2022), 1–74. DOI : <https://doi.org/10.1145/3568163>
- [19] George Gamow and Marvin Stern. 1958. *Puzzle-Math*. Viking Press.
- [20] Anat Ganor, Karthik C. S., and Dömöör Pálvölgyi. 2021. On communication complexity of fixed point computation. *ACM Transactions on Economics and Computation* 9, 4 (2021), 25:1–25:27. DOI : <https://doi.org/10.1145/3485004>
- [21] Paul W. Goldberg and Alexandros Hollender. 2021. The hairy ball problem is PPAD-complete. *Journal of Computer and System Sciences* 122 (2021), 34–62. DOI : <https://doi.org/10.1016/j.jcss.2021.05.004>
- [22] Paul W. Goldberg, Alexandros Hollender, and Warut Suksompong. 2020. Contiguous cake cutting: Hardness results and approximation algorithms. *Journal of Artificial Intelligence Research* 69 (2020), 109–141. DOI : <https://doi.org/10.1613/jair.1.12222>
- [23] Mika Göös and Toniann Pitassi. 2018. Communication lower bounds via critical block sensitivity. *SIAM Journal on Computing* 47, 5 (2018), 1778–1806. DOI : <https://doi.org/10.1137/16M1082007>
- [24] Mika Göös and Aviad Rubinstein. 2018. Near-optimal communication lower bounds for approximate nash equilibria. In *Proceedings of the 59th IEEE Symposium on Foundations of Computer Science (FOCS)*. 397–403. DOI : <https://doi.org/10.1109/FOCS.2018.00045>
- [25] Michael D. Hirsch, Christos H. Papadimitriou, and Stephen A. Vavasis. 1989. Exponential lower bounds for finding brouwer fix points. *Journal of Complexity* 5, 4 (1989), 379–416. DOI : [https://doi.org/10.1016/0885-064X\(89\)90017-4](https://doi.org/10.1016/0885-064X(89)90017-4)
- [26] Avishay Maya and Noam Nisan. 2012. Incentive compatible two player cake cutting. In *Proceedings of the 8th International Workshop on Internet and Network Economics (WINE)*. 170–183. DOI : [https://doi.org/10.1007/978-3-642-35311-6\\_13](https://doi.org/10.1007/978-3-642-35311-6_13)

- [27] Elchanan Mossel and Omer Tamuz. 2010. Truthful fair division. In *Proceedings of the 3rd International Symposium on Algorithmic Game Theory (SAGT)*, 288–299. DOI : [https://doi.org/10.1007/978-3-642-16170-4\\_25](https://doi.org/10.1007/978-3-642-16170-4_25)
- [28] Josué Ortega and Erel Segal-Halevi. 2022. Obvious manipulations in cake-cutting. *Social Choice and Welfare* 59, 4 (2022), 969–988. DOI : <https://doi.org/10.1007/s00355-022-01416-4>
- [29] Christos H. Papadimitriou. 1994. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences* 48, 3 (1994), 498–532. DOI : [https://doi.org/10.1016/S0022-0000\(05\)80063-7](https://doi.org/10.1016/S0022-0000(05)80063-7)
- [30] Ariel D. Procaccia. 2013. Cake cutting: Not just child's play. *Communications of the ACM* 56, 7 (2013), 78–87. DOI : <https://doi.org/10.1145/2483852.2483870>
- [31] Ariel D. Procaccia and Junxing Wang. 2017. A lower bound for equitable cake cutting. In *Proceedings of the 18th Conference on Economics and Computation (EC)*, 479–495. DOI : <https://doi.org/10.1145/3033274.3085107>
- [32] Jack Robertson and William Webb. 1998. *Cake-cutting Algorithms: Be Fair if You Can*. CRC Press.
- [33] Tim Roughgarden and Omri Weinstein. 2016. On the communication complexity of approximate fixed points. In *Proceedings of the 57th IEEE Symposium on Foundations of Computer Science (FOCS)*, 229–238. DOI : <https://doi.org/10.1109/FOCS.2016.32>
- [34] Erel Segal-Halevi and Shmuel Nitzan. 2019. Fair cake-cutting among families. *Social Choice and Welfare* 53, 4 (2019), 709–740. DOI : <https://doi.org/10.1007/s00355-019-01210-9>
- [35] Erel Segal-Halevi and Warut Suksompong. 2021. How to cut a cake fairly: A generalization to groups. *The American Mathematical Monthly* 128, 1 (2021), 79–83. DOI : <https://doi.org/10.1080/00029890.2021.1835338>
- [36] Erel Segal-Halevi and Warut Suksompong. 2023. Cutting a cake fairly for groups revisited. *The American Mathematical Monthly* 130, 3 (2023), 203–213. DOI : <https://doi.org/10.1080/00029890.2022.2153566>
- [37] Alexander A. Sherstov. 2014. Communication lower bounds using directional derivatives. *Journal of the ACM* 61, 6 (2014), 34:1–34:71. DOI : <https://doi.org/10.1145/2629334>
- [38] Hugo Steinhaus. 1948. The problem of fair division. *Econometrica* 16, 1 (1948), 101–104. Retrieved from <https://www.jstor.org/stable/1914289>
- [39] Walter Stromquist. 1980. How to cut a cake fairly. *The American Mathematical Monthly* 87, 8 (1980), 640–644. DOI : <https://doi.org/10.1080/00029890.1980.11995109>
- [40] Walter Stromquist. 2008. Envy-free cake divisions cannot be found by finite protocols. *The Electronic Journal of Combinatorics* 15, 1 (2008), R11. DOI : <https://doi.org/10.37236/735>
- [41] Francis Edward Su. 1999. Rental harmony: Sperner's lemma in fair division. *The American Mathematical Monthly* 106, 10 (1999), 930–942. DOI : <https://doi.org/10.1080/00029890.1999.12005142>
- [42] Biaoshuai Tao. 2022. On existence of truthful fair cake cutting mechanisms. In *Proceedings of the 23rd ACM Conference on Economics and Computation (EC)*, 404–434. DOI : <https://doi.org/10.1145/3490486.3538321>
- [43] Gerhard J. Woeginger and Jiří Sgall. 2007. On the complexity of cake cutting. *Discrete Optimization* 4, 2 (2007), 213–220. DOI : <https://doi.org/10.1016/j.disopt.2006.07.003>
- [44] Douglas R. Woodall. 1980. Dividing a cake fairly. *Journal of Mathematical Analysis and Applications* 78, 1 (1980), 233–247. DOI : [https://doi.org/10.1016/0022-247X\(80\)90225-5](https://doi.org/10.1016/0022-247X(80)90225-5)

## Appendices

### A Envy-free cake-cutting: Reduction from $n$ to $n + 1$ agents

Below, we present a reduction from  $n$  to  $n + 1$  agents for the envy-free cake-cutting problem. The reduction applies to the setting with general valuations, and also to the setting with monotone valuations. Whether there is a way to also achieve this for additive valuations is an interesting open question.

**LEMMA A.1.** *The connected  $\epsilon$ -envy-free cake-cutting problem with  $n$  agents with general valuations reduces to the same problem with  $n + 1$  agents. This reduction is efficient in all three models (query complexity, communication complexity, and computational complexity). Furthermore, the reduction maps an instance with  $n$  monotone agents to an instance with  $n + 1$  monotone agents. Finally, for the query and computational complexity models, there is also a version of the reduction that reduces instances with  $n$  identical agents to  $n + 1$  identical agents.*

**PROOF.** Let  $v_1, \dots, v_n$  denote the 1-Lipschitz continuous valuation functions of the  $n$  agents. For  $i \in [n]$  and  $a, b \in [0, 1]$ , we let

$$v'_i(a, b) := \frac{1}{3}v_i([2a, 2b] \cap [0, 1]) + \frac{2}{3}\phi\left(\left|[a, b] \cap [1/2, 1]\right|\right)$$

and

$$v'_{n+1}(a, b) := \frac{2}{3}\phi(|[a, b] \cap [1/2, 1]|)$$

where  $\phi : [0, 1/2] \rightarrow [0, 1]$  is defined as

$$\phi(t) = \begin{cases} 0 & \text{if } t \leq 1/3 \\ 6(t - 1/3) & \text{if } t \geq 1/3 \end{cases}$$

Note that  $v'_1, \dots, v'_{n+1}$  are 5-Lipschitz-continuous valuation functions (which can easily be normalized to be 1-Lipschitz continuous). Furthermore, if  $v_1, \dots, v_n$  are monotone, then so are  $v'_1, \dots, v'_{n+1}$ . Finally, the new valuations can easily be constructed/simulated from the original ones in all three models of computation.

Consider any  $\varepsilon$ -envy-free allocation  $A'_1, \dots, A'_n, A'_{n+1}$  with respect to  $v'_1, \dots, v'_{n+1}$ . We claim that, without loss of generality (and possibly by switching to a  $2\varepsilon$ -envy-free allocation), the piece  $A'_{n+1}$  obtained by agent  $n + 1$  is the rightmost piece, i.e.,  $A'_{n+1} = [a, 1]$  for some  $a \in [0, 1]$ . Indeed, if  $A'_{n+1}$  is empty, then this trivially holds. Otherwise,  $A'_{n+1} = [a, b]$  with  $a < b < 1$ . Let  $i$  be an agent who gets the rightmost piece, i.e.,  $A_i = [c, 1]$  for some  $c \geq b$ . We distinguish between the following two cases:

- $\phi(|[c, 1] \cap [1/2, 1]|) > 3\varepsilon/2$ : In that case, it must be that  $c < 2/3$ , and thus  $b < 2/3$ , which implies  $v'_{n+1}(a, b) = (2/3) \cdot \phi(|[a, b] \cap [1/2, 1]|) = 0$ . Since  $v'_{n+1}(c, 1) = (2/3) \cdot \phi(|[c, 1] \cap [1/2, 1]|) > \varepsilon$ , this contradicts the  $\varepsilon$ -envy-freeness of the division.
- $\phi(|[c, 1] \cap [1/2, 1]|) \leq 3\varepsilon/2$ : In that case,  $c \geq 1/2$  and thus  $v'_i(c, 1) = v'_{n+1}(c, 1) = (2/3) \cdot \phi(|[c, 1] \cap [1/2, 1]|) \leq \varepsilon$ . By construction of  $\phi$ , at most one of  $v'_{n+1}(a, b)$  and  $v'_{n+1}(c, 1)$  can be strictly positive. Thus, if  $v'_{n+1}(a, b) > \varepsilon$ , then  $v'_{n+1}(c, 1) = 0$ . It follows that  $v'_i(c, 1) = 0$  and  $v'_i(a, b) > \varepsilon$ , a contradiction to  $\varepsilon$ -envy-freeness. As a result, we must also have  $v'_{n+1}(a, b) \leq \varepsilon$ . By  $\varepsilon$ -envy-freeness we obtain that  $v'_{n+1}(A_j) \leq 2\varepsilon$  and  $v'_i(A_j) \leq 2\varepsilon$  for all  $j \in [n+1]$ . As a result, assigning  $A_i$  to agent  $n + 1$ , and  $A_{n+1}$  to agent  $i$  yields a  $2\varepsilon$ -envy-free allocation in which agent  $n + 1$  has the rightmost piece.

We can thus assume without loss of generality that we have an  $\varepsilon$ -envy-free allocation  $A'_1, \dots, A'_n, A'_{n+1}$  with respect to  $v'_1, \dots, v'_{n+1}$ , where  $A'_{n+1}$  is the rightmost piece. We define the allocation  $A_1, \dots, A_n$  by letting  $A_i := 2 \cdot (A'_i \cap [0, 1/2])$ . Note that this is indeed a partition of  $[0, 1]$ , because  $A'_{n+1} \subset [1/2, 1]$ . Otherwise, the rightmost piece  $A'_{n+1}$  would have value at least  $2/3$  to all agents, while any other piece would have value at most  $1/3$ .

Finally, we argue that the allocation  $A_1, \dots, A_n$  must be  $6\varepsilon$ -envy-free with respect to  $v_1, \dots, v_n$ . First of all, note that  $v'_{n+1}(A'_j) \leq \varepsilon$  for all  $j \in [n]$ . Otherwise, if  $v'_{n+1}(A'_j) > \varepsilon$  for some  $j$ , then  $v'_{n+1}(A'_{n+1}) = 0$ , and agent  $n + 1$  would be envious. Thus, it follows that  $v'_i(A'_j) \in [(1/3) \cdot v_i(A_j), (1/3) \cdot v_i(A_j) + \varepsilon]$  for all  $i, j \in [n]$ . As a result, for all  $i, j \in [n]$  we have

$$v_i(A_i) \geq 3(v'_i(A'_i) - \varepsilon) \geq 3(v'_i(A'_j) - 2\varepsilon) \geq v_i(A_j) - 6\varepsilon$$

as desired.

For the setting of identical valuations we proceed similarly. Given a valuation  $v$  shared by  $n$  agents, we define

$$v'(a, b) := \frac{1}{3}v([2a, 2b] \cap [0, 1]) + \frac{2}{3}\phi(|[a, b] \cap [1/2, 1]|)$$

to be shared by  $n + 1$  agents. Consider an  $\varepsilon$ -envy-free allocation  $A'_1, \dots, A'_n, A'_{n+1}$ , and rename the agents if needed such that  $A'_{n+1}$  is the rightmost piece. As above, we can argue that  $A'_{n+1} \subset [1/2, 1]$ , and define an allocation  $A_1, \dots, A_n$  of  $[0, 1]$ . Note that  $v'(A'_{n+1}) = (2/3) \cdot \phi(|A'_{n+1} \cap [1/2, 1]|)$ . If

$v'(A'_{n+1}) > 0$ , then  $v'(A'_j) = (1/3) \cdot v(A_j)$  for all  $j \in [n]$ , and we argue as above. If  $v'(A'_{n+1}) = 0$ , then  $v'(A'_j) \leq \varepsilon$  for all  $j \in [n]$ , and thus  $v(A_j) \leq 3\varepsilon$ , and the division is  $3\varepsilon$ -envy-free.  $\square$

## B Communication Protocols

Brânzei and Nisan [13] proved that for three agents with additive valuations an  $\varepsilon$ -envy-free allocation can be found using  $O(\log(1/\varepsilon))$  communication. This is achieved by a reduction to the *monotone-crossing* problem, which they introduce and for which they prove a  $O(\log(1/\varepsilon))$  communication bound. Their approach can easily be extended to agents with monotone valuations.

**LEMMA B.1.** *For three agents with monotone valuations, there exists a  $O(\log(1/\varepsilon))$  communication protocol that finds an  $\varepsilon$ -envy-free connected allocation.*

For completeness, we provide a proof sketch.

**PROOF SKETCH.** In the first step of the protocol, our goal is to find positions  $a < b$  such that one agent is indifferent between the three pieces resulting from cutting at  $a$  and  $b$ , while at least one of the other two agents prefers the middle piece  $[a, b]$ . In the second step, given such  $a$  and  $b$ , we then use the monotone-crossing problem to obtain a solution.

For the first step, every agent  $i$  communicates positions  $\ell_i$  and  $r_i$  such that  $v_i(0, \ell_i) = v_i(\ell_i, r_i) = v_i(r_i, 1)$ . Using some simple preprocessing we can make sure that those positions are unique and that they can be described using  $O(\log(1/\varepsilon))$  bits. Now one of the following two cases must occur:

- There exist two agents, say agents 1 and 2, such that  $\ell_1 < \ell_2$ , but  $r_2 < r_1$ . In that case, we let  $a := \ell_1$  and  $b := r_1$ . Note that agent 1 is indifferent between the three pieces when we cut at  $a$  and  $b$ . Furthermore, by monotonicity of  $v_2$ , agent 2 prefers the middle piece  $[a, b]$ . Thus,  $a$  and  $b$  satisfy the desiderata and we can move to the second step of the protocol.
- By renaming the agents if needed, we have  $\ell_1 \leq \ell_2 \leq \ell_3$  and  $r_1 \leq r_2 \leq r_3$ . In that case, we let  $a := \ell_2$  and  $b := r_2$ . Clearly, agent 2 is indifferent between the three pieces. Note that by monotonicity of  $v_1$ , agent 1 likes the leftmost piece at least as much as the rightmost piece. Similarly, agent 3 likes the rightmost piece at least as much as the leftmost piece. If one of them prefers the middle piece,  $a$  and  $b$  satisfy the desiderata and we proceed with the second step. Otherwise, giving the leftmost piece to agent 1, the rightmost piece to agent 3, and the middle piece to agent 2 yields an envy-free allocation.

Given  $a$  and  $b$  satisfying the desiderata, the second step of the protocol proceeds as follows. After renaming the agents if needed, we have that agent 1 is indifferent between the three pieces, while agent 2 prefers the middle piece. Let  $c_1$  denote the midpoint of the valuation of agent 1, i.e.,  $v_1(0, c) = v_1(c, 1)$ , and  $c_2$  denote the midpoint of the valuation of agent 1, i.e.,  $v_2(0, c) = v_2(c, 1)$ . Note that  $c_1, c_2 \in [a, b]$ .

For any  $\ell \in [a, c_1]$ , let  $r(\ell)$  denote the cut satisfying  $v_1(0, \ell) = v_1(r(\ell), 1)$ . Note that  $r(a) = b$ ,  $r(c_1) = c_1$ , and  $r(\cdot)$  decreases monotonically. Similarly, for any  $\ell \in [a, c_2]$ , let  $r'(\ell)$  denote the cut satisfying  $v_2(\ell, r'(\ell)) = \max\{v_2(0, \ell), v_2(r'(\ell), 1)\}$ . It can be shown that  $r'(a) \leq b$ ,  $r'(c_2) = 1$ , and  $r'(\cdot)$  increases monotonically. If  $c_2 \leq c_1$ , then  $r(\cdot)$  and  $r'(\cdot)$  must cross for some  $\ell \in [a, c_2]$ . If  $c_1 \leq c_2$ , then  $r(\cdot)$  and  $r'(\cdot)$  must cross for some  $\ell \in [a, c_1]$ . In either case, we can find such a crossing point (approximately) using  $O(\log(1/\varepsilon))$  communication by reducing to the monotone-crossing problem. We refer to [13] for the details. It is then easy to see that if  $r(\ell) \approx r'(\ell)$  the division  $(\ell, r(\ell))$  must be  $\varepsilon$ -envy-free.  $\square$

Our algorithm for four monotone agents from Section 5, together with the monotone-crossing problem of [13], yield the following result.

**LEMMA B.2.** *For four agents with monotone valuations, there exists a  $O(\log^2(1/\varepsilon))$  communication protocol that finds an  $\varepsilon$ -envy-free connected allocation.*

**PROOF SKETCH.** We show that the algorithm presented in Section 5 can be simulated by a communication protocol with total cost only  $O(\log^2(1/\varepsilon))$ . We think of Agent 1 as running the algorithm and communicating with the other agents whenever needed.

For the first step of the algorithm, it suffices for Agent 1 to find the equipartition of the cake into four equal parts according to its own valuation, and no communication is needed for that.

For the second step, Agent 1 performs  $O(\log(1/\varepsilon))$  steps of binary search. Every step of binary search consists in checking whether Condition A or B holds at some value  $\alpha$ . We will argue below that this only requires  $O(\log(1/\varepsilon))$  communication. Thus, the total cost of the second step is  $O(\log^2(1/\varepsilon))$ .

Finally, in the third step, Agent 1 just needs to output a division that satisfies Condition A or B at some given value  $\alpha$ . The arguments below will show that this only requires  $O(\log(1/\varepsilon))$  communication.

It remains to argue that given some value  $\alpha$ , we can, using only  $O(\log(1/\varepsilon))$  communication, check whether Condition A or B holds at value  $\alpha$ , and, if so, output a division that satisfies it. This can be shown by considering all the possible cases as in the proof of Lemma 5.3, and checking that each of them can be handled using only  $O(\log(1/\varepsilon))$  communication.

For Condition A, Agent 1 can determine the positions of all the cuts by itself, then use  $O(\log(1/\varepsilon))$  communication to share these positions with the other agents, who can then provide all the necessary information to decide whether the condition holds using  $O(1)$  communication.

For Condition B, the challenge is to determine the positions of the cuts using only  $O(\log(1/\varepsilon))$  communication. Once these positions are known,  $O(1)$  communication is again sufficient to check the condition. Below, we use the same notation as in the proof of Lemma 5.3 to argue that these cuts can be found using only  $O(\log(1/\varepsilon))$  communication.

If the pieces  $k$  and  $k'$  are adjacent, then Agent 1 can figure out the positions of two cuts, and inform Agent  $i$ . Agent  $i$  can then figure out the position of the last cut and inform everyone. This uses  $O(\log(1/\varepsilon))$  communication.

If there is one piece between pieces  $k$  and  $k'$ , then Agent 1 can figure out the position of one cut, and inform Agent  $i$ . The positions of the other two cuts can then be found using  $O(\log(1/\varepsilon))$  communication by reducing to the monotone-crossing problem [13]. We omit the details.

Finally, if  $k$  is the leftmost piece and  $k'$  is the rightmost piece, then Agent 1 and Agent  $i$  can find the positions of the leftmost and rightmost cut using  $O(\log(1/\varepsilon))$  communication again by reducing to the monotone-crossing problem [13]. We omit the details. The middle cut can then be found by Agent 1.  $\square$

Received 29 November 2023; revised 4 February 2025; accepted 27 July 2025

# The Kikuchi Hierarchy and Tensor PCA

ALEXANDER WEIN, University of California Davis, Davis, United States

AHMED EL ALAOUI, Cornell University, Ithaca, United States

CRISTOPHER MOORE, Santa Fe Institute, Santa Fe, United States

---

For the tensor principal component analysis (tensor PCA) problem, we propose a new hierarchy of increasingly powerful algorithms with increasing runtime. Our hierarchy is analogous to the sum-of-squares (SOS) hierarchy but is instead inspired by statistical physics and related algorithms such as belief propagation and AMP (approximate message passing). Our level- $\ell$  algorithm can be thought of as a linearized message-passing algorithm that keeps track of  $\ell$ -wise dependencies among the hidden variables. Specifically, our algorithms are spectral methods based on the *Kikuchi Hessian*, which generalizes the well-studied Bethe Hessian to the higher-order Kikuchi free energies.

It is known that AMP, the flagship algorithm of statistical physics, has substantially worse performance than SOS for tensor PCA. In this work, we ‘redeem’ the statistical physics approach by showing that our hierarchy gives a polynomial-time algorithm matching the performance of SOS. Our hierarchy also yields a continuum of subexponential-time algorithms, and we prove that these achieve the same (conjecturally optimal) tradeoff between runtime and statistical power as SOS. Our proofs are much simpler than prior work, and also apply to the related problem of refuting random  $k$ -XOR formulas. The results we present here apply to tensor PCA for tensors of all orders, and to  $k$ -XOR when  $k$  is even.

Our methods suggest a new avenue for systematically obtaining optimal algorithms for Bayesian inference problems, and our results constitute a step toward unifying the statistical physics and sum-of-squares approaches to algorithm design.

**CCS Concepts:** • Theory of computation → Design and analysis of algorithms; • Mathematics of computing → Probabilistic inference problems;

Additional Key Words and Phrases: Tensor PCA, CSP refutation, kikuchi matrix, message passing

**ACM Reference Format:**

Alexander Wein, Ahmed El Alaoui, and Cristopher Moore. 2025. The Kikuchi Hierarchy and Tensor PCA. *J. ACM* 72, 5, Article 35 (October 2025), 40 pages. <https://doi.org/10.1145/3762806>

---

ASW is partially supported by an Alfred P. Sloan Research Fellowship and NSF CAREER Award CCF-2338091. Most of this work was done while ASW was with the Courant Institute of Mathematical Sciences at New York University, partially supported by NSF grant DMS-1712730 and by the Simons Collaboration on Algorithms and Geometry. Most of this work was done while AEA was with Stanford University, partially supported by IIS-1741162, and ONR N00014-18-1-2729.

CM was partially supported by NSF grant BIGDATA-1838251.

Authors' Contact Information: Alexander Wein, University of California Davis, Davis, California, United States; e-mail: [aswein@ucdavis.edu](mailto:aswein@ucdavis.edu); Ahmed El Alaoui, Cornell University, Ithaca, New York, United States; e-mail: [elalaoui@cornell.edu](mailto:elalaoui@cornell.edu); Cristopher Moore, Santa Fe Institute, Santa Fe, New Mexico, United States; e-mail: [moore@santafe.edu](mailto:moore@santafe.edu).



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART35

<https://doi.org/10.1145/3762806>

## 1 Introduction

High-dimensional Bayesian inference problems are widely studied, including planted clique [3, 44], sparse PCA [45], and community detection [24, 25], just to name a few. For these types of problems, two general strategies, or *meta-algorithms*, have emerged. The first is rooted in statistical physics and includes the belief propagation (BP) algorithm [60, 75] along with variants such as **approximate message passing (AMP)** [29], and related spectral methods such as linearized BP [19, 50], and the Bethe Hessian [67]. The second meta-algorithm is the *sum-of-squares (SOS) hierarchy* [52, 59, 70], a hierarchy of increasingly powerful semidefinite programming relaxations to polynomial optimization problems, along with spectral methods inspired by it [40, 41]. Both of these meta-algorithms are known to achieve statistically-optimal performance for many problems. Furthermore, when they fail to perform a task, this is often seen as evidence that no polynomial-time algorithm can succeed. Such reasoning takes the form of free energy barriers in statistical physics [53, 54] or SOS lower bounds (e.g., Reference [14]). Thus, we generally expect both meta-algorithms to have optimal statistical performance among all *computationally efficient* algorithms.

A fundamental question is whether we can unify statistical physics and SOS, showing that the two approaches yield, or at least predict, the same performance on a large class of problems. However, one barrier to this comes from the *tensor principal component analysis* (tensor PCA) problem [65], on which the two meta-algorithms seem to have very different performance. For an integer  $p \geq 2$ , in the order- $p$  tensor PCA or *spiked tensor* problem we observe a  $p$ -fold  $n \times n \times \dots \times n$  tensor

$$Y = \lambda x_*^{\otimes p} + G$$

where the parameter  $\lambda \geq 0$  is a **signal-to-noise ratio (SNR)**,  $x_* \in \mathbb{R}^n$  is a planted signal vector with normalization  $\|x_*\| = \sqrt{n}$  drawn from a simple prior such as the uniform distribution on  $\{\pm 1\}^n$ , and  $G$  is a symmetric noise tensor with  $\mathcal{N}(0, 1)$  entries. Information-theoretically, it is possible to recover  $x_*$  given  $Y$  (in the limit  $n \rightarrow \infty$ , with  $p$  fixed) when  $\lambda \gg n^{(1-p)/2}$  [55, 65]. (Here we ignore log factors, so  $A \gg B$  can be understood to mean  $A \geq B \text{polylog}(n)$ .) However, this information-theoretic threshold corresponds to exhaustive search. We would also like to understand the *computational* threshold, i.e., for what values of  $\lambda$  there is an efficient algorithm.

The sum-of-squares hierarchy gives a polynomial-time algorithm to recover  $x_*$  when  $\lambda \gg n^{-p/4}$  [41], and SOS lower bounds suggest that no polynomial-time algorithm can do better [39, 41]. However, AMP, the flagship algorithm of statistical physics, is suboptimal for  $p \geq 3$  and fails unless  $\lambda \gg n^{-1/2}$  [65]. Various other “local” algorithms such as the tensor power method, Langevin dynamics, and gradient descent also fail below this “local” threshold  $\lambda \sim n^{-1/2}$  [8, 65]. This casts serious doubts on the optimality of the statistical physics approach.

In this article, we resolve this discrepancy and “redeem” the statistical physics approach. The *Bethe free energy* associated with AMP is merely the first level of a hierarchy of *Kikuchi free energies* [46, 47, 75]. From these Kikuchi free energies, we derive a hierarchy of increasingly powerful algorithms for tensor PCA, similar in spirit to generalized belief propagation [75]. Roughly speaking, our level- $\ell$  algorithm can be thought of as an iterative message-passing algorithm that reasons about  $\ell$ -wise dependencies among the hidden variables. As a result, it has time and space complexity  $n^{O(\ell)}$ . Specifically, the level- $\ell$  algorithm is a spectral method on a  $n^{O(\ell)} \times n^{O(\ell)}$  submatrix of (a first-order approximation of) the *Kikuchi Hessian*, i.e., the matrix of second derivatives of the Kikuchi free energy. This generalizes the *Bethe Hessian* spectral method, which has been successful in the setting of community detection [67]. We note that the Ph.D. dissertation of Saade [66] proposed the Kikuchi Hessian as a direction for future research.

For order- $p$  tensor PCA with  $p$  even, we show that level  $\ell = p/2$  of the Kikuchi hierarchy gives an algorithm that succeeds down to the SOS threshold  $\lambda \sim n^{-p/4}$ , closing the gap between SOS and

statistical physics. Furthermore, by taking  $\ell = n^\delta$  levels for various values of  $\delta \in (0, 1)$ , we obtain a continuum of subexponential-time algorithms that achieve a precise tradeoff between runtime and the signal-to-noise ratio—exactly the same tradeoff curve that SOS is known to achieve [16, 17].<sup>1</sup> We obtain similar results when  $p$  is odd, by combining a matrix related to the Kikuchi Hessian with a construction similar to [23]; see Section 6.2.

Our approach also applies to the problem of refuting random  $k$ -XOR formulas when  $k$  is even, showing that we can strongly refute random formulas with  $n$  variables and  $m \gg n^{k/2}$  clauses in polynomial time, and with a continuum of subexponential-time algorithms that succeed at lower densities. This gives a much simpler proof of the results of Reference [64], using only the matrix Chernoff bound instead of intensive moment calculations; see Section 6.1.

Our results redeem the statistical physics approach to algorithm design and give hope that the Kikuchi hierarchy provides a systematic way to derive optimal algorithms for a large class of Bayesian inference problems. We see this as a step toward unifying the statistical physics and SOS approaches. Indeed, we propose the following informal meta-conjecture: for high-dimensional inference problems with planted solutions (and related problems such as refuting random constraint satisfaction problems) the SOS hierarchy and the Kikuchi hierarchy both achieve the optimal tradeoff between runtime and statistical power.

*Concurrent and subsequent work.* After the initial appearance of this article, some related independent work has appeared. A hierarchy of algorithms similar to ours is proposed by Reference [38], but with a different motivation based on a system of quantum particles. For the problem of refuting random  $k$ -XOR formulas with  $k$  even, the work of Reference [1] gives, like us, a simplification of Reference [64], but using a different approach. Also, [18] gives an alternative “redemption” of local algorithms for tensor PCA, based on “averaged gradient descent.”

Furthermore, the conference version of the present article [74] led to a number of significant follow-up works. The “Kikuchi matrices” that we introduced have played a key role in state-of-the-art results on CSP refutation (in both random and smoothed settings) [37], locally decodable/correctable codes [4, 48, 49], as well as the resolution, up to a logarithmic factor, of Feige’s conjecture [31] on the hypergraph Moore bound in extremal combinatorics [37, 42]. Notably, for  $k$ -XOR refutation, the work of Reference [37] generalized our approach from even  $k$  to odd  $k$ , which required a more involved algorithm and analysis than we initially expected. Some of the above results use the Kikuchi matrices in conjunction with SOS, as these matrices constitute spectral SOS certificates for e.g., CSP refutation.

## 2 Preliminaries and Prior Work

### 2.1 Notation

Our asymptotic notation (e.g.,  $O(\cdot)$ ,  $o(\cdot)$ ,  $\Omega(\cdot)$ ,  $\omega(\cdot)$ ) pertains to the limit  $n \rightarrow \infty$  (large dimension) and may hide constants depending on  $p$  (tensor order), which we think of as fixed. We say an event occurs *with high probability* if it occurs with probability  $1 - o(1)$ .

A tensor  $T \in (\mathbb{R}^n)^{\otimes p}$  is an  $n \times n \times \cdots \times n$  ( $p$  times) multi-array with entries denoted by  $T_{i_1, \dots, i_p}$ , where  $i_k \in [n] := \{1, 2, \dots, n\}$ . We call  $p$  the *order* of  $T$  and  $n$  the *dimension* of  $T$ . For a vector  $u \in \mathbb{R}^n$ , the rank-1 tensor  $u^{\otimes p}$  is defined by  $(u^{\otimes p})_{i_1, \dots, i_p} = \prod_{k=1}^p u_{i_k}$ . A tensor  $T$  is *symmetric* if  $T_{i_1, \dots, i_p} = T_{i_{\pi(1)}, \dots, i_{\pi(p)}}$  for any permutation  $\pi \in S_p$ . For a symmetric tensor, if  $E = \{i_1, \dots, i_p\} \subseteq [n]$ , we will often write  $T_E := T_{i_1, \dots, i_p}$ .

---

<sup>1</sup>The strongest SOS results only apply to a variant of the spiked tensor model with Rademacher observations, but we do not expect this difference to be important; see Section 2.6.

## 2.2 The Spiked Tensor Model

A general formulation of the spiked tensor model is as follows. For an integer  $p \geq 2$ , let  $\tilde{G} \in (\mathbb{R}^n)^{\otimes p}$  be an order- $p$  tensor with entries sampled i.i.d. from  $\mathcal{N}(0, 1)$ . We then symmetrize this tensor and obtain a symmetric tensor  $G$ ,

$$G := \frac{1}{\sqrt{p!}} \sum_{\pi \in S_p} \tilde{G}^\pi,$$

where  $S_p$  is the symmetric group of permutations of  $[p]$ , and  $\tilde{G}_{i_1, \dots, i_p}^\pi := \tilde{G}_{i_{\pi(1)}, \dots, i_{\pi(p)}}$ . Note that if  $i_1, \dots, i_p$  are distinct then  $G_{i_1, \dots, i_p} \sim \mathcal{N}(0, 1)$ . We draw a ‘signal’ vector or ‘spike’  $x_* \in \mathbb{R}^n$  from a prior distribution  $P_x$  supported on the sphere  $\mathcal{S}^{n-1}(\sqrt{n}) = \{x \in \mathbb{R}^n : \|x\| = \sqrt{n}\}$ . Then, we let  $Y \in (\mathbb{R}^n)^{\otimes p}$  be the tensor

$$Y = \lambda x_*^{\otimes p} + G. \quad (1)$$

We will mostly focus on the *Rademacher-spiked model* where  $x_*$  is uniform in  $\{\pm 1\}^n$ , i.e.,  $P_x = 2^{-n} \prod_i (\delta_{-1}(x_i) + \delta_{+1}(x_i))$ . We will sometimes state results without specifying the prior  $P_x$ , in which case the result holds for *any* prior normalized so that  $\|x_*\| = \sqrt{n}$ . Let  $\mathbb{P}_\lambda$  denote the law of the tensor  $Y$ . The parameter  $\lambda = \lambda(n)$  may depend on  $n$ . We will consider the limit  $n \rightarrow \infty$  with  $p$  held fixed.

Our algorithms will depend only on the entries  $Y_{i_1, \dots, i_p}$  where the indices  $i_1, \dots, i_p$  are distinct: that is, on the collection

$$\{Y_E = \lambda x_*^E + G_E : E \subseteq [n], |E| = p\},$$

where  $G_E \sim \mathcal{N}(0, 1)$  and for a vector  $x \in \mathbb{R}^n$  we write  $x^E = \prod_{i \in E} x_i$ .

Perhaps one of the simplest statistical tasks is binary hypothesis testing. In our case this amounts to, given a tensor  $Y$  as input with the promise that it was sampled from  $\mathbb{P}_\lambda$  with  $\lambda \in \{0, \bar{\lambda}\}$ , determining whether  $\lambda = 0$  or  $\lambda = \bar{\lambda}$ . We refer to  $\mathbb{P}_\lambda$  for  $\lambda > 0$  as the *planted* distribution, and  $\mathbb{P}_0$  as the *null* distribution.

**Definition 1.** We say that an algorithm (or test)  $f : (\mathbb{R}^n)^{\otimes p} \rightarrow \{0, 1\}$  achieves *strong detection* between  $\mathbb{P}_0$  and  $\mathbb{P}_\lambda$  if

$$\lim_{n \rightarrow \infty} \mathbb{P}_\lambda(f(Y) = 1) = 1 \quad \text{and} \quad \lim_{n \rightarrow \infty} \mathbb{P}_0(f(Y) = 0) = 1.$$

Additionally, we say that  $f$  achieves *weak detection* between  $\mathbb{P}_0$  and  $\mathbb{P}_\lambda$  if the sum of Type-I and Type-II errors remains strictly below 1:

$$\limsup_{n \rightarrow \infty} \{\mathbb{P}_0(f(Y) = 1) + \mathbb{P}_\lambda(f(Y) = 0)\} < 1.$$

An additional goal is to recover the planted vector  $x_*$ . Note that when  $p$  is even,  $x_*$  and  $-x_*$  have the same posterior probability. Thus, our goal is to recover  $x_*$  up to a sign.

**Definition 2.** The *normalized correlation* between vectors  $\hat{x}, x \in \mathbb{R}^n$  is

$$\text{corr}(\hat{x}, x) = \frac{|\langle \hat{x}, x \rangle|}{\|\hat{x}\| \|x\|}.$$

**Definition 3.** An estimator  $\hat{x} = \hat{x}(Y)$  achieves *weak recovery* if  $\text{corr}(\hat{x}, x_*)$  is lower-bounded by a strictly positive constant—and we write  $\text{corr}(\hat{x}, x_*) = \Omega(1)$ —with high probability, and achieves *strong recovery* if  $\text{corr}(\hat{x}, x_*) = 1 - o(1)$  with high probability.

We expect that strong detection and weak recovery are generally equally difficult, although formal implications are not known in either direction. We will see in Section 2.5 that in some regimes, weak recovery and strong recovery are equivalent.

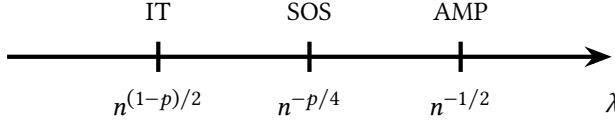


Fig. 1. Thresholds for tensor PCA: (IT) Information-theoretic threshold; (SOS) Threshold for the best known poly-time algorithms, including sum-of-squares and certain spectral methods; (AMP) Threshold for approximate message passing and other “local” algorithms, which are suboptimal.

*The matrix case.* When  $p = 2$ , the spiked tensor model reduces to the *spiked Wigner model*. We know from random matrix theory that when  $\lambda = \hat{\lambda}/\sqrt{n}$  with  $\hat{\lambda} > 1$ , strong detection is possible by thresholding the maximum eigenvalue of  $Y$ , and weak recovery is achieved by PCA, i.e., taking the leading eigenvector [15, 32]. For many spike priors including Rademacher, strong detection and weak recovery are statistically impossible when  $\hat{\lambda} < 1$  [26, 56, 62], so there is a sharp phase transition at  $\hat{\lambda} = 1$ . (Note that weak detection is still possible below  $\hat{\lambda} = 1$  [30].) A more sophisticated algorithm is AMP [26, 29, 34, 53], which can be thought of as a modification of the matrix power method which uses certain nonlinear transformations to exploit the structure of the spike prior. For many spike priors including Rademacher, AMP is known to achieve the information-theoretically optimal correlation with the true spike [26, 27]. However, like PCA, AMP achieves zero correlation asymptotically when  $\hat{\lambda} < 1$ . For certain spike priors (e.g., sparse priors), statistical-computational gaps can appear in which it is information-theoretically possible to succeed for some  $\hat{\lambda} < 1$  but we do not expect that polynomial-time algorithms can do so [13, 53, 54].

*The tensor case.* The tensor case  $p \geq 3$  was introduced by Reference [65], who also proposed various algorithms. Information-theoretically, there is a sharp phase transition similar to the matrix case: for many spike priors  $P_X$  including Rademacher, if  $\lambda = \hat{\lambda} n^{(1-p)/2}$  then weak recovery is possible when  $\hat{\lambda} > \lambda_c$  and impossible when  $\hat{\lambda} < \lambda_c$  for a particular constant  $\lambda_c = \lambda_c(p, P_X)$  depending on  $p$  and  $P_X$  [55]. Strong detection undergoes the same transition, being possible if  $\lambda > \lambda_c$  and impossible otherwise [21, 22, 43] (see also Reference [61]). In fact, it is shown in these works that even weak detection is impossible below  $\lambda_c$ , in sharp contrast with the matrix case. There are polynomial-time algorithms (e.g., SOS) that succeed at both strong detection and strong recovery when  $\lambda \gg n^{-p/4}$  for any spike prior [40, 41, 65], which is above the information-theoretic threshold by a factor that diverges with  $n$ . There are also SOS lower bounds suggesting that (for many priors) no polynomial-time algorithm can succeed at strong detection or weak recovery when  $\lambda \ll n^{-p/4}$  [39, 41]. Thus, unlike the matrix case, we expect a large statistical-computational gap, i.e., a “possible-but-hard” regime when  $n^{(1-p)/2} \ll \lambda \ll n^{-p/4}$ . Thresholds for tensor PCA are depicted in Figure 1 and discussed further in the following subsections.

### 2.3 The Tensor Power Method and Local Algorithms

Various algorithms have been proposed and analyzed for tensor PCA [5, 8, 40, 41, 65]. We will present two such algorithms that are simple, representative, and will be relevant to the discussion. The first is the *tensor power method* [6, 7, 65].

*Algorithm 4.* (Tensor Power Method) For a vector  $u \in \mathbb{R}^n$  and a tensor  $Y \in (\mathbb{R}^n)^{\otimes p}$ , let  $Y\{u\} \in \mathbb{R}^n$  denote the vector with entries

$$(Y\{u\})_i = \sum_{j_1, \dots, j_{p-1}} Y_{i, j_1, \dots, j_{p-1}} u_{j_1} \cdots u_{j_{p-1}}, \quad i \in [n].$$

The *tensor power method* begins with an initial guess  $u \in \mathbb{R}^n$  (e.g., chosen at random) and repeatedly iterates the update rule  $u \leftarrow Y\{u\}$  until  $u/\|u\|$  converges.

The tensor power method appears to only succeed when  $\lambda \gg n^{-1/2}$  [65], which is worse than the SOS threshold  $\lambda \sim n^{-p/4}$ . The AMP algorithm of Reference [65] is a more sophisticated variant of the tensor power method, but AMP also fails unless  $\lambda \gg n^{-1/2}$  [65]. Two other related algorithms, gradient descent and Langevin dynamics, also fail unless  $\lambda \gg n^{-1/2}$  [8]. Following [8], we refer to all of these algorithms (tensor power method, AMP, gradient descent, Langevin dynamics) as *local algorithms*, and we refer to the corresponding threshold  $\lambda \sim n^{-1/2}$  as the *local threshold*. Here “local” is not a precise notion, but roughly speaking, local algorithms keep track of a current guess for  $x_*$  and iteratively update it to a nearby vector that is more favorable in terms of e.g., the log-likelihood. This discrepancy between local algorithms and SOS is what motivated the current work.

## 2.4 The Tensor Unfolding Algorithm

We have seen that local algorithms do not seem able to reach the SOS threshold. Let us now describe one of the simplest algorithms that does reach this threshold: *tensor unfolding*. Tensor unfolding was first proposed by Reference [65], where it was shown to succeed when  $\lambda \gg n^{-\lfloor p/2 \rfloor / 2}$  and conjectured to succeed when  $\lambda \gg n^{-p/4}$  (the SOS threshold). For the case  $p = 3$ , the same algorithm was later reinterpreted as a spectral relaxation of SOS, and proven to succeed<sup>2</sup> when  $\lambda \gg n^{-3/4} = n^{-p/4}$  [41], confirming the conjecture of Reference [65]. We now present the tensor unfolding method, restricting to the case  $p = 3$  for simplicity. There is a natural extension to all  $p$  [65], and (a close variant of) this algorithm will in fact appear as level  $\ell = \lfloor p/2 \rfloor$  in our hierarchy of algorithms (see Section 3 and Appendix B).

*Algorithm 5.* (Tensor Unfolding) Given an order-3 tensor  $Y \in (\mathbb{R}^n)^{\otimes 3}$ , flatten it to an  $n \times n^2$  matrix  $M$ , i.e., let  $M_{i,jk} = Y_{ijk}$ . Compute the leading eigenvector of  $MM^\top$ .

If we use the matrix power method to compute the leading eigenvector, we can restate the tensor unfolding method as an iterative algorithm: keep track of state vectors  $u \in \mathbb{R}^n$  and  $v \in \mathbb{R}^{n^2}$ , initialize  $u$  randomly, and alternate between applying the update steps  $v \leftarrow M^\top u$  and  $u \leftarrow Mv$ . We will see later (see Section 4.1) that this can be interpreted as a message-passing algorithm between singleton indices, represented by  $u$ , and pairs of indices, represented by  $v$ . Thus, tensor unfolding is not “local” in the sense of Section 2.3 because it keeps a state of size  $O(n^2)$  (keeping track of pairwise information) instead of size  $O(n)$ . We can, however, think of it as local on a “lifted” space, and this allows it to surpass the local threshold.

Other methods have also been shown to achieve the SOS threshold  $\lambda \sim n^{-p/4}$ , including SOS itself and various spectral methods inspired by it References [40, 41].

## 2.5 Boosting and Linearization

One fundamental difference between the matrix case ( $p = 2$ ) and tensor case ( $p \geq 3$ ) arise from the following boosting property. The following result, implicit in Reference [65], shows that if  $\lambda$  is substantially above the information-theoretic threshold (i.e.,  $\lambda \gg n^{(1-p)/2}$ ) then weak recovery can be boosted to strong recovery via a single power iteration. We give a proof in Appendix C.

**PROPOSITION 6.** *Let  $Y \sim \mathbb{P}_\lambda$  with any spike prior  $P_x$  supported on  $\mathcal{S}^{n-1}(\sqrt{n})$ . Suppose we have an initial guess  $u \in \mathbb{R}^n$  satisfying  $\text{corr}(u, x_*) \geq \tau$ . Obtain  $\hat{x}$  from  $u$  via a single iteration of the tensor*

<sup>2</sup>The analysis of Reference [41] applies to a close variant of the spiked tensor model in which the noise tensor is asymmetric. We do not expect this difference to be important.

power method:  $\hat{x} = Y\{u\}$ . There exists a constant  $c = c(p) > 0$  such that with high probability,

$$\text{corr}(\hat{x}, x_*) \geq 1 - c\lambda^{-1}\tau^{1-p}n^{(1-p)/2}.$$

In particular, if  $\tau > 0$  is any constant and  $\lambda = \omega(n^{(1-p)/2})$  then  $\text{corr}(\hat{x}, x) = 1 - o(1)$ .

For  $p \geq 3$ , since we do not expect polynomial-time algorithms to succeed when  $\lambda = O(n^{(1-p)/2})$ , this implies an “all-or-nothing” phenomenon: for a given  $\lambda = \lambda(n)$ , the optimal polynomial-time algorithm will either achieve correlation that is asymptotically 0 or asymptotically 1. This is in stark contrast to the matrix case where, for  $\lambda = \hat{\lambda}/\sqrt{n}$ , the optimal correlation is a constant (in  $[0, 1]$ ) depending on both  $\hat{\lambda}$  and the spike prior  $P_x$ . This contrast is not due to Proposition 6 itself—which applies equally well to  $p = 2$ —but due to the large statistical-computational gap present in the tensor case, as this means the regime of interest lies well above the information-theoretic threshold.

This boosting result substantially simplifies things when  $p \geq 3$  because it implies that the only important question is identifying the threshold for weak recovery, instead of trying to achieve the optimal correlation. Heuristically, since we only want to attain the optimal threshold, statistical physics suggests that we can use a simple “linearized” spectral algorithm instead of a more sophisticated nonlinear algorithm. To illustrate this in the matrix case ( $p = 2$ ), one needs to use AMP in order to achieve optimal correlation, but one can achieve the optimal threshold using linearized AMP, which boils down to computing the top eigenvector. In the related setting of community detection in the stochastic block model, one needs to use belief propagation to achieve optimal correlation [24, 25, 57], but one can achieve the optimal threshold using a linearized version of belief propagation, which is a spectral method on the non-backtracking walk matrix [19, 50] or the related *Bethe Hessian* [67]. Our spectral methods for tensor PCA are based on the *Kikuchi Hessian*, which is a generalization of the Bethe Hessian.

## 2.6 Subexponential-Time Algorithms for Tensor PCA

The degree- $\ell$  sum-of-squares algorithm is a large semidefinite program that requires runtime  $n^{O(\ell)}$  to solve. Oftentimes the regime of interest is when  $\ell$  is constant, so that the algorithm runs in polynomial time. However, one can also explore the power of subexponential-time algorithms by letting  $\ell = n^\delta$  for  $\delta \in (0, 1)$ , which gives an algorithm with runtime roughly  $2^{n^\delta}$ . Results of this type are known for tensor PCA [16, 17, 64]. The strongest such results are for a different variant of tensor PCA, which we now define.

*Definition 7.* In the *order- $p$  discrete spiked tensor model* with spike prior  $P_x$  supported on  $\mathcal{S}^{n-1}(\sqrt{n})$ , and SNR parameter  $\lambda \geq 0$ , we draw a spike  $x_* \sim P_x$  and then for each  $1 \leq i_1 \leq \dots \leq i_p \leq n$ , we independently observe a  $\{\pm 1\}$ -valued random variable  $Y_{i_1, \dots, i_p}$  with  $\mathbb{E}[Y_{i_1, \dots, i_p}] = \lambda(x_*^{\otimes p})_{i_1, \dots, i_p}$ .

This model differs from our usual one in that the observations are conditionally Rademacher instead of Gaussian, but we do not believe this makes an important difference. However, for technical reasons, the known SOS results are strongest in this discrete setting.

**THEOREM 8 ([16, 41]).** *For any  $1 \leq \ell \leq n$ , there is an algorithm with runtime  $n^{O(\ell)}$  that achieves strong detection and strong recovery in the order- $p$  discrete spiked tensor model (with any spike prior) whenever*

$$\lambda \geq \ell^{1/2-p/4}n^{-p/4}\text{polylog}(n).$$

The work of Reference [16] shows how to certify an upper bound on the injective norm of a random  $\{\pm 1\}$ -valued tensor, which immediately implies the algorithm for strong detection. When

combined with [41], this can also be made into an algorithm for strong recovery (see Lemma 4.4 of Reference [41]). Similar (but weaker) SOS results are also known for the standard spiked tensor model (see [64] and *arXiv* version 1 of Reference [16]), and we expect that Theorem 8 also holds for this case. (Curiously, the certification-to-recovery reduction within SOS is not known in the related setting of CSPs.)

When  $\ell = n^\delta$  for  $\delta \in (0, 1)$ , Theorem 8 implies that we have an algorithm with runtime  $n^{O(n^\delta)} = 2^{n^{\delta+o(1)}}$  that succeeds when  $\lambda \gg n^{\delta/2-p\delta/4-p/4}$ . Note that this interpolates smoothly between the polynomial-time threshold ( $\lambda \sim n^{-p/4}$ ) when  $\delta = 0$ , and the information-theoretic threshold ( $\lambda \sim n^{(1-p)/2}$ ) when  $\delta = 1$ . We will prove (for  $p$  even) that our algorithms achieve this same tradeoff, and we expect this tradeoff to be optimal.<sup>3</sup>

### 3 Main Results

In this section, we present our main results about detection and recovery in the spiked tensor model. We propose a hierarchy of spectral methods, which are directly derived from the hierarchy of *Kikuchi free energies*. Specifically, the symmetric difference matrix defined below appears (approximately) as a submatrix of the Hessian of the Kikuchi free energy. The full details of this derivation are given in Section 4 and Appendix D. For now we simply state the algorithms and results.

We will restrict our attention to the Rademacher-spiked tensor model, which is the setting in which we derived our algorithms. However, we show in Appendix A that the same algorithm works for a large class of priors (at least for strong detection). Furthermore, we show in Section 6.1 that the same algorithm can also be used for refuting random  $k$ -XOR formulas (when  $k$  is even).

We will also restrict to the case where the tensor order  $p$  is even. The case of odd  $p$  is discussed in Appendix B, where we give an algorithm derived from the Kikuchi Hessian and conjecture that it achieves optimal performance. We are unable to prove this, but we are able to prove that optimal results are attained by a related algorithm (see Section 6.2).

Our approach requires the introduction of two matrices:

*The symmetric difference matrix of order  $\ell$ .* Let  $p$  be even and let  $Y \in (\mathbb{R}^n)^{\otimes p}$  be the observed order- $p$  symmetric tensor. We will only use the entries  $Y_{i_1, \dots, i_p}$  for which the indices  $i_1, \dots, i_p$  are distinct; we denote such entries by  $Y_E$  where  $E \subseteq [n]$  with  $|E| = p$ . Fix an integer  $\ell \in [p/2, n - p/2]$ , and consider the symmetric  $\binom{n}{\ell} \times \binom{n}{\ell}$  matrix  $M$  indexed by sets  $S \subseteq [n]$  of size  $\ell$ , having entries

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Here  $\triangle$  denotes the symmetric difference between sets. The leading eigenvector of  $M$  is intended to be an estimate of  $(x_*^S)_{|S|=\ell}$  where  $x^S := \prod_{i \in S} x_i$ . The following ‘voting’ matrix is a natural rounding scheme to extract an estimate of  $x_*$  from such a vector.

*The voting matrix.* To a vector  $v \in \mathbb{R}^{\binom{n}{\ell}}$  we associate the following symmetric  $n \times n$  voting matrix  $V(v)$  having entries

$$V_{ii}(v) = 0 \quad \forall i \in [n], \quad \text{and} \quad V_{ij}(v) = \frac{1}{2} \sum_{S,T \in \binom{[n]}{\ell}} v_S v_T \mathbf{1}_{S \triangle T = \{i,j\}} \quad \forall i \neq j. \quad (3)$$

---

<sup>3</sup>One form of evidence suggesting that this tradeoff is optimal is based on the *low-degree likelihood ratio*; see Reference [51]. There is also an SOS lower bound [63].

Let us define the important quantity

$$d_\ell := \binom{n-\ell}{p/2} \binom{\ell}{p/2}. \quad (4)$$

This is the number of sets  $T$  of size  $\ell$  such that  $|S \Delta T| = p$  for a given set  $S$  of size  $\ell$ . (Note it is important for  $p$  to be even here.) Now we are in position to formulate our algorithms for detection and recovery.

*Algorithm 9 (Detection for even  $p$ ).*

- (1) Compute the top eigenvalue  $\lambda_{\max}(M)$  of the symmetric difference matrix  $M$ .
- (2) Reject the null hypothesis  $\mathbb{P}_0$  (i.e., return ‘1’) if  $\lambda_{\max}(M) \geq \lambda d_\ell / 2$ .

We will see in Section 5 that the quantity  $\lambda d_\ell / 2$  is half the largest eigenvalue of ‘the signal part’ of the matrix  $M$ . This intuitively justifies its appearance in the above algorithm.

*Algorithm 10 (Recovery for even  $p$ ).*

- (1) Compute a (unit-norm) leading eigenvector<sup>4</sup>  $v_{\text{top}} \in \mathbb{R}^{\binom{n}{\ell}}$  of  $M$ .
- (2) Form the associated voting matrix  $V(v_{\text{top}})$ .
- (3) Compute a leading eigenvector  $\hat{x}$  of  $V(v_{\text{top}})$ , and output  $\hat{x}$ .

The next two theorems characterize the performance of Algorithms 9 and 10 for the strong detection and recovery tasks, respectively. The proofs can be found in Section 5.

**THEOREM 11.** *Consider the Rademacher-spiked tensor model with  $p$  even. For all  $\lambda \geq 0$  and  $\ell \in [p/2, n - p/2]$ , we have*

$$\mathbb{P}_0 \left( \lambda_{\max}(M) \geq \frac{\lambda d_\ell}{2} \right) \vee \mathbb{P}_\lambda \left( \lambda_{\max}(M) \leq \frac{\lambda d_\ell}{2} \right) \leq 2n^\ell e^{-\lambda^2 d_\ell / 8}.$$

(Here,  $a \vee b = \max\{a, b\}$ .) Therefore, Algorithm 9 achieves strong detection between  $\mathbb{P}_0$  and  $\mathbb{P}_\lambda$  if  $\frac{1}{8}\lambda^2 d_\ell - \ell \log n \rightarrow +\infty$  as  $n \rightarrow +\infty$ .

**THEOREM 12.** *Consider the Rademacher-spiked tensor model with  $p$  even. Let  $\hat{x} \in \mathbb{R}^n$  be the output of Algorithm 10. There exists an absolute constant  $c_0 > 0$  such that for all  $\epsilon > 0$  and  $\delta \in (0, 1)$ , if  $\ell \leq n\epsilon^2$  and  $\lambda \geq c_0\epsilon^{-4}\sqrt{\log(n^\ell/\delta)/d_\ell}$ , then  $\text{corr}(\hat{x}, x_*) \geq 1 - c_0\epsilon$  with probability at least  $1 - \delta$ .*

**Remark 13.** If  $\ell = o(n)$ , we have  $d_\ell = \Theta(n^{p/2} \ell^{p/2})$ , and so the above theorems imply that strong detection and strong recovery are possible as soon as  $\lambda \gg \ell^{-(p-2)/4} n^{-p/4} \sqrt{\log n}$ . Comparing with Theorem 8, this scaling coincides with the guarantees achieved by the level- $\ell$  SOS algorithm of Reference [16], up to a possible discrepancy in logarithmic factors.

Due to the particularly simple structure of the symmetric difference matrix  $M$  (in particular, the fact that its entries are simply entries of  $Y$ ), the proof of detection (Theorem 11) follows from a straightforward application of the matrix Chernoff bound. In contrast, the corresponding SOS results [16, 17, 64] work with more complicated matrices involving high powers of the entries of  $Y$ , and the analysis is much more involved.

Our proof of recovery is unusual in that the signal component of  $M$ , call it  $X$ , is not rank-one (see Equation (14)); it even has a vanishing spectral gap when  $\ell \gg 1$ . Thus, the leading eigenvector of  $M$  does not correlate well with the leading eigenvector of  $X$ . While this may seem to render recovery hopeless at first glance, this is not the case, due to the fact that *many* eigenvectors (actually,

<sup>4</sup>We define a leading eigenvector to be an eigenvector whose eigenvalue is maximal (although our results still hold for an eigenvector whose eigenvalue is maximal in absolute value).

eigenspaces) of  $X$  contain non-trivial information about the spike  $x_*$ , as opposed to only the top one. We prove this by exploiting the special structure of  $X$  through the *Johnson scheme*, and using tools from Fourier analysis on a slice of the hypercube, in particular a Poincaré-type inequality by Reference [33].

*Removing the logarithmic factor.* Both Theorems 11 and 12 involve a logarithmic factor in  $n$  in the lower bound on SNR  $\lambda$ . These logarithmic factors are an artifact of the matrix Chernoff bound, and we believe they can be removed. The analysis of Reference [41] removes the logarithmic factors for the tensor unfolding algorithm, which is essentially the case  $p = 3$  and  $\ell = 1$  of our algorithm. This suggests the following precise conjecture on the power of polynomial-time algorithms.

**CONJECTURE 14.** *Fix  $p$  and let  $\ell$  be constant (not depending on  $n$ ). There exists a constant  $c_p(\ell) > 0$  with  $c_p(\ell) \rightarrow 0$  as  $\ell \rightarrow \infty$  (with  $p$  fixed) such that if  $\lambda \geq c_p(\ell)n^{-p/4}$  then Algorithm 9 and Algorithm 10 (which run in time  $n^{O(\ell)}$ ) achieve strong detection and strong recovery, respectively. Specifically, we expect  $c_p(\ell) \sim \ell^{-(p-2)/4}$  for large  $\ell$ .*

*Followup work:* The above conjecture was partially settled in the recent work [12] for  $\ell \in [p/2, 3p/4]$ , where the authors show a sharp estimate for the first eigenvalue of the matrix  $M/\sqrt{d_\ell}$  by relating it to a corresponding free probability model which is amenable to sharp analysis; a line of inquiry yielding remarkable results initiated in Reference [11]. In particular, the authors show that the test described in Algorithm 9 achieves strong detection for  $\lambda \geq (1 + \epsilon)d_\ell^{-1/2}$  when  $p/2 \leq \ell < 3p/4$  for any  $\epsilon > 0$ , see [12, Section 3.3].

## 4 Motivating the Symmetric Difference Matrices

In this section, we motivate the symmetric difference matrices used in our algorithms. In Section 4.1, we give some high-level intuition. In Section 4.2, we give a more principled derivation based on the Kikuchi Hessian, with many of the calculations deferred to Appendix D.

### 4.1 Intuition: Higher-Order Message-Passing and Maximum Likelihood

We will give two related justifications for the symmetric difference matrices. First, we will see how our algorithms can be thought of as iterative message-passing procedures among subsets of size  $\ell$ . Second, we will see that the symmetric difference matrix is a submatrix of some exponential-size matrix whose leading eigenvector can exactly compute the maximum likelihood estimator.

As stated previously, our algorithms will choose to ignore the entries  $Y_{i_1, \dots, i_p}$  for which  $i_1, \dots, i_p$  are not distinct; these entries turn out to be unimportant asymptotically. We restrict to the Rademacher-spiked tensor model, as this yields a clean and simple derivation. The posterior distribution for the spike  $x_*$  given the observed tensor  $Y$  is

$$\begin{aligned} \mathbb{P}(x | Y) &\propto \exp \left\{ -\frac{1}{2} \sum_{i_1 < \dots < i_p} \left( Y_{i_1, \dots, i_p} - \lambda x_{i_1} \cdots x_{i_p} \right)^2 \right\} \\ &\propto \exp \left\{ \lambda \sum_{i_1 < \dots < i_p} Y_{i_1, \dots, i_p} x_{i_1} \cdots x_{i_p} \right\} = \exp \left\{ \lambda \sum_{|E|=p} Y_E x^E \right\}, \end{aligned} \quad (5)$$

over the domain  $x \in \{\pm 1\}^n$ . We take  $p$  to be even; a similar derivation works for odd  $p$ . Now fix  $\ell \in [p/2, n - p/2]$ . We can write the above as

$$\mathbb{P}(x | Y) \propto \exp \left\{ \frac{\lambda}{N} \sum_{|S \triangle T|=p} Y_{S \triangle T} x^S x^T \right\}, \quad (6)$$

where the sum is over ordered pairs  $(S, T)$  of sets  $S, T \subseteq [n]$  with  $|S| = |T| = \ell$  and  $|S \Delta T| = p$ , and where  $N = d_\ell(\binom{n}{\ell})/\binom{p}{\ell}$  is the number of terms  $(S, T)$  with a given symmetric difference  $E$ .

A natural algorithm to maximize the log-likelihood is the following. For each  $S \subseteq [n]$  of size  $|S| = \ell$ , keep track of a variable  $u_S \in \mathbb{R}$ , which is intended to be an estimate of  $x_*^S := \prod_{i \in S} (x_*)_i$ . Note that there are consistency constraints that  $(x_*^S)_{|S|=\ell}$  must obey, such as  $x_*^S x_*^T x_*^V = 1$  when  $S \Delta T \Delta V = \emptyset$ ; we will relax the problem and will not require our vector  $u = (u_S)_{|S|=\ell}$  to obey such constraints. Instead, we simply attempt to maximize

$$\frac{1}{\|u\|^2} \sum_{|S \Delta T|=p} Y_{S \Delta T} u_S u_T, \quad (7)$$

over all  $u \in \mathbb{R}^{\binom{n}{\ell}}$ . To do this, we iterate the update equations

$$u_S \leftarrow \sum_{T : |S \Delta T|=p} Y_{S \Delta T} u_T. \quad (8)$$

We call  $S$  and  $T$  *neighbors* if  $|S \Delta T| = p$ . Intuitively, each neighbor  $T$  of  $S$  sends a message  $m_{T \rightarrow S} := Y_{S \Delta T} u_T$  to  $S$ , indicating  $T$ 's opinion about  $u_S$ . We update  $u_S$  to be the sum of all incoming messages.

Now note that the sum in (7) is simply  $\|u\|^{-2} u^\top M u$  where  $M$  is the symmetric difference matrix, and (8) can be written as

$$u \leftarrow Mu.$$

Thus, this natural message-passing scheme is precisely power iteration against  $M$ , and so we should take the leading eigenvector  $v_{\text{top}}$  of  $M$  as our estimate of  $(x_*^S)_{|S|=\ell}$  (up to a scaling factor). Finally, defining our voting matrix  $V(v_{\text{top}})$  and taking its leading eigenvector is a natural method for rounding  $v_{\text{top}}$  to a vector of the form  $u^x$  where  $u_S^x = x^S$ , thus restoring the consistency constraints we ignored before.

For insight on why this ought to work, it is instructive to note that if we carry out this procedure on *all* subsets  $S \subseteq [n]$  then this works as intended, and no rounding is necessary: consider the  $2^n \times 2^n$  matrix  $\bar{M}_{S,T} = Y_{S \Delta T} 1_{|S \Delta T|=p}$ . It is easy to verify that the eigenvectors of  $\bar{M}$  are precisely the Fourier basis vectors on the hypercube, namely vectors of the form  $u^x$  where  $u_S^x = x^S$  and  $x \in \{\pm 1\}^n$ . Moreover, the eigenvalue associated to  $u^x$  is

$$\frac{1}{2^n} (u^x)^\top \bar{M} u^x = \frac{1}{2^n} \sum_{S,T \subseteq [n] : |S \Delta T|=p} Y_{S \Delta T} x^S x^T = \sum_{|E|=p} Y_E x^E.$$

This is the expression in the log-likelihood in (5). Thus, the leading eigenvector of  $\bar{M}$  is exactly  $u^x$  where  $x$  is the maximum likelihood estimate of  $x_*$ .

This procedure succeeds all the way down to the information-theoretic threshold  $\lambda \sim n^{(1-p)/2}$ , but takes exponential time. Our contribution can be viewed as showing that even when we restrict to the submatrix  $M$  of  $\bar{M}$  supported on subsets of size  $\ell$ , the leading eigenvector still allows us to recover  $x_*$  whenever the SNR is sufficiently large. Proving this requires us to perform Fourier analysis over a slice of the hypercube rather than the simpler setting of the entire hypercube, which we do by appealing to Johnson schemes and some results of Reference [33].

## 4.2 Variational Inference and Kikuchi Free Energy

We now introduce the *Kikuchi approximations to the free energy* (or simply the *Kikuchi free energies*) of the above posterior (5) [46, 47], the principle from which our algorithms are derived. For concreteness, we restrict to the case of the Rademacher-spiked tensor model, but the Kikuchi free energies can be defined for general graphical models [75].

The posterior distribution in (5) is a Gibbs distribution  $\mathbb{P}(x \mid Y) \propto e^{-\beta H(x)}$  with random Hamiltonian  $H(x) := -\lambda \sum_{i_1 < \dots < i_p} Y_{i_1 \dots i_p} x_{i_1} \dots x_{i_p}$ , and inverse temperature  $\beta = 1$ . We let  $Z_n(\beta; Y) := \sum_{x \in \{\pm 1\}^n} e^{-\beta H(x)}$  be the partition function of the model, and denote by  $F_n(\beta; Y) := -\frac{1}{\beta} \log Z_n(\beta; Y)$  its *free energy*. It is a classical fact that the Gibbs distribution has the following variational characterization. Fix a finite domain  $\Omega$  (e.g.,  $\{\pm 1\}^n$ ),  $\beta > 0$  and  $H : \Omega \rightarrow \mathbb{R}$ . Consider the optimization problem

$$\inf_{\mu} F(\mu), \quad (9)$$

where the infimum is over probability distributions  $\mu$  supported on  $\Omega$ , and define the *free energy functional*  $F$  of  $\mu$  by

$$F(\mu) := \mathbb{E}_{x \sim \mu} [H(x)] - \frac{1}{\beta} \bar{\mathcal{S}}(\mu), \quad (10)$$

where  $\bar{\mathcal{S}}(\mu)$  is the Shannon entropy of  $\mu$ , i.e.,  $\bar{\mathcal{S}}(\mu) = -\sum_{x \in \Omega} \mu(x) \log \mu(x)$ . Then, the unique optimizer of (9) is the Gibbs distribution  $\mu^*(x) \propto \exp(-\beta H(x))$ . If we specialize this statement to our setting,  $\mu^* = \mathbb{P}(\cdot \mid Y)$  and  $F_n(1; Y) = F(\mu^*)$ . We refer to Reference [72] for more background.

In light of the above variational characterization, a natural algorithmic strategy to learn the posterior distribution is to minimize the free energy functional  $F(\mu)$  over distributions  $\mu$ . However, this is *a priori* intractable because (for a high-dimensional domain such as  $\Omega = \{\pm 1\}^n$ ) an exponential number of parameters are required to represent  $\mu$ . The idea underlying the *belief propagation* algorithm [60, 75] is to work only with locally-consistent marginals, or *beliefs*, instead of a complete distribution  $\mu$ . Standard belief propagation works with beliefs on singleton variables and on pairs of variables. The *Bethe free energy* is a proxy for the free energy that only depends on these beliefs, and belief propagation is a certain procedure that iteratively updates the beliefs in order to locally minimize the Bethe free energy. The level- $r$  *Kikuchi free energy* is a generalization of the *Bethe free energy* that depends on  $r$ -wise beliefs and gives (in principle) increasingly better approximations of  $F(\mu^*)$  as  $r$  increases. Our algorithms are based on the principle of locally minimizing Kikuchi free energy, which we define next.

We now define the level- $r$  Kikuchi approximation to the free energy. We require  $r \geq p$ , i.e., the Kikuchi level needs to be at least as large as the interactions present in the data (although the  $r < p$  case could be handled by defining a modified graphical model with auxiliary variables). The Bethe free energy is the case  $r = 2$ .

For  $S \subseteq [n]$  with  $0 < |S| \leq r$ , let  $b_S : \{\pm 1\}^S \rightarrow \mathbb{R}$  denote the *belief* on  $S$ , which is a probability mass function over  $\{\pm 1\}^{|S|}$  representing our belief about the joint distribution of  $x_S := \{x_i\}_{i \in S}$ . Let  $b = \{b_S : S \in \binom{[n]}{\leq r}\}$  denote the set of beliefs on  $s$ -wise interactions for all  $s \leq r$ . Following [75], the Kikuchi free energy is a real-valued functional  $\mathcal{K}$  of  $b$  having the form  $\mathcal{E} - \frac{1}{\beta} \mathcal{S}$  (in our case,  $\beta = 1$ ). Here, the first term is the ‘energy’ term

$$\mathcal{E}(b) = -\lambda \sum_{|S|=p} Y_S \sum_{x_S \in \{\pm 1\}^S} b_S(x_S) x^S.$$

where, recall,  $x^S := \prod_{i \in S} x_i$ . (This is a proxy for the term  $\mathbb{E}_{x \sim \mu} [H(x)]$  in (10).) The second term in  $\mathcal{K}$  is the ‘entropy’ term

$$\mathcal{S}(b) = \sum_{0 < |S| \leq r} c_S \mathcal{S}_S(b), \quad \mathcal{S}_S(b) = - \sum_{x_S \in \{\pm 1\}^S} b_S(x_S) \log b_S(x_S), \quad (11)$$

where the *overcounting numbers* are  $c_S := \sum_{T \supseteq S, |T| \leq r} (-1)^{|T \setminus S|}$ . These are defined so that for any  $S \subseteq [n]$  with  $0 < |S| \leq r$ ,

$$\sum_{T \supseteq S, |T| \leq r} c_T = 1, \quad (12)$$

which corrects for overcounting the contribution of every variable  $x_i$  to the entropy. The simplest case is when only beliefs on single marginals are considered:  $r = 1$ . In this case the entropy term becomes  $\mathcal{S}(b) = \sum_{i=1}^n -b(x_i) \log b(x_i)$ , and the resulting approximation to the optimization problem (9) is known as the *naive mean-field* approximation, which effectively optimizes over product measures  $\mu(x) = \prod_{i=1}^n \mu_i(x_i)$ . The case  $r = 2$  amounts to considering pair interactions, it considers distributions  $\mu$  which can be approximated by an *acyclic* Markov random field ‘at the level of the entropy’: a Markov random field on a tree  $T = (V = [n], E)$  (a graph without cycles) can be written in terms of its vertex and edge marginals as  $\mu(x) = \prod_{i \in V} \mu(x_i) \prod_{(i,j) \in E} \mu(x_i, x_j) / [\mu(x_i) \mu(x_j)]$ . The Shannon entropy of  $\mu$  can be readily computed and is given by the expression  $\tilde{\mathcal{S}}(\mu) = \sum_{(i,j) \in E} \mathcal{S}_{\{i,j\}}(\mu) - \sum_{i \in V} (d_i - 1) \mathcal{S}_{\{i\}}(\mu) = \mathcal{S}(\mu)$  where  $d_i$  is the degree of vertex  $i$  in  $T$ , and in writing  $\mathcal{S}(\mu)$ ,  $\mathcal{S}_{\{i\}}(\mu)$  and  $\mathcal{S}_{\{i,j\}}(\mu)$ , we identify the probability distribution  $\mu$  with its collection of marginals  $\mu_S = \mu((x_i)_{i \in S})$ ,  $S \subseteq [n]$ . The Bethe approximation consists in ignoring the cycles in the graph—in our case the complete graph—and using an entropy of the previous form. The origin of the overcounting terms  $c_S$  in this simple case is as follows: heuristically, in the pairwise terms  $\sum_{(i,j) \in E} \mathcal{S}_{\{i,j\}}$  defining  $\mathcal{S}(b)$  in Equation (11), each vertex  $i$  ‘contributes’  $d_i$  times to the entropy, so its marginal entropy must be subtracted  $d_i - 1$  times so that every vertex ‘contributes’ once. A generalization of this approach to larger  $r$  can be done via the language of hypergraphs and using hypertrees as a basis for the approximation. We do not attempt further elaboration here and refer to the excellent book [72, Section 4.2] for a clear and full exposition of this formalism.

Notice that  $\mathcal{E}$  and  $\mathcal{S}$  each take the form of an “expectation” with respect to the beliefs  $b_S$ ; these would be actual expectations were the beliefs the marginals of an actual probability distribution. This situation is to be compared with the notion of a *pseudo-expectation*, which plays a central role in the theory underlying the sum-of-squares algorithm.

Our algorithms are based on the *Kikuchi Hessian*, a generalization of the Bethe Hessian matrix that was introduced in the setting of community detection [67]. The Bethe Hessian is the Hessian of the Bethe free energy with respect to the moments of the beliefs, evaluated at belief propagation’s so-called “uninformative fixed point.” The bottom eigenvector of the Bethe Hessian is a natural estimator for the planted signal because it represents the best direction for local improvement of the Bethe free energy, starting from belief propagation’s uninformative starting point. We generalize this method and compute the analogous Kikuchi Hessian matrix. The full derivation is given in Appendix D. The order- $\ell$  symmetric difference matrix (2) (approximately) appears as a submatrix of the level- $r$  Kikuchi Hessian whenever  $r \geq \ell + p/2$ .

## 5 Analysis of Symmetric Difference and Voting Matrices

We adopt the notation  $x^S := \prod_{i \in S} x_i$  for  $x \in \{\pm 1\}^n$  and  $S \subseteq [n]$ . Recall the matrix  $M$  indexed by sets  $S \subseteq [n]$  of size  $\ell$ , having entries

$$M_{S,T} = Y_{S \triangle T} 1_{|S \triangle T|=p} \quad \text{where} \quad Y_{S \triangle T} = \lambda x_*^{S \triangle T} + G_{S \triangle T}. \quad (13)$$

First, observe that we can restrict our attention to the case where the spike is the all-ones vector  $x_* = \mathbb{1}$  without loss of generality. To see this, conjugate  $M$  by a diagonal matrix  $D$  with diagonal entries  $D_{S,S} = x_*^S$  and obtain  $(M')_{S,T} = (D^{-1}MD)_{S,T} = Y'_{S \triangle T} 1_{|S \triangle T|=p}$  where  $Y'_{S \triangle T} = x_*^S x_*^{T \triangle S} Y_{S \triangle T} = x_*^{S \triangle T} Y_{S \triangle T} = \lambda + G'_{S \triangle T}$  where  $G'_{S \triangle T} = x_*^S x_*^{T \triangle S} G_{S \triangle T}$ . By symmetry of the Gaussian distribution,  $(G'_E)_{|E|=p}$  are i.i.d.  $\mathcal{N}(0, 1)$  random variables. Therefore, the two matrices have the same spectrum and the eigenvectors of  $M$  can be obtained from those of  $M'$  by pre-multiplying by  $D$ . Similarly, the case

$x_* = \mathbb{1}$  is sufficient for the analysis of the voting matrices. So from now on we write

$$M = \lambda X + Z, \quad (14)$$

where  $X_{S,T} = 1_{|S \Delta T|=p}$  and  $Z_{S,T} = G_{S \Delta T} 1_{|S \cap T|=p}$ , where  $(G_E)_{|E|=p}$  is a collection of i.i.d.  $\mathcal{N}(0, 1)$  r.v.'s.

### 5.1 Structure of $X$

The matrix  $X$  is the adjacency matrix of a regular graph  $J_{n,\ell,p}$  on  $\binom{n}{\ell}$  vertices, where vertices are represented by sets, and two sets  $S$  and  $T$  are connected by an edge if  $|S \Delta T| = p$ , or equivalently  $|S \cap T| = \ell - p/2$ . This matrix belongs to the Bose-Mesner algebra of the  $(n, \ell)$ -Johnson association scheme (see for instance [35, 69]). This is the algebra of  $\binom{n}{\ell} \times \binom{n}{\ell}$  real- or complex-valued symmetric matrices where the entry  $X_{S,T}$  depends only on the size of the intersection  $|S \cap T|$ . In addition to this set of matrices being an algebra, it is a *commutative* algebra, which means that all such matrices are simultaneously diagonalizable and share the same eigenvectors.

Filmus [33] provides a common eigenbasis for this algebra: for  $0 \leq m \leq \ell$ , let  $\varphi = (a_1, b_1, \dots, a_m, b_m)$  be a sequence of  $2m$  distinct elements of  $[n]$ . Let  $|\varphi| = 2m$  denote its total length. Now define a vector  $u^\varphi \in \mathbb{R}^{\binom{n}{\ell}}$  having coordinates

$$u_S^\varphi = \prod_{i=1}^m (1_{a_i \in S} - 1_{b_i \in S}), \quad |S| = \ell.$$

In the case  $m = 0$ ,  $\varphi$  is the empty sequence  $\emptyset$  and we have  $u^\emptyset = \mathbb{1}$  (the all-ones vector).

**PROPOSITION 15.** *Each  $u^\varphi$  is an eigenvector of  $X$ . Furthermore, the linear space  $\mathcal{Y}_m := \text{span}\{u^\varphi : |\varphi| = 2m\}$  for  $0 \leq m \leq \ell$  is an eigenspace of  $X$  (i.e., all vectors  $u^\varphi$  with sequences  $\varphi$  of length of  $2m$  have the same eigenvalue  $\mu_m$ ). Lastly  $\mathbb{R}^{\binom{n}{\ell}} = \bigoplus_{m=0}^{\ell} \mathcal{Y}_m$ , and  $\dim \mathcal{Y}_m = \binom{n}{m} - \binom{n}{m-1}$ . (By convention,  $\binom{n}{-1} = 0$ .)*

**PROOF.** The first two statements are the content of Lemma 4.3 in Reference [33]. The dimension of  $\mathcal{Y}_m$  is given in Lemma 2.1 in Reference [33].  $\square$

We note that  $(u^\varphi)_{|\varphi|=2m}$  are not linearly independent; an orthogonal basis, called the Young basis, consisting of linear combinations of the  $u^\varphi$ 's is given explicitly in Reference [33].

We see from the above proposition that  $X$  has  $\ell + 1$  distinct eigenvalues  $\mu_0 \geq \mu_1 \geq \dots \geq \mu_\ell$ , each one corresponding to the eigenspace  $\mathcal{Y}_m$ . The first eigenvalue is the degree of the graph  $J_{n,\ell,p}$ :

$$\mu_0 = d_\ell = \binom{\ell}{p/2} \binom{n-\ell}{p/2}. \quad (15)$$

We provide an explicit formula for all the remaining eigenvalues:

**LEMMA 16.** *The eigenvalues of  $X$  are as follows:*

$$\mu_m = \sum_{s=0}^{\min(m, p/2)} (-1)^s \binom{m}{s} \binom{\ell-m}{p/2-s} \binom{n-\ell-m}{p/2-s}, \quad 0 \leq m \leq \ell. \quad (16)$$

**PROOF.** These are the so-called Eberlein polynomials, which are polynomials in  $m$  of degree  $p$  (see, e.g., Reference [69]). We refer to Reference [20] for formulae in more general contexts, but we give a proof here for completeness. Let  $A = \{a_1, \dots, a_m\}$  and  $B = \{b_1, \dots, b_m\}$ . Note that  $u_S^\varphi$  is nonzero if and only if  $|S \cap \{a_i, b_i\}| = 1$  for each  $1 \leq i \leq m$ . By symmetry, we can assume that  $A \subseteq S$  and  $S \cap B = \emptyset$ . Then,

$$\mu_n = \sum_{T : |T|=\ell, |S \Delta T|=p, |T \cap \{a_i, b_i\}|=1 \forall i} (-1)^{|T \cap B|}.$$

Letting  $s = |T \cap B|$ , we will count the number of sets  $T$  appearing in the sum for a fixed value of  $s$ . First, there are  $\binom{m}{s}$  ways to choose which elements of  $A \sqcup B$  belong to  $T$ . Next, to achieve  $|S \setminus T| = p$ ,  $T$  must include exactly  $p/2 - s$  elements of  $\overline{S} \cup \overline{B}$ , and there are  $\binom{n-\ell-m}{p/2-s}$  ways to choose these. Finally, the remaining  $\ell - m - p/2 + s$  elements of  $T$  must come from  $S \setminus A$ , and there are  $\binom{\ell-m}{\ell-m-p/2+s} = \binom{\ell-m}{p/2-s}$  ways to choose these. Also note that we must have  $s \leq m$  and  $s \leq p/2$ , giving the bounds for the sum.  $\square$

As the following lemma shows, the succeeding eigenvalues decay rapidly with  $m$ .

LEMMA 17. *Let  $3 \leq p \leq \sqrt{n}$  and let  $\ell < n/p^2$ . For all  $0 \leq m \leq \ell - p/2$  it holds that*

$$\frac{|\mu_m|}{\mu_0} \leq \left(1 - \frac{m}{\ell}\right)^{p/2}.$$

PROOF. The terms in (16) have alternating signs. We will show that they decrease in absolute value beyond the first nonzero term, so that this gives a bound on  $\mu_m$ . Since  $m \leq \ell - p/2$ , the  $s = 0$  term is positive. The  $(s+1)$ st term divided by the  $s$ th term is, in absolute value,

$$\begin{aligned} \frac{\binom{m}{s+1} \binom{\ell-m}{p/2-s-1} \binom{n-\ell-m}{p/2-s-1}}{\binom{m}{s} \binom{\ell-m}{p/2-s} \binom{n-\ell-m}{p/2-s}} &= \left(\frac{m-s}{s+1}\right) \left(\frac{p/2-s}{\ell-m-p/2+s+1}\right) \left(\frac{p/2-s}{n-\ell-m-p/2+s+1}\right) \\ &\leq \frac{m(p/2)(p/2)}{(\ell-m-p/2+1)(n-\ell-m-p/2+1)} \\ &\leq \frac{(\ell-p/2)(p/2)(p/2)}{n-2\ell+1} \quad \text{since } m \leq \ell - p/2 \\ &\leq \frac{\ell p^2/4}{n-2\ell} < 1. \end{aligned}$$

We used the assumptions  $3 \leq p \leq \sqrt{n}$  and  $\ell p^2 \leq n$  to obtain the last bound. It follows that the  $s = 0$  term is an upper bound,

$$\mu_m \leq \binom{\ell-m}{p/2} \binom{n-\ell-m}{p/2},$$

and so

$$\frac{\mu_m}{\mu_0} \leq \binom{\ell-m}{p/2} / \binom{\ell}{p/2} \leq \left(\frac{\ell-m}{\ell}\right)^{p/2}. \quad \square$$

## 5.2 Proof of Strong Detection

Here, we prove our strong detection result, Theorem 11. The proof does not exploit the full details of the structure exhibited above. Instead, the proof is a straightforward application of the matrix Chernoff bound for Gaussian series [58] (see also Theorem 4.1 of Reference [71]):

THEOREM 18. *Let  $\{A_i\}$  be a finite sequence of fixed symmetric  $d \times d$  matrices, and let  $\{\xi_i\}$  be independent  $\mathcal{N}(0, 1)$  random variables. Let  $\Sigma = \sum_i \xi_i A_i$ . Then, for all  $t \geq 0$ ,*

$$\mathbb{P}(\|\Sigma\|_{\text{op}} \geq t) \leq 2de^{-t^2/2\sigma^2} \quad \text{where } \sigma^2 = \|\mathbb{E}[\Sigma^2]\|_{\text{op}}.$$

Furthermore, the same bound holds if  $\{\xi_i\}$  is a sequence of independent Rademacher random variables, i.e.,  $\pm 1$ -valued with equal probabilities.

Let us first write  $M$  in the form of a Gaussian series. For a set  $E \in \binom{[n]}{p}$ , define the  $\binom{n}{\ell} \times \binom{n}{\ell}$  matrix  $A_E$  as

$$(A_E)_{S,T} = 1_{S \triangle T = E}.$$

It is immediate that for  $\lambda = 0$ ,  $M = Z = \sum_{|E|=p} g_E A_E$  where  $(g_E)_{|E|=p}$  is a collection of i.i.d.  $\mathcal{N}(0, 1)$  random variables. The second moment of this random matrix is

$$\mathbb{E}[M^2] = \sum_{E : |E|=p} A_E^2 = d_\ell I,$$

since  $A_E^2$  is the diagonal matrix with  $(A_E)_{S,S} = 1_{|S \triangle E|=p}$ , and summing over all  $E$  gives  $d_\ell$  on the diagonal. The operator norm of the second moment is then  $d_\ell$ . It follows that for all  $t \geq 0$ ,

$$\mathbb{P}_0(\lambda_{\max}(M) \geq t) \leq 2 \binom{n}{\ell} e^{-t^2/2d_\ell} \leq 2n^\ell e^{-t^2/2d_\ell}. \quad (17)$$

Now letting  $t = \frac{\lambda d_\ell}{2}$  yields the first statement of the theorem.

As for the second statement, we have  $\lambda_{\max}(M) \geq \|X\|_{\text{op}} - \|Z\|_{\text{op}} = \lambda d_\ell - \|Z\|_{\text{op}}$  where  $Z$  is defined in (14). Applying the same bound we have

$$\mathbb{P}_\lambda\left(\lambda_{\max}(M) \leq \frac{\lambda d_\ell}{2}\right) \leq \mathbb{P}\left(\|Z\|_{\text{op}} \geq \frac{\lambda d_\ell}{2}\right) \leq 2n^\ell e^{-\lambda^2 d_\ell / 8}.$$

### 5.3 Proof of Strong Recovery

Here, we prove our strong recovery result, Theorem 12. Let  $v_0 = v_{\text{top}}(M)$  be a unit-norm leading eigenvector of  $M$ . For a fixed  $m \in [\ell]$  to be determined later on, we write the orthogonal decomposition  $v_0 = v^{(m)} + v^\perp$ , where  $v^{(m)} \in \bigoplus_{s \leq m} \mathcal{Y}_s$ , and  $v^\perp$  in the orthogonal complement. The goal is to first show that if  $m$  is proportional to  $\ell$  then  $v^\perp$  has small Euclidean norm, so that  $v_0$  and  $v^{(m)}$  have high inner product. The second step of the argument is to approximate the voting matrix  $V(v_0)$  by  $V(v^{(m)})$ , and then use Fourier-analytic tools to reason about the latter.

Let us start with the first step.

LEMMA 19. *With  $Z$  defined as in (14), we have*

$$\|v^\perp\|^2 \leq 2 \frac{\|Z\|_{\text{op}}}{\lambda(\mu_0 - \mu_{m+1})}.$$

PROOF. Let us absorb the factor  $\lambda$  in the definition of the matrix  $X$ . Let  $\{u_0, \dots, u_d\}$  be a set of eigenvectors of  $X$  which also form an orthogonal basis for  $\bigoplus_{s \leq m} \mathcal{Y}_s$ , with  $u_0$  being the top eigenvector of  $X$  ( $u_0$  is the normalized all-ones vector). We start with the inequality  $u_0^\top M u_0 \leq v_0^\top M v_0$ . The left-hand side of the inequality is  $\mu_0 + u_0^\top Z u_0$ . The right-hand side is  $v_0^\top X v_0 + v_0^\top Z v_0$ . Moreover  $v_0^\top X v_0 = v_0^\top X v^{(m)} + v_0^\top X v^\perp$ . Since  $v^{(m)} \in \bigoplus_{s \leq m} \mathcal{Y}_s$ , by Proposition 15,  $X v^{(m)}$  belongs to the space as well, and therefore  $v_0^\top X v^{(m)} = v^{(m)\top} X v^{(m)}$ . Similarly  $v_0^\top X v^\perp = (v^\perp)^\top X v^\perp$ , so  $v_0^\top X v_0 = v^{(m)\top} X v^{(m)} + (v^\perp)^\top X v^\perp$ . Therefore the inequality becomes

$$\mu_0 + u_0^\top Z u_0 \leq v^{(m)\top} X v^{(m)} + (v^\perp)^\top X v^\perp + v_0^\top Z v_0.$$

Since  $v^\perp$  is orthogonal to the top  $m$  eigenspaces of  $X$  we have  $(v^\perp)^\top X v^\perp \leq \mu_{m+1} \|v^\perp\|_2^2$ . Moreover,  $v^{(m)\top} X v^{(m)} \leq \mu_0 \|v^{(m)}\|^2$ , hence

$$\mu_0 + u_0^\top Z u_0 \leq \mu_0 \|v^{(m)}\|^2 + \mu_{m+1} \|v^\perp\|^2 + v_0^\top Z v_0.$$

By rearranging and applying the triangle inequality we get,

$$(\mu_0 - \mu_{m+1}) \|v^\perp\|^2 \leq |v_0^\top Z v_0| + |u_0^\top Z u_0| \leq 2\|Z\|_{\text{op}}. \quad \square$$

Combining this fact with Lemma 17 and recalling that  $\mu_0 = d_\ell$ , we obtain the following result:

LEMMA 20. For any  $\epsilon > 0$  and  $\delta \in (0, 1)$ , if  $\lambda \geq \epsilon^{-1} \sqrt{2 \log(n^\ell/\delta)/d_\ell}$ , and  $m + 1 \leq \ell - p/2$  then

$$\|v^\perp\|^2 \leq \epsilon \frac{\ell}{m+1}$$

with probability at least  $1 - \delta$ .

PROOF. Using Lemma 17 implies  $\frac{1}{\mu_0 - \mu_{m+1}} \leq \frac{1}{\mu_0} \frac{1}{1 - (1 - \frac{m+1}{\ell})^{p/2}} \leq \frac{1}{\mu_0} \cdot \frac{\ell}{m+1}$ . To obtain the last bound we use the inequalities  $\frac{1}{1 - (1-x)^\alpha} \leq \frac{1}{1 - e^{-\alpha x}} < \frac{1}{x}$ , the last one, obtained by simple calculus, is valid for all  $\alpha > 1$  and  $0 < x < x_0(\alpha) = \alpha^{-1} \log \alpha$ . We let  $x = (m+1)/\ell$  and  $\alpha = p/2 > 1$ . Therefore, Lemma 19 implies

$$\|v^\perp\|^2 \leq \frac{\|Z\|_{\text{op}}}{\lambda d_\ell} \frac{\ell}{m+1}.$$

The operator norm of the noise can be bounded by a matrix Chernoff bound [58, 71], similarly to our argument in the proof of detection: for all  $t \geq 0$ ,

$$\mathbb{P}(\|Z\|_{\text{op}} \geq t) \leq n^\ell e^{-t^2/2d_\ell}.$$

Therefore, letting  $\lambda \geq \epsilon^{-1} \sqrt{2 \log(n^\ell/\delta)/d_\ell}$  we obtain the desired result.  $\square$

**5.3.1 Analysis of the Voting Matrix.** Recall that the voting matrix  $V(v)$  of a vector  $v \in \mathbb{R}^{\binom{n}{\ell}}$  has zeros on the diagonal, and off-diagonal entries

$$\begin{aligned} V_{ij}(v) &= \frac{1}{2} \sum_{S, T \in \binom{[n]}{\ell}} v_S v_T 1_{S \Delta T = \{i, j\}} \\ &= \sum_{S \in \binom{[n]}{\ell}} v_S v_{S \Delta \{i, j\}} 1_{i \in S, j \notin S}, \quad 1 \leq i \neq j \leq n. \end{aligned}$$

It will be more convenient in our analysis to work with  $V(v^{(m)})$  instead of  $V(v_0)$ . To this end, we produce the following approximation result:

LEMMA 21. Let  $u, e \in \mathbb{R}^{\binom{n}{\ell}}$  and  $v = u + e$ . Then

$$\|V(v) - V(u)\|_F^2 \leq 3\ell^2 \|e\|^2 (2\|u\|^2 + \|e\|^2).$$

In particular,

$$\|V(v_0) - V(v^{(m)})\|_F^2 \leq 9\ell^2 \|v^\perp\|^2.$$

PROOF. Let us introduce the shorthand notation  $\langle u, v \rangle_{ij} := \sum_{S \in \binom{[n]}{\ell}} u_S v_{S \Delta \{i, j\}} 1_{i \in S, j \notin S}$ . We have

$$\begin{aligned} \|V(v) - V(u)\|_F^2 &= \sum_{i,j} (V_{ij}(u+e) - V_{ij}(u))^2 \\ &= \sum_{i,j} (\langle u+e, u+e \rangle_{ij} - \langle u, u \rangle_{ij})^2 \\ &= \sum_{i,j} (\langle u, e \rangle_{ij} + \langle e, u \rangle_{ij} + \langle e, e \rangle_{ij})^2 \\ &\leq 3 \sum_{i,j} (\langle u, e \rangle_{ij}^2 + \langle e, u \rangle_{ij}^2 + \langle e, e \rangle_{ij}^2), \end{aligned} \tag{18}$$

where the last step uses the bound  $(a + b + c)^2 \leq 3(a^2 + b^2 + c^2)$ . Now we expand

$$\begin{aligned}
\sum_{i,j} \langle u, e \rangle_{ij}^2 &= \sum_{i,j} \left( \sum_{S : |S|=\ell, i \in S, j \notin S} u_S e_{S \setminus \{i,j\}} \right)^2 \\
&\leq \sum_{i,j} \left( \sum_{S : |S|=\ell, i \in S, j \notin S} u_S^2 \right) \left( \sum_{S : |S|=\ell, i \in S, j \notin S} e_{S \setminus \{i,j\}}^2 \right) \quad (\text{Cauchy-Schwarz}) \\
&\leq \sum_{i,j} \left( \sum_{S : |S|=\ell, i \in S} u_S^2 \right) \left( \sum_{T : |T|=\ell, j \in T} e_T^2 \right) \\
&= \left( \sum_i \sum_{S : |S|=\ell, i \in S} u_S^2 \right) \left( \sum_j \sum_{T : |T|=\ell, j \in T} e_T^2 \right) \\
&= \left( \ell \sum_{|S|=\ell} u_S^2 \right) \left( \ell \sum_{|T|=\ell} e_T^2 \right) \\
&= \ell^2 \|u\|^2 \|e\|^2.
\end{aligned} \tag{19}$$

Plugging this back into (18) yields the desired result. To obtain  $\|V(v_0) - V(v^{(m)})\|_F^2 \leq 9\ell^2 \|v^\perp\|^2$  we just bound  $2\|v^{(m)}\|^2 + \|v^\perp\|^2$  by 3.  $\square$

Let us also state the following lemma, which will be need later on:

LEMMA 22. For  $u \in \mathbb{R}^{\binom{n}{\ell}}$ ,  $\|V(u)\|_F^2 \leq \ell^2 \|u\|^4$ .

PROOF. Note that  $\|V(u)\|_F^2 = \sum_{i,j} V_{ij}(u)^2 = \sum_{i,j} \langle u, u \rangle_{ij}^2$  and so the desired result follows immediately from (19).  $\square$

Next, in the main technical part of the proof, we show that  $V(v^{(m)})$  is close to a multiple of the all-ones matrix in Frobenius norm:

PROPOSITION 23. Let  $\hat{\mathbb{1}} = \mathbb{1}/\sqrt{n}$  and  $\alpha = \ell\|v^{(m)}\|^2$ . Then

$$\|V(v^{(m)}) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F^2 \leq 2 \left( \frac{m}{\ell} + \frac{\ell}{n} \right) \alpha^2.$$

Before proving the above proposition, let us put the results together and prove our recovery result.

5.3.2 *Proof of Theorem 12.* For  $\epsilon, \delta > 0$ , assume  $\lambda \geq \epsilon^{-1} \sqrt{2 \log(n^\ell/\delta)/d_\ell}$  and  $m+1 \leq \ell - p/2$ . By Lemma 20 and Lemma 21 we have

$$\|V(v_0) - V(v^{(m)})\|_F^2 \leq 9\ell^2 \frac{\ell}{m+1} \epsilon$$

with probability at least  $1 - \delta$ . Moreover, by Proposition 23, we have

$$\|V(v^{(m)}) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F^2 \leq 2\eta\alpha^2,$$

with  $\alpha = \ell\|v^{(m)}\|^2 \leq \ell$  and  $\eta = \frac{m}{\ell} + \frac{\ell}{n}$ . Therefore, by a triangle inequality we have

$$\begin{aligned}
\|V(v_0) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F &\leq 3\ell \sqrt{\frac{\ell\epsilon}{m+1}} + \sqrt{2\eta}\alpha \\
&\leq 3\ell \left( \sqrt{\frac{\ell\epsilon}{m+1}} + \sqrt{\frac{m}{\ell} + \frac{\ell}{n}} \right),
\end{aligned}$$

with probability at least  $1 - \delta$ . Now let us choose a value of  $m$  that achieves a good tradeoff of the above two error terms: we take  $m = \lfloor \ell\sqrt{\epsilon} \rfloor$  if this choice satisfies the condition  $m + 1 \leq \ell - p/2$ , i.e., if  $\ell \geq (p/2 + 1)/(1 - \sqrt{\epsilon})$ . Otherwise, we take  $m = 0$ . Use the inequality  $\sqrt{a + b} \leq \sqrt{a} + \sqrt{b}$  for positive  $a, b$ , we obtain

$$\|V(v_0) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F \leq 3\ell \max \left\{ \left( 2\epsilon^{1/4} + \sqrt{\frac{\ell}{n}} \right), c(p) \left( \sqrt{\epsilon} + \sqrt{\frac{1}{n}} \right) \right\}, \quad (20)$$

$$\leq 6\ell \left( \epsilon^{1/4} + \sqrt{\frac{\ell}{n}} \right), \quad (21)$$

where  $c(p)$  is a constant depending only on  $p$  when  $\epsilon$  is bounded away from 1. (The first term in the above upper bound corresponds to the case  $m = \lfloor \ell\sqrt{\epsilon} \rfloor$ ,  $\ell \geq (p/2 + 1)/(1 - \sqrt{\epsilon})$ , and the second term to  $m = 0$ ,  $\ell \leq (p/2 + 1)/(1 - \sqrt{\epsilon})$ .) For small  $\epsilon$  and large  $n$  the first term in (20) dominates, leading to the bound (21).

Now let  $\hat{x}$  be a leading eigenvector of  $V(v_0)$ , and let  $R = V(v_0) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top$ . Since  $\hat{x}^\top V(v_0) \hat{x} \geq \hat{\mathbb{1}}^\top V(v_0) \hat{\mathbb{1}}$ , we have

$$\alpha \langle \hat{x}, \hat{\mathbb{1}} \rangle^2 + \hat{x}^\top R \hat{x} \geq \alpha + \hat{\mathbb{1}}^\top R \hat{\mathbb{1}}.$$

Therefore

$$\langle \hat{x}, \hat{\mathbb{1}} \rangle^2 \geq 1 - 2 \frac{\|R\|_{\text{op}}}{\alpha}.$$

Since  $\alpha = \ell(1 - \|v^\perp\|^2)$ , and  $\|v^\perp\|^2 \leq \epsilon \frac{\ell}{m+1} \leq \max\{\sqrt{\epsilon}, c(p)\epsilon\} \leq \sqrt{\epsilon}$  for small  $\epsilon$ , the bound (21) (together with  $\|R\|_{\text{op}} \leq \|R\|_F$ ) implies

$$\langle \hat{x}, \hat{\mathbb{1}} \rangle^2 \geq 1 - 12 \frac{\epsilon^{1/4} + \sqrt{\ell/n}}{1 - \sqrt{\epsilon}}. \quad (22)$$

To conclude the proof of our theorem, we let  $\ell \leq n\sqrt{\epsilon}$ ,  $\epsilon < 1/16$  and then replace  $\epsilon$  by  $\epsilon^4$ : we obtain  $\langle \hat{x}, \hat{\mathbb{1}} \rangle^2 \geq 1 - 48\epsilon$  with probability at least  $1 - \delta$  if  $\lambda \geq \epsilon^{-4} \sqrt{2 \log(n^\ell/\delta)/d_\ell}$ .

**5.3.3 Proof of Proposition 23.** Let  $\alpha = \ell\|v^{(m)}\|^2$ . By Lemma 22 we have  $\|V(v^{(m)})\|_F^2 \leq \alpha^2$ . Therefore

$$\begin{aligned} \|V(v^{(m)}) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F^2 &= \|V(v^{(m)})\|_F^2 - \frac{2}{n} \alpha \sum_{i,j=1}^n V_{ij}(v^{(m)}) + \alpha^2 \\ &\leq 2\alpha^2 - \frac{2}{n} \alpha \sum_{i,j=1}^n V_{ij}(v^{(m)}). \end{aligned} \quad (23)$$

Now we need a lower bound on  $\sum_{i,j=1}^n V_{ij}(v^{(m)})$ . This will crucially rely on the fact that  $v^{(m)}$  lies in the span of the top  $m$  eigenspaces of  $X$ :

**LEMMA 24.** For a fixed  $m \leq \ell$ , let  $v \in \bigoplus_{s=0}^m \mathcal{Y}_s$ . Then

$$\sum_{i,j=1}^n V_{ij}(v) \geq ((\ell - m)n - \ell^2) \|v\|^2.$$

We substitute the result of the above lemma in (23) to obtain

$$\begin{aligned} \|V(v^{(m)}) - \alpha \hat{\mathbb{1}} \hat{\mathbb{1}}^\top\|_F^2 &\leq 2\alpha^2 \left( 1 - \frac{\ell - m}{\ell} + \frac{\ell}{n} \right) \\ &= 2\alpha^2 \left( \frac{m}{\ell} + \frac{\ell}{n} \right), \end{aligned}$$

as desired.

**5.3.4 A Poincaré Inequality on a Slice of the Hypercube.** To prove Lemma 24, we need some results on Fourier analysis on the slice of the hypercube  $\binom{[n]}{\ell}$ . Following [33], we define the following. First, given a function  $f : \binom{[n]}{\ell} \mapsto \mathbb{R}$ , we define its expectation as its average value over all sets of size  $\ell$ , and write

$$\mathbb{E}_{|S|=\ell}[f(S)] := \frac{1}{\binom{n}{\ell}} \sum_{|S|=\ell} f(S).$$

We also define its variance as

$$\mathbb{V}[f] := \mathbb{E}_{|S|=\ell}[f(S)^2] - \mathbb{E}_{|S|=\ell}[f(S)]^2.$$

Moreover, we identify a vector  $u \in \mathbb{R}^{\binom{n}{\ell}}$  with a function on sets of size  $\ell$  in the obvious way:  $f(S) = u_S$ .

**Definition 25.** For  $u \in \mathbb{R}^{\binom{n}{\ell}}$  and  $i, j \in [n]$ , let  $u^{(ij)}$  denote the vector having coordinates

$$u_S^{(ij)} = \begin{cases} u_{S \Delta \{i,j\}} & \text{if } |S \cap \{i, j\}| = 1 \\ u_S & \text{otherwise.} \end{cases}$$

(The operation  $u \mapsto u^{(ij)}$  exchanges the roles of  $i$  and  $j$  whenever possible.)

**Definition 26.** For  $u \in \mathbb{R}^{\binom{n}{\ell}}$  and  $i, j \in [n]$ , define the *influence* of the pair  $(i, j)$  as

$$\text{Inf}_{ij}[u] := \frac{1}{2} \mathbb{E}_{|S|=\ell} \left[ \left( u_S^{(ij)} - u_S \right)^2 \right],$$

and the *total influence* as

$$\text{Inf}[u] := \frac{1}{n} \sum_{i < j} \text{Inf}_{ij}[u].$$

With this notation, our main tool is a version of Poincaré's inequality on  $\binom{[n]}{\ell}$ :

**LEMMA 27 (LEMMA 5.6 IN [33]).** For  $v \in \bigoplus_{s=0}^m \mathcal{Y}_s$  we have

$$\text{Inf}[v] \leq m \mathbb{V}[v].$$

**PROOF OF LEMMA 24.** We have  $m \mathbb{E}_{|S|=\ell}[v_S^2] \geq m \mathbb{V}[v] \geq \text{Inf}[v] = \frac{1}{n} \sum_{i < j} \text{Inf}_{ij}[v]$ , and

$$\begin{aligned} 2 \text{Inf}_{ij}[v] &= \mathbb{E}_{|S|=\ell} \left[ \left( v_S^{(ij)} - v_S \right)^2 \right] \\ &= \mathbb{E}_{|S|=\ell} \left[ 1_{|S \cap \{i,j\}|=1} \left( v_S^{(ij)} - v_S \right)^2 \right] \\ &= \mathbb{E}_{|S|=\ell} \left[ 1_{|S \cap \{i,j\}|=1} \left( \left( v_S^{(ij)} \right)^2 - 2v_S^{(ij)} v_S + v_S^2 \right) \right]. \end{aligned}$$

Since  $\mathbb{E}_{|S|=\ell}[1_{|S \cap \{i,j\}|=1}(v_S^{(ij)})^2] = \mathbb{E}_{|S|=\ell}[1_{|S \cap \{i,j\}|=1}v_S^2]$ , the above is equal to

$$2 \mathbb{E}_{|S|=\ell} \left[ 1_{|S \cap \{i,j\}|=1} \left( v_S^2 - v_S^{(ij)} v_S \right) \right].$$

Now,

$$\begin{aligned}
\frac{1}{n} \sum_{i < j} \text{Inf}_{ij}[v] &= \frac{1}{n} \sum_{i < j} \binom{n}{\ell}^{-1} \sum_{S : |S|=\ell, |S \cap \{i,j\}|=1} \left( v_S^2 - v_S^{(ij)} v_S \right) \\
&= \frac{1}{n} \binom{n}{\ell}^{-1} \left[ \ell(n-\ell) \sum_{|S|=\ell} v_S^2 - \sum_{i,j : i \neq j} \sum_{S : |S|=\ell, i \in S, j \notin S} v_S v_{S \setminus \{i,j\}} \right] \\
&= \frac{1}{n} \binom{n}{\ell}^{-1} \left[ \ell(n-\ell) \sum_{|S|=\ell} v_S^2 - \sum_{i,j} V_{ij}(v) \right] \\
&= \frac{1}{n} \binom{n}{\ell}^{-1} \left[ \ell(n-\ell) \|v\|^2 - \sum_{i,j} V_{ij}(v) \right].
\end{aligned}$$

Therefore

$$m\|v\|^2 \geq \frac{1}{n} \left( \ell(n-\ell) \|v\|^2 - \sum_{i,j} V_{ij}(v) \right),$$

and so

$$\sum_{i,j} V_{ij}(v) \geq (\ell(n-\ell) - mn) \|v\|^2 = ((\ell-m)n - \ell^2) \|v\|^2.$$

as desired.  $\square$

## 6 Extensions

### 6.1 Refuting Random $k$ -XOR Formulas for $k$ Even

Our symmetric difference matrices can be used to give a simple algorithm and proof for a related problem: strongly refuting random  $k$ -XOR formulas (see References [2, 64] and references therein). This is essentially a variant of the spiked tensor problem with sparse Rademacher observations instead of Gaussian ones. It is known [64] that this problem exhibits a smooth tradeoff between subexponential runtime and the number of constraints required, but the proof of [64] involves intensive moment calculations. When  $k$  is even, we will give a simple algorithm and a simple proof using the matrix Chernoff bound that achieves the same tradeoff. SOS lower bounds suggest that this tradeoff is optimal [36, 68].

After the initial appearance of our work, the follow-up work [37] extended our approach to the case where  $k$  is odd. Surprisingly, they showed that the construction we used for odd-order tensor PCA in Section 6.2 does not work as is, and requires an additional “row pruning” step.

**6.1.1 Setup.** Let  $x_1, \dots, x_n$  be  $\{\pm 1\}$ -valued variables. A  $k$ -XOR formula  $\Phi$  with  $m$  constraints is specified by a sequence of subsets  $U_1, \dots, U_m$  with  $U_i \subseteq [n]$  and  $|U_i| = k$ , along with values  $b_1, \dots, b_m$  with  $b_i \in \{\pm 1\}$ . For  $1 \leq i \leq m$ , constraint  $i$  is satisfied if  $x^{U_i} = b_i$ , where  $x^{U_i} := \prod_{j \in U_i} x_j$ . We write  $P_\Phi(x)$  for the number of constraints satisfied by  $x$ . We will consider a uniformly random  $k$ -XOR formula in which each  $U_i$  is chosen uniformly and independently from the  $\binom{n}{k}$  possible  $k$ -subsets, and each  $b_i$  is chosen uniformly and independently from  $\{\pm 1\}$ .

Given a formula  $\Phi$ , the goal of strong refutation is to certify an upper bound on the number of constraints that can be satisfied. In other words, our algorithm should output a bound  $B = B(\Phi)$  such that for every formula  $\Phi$ ,  $\max_{x \in \{\pm 1\}^n} P_\Phi(x) \leq B(\Phi)$ . (Note that this must *always* be satisfied, not merely with high probability.) At the same time, we want the bound  $B$  to be small with high probability over a random  $\Phi$ . Since a random assignment  $x$  will satisfy roughly half the constraints, the best bound we can hope for is  $B = \frac{m}{2}(1 + \varepsilon)$  with  $\varepsilon > 0$  small.

**6.1.2 Algorithm.** Let  $k \geq 2$  be even and let  $\ell \geq k/2$ . Given a  $k$ -XOR formula  $\Phi$ , construct the order- $\ell$  symmetric difference matrix  $M \in \mathbb{R}^{\binom{[n]}{\ell} \times \binom{[n]}{\ell}}$  as follows. For  $S, T \subseteq [n]$  with  $|S| = |T| = \ell$  and  $i \in [n]$ , let

$$M_{S,T}^{(i)} = \begin{cases} b_i & \text{if } S \triangle T = U_i \\ 0 & \text{otherwise} \end{cases}$$

and let

$$M = \sum_{i=1}^m M^{(i)}.$$

Define the parameter

$$d_\ell := \binom{n-\ell}{k/2} \binom{\ell}{k/2},$$

which, for any fixed  $|S| = \ell$ , is the number of sets  $|T| = \ell$  such that  $|S \triangle T| = k$ . For an assignment  $x \in \{\pm 1\}^n$ , let  $u^x \in \mathbb{R}^{\binom{[n]}{\ell}}$  be defined by  $u_S^x = x^S$  for all  $|S| = \ell$ . We have

$$\|M\| \geq \frac{(u^x)^\top M u^x}{\|u^x\|^2} = \binom{n}{\ell}^{-1} \sum_{i=1}^m \sum_{S \triangle T = U_i} x^{U_i} b_i = d_\ell \binom{n}{k}^{-1} (2P_\Phi(x) - m)$$

since for any fixed  $U_i$  (of size  $k$ ), the number of  $(S, T)$  pairs such that  $S \triangle T = U_i$  is  $\binom{n}{\ell} d_\ell \binom{n}{k}^{-1}$ . Thus, we can perform strong refutation by computing  $\|M\|$ :

$$P_\Phi(x) \leq \frac{m}{2} + \frac{1}{2d_\ell} \binom{n}{k} \|M\|. \quad (24)$$

**THEOREM 28.** Let  $k \geq 2$  be even and let  $k/2 \leq \ell \leq n-k/2$ . Let  $\beta \in (0, 1)$ . If

$$m \geq \frac{4e^2 \binom{n}{k} \log \binom{n}{\ell}}{\beta^2 d_\ell} \quad (25)$$

then  $\|M\|$  certifies

$$P_\Phi(x) \leq \frac{m}{2}(1 + \beta)$$

with probability at least  $1 - 3 \binom{n}{\ell}^{-1}$  over a uniformly random  $k$ -XOR formula  $\Phi$  with  $m$  constraints.

If  $k$  is constant and  $\ell = n^\delta$  with  $\delta \in (0, 1)$ , the condition (25) becomes

$$m \geq O(\beta^{-2} n^{k/2} \ell^{1-k/2} \log n) = O(\beta^{-2} n^{k/2 + \delta(1-k/2)} \log n),$$

matching the result of [64]. In fact, our result is tighter by polylog factors.

**6.1.3 Binomial Tail Bound.** The main ingredients in the proof of Theorem 28 will be the matrix Chernoff bound (Theorem 18) and the following standard binomial tail bound.

**PROPOSITION 29.** Let  $X \sim \text{Binomial}(n, p)$ . For  $p < \frac{u}{n} < 1$ ,

$$\mathbb{P}(X \geq u) \leq \exp \left[ -u \left( \log \left( \frac{u}{pn} \right) - 1 \right) \right].$$

**PROOF.** We begin with the standard Binomial tail bound [9]

$$\mathbb{P}(X \geq u) \leq \exp \left( -n D \left( \frac{u}{n} \parallel p \right) \right)$$

for  $p < \frac{u}{n} < 1$ , where

$$D(a \| p) := a \log\left(\frac{a}{p}\right) + (1-a) \log\left(\frac{1-a}{1-p}\right).$$

Since  $\log(x) \geq 1 - 1/x$ ,

$$(1-a) \log\left(\frac{1-a}{1-p}\right) \geq (1-a)\left(1 - \frac{1-p}{1-a}\right) = p - a \geq -a,$$

and the desired result follows.  $\square$

#### 6.1.4 Proof.

**PROOF OF THEOREM 28.** We need to bound  $\|M\|$  with high probability over a uniformly random  $k$ -XOR formula  $\Phi$ . First, fix the subsets  $U_1, \dots, U_m$  and consider the randomness of the signs  $b_i$ . We can write  $M$  as a Rademacher series

$$M = \sum_{i=1}^m b_i A^{(i)}$$

where

$$A_{S,T}^{(i)} = \mathbb{1}_{S \triangle T = U_i}.$$

By the matrix Chernoff bound (Theorem 18),

$$\mathbb{P}(\|M\| \geq t) \leq 2 \binom{n}{\ell} e^{-t^2/2\sigma^2} = 2 \exp\left(\log\binom{n}{\ell} - \frac{t^2}{2\sigma^2}\right)$$

where

$$\sigma^2 = \left\| \sum_{i=1}^m (A^{(i)})^2 \right\|.$$

In particular,

$$\mathbb{P}\left(\|M\| \geq 2\sqrt{\sigma^2 \log\binom{n}{\ell}}\right) \leq 2\binom{n}{\ell}^{-1}. \quad (26)$$

Now we will bound  $\sigma^2$  with high probability over the random choice of  $U_1, \dots, U_m$ . We have

$$\sum_{i=1}^m (A^{(i)})^2 = \text{diag}(D)$$

where  $D_S$  is the number of  $i$  for which  $|S \triangle U_i| = \ell$ . This means  $\sigma^2 = \max_{|S|=\ell} D_S$ . For fixed  $S \subseteq [n]$  with  $|S| = \ell$ , the number of sets  $U \subseteq [n]$  with  $|U| = k$  such that  $|S \triangle U| = \ell$  is  $d_\ell$  and so  $D_S \sim \text{Binomial}(m, p)$  with  $p := d_\ell \binom{n}{k}^{-1}$ . Using the Binomial tail bound (Proposition 29) and a union bound over  $S$ ,

$$\mathbb{P}(\sigma^2 \geq u) \leq \binom{n}{\ell} \exp\left[-u\left(\log\left(\frac{u}{pm}\right) - 1\right)\right] = \exp\left[\log\binom{n}{\ell} - u\left(\log\left(\frac{u}{pm}\right) - 1\right)\right].$$

Provided

$$\frac{u}{pm} \geq e^2 \quad \text{and} \quad u \geq 2 \log\binom{n}{\ell}, \quad (27)$$

we have

$$\mathbb{P}(\sigma^2 \geq u) \leq \binom{n}{\ell}^{-1}.$$

Let  $\beta \in (0, 1)$ . From (24), to certify  $P_\Phi(x) \leq \frac{m}{2}(1 + \beta)$  it suffices to have  $\|M\| \leq \beta m d_\ell \binom{n}{\ell}^{-1} = \beta p m$ . Therefore, from (26), it suffices to have

$$\sigma^2 \leq \frac{\beta^2 p^2 m^2}{4 \log \binom{n}{\ell}}.$$

From (27), this will occur provided

$$\frac{\beta^2 p^2 m^2}{4 \log \binom{n}{\ell}} \geq e^2 p m \Leftrightarrow m \geq \frac{4e^2 \log \binom{n}{\ell}}{\beta^2 p} \quad (28)$$

and

$$\frac{\beta^2 p^2 m^2}{4 \log \binom{n}{\ell}} \geq 2 \log \binom{n}{\ell} \Leftrightarrow m \geq \frac{2\sqrt{2} \log \binom{n}{\ell}}{\beta p}. \quad (29)$$

Note that (29) is subsumed by (28). This completes the proof.  $\square$

## 6.2 Odd-Order Tensors

We return now to tensor PCA rather than  $k$ -XOR. When the tensor order  $p$  is odd, the Kikuchi Hessian suggests a natural algorithm for tensor PCA (see Appendix B) but we are unfortunately unable to give a tight analysis of it. Here, we present a related algorithm for which we are able to give a better analysis, matching SOS. The idea of the algorithm is to use a construction from the SOS literature that transforms an order- $p$  tensor (with  $p$  odd) into an order-2( $p - 1$ ) tensor via the Cauchy–Schwarz inequality [23]. We then apply a variant of our symmetric difference matrix to the resulting even-order tensor. A similar construction was given independently in the recent work [38] and shown to give optimal performance for all  $\ell \leq n^\delta$  for a certain constant  $\delta > 0$ . The proof we give here applies to the full range of  $\ell$  values:  $\ell \ll n$ . Our proof uses a certain variant of the matrix Bernstein inequality combined with some fairly simple moment calculations.

**6.2.1 Setup.** For simplicity, we consider the following certification version of the tensor PCA problem. Let  $p \geq 3$  be odd and let  $Y \in (\mathbb{R}^n)^{\otimes p}$  be an asymmetric tensor with i.i.d. Rademacher (uniform  $\pm 1$ ) entries. Our goal is to certify an upper bound on the *Rademacher injective norm*, defined as

$$\|Y\|_{\pm} := \max_{x \in \{\pm 1\}^n / \sqrt{n}} |\langle Y, x^{\otimes p} \rangle|.$$

The true value is  $O(\sqrt{n})$  with high probability. In time  $n^\ell$  (where  $\ell = n^\delta$  with  $\delta \in (0, 1)$ ) we will certify the bound  $\|Y\|_{\pm} \leq n^{p/4} \ell^{1/2-p/4} \text{polylog}(n)$ , matching the results of References [16, 17]. Following Lemma 4.4 of [41], such certification results can be transformed (via sum-of-squares) into recovery results for tensor PCA under the optimal condition  $\lambda \geq \ell^{1/2-p/4} n^{-p/4} \text{polylog}(n)$ . However, our specific result would apply to a strange variant of tensor PCA where both the signal and noise are Rademacher. To handle Gaussian noise, one would need to take  $Y$  Gaussian in our result, but we do not attempt this here. To certify a bound on the *injective norm* instead of the Rademacher injective norm (where  $x$  is constrained to the sphere instead of the hypercube), one should use the basis-invariant version of the symmetric difference matrices given by Reference [38] (but again, we do not attempt this here).

**6.2.2 Algorithm.** We will use a trick from [23] which is often used in the sum-of-squares literature. For any  $\|x\| = 1$ , we have by the Cauchy–Schwarz inequality,

$$\langle Y, x^{\otimes p} \rangle^2 \leq \|x\|^2 \langle T, x^{\otimes 4q} \rangle = \langle T, x^{\otimes 4q} \rangle$$

where  $p = 2q + 1$  and  $T_{abcd} := \sum_{e \in [n]} Y_{ace} Y_{bde}$  where  $a, b, c, d \in [n]^q$ . We have  $\mathbb{E}[T]_{abcd} = n \cdot 1\{ac = bd\}$  and so  $\langle \mathbb{E}[T], x^{\otimes 4q} \rangle = n \sum_{ac} (x^a x^c)^2 = n$ . Let  $\tilde{T} = T - \mathbb{E}[T]$ , i.e.,  $\tilde{T}_{abcd} = T_{abcd} \cdot 1\{ac \neq bd\}$ . (The symbols  $ab$  and  $cd$  in the indicators are interpreted as concatenation of strings.) For some  $\ell \geq 2q$ , define the  $n^\ell \times n^\ell$  matrix  $M$  as follows. For  $S, T \in [n]^\ell$ ,

$$M_{S,T} := \sum_{abcd} \tilde{T}_{abcd} N_{ab,cd}^{-1} \cdot 1\{S \xrightarrow{ab,cd} T\}$$

where  $S \xrightarrow{ab,cd} T$  roughly means that  $S$  is obtained from  $T$  by replacing  $ab$  by  $cd$ , or  $cd$  by  $ab$ ; the formal definition is given below. Also,  $N_{ab,cd}$  denotes the number of  $(S, T)$  pairs for which  $S \xrightarrow{ab,cd} T$ .

**Definition 30.** For  $S, T \in [n]^\ell$  and  $a, b, c, d \in [n]^q$ , we write  $S \xrightarrow{ab,cd} T$  if there are distinct indices  $i_1, \dots, i_{2q} \in [\ell]$  such that either: (i)  $S_{i_j} = (ab)_j$  and  $T_{i_j} = (cd)_j$  for all  $j \in [2q]$ , the values in  $a, b, c, d$  do not appear anywhere else in  $S$  or  $T$ , and  $S, T$  are identical otherwise:  $S_i = T_i$  for all  $i \notin \{i_1, \dots, i_{2q}\}$ ; or (ii) the same holds but with  $ab$  and  $cd$  interchanged.

Note that

$$N_{ab,cd} \geq \bar{N} := \binom{\ell}{2q} (n - 4q)^{\ell - 2q}. \quad (30)$$

The above construction ensures that

$$n^\ell (x^{\otimes \ell})^\top M(x^{\otimes \ell}) = n^{2q} \langle \tilde{T}, x^{\otimes 4q} \rangle \quad \text{for all } x \in \{\pm 1\}^n / \sqrt{n}.$$

This means we can certify an upper bound on  $\|Y\|_\pm$  by computing  $\|M\|$ :

$$\|Y\|_\pm \leq \sqrt{\langle T, x^{\otimes 4q} \rangle} \leq \sqrt{\langle \mathbb{E}[T], x^{\otimes 4q} \rangle} + \sqrt{\langle \tilde{T}, x^{\otimes 4q} \rangle} \leq n^{1/2} + n^{\ell/2 - q} \|M\|^{1/2}.$$

**THEOREM 31.** Let  $p \geq 3$  be odd and let  $p - 1 \leq \ell \leq \min\{\frac{n}{4(p-1)}, \frac{n}{8 \log n}\}$ . Then  $\|M\|$  certifies

$$\|Y\|_\pm \leq n^{1/2} + 8p^p \ell^{1/2 - p/4} n^{p/4} (\log n)^{1/4}$$

with probability at least  $1 - n^{-\ell}$  over an i.i.d. Rademacher  $Y$ .

**6.2.3 Proof.** We will use the following variant of the matrix Bernstein inequality; this is a special case ( $A_k = R \cdot I$ ) of [71], Theorem 6.2.

**THEOREM 32 (MATRIX BERNSTEIN).** Consider a finite sequence  $\{X_i\}$  of independent random symmetric  $d \times d$  matrices. Suppose  $\mathbb{E}[X_i] = 0$  and  $\|\mathbb{E}[X_i^r]\| \leq \frac{r!}{2} R^r$  for  $r = 2, 3, 4, \dots$ . Then

$$\Pr \left\{ \left\| \sum_{i=1}^n X_i \right\| \geq t \right\} \leq d \cdot \exp \left( \frac{-t^2/2}{nR^2 + Rt} \right).$$

For  $e \in [n]$ , let

$$M_{S,T}^{(e)} := \sum_{abcd} Y_{ace} Y_{bde} N_{ab,cd}^{-1} \cdot 1\{S \xrightarrow{ab,cd} T\} \cdot 1\{ac \neq bd\}.$$

We will apply Theorem 32 to the sum  $M = \sum_e M^{(e)}$ . Note that  $\mathbb{E}[M^{(e)}] = 0$ . To bound the moments  $\|\mathbb{E}[(M^{(e)})^r]\|$ , we will use the following basic fact.

**LEMMA 33.** If  $A$  is a symmetric matrix,

$$\|A\| \leq \max_j \sum_i |A_{ij}|.$$

PROOF. Let  $v$  be the leading eigenvector of  $A$  so that  $Av = \lambda v$  where  $\|A\| = |\lambda|$ . Normalize  $v$  so that  $\|v\|_1 = 1$ . Then  $\|Av\|_1 = |\lambda| \cdot \|v\|_1$  and so

$$\|A\| = |\lambda| \cdot \|v\|_1 = \sum_i \left| \sum_j A_{ij} v_j \right| \leq \sum_{ij} |A_{ij}| \cdot |v_j| \leq \sum_j |v_j| \cdot \sum_i |A_{ij}| \leq \|v\|_1 \cdot \max_j \sum_i |A_{ij}|. \quad \square$$

PROOF OF THEOREM 31. For any fixed  $e$ , we have by Lemma 33,

$$\mathbb{E}[(M^{(e)})^r] \leq \max_S \sum_T |\mathbb{E}[(M^{(e)})^r]_{S,T}| =: \max_S h(r, e, S).$$

Let  $\pi$  denote a “path” of the form

$$\pi = (S_0, a_1, b_1, c_1, d_1, S_1, a_2, b_2, c_2, d_2, S_2, \dots, a_r, b_r, c_r, d_r, S_r)$$

such that  $S_0 = S$ ,  $(a_i, c_i) \neq (b_i, d_i)$ , and  $S_{i-1} \xleftrightarrow{a_i b_i, c_i d_i} S_i$ . Then, we have

$$h(r, e, S) = \sum_{\pi} \mathbb{E} \prod_{i=1}^r Y_{a_i c_i e} Y_{b_i d_i e} N_{a_i b_i, c_i d_i}^{-1}.$$

Among tuples of the form  $(a_i, c_i)$  and  $(b_i, d_i)$ , each must occur an even number of times (or else the term associated with  $\pi$  is 0). There are  $2r$  such tuples, so there are  $\binom{2r}{r} r! 2^{-r}$  ways to pair them up (via a perfect matching on  $2r$  elements). Once  $S_{i-1}$  is chosen, there are at most  $2(\ell n)^q$  choices for  $(a_i, c_i)$ —where the factor of 2 accounts for the possibility of interchanging  $ab$  and  $cd$ —and the same is true for  $(b_i, d_i)$ . However, the second occurrence of  $(a_i, c_i)$  or  $(b_i, d_i)$  within a pair has only 1 choice. Once  $S_{i-1}, a_i, b_i, c_i, d_i$  are chosen, there are at most  $(2q)!$  possible choices for  $S_i$  (the worst case being e.g. when  $a_i, b_i$  are all 1’s, but  $c_i, d_i$  have distinct elements). This means

$$h(r, e, S) \leq \binom{2r}{r} r! 2^{-r} [2(\ell n)^q \cdot (2q)!]^r \bar{N}^{-r}.$$

where  $\bar{N}$  is defined in (30). Since  $\binom{2r}{r} \leq 4^r$ , we can apply Theorem 32 with  $R = 8(2q)!(\ell n)^q \bar{N}^{-1}$ . This yields

$$\Pr \{\|M\| \geq t\} \leq n^\ell \cdot \exp \left( \frac{-t^2/2}{nR^2 + Rt} \right).$$

Let  $t = R\sqrt{8\ell n \log n}$ . Provided  $\ell \leq n/(8 \log n)$  we have  $Rt \leq nR^2$  and so

$$\Pr \{\|M\| \geq R\sqrt{8\ell n \log n}\} \leq \exp \left( \ell \log n - \frac{t^2}{4nR^2} \right) = \exp(-\ell \log n) = n^{-\ell}.$$

Thus, with high probability we certify

$$\begin{aligned} \|Y\|_{\pm} &\leq n^{1/2} + n^{\ell/2-q} \|M\|^{1/2} \\ &\leq n^{1/2} + n^{\ell/2-q} R^{1/2} (8\ell n \log n)^{1/4} \\ &= n^{1/2} + n^{\ell/2-q} \sqrt{8(2q)!} (\ell n)^{q/2} \bar{N}^{-1/2} (8\ell n \log n)^{1/4} \\ &= n^{1/2} + 8^{3/4} \sqrt{(2q)!} n^{\ell/2-q} \bar{N}^{-1/2} (\ell n)^{1/4+q/2} (\log n)^{1/4}. \end{aligned}$$

We have the following bound on  $\bar{N}$ :

$$\bar{N} = \binom{\ell}{2q} (n - 4q)^{\ell-2q}$$

$$\begin{aligned}
&= n^\ell \cdot \frac{\binom{\ell}{2q} (n - 4q)^{\ell-2q}}{n^\ell} \\
&\geq n^\ell \cdot \frac{\ell^{2q}}{(2q)^{2q}} \cdot \frac{(n - 4q)^{\ell-2q}}{n^\ell} \\
&= \frac{n^\ell}{(2q)^{2q}} \cdot \left(\frac{n - 4q}{n}\right)^{\ell-2q} \left(\frac{\ell}{n}\right)^{2q} \\
&= \frac{n^\ell}{(2q)^{2q}} \cdot \left(1 - \frac{4q}{n}\right)^{\ell-2q} \left(\frac{\ell}{n}\right)^{2q} \\
&\geq \frac{n^\ell}{p^p} \cdot \left(1 - (\ell - 2q) \frac{4q}{n}\right) \left(\frac{\ell}{n}\right)^{2q} \\
&\geq \frac{n^\ell}{p^p} \cdot \left(1 - \frac{4q\ell}{n}\right) \left(\frac{\ell}{n}\right)^{2q} \\
&\geq \frac{n^\ell}{2p^p} \left(\frac{\ell}{n}\right)^{2q}
\end{aligned}$$

provided  $\ell \leq n/(8q)$ , i.e.,  $\ell \leq n/[4(p - 1)]$ . Therefore, we certify

$$\begin{aligned}
\|\mathbb{Y}\|_{\pm} &\leq n^{1/2} + 2^{1/2} \cdot 8^{3/4} \sqrt{(2q)!} p^{p/2} \ell^{1/4+q/2} n^{1/4-q/2} (\ell/n)^{-q} (\log n)^{1/4} \\
&\leq n^{1/2} + 8p^p \ell^{1/2-p/4} n^{p/4} (\log n)^{1/4},
\end{aligned}$$

as desired.  $\square$

## 7 Conclusion

We have presented a hierarchy of spectral algorithms for tensor PCA, inspired by variational inference and statistical physics. In particular, the core idea of our approach is to locally minimize the Kikuchi free energy. We specifically implemented this via the Kikuchi Hessian, but there may be many other viable approaches to minimizing the Kikuchi free energy such as generalized belief propagation [75]. Broadly speaking, we conjecture that for many average-case problems, algorithms based on Kikuchi free energy and algorithms based on sum-of-squares should both achieve the optimal tradeoff between runtime and statistical power. One direction for further work is to verify that this analogy holds for problems other than tensor PCA; in particular, we show here that it also applies to refuting random  $k$ -XOR formulas when  $k$  is even. Some other models to consider in the future could be planted clique or sparse PCA, which also have smooth tradeoffs between SNR and runtime [3, 28].

Perhaps one benefit of the Kikuchi hierarchy over the sum-of-squares hierarchy is that it has allowed us to *systematically* obtain spectral methods, simply by computing a certain Hessian matrix. Furthermore, the algorithms we obtained are simpler than their SOS counterparts. We are hopeful that the Kikuchi hierarchy will provide a roadmap for systematically deriving simple and optimal algorithms for a large class of problems.

## Acknowledgments

We thank Alex Russell for suggesting the matrix Chernoff bound (Theorem 18). For helpful discussions, we thank Afonso Bandeira, Sam Hopkins, Pravesh Kothari, Florent Krzakala, Tselil Schramm, Jonathan Shi, and Lenka Zdeborová. This project started during the workshop *Spin Glasses and Related Topics* held at the Banff International Research Station (BIRS) in the Fall of 2018. We thank

our hosts at BIRS as well as the workshop organizers: Antonio Auffinger, Wei-Kuo Chen, Dmitry Panchenko, and Lenka Zdeborová.

## References

- [1] Kwangjun Ahn. 2020. A simpler strong refutation of random k-XOR. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.
- [2] Sarah R Allen, Ryan O'Donnell, and David Witmer. 2015. How to refute a random CSP. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. IEEE, 689–708.
- [3] Noga Alon, Michael Krivelevich, and Benny Sudakov. 1998. Finding a large hidden clique in a random graph. *Random Structures & Algorithms* 13, 3-4 (1998), 457–466.
- [4] Omar Alrabiah, Venkatesan Guruswami, Pravesh K Kothari, and Peter Manohar. 2023. A near-cubic lower bound for 3-query locally decodable codes from semirandom CSP refutation. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 1438–1448.
- [5] Anima Anandkumar, Yuan Deng, Rong Ge, and Hossein Mobahi. 2017. Homotopy analysis for tensor PCA. *Conference on Learning Theory*. PMLR, 79–104.
- [6] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. 2014. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research* 15, 1 (2014), 2773–2832.
- [7] Animashree Anandkumar, Rong Ge, and Majid Janzamin. 2017. Analyzing tensor power method dynamics in overcomplete regime. *The Journal of Machine Learning Research* 18, 1 (2017), 752–791.
- [8] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. 2020. Algorithmic thresholds for tensor PCA. *The Annals of Probability* 48, 4 (2020), 2052–2087.
- [9] Richard Arratia and Louis Gordon. 1989. Tutorial on large deviations for the binomial distribution. *Bulletin of Mathematical Biology* 51, 1 (1989), 125–131.
- [10] Antonio Auffinger, Gérard Ben Arous, and Jiří Černý. 2013. Random matrices and complexity of spin glasses. *Communications on Pure and Applied Mathematics* 66, 2 (2013), 165–201.
- [11] Afonso S Bandeira, March T Boedihardjo, and Ramon van Handel. 2023. Matrix concentration inequalities and free probability. *Inventiones Mathematicae* 234, 1 (2023), 419–487.
- [12] Afonso S Bandeira, Giorgio Cipolloni, Dominik Schröder, and Ramon van Handel. 2024. Matrix concentration inequalities and free probability II. two-sided bounds and applications. arXiv:2406.11453. Retrieved from <https://arxiv.org/abs/2406.11453>
- [13] Jess Banks, Christopher Moore, Roman Vershynin, Nicolas Verzelen, and Jiaming Xu. 2018. Information-theoretic bounds and phase transitions in clustering, sparse PCA, and submatrix localization. *IEEE Transactions on Information Theory* 64, 7 (2018), 4872–4894.
- [14] Boaz Barak, Samuel B. Hopkins, Jonathan Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. 2016. A nearly tight sum-of-squares lower bound for the planted clique problem. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 428–437.
- [15] Florent Benaych-Georges and Raj Rao Nadakuditi. 2011. The eigenvalues and eigenvectors of finite, low rank perturbations of large random matrices. *Advances in Mathematics* 227, 1 (2011), 494–521.
- [16] Vijay Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. 2016. Sum-of-squares certificates for maxima of random tensors on the sphere. arXiv:1605.00903. Retrieved from <https://arxiv.org/abs/1605.00903>
- [17] Vijay VSP Bhattiprolu, Mrinalkanti Ghosh, Venkatesan Guruswami, Euiwoong Lee, and Madhur Tulsiani. 2016. Multiplicative approximations for polynomial optimization over the unit sphere. In *Electronic Colloquium on Computational Complexity (ECCC)*, Vol. 23. 1.
- [18] Giulio Biroli, Chiara Cammarota, and Federico Ricci-Tersenghi. 2020. How to iron out rough landscapes and get optimal performances: Averaged gradient descent and its application to tensor PCA. *Journal of Physics A: Mathematical and Theoretical* 53, 17 (2020), 174003.
- [19] Charles Bordenave, Marc Lelarge, and Laurent Massoulié. 2015. Non-backtracking spectrum of random graphs: Community detection and non-regular ramanujan graphs. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. IEEE, 1347–1357.
- [20] Amanda Burcroff. 2017. *Johnson Schemes and Certain Matrices with Integral Eigenvalues*. Technical Report. University of Michigan.
- [21] Wei-Kuo Chen. 2019. Phase transition in the spiked random tensor with rademacher prior. *The Annals of Statistics* 47, 5 (2019), 2734–2756.
- [22] Wei-Kuo Chen, Madeline Handschy, and Gilad Lerman. 2018. Phase transition in random tensors with multiple spikes. arXiv:1809.06790. Retrieved from <https://arxiv.org/abs/1809.06790>

- [23] Amin Coja-Oghlan, Andreas Goerdt, and André Lanka. 2004. Strong refutation heuristics for random k-SAT. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 310–321.
- [24] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. 2011. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E* 84, 6 (2011), 066106.
- [25] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. 2011. Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters* 107, 6 (2011), 065701.
- [26] Yash Deshpande, Emmanuel Abbe, and Andrea Montanari. 2015. Asymptotic mutual information for the two-groups stochastic block model. arXiv:1507.08685. Retrieved from <https://arxiv.org/abs/1507.08685>
- [27] Mohamad Dia, Nicolas Macris, Florent Krzakala, Thibault Lesieur, and Lenka Zdeborová. 2016. Mutual information for symmetric rank-one matrix estimation: A proof of the replica formula. In *Advances in Neural Information Processing Systems*. 424–432.
- [28] Yunzi Ding, Dmitriy Kunisky, Alexander S. Wein, and Afonso S. Bandeira. 2024. Subexponential-time algorithms for sparse PCA. *Foundations of Computational Mathematics* 24, 3 (2024), 865–914.
- [29] David L Donoho, Arian Maleki, and Andrea Montanari. 2009. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences* 106, 45 (2009), 18914–18919.
- [30] Ahmed El Alaoui, Florent Krzakala, and Michael Jordan. 2020. Fundamental limits of detection in the spiked wigner model. *The Annals of Statistics* 48, 2 (2020), 863–885.
- [31] Uriel Feige. 2008. Small linear dependencies for binary vectors of low weight. In *Building Bridges: Between Mathematics and Computer Science*. Springer, 283–307.
- [32] Delphine Féral and Sandrine Péché. 2007. The largest eigenvalue of rank one deformation of large wigner matrices. *Communications in Mathematical Physics* 272, 1 (2007), 185–228.
- [33] Yuval Filmus. 2016. An orthogonal basis for functions over a slice of the boolean hypercube. *The Electronic Journal of Combinatorics* 23, 1 (2016), P1–23.
- [34] Alyson K Fletcher and Sundeep Rangan. 2018. Iterative reconstruction of rank-one matrices in noise. *Information and Inference: A Journal of the IMA* 7, 3 (2018), 531–562.
- [35] Chris Godsil and Sung Y Song. 2010. Association schemes. *Handbook of Combinatorial Designs*, (2010), 325–330.
- [36] Dima Grigoriev. 2001. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science* 259, 1-2 (2001), 613–622.
- [37] Venkatesan Guruswami, Pravesh K Kothari, and Peter Manohar. 2022. Algorithms and certificates for boolean CSP refutation: Smoothed is no harder than random. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. 678–689.
- [38] Matthew B. Hastings. 2020. Classical and quantum algorithms for tensor principal component analysis. *Quantum* 4 (2020), 237. <https://doi.org/10.22331/q-2020-02-27-237>
- [39] Samuel B. Hopkins, Pravesh K Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. 2017. The power of sum-of-squares for detecting hidden structures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 720–731.
- [40] Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. 2016. Fast spectral algorithms from sum-of-squares proofs: Tensor decomposition and planted sparse vectors. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*. ACM, 178–191.
- [41] Samuel B. Hopkins, Jonathan Shi, and David Steurer. 2015. Tensor principal component analysis via sum-of-square proofs. In *Conference on Learning Theory*. 956–1006.
- [42] Jun-Ting Hsieh, Pravesh K Kothari, and Sidhanth Mohanty. 2023. A simple and sharper proof of the hypergraph moore bound. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2324–2344.
- [43] Aukosh Jagannath, Patrick Lopatto, and Leo Miolane. 2020. Statistical thresholds for tensor PCA. *The Annals of Applied Probability* 30, 4 (2020), 1910–1933.
- [44] Mark Jerrum. 1992. Large cliques elude the metropolis process. *Random Structures & Algorithms* 3, 4 (1992), 347–359.
- [45] Iain M. Johnstone and Arthur Yu Lu. 2009. On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association* 104, 486 (2009), 682–693.
- [46] Ryoichi Kikuchi. 1951. A theory of cooperative phenomena. *Phys. Rev.* 81, 6 (1951), 988.
- [47] Ryoichi Kikuchi. 1994. Special issue in honor of R. Kikuchi. *Progr. Theor. Phys. Suppl* 115 (1994), 1–26.
- [48] Pravesh K Kothari and Peter Manohar. 2024. An exponential lower bound for linear 3-query locally correctable codes. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 776–787.
- [49] Pravesh K Kothari and Peter Manohar. 2024. Superpolynomial lower bounds for smooth 3-LCCs and sharp bounds for designs. arXiv:2404.06513. Retrieved from <https://arxiv.org/abs/2404.06513>
- [50] Florent Krzakala, Cristopher Moore, Elchanan Mossel, Joe Neeman, Allan Sly, Lenka Zdeborová, and Pan Zhang. 2013. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences* 110, 52 (2013), 20935–20940.

- [51] Dmitriy Kunisky, Alexander S. Wein, and Afonso S. Bandeira. 2019. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio. *ISAAC Congress (International Society for Analysis, its Applications and Computation)*. Springer, 1–50.
- [52] Jean B. Lasserre. 2001. An explicit exact SDP relaxation for nonlinear 0-1 programs. In *International Conference on Integer Programming and Combinatorial Optimization*. Springer, 293–303.
- [53] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. 2015. MMSE of probabilistic low-rank matrix estimation: Universality with respect to the output channel. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 680–687.
- [54] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. 2015. Phase transitions in sparse PCA. In *2015 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 1635–1639.
- [55] Thibault Lesieur, Léo Miolane, Marc Lelarge, Florent Krzakala, and Lenka Zdeborová. 2017. Statistical and computational phase transitions in spiked tensor estimation. In *2017 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 511–515.
- [56] Andrea Montanari, Daniel Reichman, and Ofer Zeitouni. 2015. On the limitation of spectral methods: From the gaussian hidden clique problem to rank-one perturbations of Gaussian tensors. In *Advances in Neural Information Processing Systems*. 217–225.
- [57] Elchanan Mossel, Joe Neeman, and Allan Sly. 2014. Belief propagation, robust reconstruction and optimal recovery of block models. In *Conference on Learning Theory*. 356–370.
- [58] Roberto Oliveira. 2010. Sums of random hermitian matrices and an inequality by rudelson. *Electronic Communications in Probability* 15 (2010), 203–212.
- [59] Pablo A Parrilo. 2000. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. Ph. D. Dissertation. California Institute of Technology.
- [60] Judea Pearl. 1986. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence* 29, 3 (1986), 241–288.
- [61] Amelia Perry, Alexander S. Wein, and Afonso S. Bandeira. 2020. Statistical limits of spiked tensor models. *Ann. Inst. H. Poincaré Probab. Statist.* 56, 1 (2020), 230–264.
- [62] Amelia Perry, Alexander S. Wein, Afonso S. Bandeira, and Ankur Moitra. 2018. Optimality and sub-optimality of PCA I: Spiked random matrix models. *The Annals of Statistics* 46, 5 (2018), 2416–2451.
- [63] Aaron Potechin and Goutham Rajendran. 2022. Sub-exponential time sum-of-squares lower bounds for principal components analysis. *Advances in Neural Information Processing Systems* 35 (2022), 35724–35740. <https://dl.acm.org/doi/10.5555/3600270.3602859>
- [64] Prasad Raghavendra, Satish Rao, and Tselil Schramm. 2017. Strongly refuting random CSPs below the spectral threshold. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 121–131.
- [65] Emile Richard and Andrea Montanari. 2014. A statistical model for tensor PCA. In *Advances in Neural Information Processing Systems*. 2897–2905.
- [66] Alaa Saade. 2016. Spectral inference methods on sparse graphs: Theory and applications. arXiv:1610.04337. Retrieved from <https://arxiv.org/abs/1610.04337>
- [67] Alaa Saade, Florent Krzakala, and Lenka Zdeborová. 2014. Spectral clustering of graphs with the bethe hessian. In *Advances in Neural Information Processing Systems*. 406–414.
- [68] Grant Schoenebeck. 2008. Linear level lasserre lower bounds for certain k-CSPs. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 593–602.
- [69] Alexander Schrijver. 1979. A comparison of the delsarte and Lovász bounds. *IEEE Trans. Information Theory* 25, 4 (1979), 425–429.
- [70] Naum Z Shor. 1987. Class of global minimum bounds of polynomial functions. *Cybernetics* 23, 6 (1987), 731–734.
- [71] Joel A Tropp. 2012. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics* 12, 4 (2012), 389–434.
- [72] Martin J. Wainwright and Michael I. Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1, 1–2 (2008), 1–305.
- [73] William C Waterhouse. 1990. The absolute-value estimate for symmetric multilinear forms. *Linear Algebra Appl.* 128 (1990), 97–105. <https://www.sciencedirect.com/science/article/pii/002437959090284j>
- [74] Alexander S Wein, Ahmed El Alaoui, and Christopher Moore. 2019. The kikuchi hierarchy and tensor PCA. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 1446–1468.
- [75] Jonathan S Yedidia, William T Freeman, and Yair Weiss. 2003. Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium* 8 (2003), 236–239. <https://dl.acm.org/doi/10.5555/779343.779352>

## Appendices

### A Detection for General Priors

While we have mainly focused on the Rademacher-spiked tensor model, we now show that our algorithm works just as well (at least for detection) for a much larger class of spike priors.

**THEOREM 34.** *Let  $p \geq 2$  be even. Consider the spiked tensor model with a spike prior  $P_x$  that draws the entries of  $x_*$  i.i.d. from some distribution  $\pi$  on  $\mathbb{R}$  (which does not depend on  $n$ ), normalized so that  $\mathbb{E}[\pi^2] = 1$ . There is a constant  $C$  (depending on  $p$  and  $\pi$ ) such that if  $\lambda \geq C\ell^{1/2}d_\ell^{-1/2}\sqrt{\log n}$  then Algorithm 9 achieves strong detection.*

**PROOF.** From (17), we have  $\|Z\|_{\text{op}} = O(\sqrt{\ell d_\ell \log n})$  with high probability, and so it remains to give a lower bound on  $\|X\|_{\text{op}}$ . Letting  $u_S = \prod_{i \in S} \text{sgn}((x_*)_i)$  for  $|S| = \ell$ ,

$$\|X\|_{\text{op}} \geq \frac{u^\top Xu}{\|u\|^2} = \binom{n}{\ell}^{-1} u^\top Xu$$

where

$$\begin{aligned} u^\top Xu &= \sum_{|S \triangle T|=p} u_S X_{S,T} u_T \\ &= \sum_{|S \triangle T|=p} x_*^{S \triangle T} \prod_{i \in S} \text{sgn}((x_*)_i) \prod_{i \in T} \text{sgn}((x_*)_i) \\ &= \sum_{|S \triangle T|=p} |x_*^{S \triangle T}|. \end{aligned}$$

We have

$$\mathbb{E}[u^\top Xu] = \binom{n}{\ell} d_\ell (\mathbb{E}|\pi|)^p = C(\pi, p) \binom{n}{\ell} d_\ell, \quad (31)$$

and

$$\begin{aligned} \text{Var}[u^\top Xu] &= \text{Var} \left[ \sum_{|S \triangle T|=p} |x_*^{S \triangle T}| \right] \\ &= \sum_{|S \triangle T|=p} \sum_{|S' \triangle T'|=p} \text{Cov}(|x_*^{S \triangle T}|, |x_*^{S' \triangle T'}|). \end{aligned} \quad (32)$$

We have

$$\begin{aligned} \text{Cov}(|x_*^{S \triangle T}|, |x_*^{S' \triangle T'}|) &\leq \sqrt{\text{Var}(|x_*^{S \triangle T}|) \text{Var}(|x_*^{S' \triangle T'}|)} \\ &= \text{Var}(|x_*^{S \triangle T}|) \leq \mathbb{E}[|x_*^{S \triangle T}|^2] = (\mathbb{E}[\pi^2])^p = 1. \end{aligned}$$

Also,  $\text{Cov}(|x_*^{S \triangle T}|, |x_*^{S' \triangle T'}|) = 0$  unless  $S \triangle T$  and  $S' \triangle T'$  have nonempty intersection. Using Lemma 35 (below), the fraction of terms in (32) that are nonzero is at most  $p^2/n$  and so

$$\text{Var}[u^\top Xu] \leq \left[ \binom{n}{\ell} d_\ell \right]^2 \frac{p^2}{n}. \quad (33)$$

By Chebyshev's inequality, it follows from (31) and (33) that  $u^\top Xu \geq \frac{1}{2}C(\pi, p)\binom{n}{\ell}d_\ell$  with probability at least  $1 - \frac{4p^2}{C(\pi, p)^2 n}$ . This implies  $\|X\|_{\text{op}} \geq \frac{1}{2}C(\pi, p)d_\ell$  with the same probability, and so we have strong detection provided  $\lambda \geq c_0\ell^{1/2}d_\ell^{-1/2}\sqrt{\log n}$  for a particular constant  $c_0 = c_0(\pi, p)$ .  $\square$

Above, we made use of the following lemma.

**LEMMA 35.** *Fix  $A \subseteq [n]$  with  $|A| = a$ . Let  $B$  be chosen uniformly at random from all subsets of  $[n]$  of size  $b$ . Then  $\mathbb{P}(A \cap B \neq \emptyset) \leq \frac{ab}{n}$ .*

**PROOF.** Each element of  $A$  will lie in  $B$  with probability  $b/n$ , so the result follows by a union bound over the elements of  $A$ .  $\square$

## B The Odd- $p$ Case

When  $p$  is odd, the Kikuchi Hessian still gives rise to a spectral algorithm. While we conjecture that this algorithm is optimal, we unfortunately only know how to prove suboptimal results for it. (However, we can prove optimal results for a related algorithm; see Section 6.2.) We now state the algorithm and its conjectured performance.

Let  $p$  be odd and fix an integer  $\ell \in [|p/2], n - |p/2|]$ . Consider the symmetric difference matrix  $M \in \mathbb{R}^{\binom{n}{\ell} \times \binom{n}{\ell+1}}$  with entries

$$M_{S,T} = \begin{cases} Y_{S \Delta T} & \text{if } |S \Delta T| = p, \\ 0 & \text{otherwise,} \end{cases}$$

where  $S, T \subseteq [n]$  with  $|S| = \ell$  and  $|T| = \ell + 1$ .

**Algorithm 36 (Recovery for odd  $p$ ).** Let  $u$  be a (unit-norm) top left-singular vector of  $M$  and let  $v = M^\top u$  be the corresponding top right-singular vector. Output  $\hat{x} = \hat{x}(Y) \in \mathbb{R}^n$ , defined by

$$\hat{x}_i = \sum_{S \in \binom{[n]}{\ell}, T \in \binom{[n]}{\ell+1}} u_S v_T \mathbf{1}_{S \Delta T = \{i\}}, \quad i \in [n].$$

Notice that the rounding step consisting in extracting an  $n$ -dimensional vector  $\hat{x}$  from the singular vectors of  $M$  is slightly simpler than the even- $p$  case, in that it does not require forming a voting matrix. We conjecture that, like the even case, this algorithm matches the performance of SOS.

**CONJECTURE 37.** *Consider the Rademacher-spiked tensor model with  $p \geq 3$  odd. If*

$$\lambda \gg \ell^{-(p-2)/4} n^{-p/4}$$

*then (i) there is a threshold  $\tau = \tau(n, p, \ell, \lambda)$  such that strong detection can be achieved by thresholding the top singular value of  $M$  at  $\tau$ , and (ii) Algorithm 36 achieves strong recovery.*

Similarly to the proof of Theorem 12, the matrix Chernoff bound (Theorem 18) can be used to show that strong recovery is achievable when  $\lambda \gg \ell^{-(p-1)/4} n^{-(p-1)/4}$ , which is weaker than SOS when  $\ell \ll n$ . We now explain the difficulties involved in improving this. We can decompose  $M$  into a signal part and a noise part:  $M = \lambda X + Z$ . In the regime of interest,  $\ell^{-(p-2)/4} n^{-p/4} \ll \lambda \ll \ell^{-(p-1)/4} n^{-(p-1)/4}$ , the signal term is smaller in operator norm than the noise term, i.e.,  $\lambda \|X\|_{\text{op}} \ll \|Z\|_{\text{op}}$ . While at first sight this would seem to suggest that detection and recovery are hopeless, we actually expect that  $\lambda X$  still affects the top singular value and singular vectors of  $M$ . This phenomenon is already present in the analysis of tensor unfolding (the case  $p = 3, \ell = 1$ ) [41], but it seems that new ideas are required to extend the analysis beyond this case.

## C Proof of Boosting

In this section, we prove Proposition 6 which we restate here:

**PROPOSITION 38 (PROPOSITION 6 RESTATED).** *Let  $Y \sim \mathbb{P}_\lambda$  with any spike prior  $P_x$  supported on  $S^{n-1}(\sqrt{n})$ . Suppose we have an initial guess  $u \in \mathbb{R}^n$  satisfying  $\text{corr}(u, x_*) \geq \tau$ . Obtain  $\hat{x}$  from  $u$  via a*

single iteration of the tensor power method:  $\hat{x} = Y\{u\}$ . There exists a constant  $c = c(p) > 0$  such that with high probability,

$$\text{corr}(\hat{x}, x_*) \geq 1 - c\lambda^{-1}\tau^{1-p}n^{(1-p)/2}.$$

In particular, if  $\tau > 0$  is any constant and  $\lambda = \omega(n^{(1-p)/2})$  then  $\text{corr}(\hat{x}, x) = 1 - o(1)$ .

**Definition 39.** For a tensor  $G \in (\mathbb{R}^n)^{\otimes p}$ , the *injective tensor norm* is

$$\|G\|_{\text{inj}} := \max_{\|u^{(1)}\| = \dots = \|u^{(p)}\| = 1} \sum_{i_1, \dots, i_p} G_{i_1, \dots, i_p} u_{i_1}^{(1)} \cdots u_{i_p}^{(p)},$$

where  $u^{(j)} \in \mathbb{R}^n$ . For a symmetric tensor  $G$ , it is known [73] that equivalently,

$$\|G\|_{\text{inj}} = \max_{\|u\|=1} \left| \sum_{i_1, \dots, i_p} G_{i_1, \dots, i_p} u_{i_1} \cdots u_{i_p} \right|.$$

**PROOF OF PROPOSITION 6.** Write  $\hat{x} = \lambda\langle u, x_* \rangle^{p-1}x_* + \Delta$  where  $\|\Delta\| \leq \|G\|_{\text{inj}}\|u\|^{p-1}$ . We have

$$|\langle \hat{x}, x_* \rangle| \geq \lambda|\langle u, x_* \rangle|^{p-1}\|x_*\|^2 - \|\Delta\|\|x_*\|,$$

and

$$\|\hat{x}\| \leq \lambda|\langle u, x_* \rangle|^{p-1}\|x_*\| + \|\Delta\|,$$

and so

$$\begin{aligned} \text{corr}(\hat{x}, x_*) &= \frac{|\langle \hat{x}, x_* \rangle|}{\|\hat{x}\|\|x_*\|} \\ &\geq \frac{\lambda|\langle u, x_* \rangle|^{p-1}\|x_*\| - \|\Delta\|}{\lambda|\langle u, x_* \rangle|^{p-1}\|x_*\| + \|\Delta\|} \\ &= 1 - \frac{2\|\Delta\|}{\lambda|\langle u, x_* \rangle|^{p-1}\|x_*\| + \|\Delta\|} \\ &\geq 1 - \frac{2\|\Delta\|}{\lambda|\langle u, x_* \rangle|^{p-1}\|x_*\|} \\ &\geq 1 - \frac{2\|G\|_{\text{inj}}\|u\|^{p-1}}{\lambda|\langle u, x_* \rangle|^{p-1}\|x_*\|} \\ &\geq 1 - \frac{2\|G\|_{\text{inj}}}{\lambda\tau^{p-1}\|x_*\|^p}. \end{aligned}$$

Our prior  $P_x$  is supported on the sphere of radius  $\sqrt{n}$ , so  $\|x_*\| = \sqrt{n}$ . We need to control the injective norm of the tensor  $G$ . To this end we use Theorem 2.12 in Reference [10] (see also Lemma 2.1 of [65]): there exists a constant  $c(p) > 0$  (called  $E_0(p)$  in [10]) such that for all  $\epsilon > 0$ ,

$$\mathbb{P} \left( \sqrt{\frac{p}{n}}\|G\|_{\text{inj}} \geq c(p) + \epsilon \right) \longrightarrow 0 \quad \text{as } n \rightarrow \infty.$$

Letting  $\epsilon = c(p)$  we obtain

$$\text{corr}(\hat{x}, x_*) \geq 1 - 4 \frac{c(p)}{\sqrt{p}} \frac{n^{(1-p)/2}}{\lambda\tau^{p-1}},$$

with probability tending to 1 as  $n \rightarrow \infty$ . □

## D Computing the Kikuchi Hessian

In Section 4, we defined the Kikuchi free energy and explained the high level idea of how the symmetric difference matrices are derived from the Kikuchi Hessian. We now carry out the Kikuchi Hessian computation in full detail. This is a heuristic (non-rigorous) computation, but we believe these methods are important as we hope they will be useful for systematically obtaining optimal spectral methods for a wide variety of problems.

### D.1 Derivatives of Kikuchi Free Energy

Following [67], we parametrize the beliefs in terms of the moments  $m_S = \mathbb{E}[x^S]$ . Specifically,

$$b_S(x_S) = \frac{1}{2^{|S|}} \left( 1 + \sum_{\emptyset \subset T \subseteq S} m_T x^T \right). \quad (34)$$

We imagine  $m_T$  are close enough to zero so that  $b_S$  is a positive measure. One can check that these beliefs indeed have the prescribed moments: for  $T \subseteq S$ ,

$$\sum_{x_S} b_S(x_S) x^T = m_T.$$

Thus we can think of the Kikuchi free energy  $\mathcal{K}$  as a function of the moments  $\{m_S\}_{0 < |S| \leq r}$ . This parametrization forces the beliefs to be consistent, i.e., if  $T \subseteq S$  then the marginal distribution  $b_S|T$  is equal to  $b_T$ .

We now compute first and second derivatives of  $\mathcal{K} = \mathcal{E} - \mathcal{S}$  with respect to the moments  $m_S$ . First, the energy term:

$$\mathcal{E} = -\lambda \sum_{|S|=p} Y_S m_S$$

$$\frac{\partial \mathcal{E}}{\partial m_S} = \begin{cases} -\lambda Y_S & \text{if } |S| = p \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial^2 \mathcal{E}}{\partial m_S \partial m_{S'}} = 0.$$

Now the entropy term:

$$\frac{\partial \mathcal{S}_S}{\partial b_S(x_S)} = -\log b_S(x_S) - 1.$$

From (34), for  $\emptyset \subset T \subseteq S$ ,

$$\frac{\partial b_S(x_S)}{\partial m_T} = \frac{x^T}{2^{|S|}}$$

and so

$$\begin{aligned} \frac{\partial \mathcal{S}_S}{\partial m_T} &= \sum_{x_S} \frac{\partial \mathcal{S}_S}{\partial b_S(x_S)} \cdot \frac{\partial b_S(x_S)}{\partial m_T} \\ &= -2^{-|S|} \sum_{x_S} x^T [\log b_S(x_S) + 1] \\ &= -2^{-|S|} \sum_{x_S} x^T \log b_S(x_S). \end{aligned} \quad (35)$$

For  $\emptyset \subset T \subseteq S$ ,  $\emptyset \subset T' \subseteq S$ ,

$$\begin{aligned}\frac{\partial^2 \mathcal{S}_S}{\partial m_T \partial m_{T'}} &= -2^{-|S|} \sum_{x_S} x^T b_S(x_S)^{-1} \cdot \frac{\partial b_S(x_S)}{\partial m_{T'}} \\ &= -2^{-|S|} \sum_{x_S} x^T x^{T'} b_S(x_S)^{-1} \\ &= -2^{-|S|} \sum_{x_S} x^{T \triangle T'} b_S(x_S)^{-1}.\end{aligned}$$

Finally, if  $T \not\subseteq S$  then  $\frac{\partial \mathcal{S}_S}{\partial m_T} = 0$ .

## D.2 The Case $r = p$

We first consider the simplest case, where  $r$  is as small as possible:  $r = p$ . (We need to require  $r \geq p$  in order to express the energy term in terms of the beliefs.)

**D.2.1 Trivial Stationary Point.** There is a “trivial stationary point” of the Kikuchi free energy where the beliefs only depend on local information. Specifically, if  $|S| < p$  then  $b_S$  is the uniform distribution over  $\{\pm 1\}^{|S|}$ , and if  $|S| = p$  then

$$b_S(x_S) \propto \exp(\lambda Y_S x^S)$$

i.e.,

$$b_S(x_S) = \frac{1}{Z_S} \exp(\lambda Y_S x^S) \quad (36)$$

where

$$Z_S = \sum_{x_S} \exp(\lambda Y_S x^S).$$

Note that these beliefs are consistent (if  $T \subseteq S$  with  $|S| \leq p$  then  $b_S|T = b_T$ ) and so there is a corresponding set of moments  $\{m_S\}_{|S| \leq p}$ .

We now check that this is indeed a stationary point of the Kikuchi free energy. Using (35) and (36) we have for  $\emptyset \subset T \subseteq S$  and  $|S| \leq p$ ,

$$\begin{aligned}\frac{\partial \mathcal{S}_S}{\partial m_T} &= -2^{-|S|} \sum_{x_S} x^T \log b_S(x_S) \\ &= -2^{-|S|} \sum_{x_S} x^T [-\log Z_S + \lambda 1_{|S|=p} Y_S x^S] \\ &= \begin{cases} -\lambda Y_T & \text{if } |T| = p \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

Thus if  $|T| < p$  we have  $\frac{\partial \mathcal{K}}{\partial m_T} = 0$ , and if  $|T| = p$  we have

$$\begin{aligned}\frac{\partial \mathcal{K}}{\partial m_T} &= \frac{\partial}{\partial m_T} \left[ \mathcal{E} - \sum_{0 < |S| \leq p} c_S \mathcal{S}_S \right] \\ &= -\lambda Y_T + c_T \lambda Y_T \\ &= 0\end{aligned}$$

This confirms that we indeed have a stationary point.

**D.2.2 Hessian.** We now compute the Kikuchi Hessian, the matrix indexed by subsets  $0 < |T| \leq p$  with entries  $H_{T,T'} = \frac{\partial^2 \mathcal{K}}{\partial m_T \partial m_{T'}}$ , evaluated at the trivial stationary point. Similarly to the Bethe Hessian [67], we expect the bottom eigenvector of the Kikuchi Hessian to be a good estimate for the (moments of) the true signal. This is because this bottom eigenvector indicates the best local direction for improving the Kikuchi free energy, starting from the trivial stationary point. If all eigenvalues of  $H$  are positive then the trivial stationary point is a local minimum and so an algorithm acting locally on the beliefs should not be able to escape from it, and should not learn anything about the signal. On the other hand, a negative eigenvalue (or even an eigenvalue close to zero) indicates a (potential) direction for improvement.

*Remark 40.* When  $p$  is odd, we cannot hope for a substantially negative eigenvalue because  $x_*$  and  $-x_*$  are *not* equally-good solutions and so the Kikuchi free energy should be locally cubic instead of quadratic. Still, we believe that the bottom eigenvector of the Kikuchi Hessian (which will have eigenvalue close to zero) yields a good algorithm. For instance, we will see in the next section that this method yields a close variant of tensor unfolding when  $r = p = 3$ .

Recall that for  $\emptyset \subset T \subseteq S$ , and  $\emptyset \subset T' \subseteq S$ ,

$$\frac{\partial^2 \mathcal{S}_S}{\partial m_T \partial m_{T'}} = -2^{-2|S|} \sum_{x_S} x^{T \triangle T'} b_S(x_S)^{-1}.$$

If  $|S| < p$  then  $b_S$  is uniform on  $\{\pm 1\}^{|S|}$  (at the trivial stationary point) and so  $\frac{\partial^2 \mathcal{S}_S}{\partial m_T \partial m_{T'}} = -\mathbb{1}_{T=T'}$ . If  $|S| = p$  then  $b_S(x_S) = \frac{1}{Z_S} \exp(\lambda Y_S x^S)$  where  $Z_S = \sum_{x_S} \exp(\lambda Y_S x^S) = 2^{|S|} \cosh(\lambda Y_S)$ , and so

$$\begin{aligned} \frac{\partial^2 \mathcal{S}_S}{\partial m_T \partial m_{T'}} &= -2^{-2|S|} \sum_{x_S} x^{T \triangle T'} b_S(x_S)^{-1} \\ &= -2^{-2|S|} \sum_{x_S} x^{T \triangle T'} Z_S \exp(-\lambda Y_S x^S) \\ &= -2^{-|S|} \sum_{x_S} x^{T \triangle T'} \cosh(\lambda Y_S) \left( 1 - \lambda Y_S x^S + \frac{1}{2!} \lambda^2 Y_S^2 - \frac{1}{3!} \lambda^3 Y_S^3 x^S + \dots \right) \\ &= -2^{-|S|} \sum_{x_S} x^{T \triangle T'} \cosh(\lambda Y_S) (\cosh(\lambda Y_S) - \sinh(\lambda Y_S) x^S) \end{aligned}$$

since  $\cosh(x) = 1 + \frac{1}{2!}x^2 + \frac{1}{4!}x^4 + \dots$  and  $\sinh(x) = x + \frac{1}{3!}x^3 + \frac{1}{5!}x^5 + \dots$

$$\begin{aligned} &= \begin{cases} -\cosh^2(\lambda Y_S) & \text{if } T = T', T \subseteq S \\ \cosh(\lambda Y_S) \sinh(\lambda Y_S) & \text{if } T \triangle T' = S, T \cup T' \subseteq S \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} -\cosh^2(\lambda Y_S) & \text{if } T = T', T \subseteq S \\ \cosh(\lambda Y_S) \sinh(\lambda Y_S) & \text{if } T \sqcup T' = S \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where  $\sqcup$  denotes disjoint union. (Note that we have replaced  $\triangle$  with  $\sqcup$  due to the restriction  $T, T' \subseteq S$ ) We can now compute the Hessian:

$$\begin{aligned}
H_{T,T'} &= \frac{\partial^2 \mathcal{K}}{\partial m_T \partial m_{T'}} \\
&= - \sum_{\substack{S \supseteq T \sqcup T' \\ |S| \leq p}} c_S \frac{\partial^2 \mathcal{S}_S}{\partial m_T \partial m_{T'}} \\
&= \begin{cases} \sum_{\substack{S \supseteq T \\ |S| < p}} c_S + \sum_{\substack{S \supseteq T \\ |S|=p}} \cosh^2(\lambda Y_S) & \text{if } T = T' \\ -\cosh(\lambda Y_{T \sqcup T'}) \sinh(\lambda Y_{T \sqcup T'}) & \text{if } |T \sqcup T'| = p \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} 1 + \sum_{\substack{S \supseteq T \\ |S|=p}} [\cosh^2(\lambda Y_S) - 1] & \text{if } T = T' \\ -\cosh(\lambda Y_{T \sqcup T'}) \sinh(\lambda Y_{T \sqcup T'}) & \text{if } |T \sqcup T'| = p \\ 0 & \text{otherwise} \end{cases}
\end{aligned} \tag{37}$$

where we used (12) in the last step. Suppose  $\lambda \ll 1$  (since otherwise tensor PCA is very easy). If  $T = T'$  then, using the cosh Taylor series, we have the leading-order approximation

$$\begin{aligned}
H_{T,T} &\approx 1 + \sum_{\substack{S \supseteq T \\ |S|=p}} \lambda^2 Y_S^2 \\
&\approx 1 + \binom{n - |T|}{p - |T|} \lambda^2 \mathbb{E}[Y_S^2] \\
&\approx 1 + \frac{n^{p-|T|}}{(p - |T|)!} \lambda^2.
\end{aligned}$$

Using the Taylor series for  $\cosh \cdot \sinh$ , this means  $H \approx \tilde{H}$  where

$$\tilde{H}_{T,T'} = \begin{cases} 1 \vee \frac{n^{p-|T|}}{(p - |T|)!} \lambda^2 & \text{if } T = T' \\ -\lambda Y_{T \sqcup T'} & \text{if } |T \sqcup T'| = p \\ 0 & \text{otherwise.} \end{cases}$$

*D.2.3 The Case  $r = p = 3$ .* We now restrict to the case  $r = p = 3$  and show that the Kikuchi Hessian recovers (a close variant of) the tensor unfolding method. Recall that in this case the computational threshold is  $\lambda \sim n^{-3/4}$  and so we can assume  $\lambda \ll n^{-1/2}$  (or else the problem is easy). We have

$$\tilde{H}_{T,T'} = \begin{cases} \frac{1}{2} n^2 \lambda^2 & \text{if } T = T' \text{ with } |T| = 1 \\ 1 & \text{if } T = T' \text{ with } |T| \in \{2, 3\} \\ -\lambda Y_{T \sqcup T'} & \text{if } |T \sqcup T'| = 3 \\ 0 & \text{otherwise.} \end{cases}$$

This means we can write

$$\tilde{H} = \begin{pmatrix} \alpha I & -\lambda M & 0 \\ -\lambda M^\top & I & 0 \\ 0 & 0 & I \end{pmatrix}$$

where  $\alpha = \frac{1}{2} n^2 \lambda^2$  and  $M$  is the  $n \times \binom{n}{2}$  flattening of  $Y$ , i.e.,  $M_{i,\{j,k\}} = Y_{ijk} \mathbf{1}_{\{i,j,k \text{ distinct}\}}$ .

Since we are looking for the minimum eigenvalue of  $\tilde{H}$ , we can restrict ourselves to the submatrix  $\tilde{H}^{\leq 2}$  indexed by sets of size 1 and 2. We have

$$\tilde{H}^{\leq 2} = \begin{pmatrix} \alpha I & -\lambda M \\ -\lambda M^\top & I \end{pmatrix}.$$

An eigenvector  $[u \ v]^\top$  of  $\tilde{H}^{\leq 2}$  with eigenvalue  $\beta$  satisfies

$$\alpha u - \lambda M v = \beta u \quad \text{and} \quad -\lambda M^\top u + v = \beta v$$

which implies  $(1 - \beta)v = \lambda M^\top u$  and so  $\lambda^2 M M^\top u = (\alpha - \beta)(1 - \beta)u$ . This means either  $u$  is an eigenvector of  $\lambda^2 M M^\top$  with eigenvalue  $(\alpha - \beta)(1 - \beta)$ , or  $u = 0$  and  $\beta \in \{1, \alpha\}$ . Conversely, if  $u$  is an eigenvector of  $\lambda^2 M M^\top$  with eigenvalue  $(\alpha - \beta)(1 - \beta) \neq 0$ , then  $[u \ v]^\top$  with  $v = (1 - \beta)^{-1} \lambda M^\top u$  is an eigenvector of  $\tilde{H}^{\leq 2}$  with eigenvalue  $\beta$ . Letting  $\mu_1 > \dots > \mu_n > 0$  be the eigenvalues of  $\lambda^2 M M^\top$ ,  $\tilde{H}^{\leq 2}$  has  $2n$  eigenvalues of the form

$$\frac{\alpha + 1 \pm \sqrt{(\alpha - 1)^2 + 4\mu_i}}{2}$$

and the remaining eigenvalues are  $\alpha$  or 1. Thus, the  $u$ -part of the bottom eigenvector of  $\tilde{H}^{\leq 2}$  is precisely the leading eigenvector of  $M M^\top$ . This is a close variant of the tensor unfolding spectral method (see Section 2.4), and we expect that its performance is essentially identical.

### D.3 The General Case: $r \geq p$

One difficulty when  $r > p$  is that there is no longer a trivial stationary point that we can write down in closed form. There is, however, a natural guess for “uninformative” beliefs that only depend on the local information: for  $0 < |S| \leq r$ ,

$$b_S(x_S) = \frac{1}{Z_S} \exp \left( \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right)$$

for the appropriate normalizing factor  $Z_S$ . Unfortunately, these beliefs are not quite consistent, so we need separate moments for each set  $S$ :

$$m_T^{(S)} = \mathbb{E}_{x_S \sim b_S} [x^T].$$

Provided  $\lambda \ll 1$ , we can check that  $m_T^{(S)} \approx m_T^{(S')}$  to first order, and so the above beliefs are at least approximately consistent:

$$Z_S = \sum_{x_S} \exp \left( \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \approx \sum_{x_S} \left( 1 + \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) = 2^{|S|}$$

and so

$$\begin{aligned}
m_T^{(S)} &= \sum_{x_S} b_S(x_S) x^T \\
&= \frac{1}{Z_S} \sum_{x_S} x^T \exp \left( \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \\
&\approx \frac{1}{Z_S} \sum_{x_S} x^T \left( 1 + \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \\
&= \begin{cases} \lambda Y_T & \text{if } |T| = p \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

which does not depend on  $S$ . Thus, we will ignore the slight inconsistencies and carry on with the derivation. As above, the important calculation is, for  $T, T' \subseteq S$ ,

$$\begin{aligned}
\frac{\partial^2 \mathcal{S}_S}{\partial m_T^{(S)} \partial m_{T'}^{(S)}} &= -2^{-2|S|} \sum_{x_S} x^{T \Delta T'} b_S(x_S)^{-1} \\
&= -2^{-2|S|} \sum_{x_S} x^{T \Delta T'} Z_S \exp \left( -\lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \\
&\approx -2^{-|S|} \sum_{x_S} x^{T \Delta T'} \exp \left( -\lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \\
&\approx -2^{-|S|} \sum_{x_S} x^{T \Delta T'} \left( 1 - \lambda \sum_{\substack{U \subseteq S \\ |U|=p}} Y_U x^U \right) \\
&= \begin{cases} -1 & \text{if } T = T' \\ \lambda Y_{T \Delta T'} & \text{if } |T \Delta T'| = p \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Analogous to (37), we compute the Kikuchi Hessian

$$H_{T, T'} := - \sum_{\substack{S \supseteq T \cup T' \\ |S| \leq r}} c_S \frac{\partial^2 \mathcal{S}_S}{\partial m_T^{(S)} \partial m_{T'}^{(S)}}.$$

If we fix  $\ell_1, \ell_2$ , the submatrix  $H^{(\ell_1, \ell_2)} = (H_{T, T'})_{|T|=\ell_1, |T'|=\ell_2}$  takes the form

$$H^{(\ell_1, \ell_2)} \approx a(\ell_1, \ell_2) \mathbf{1}_{\ell_1=\ell_2} I - b(\ell_1, \ell_2) M^{(\ell_1, \ell_2)}$$

for certain scalars  $a(\ell_1, \ell_2)$  and  $b(\ell_1, \ell_2)$ , where  $I$  is the identity matrix and  $M^{(\ell_1, \ell_2)} \in \mathbb{R}^{\binom{[n]}{\ell_1} \times \binom{[n]}{\ell_2}}$  is the symmetric difference matrix

$$M_{T, T'}^{(\ell_1, \ell_2)} = \begin{cases} Y_{T \Delta T'} & \text{if } |T \Delta T'| = p \\ 0 & \text{otherwise.} \end{cases}$$

Instead of working with the entire Kikuchi Hessian, we choose to work instead with  $M^{(\ell,\ell)}$ , which (when  $p$  is even) appears as a diagonal block of the Kikuchi Hessian whenever  $r \geq \ell + p/2$  (since there must exist  $|T| = |T'| = \ell$  with  $|T \cup T'| \leq r$  and  $|T \triangle T'| = p$ ). Our theoretical results (see Section 3) show that indeed  $M^{(\ell,\ell)}$  yields algorithms matching the (conjectured optimal) performance of sum-of-squares. When  $p$  is odd,  $M^{(\ell,\ell)} = 0$  and so we propose to instead focus on  $M^{(\ell,\ell+1)}$ ; see Appendix B.

Received 23 August 2024; revised 17 May 2025; accepted 2 August 2025

# Optimal Multi-Distribution Learning

**ZIHAN ZHANG**, Department of Electrical and Computer Engineering, Princeton University, Princeton, United States

**WENHAO ZHAN**, Department of Electrical and Computer Engineering, Princeton University, Princeton, United States

**YUXIN CHEN**, Department of Statistics and Data Science, University of Pennsylvania, Philadelphia, United States

**SIMON S. DU**, Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, United States

**JASON LEE**, Department of Electrical and Computer Engineering, Princeton University, Princeton, United States

---

Multi-distribution learning (MDL), which seeks to learn a shared model that minimizes the worst-case risk across  $k$  distinct data distributions, has emerged as a unified framework in response to the evolving demand for robustness, fairness, multi-group collaboration, and so on. Achieving data-efficient MDL necessitates adaptive sampling, also called on-demand sampling, throughout the learning process. However, there exist substantial gaps between the state-of-the-art upper and lower bounds on the optimal sample complexity. Focusing on a hypothesis class of Vapnik–Chervonenkis (VC) dimension  $d$ , we propose a novel algorithm that yields an  $\epsilon$ -optimal randomized hypothesis with a sample complexity on the order of  $\frac{d+k}{\epsilon^2}$  (modulo some logarithmic factor), matching the best-known lower bound. Our algorithmic ideas and theory are further extended to accommodate Rademacher classes. The proposed algorithms are oracle-efficient, which access the hypothesis class solely through an empirical risk minimization oracle. Additionally, we establish the necessity of improper learning, revealing a large sample size barrier when only deterministic, proper hypotheses are permitted. These findings resolve three open problems presented in COLT 2023 (i.e., Awasthi et al. [4, Problems 1, 3, and 4]).

CCS Concepts: • Theory of computation → Online learning theory; Sample complexity and generalization bounds;

Additional Key Words and Phrases: Multi-distribution Learning; On-demand Sampling; Game Dynamics; VC Classes; Rademacher Classes; Oracle Efficiency

---

YC is supported in part by the Alfred P. Sloan Research Fellowship, the NSF grants CCF-1907661, DMS-2014279, IIS-2218713 and IIS-2218773. JDL acknowledges support of the ARO under MURI Award W911NF-11-1-0304, the Sloan Research Fellowship, NSF CCF 2002272, NSF IIS 2107304, NSF CIF 2212262, ONR Young Investigator Award, and NSF CAREER Award 2144994.

Authors' Contact Information: Zihan Zhang, Department of Electrical and Computer Engineering, Princeton University, Princeton, New Jersey, United States; e-mail: zsubfunc@outlook.com; Wenhao Zhan, Department of Electrical and Computer Engineering, Princeton University, Princeton, New Jersey, United States; e-mail: wenhao.zhan@princeton.edu; Yuxin Chen, Department of Statistics and Data Science, University of Pennsylvania, Philadelphia, Pennsylvania, United States; e-mail: yuxinc@wharton.upenn.edu; Simon S. Du, Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, Washington, United States; e-mail: ssdu@cs.washington.edu; Jason Lee, Department of Electrical and Computer Engineering, Princeton University, Princeton, New Jersey, United States; e-mail: jasonlee88@gmail.com.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART36

<https://doi.org/10.1145/3760256>

**ACM Reference Format:**

Zihan Zhang, Wenhao Zhan, Yuxin Chen, Simon S. Du, and Jason Lee. 2025. Optimal Multi-Distribution Learning. *J. ACM* 72, 5, Article 36 (October 2025), 71 pages. <https://doi.org/10.1145/3760256>

## 1 Introduction

Driven by the growing need of robustness, fairness, and multi-group collaboration in machine learning practice, the **multi-distribution learning (MDL)** framework has emerged as a unified solution in response to these evolving demands [4, 6, 20, 30]. Setting the stage, imagine that we are interested in a collection of  $k$  unknown data distributions  $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^k$  supported on  $\mathcal{X} \times \mathcal{Y}$ , where  $\mathcal{X}$  (respectively  $\mathcal{Y}$ ) stands for the instance (respectively label) space. Given a hypothesis class  $\mathcal{H}$  and a prescribed loss function<sup>1</sup>  $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \rightarrow [-1, 1]$ , we are asked to identify a (possibly randomized) hypothesis  $\hat{h}$  achieving near-optimal *worst-case* loss across these data distributions, namely,<sup>2</sup>

$$\max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, \hat{h}} [\ell(\hat{h}, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon, \quad (1)$$

with  $\varepsilon \in (0, 1]$  a target accuracy level. In light of the unknown nature of these data distributions, the learning process is often coupled with data collection, allowing the learner to sample from  $\{\mathcal{D}_i\}_{i=1}^k$ . The performance of a learning algorithm is then gauged by its sample complexity — the number of samples required to fulfil Equation (1). Our objective is to design a learning paradigm that achieves the optimal sample complexity.

The MDL framework described above, which can be viewed as an extension of agnostic learning [8, 40] tailored to multiple data distributions, has found a wealth of applications across multiple domains. Here, we highlight a few representative examples, and refer the interested reader to Haghtalab et al. [20] and the references therein for more extensive discussions.

- *Collaborative and agnostic federated learning.* In the realm of collaborative and agnostic federated learning [5–7, 11, 12, 14, 30, 31], a group of  $k$  agents, each having access to distinct data sources as characterized by different data distributions  $\{\mathcal{D}_i\}_{i=1}^k$ , aim at learning a shared prediction model that ideally would achieve low risk for each of their respective data sources. A sample-efficient MDL paradigm would help unleash the potential of collaboration and information sharing in jointly learning a complicated task.
- *Min-max fairness in learning.* The MDL framework is well-suited to scenarios requiring fairness across multiple groups [14, 16, 33]. For instance, in situations where multiple subpopulations with distinct data distributions exist, a prevailing objective is to ensure that the learned model does not adversely impact any of these subpopulations. One criterion designed to meet this objective, known as “min-max fairness” in the literature [1, 30], plays a pivotal role in mitigating the worst disadvantage experienced by any particular subpopulation.
- *Distributionally robust optimization/learning.* Another context where MDL naturally finds applications is group **distributionally robust optimization and learning (DRO/DRL)**. Group DRO and DRL aim at developing algorithms that offer robust performance guarantees across a finite number of possible distributional models [12, 22, 25, 35, 36, 43–45], and have garnered substantial attention recently due to the pervasive need for robustness in modern

<sup>1</sup>For example, for each hypothesis  $h \in \mathcal{H}$  and each datapoint  $(x, y) \in \mathcal{X} \times \mathcal{Y}$ , we employ  $\ell(h, (x, y))$  to measure the risk of using hypothesis  $h$  to predict  $y$  based on  $x$ .

<sup>2</sup>Here, the expectation on the left-hand side of Equation (1) is taken over the randomness of both the datapoints  $(x, y)$  and the (randomized) hypothesis  $\hat{h}$ .

decision-making [3, 10, 20, 26]. When applying MDL to the context of group DRO/DRL, the resultant sample complexity reflects the price that needs to be paid for learning a robust solution.

The MDL framework is also closely related to other topics like multi-source domain adaptation, maximum aggregation, to name just a few [9, 19, 28, 47].

In contrast to single-distribution learning, achieving data-efficient MDL necessitates adaptive sampling throughout the learning process, also known as on-demand sampling [20]. More specifically, pre-determining a sample-size budget for each distribution beforehand and sampling non-adaptively could result in loss of sample efficiency, as we lack knowledge about the complexity of learning each distribution before the learning process begins. The question then comes down to how to optimally adapt the online sampling strategy to effectively tackle diverse data distributions.

**Inadequacy of prior results.** The sample complexity of MDL has been explored in a strand of recent works under various settings. Consider first the case where the hypothesis class  $\mathcal{H}$  comprises a *finite* number of hypotheses. If we sample non-adaptively and draw the same number of samples from each individual distribution  $\mathcal{D}_i$ , then this results in a total sample size exceeding the order of  $\frac{k \log(|\mathcal{H}|)}{\epsilon^2}$  (given that learning each distribution requires a sample size at least on the order of  $\frac{\log(|\mathcal{H}|)}{\epsilon^2}$ ). Fortunately, this sample size budget can be significantly reduced with the aid of adaptive sampling. In particular, the state-of-the-art approach, proposed by Haghtalab et al. [20], accomplishes the objective Equation (1) with probability at least  $1 - \delta$  using  $O(\frac{\log(|\mathcal{H}|) + k \log(k/\delta)}{\epsilon^2})$  samples. In comparison to agnostic learning on a single distribution, it only incurs an extra *additive cost* of  $k \log(k/\delta)/\epsilon^2$  as opposed to a multiplicative factor in  $k$ , thus underscoring the importance of adaptive sampling.

A more challenging scenario arises when  $\mathcal{H}$  has a finite **Vapnik–Chervonenkis (VC)** dimension  $d$ . The sample complexity for VC classes has only been settled for the realizable case [6, 11, 31], a special scenario where it is feasible to achieve zero mean loss. For the general non-realizable case, the best-known lower bound for such VC classes is [20]<sup>3</sup>

$$\tilde{\Omega}\left(\frac{d+k}{\epsilon^2}\right), \quad (2)$$

which serves as a theoretical benchmark. By first collecting  $\tilde{O}(\frac{dk}{\epsilon})$  samples to help construct a cover of  $\mathcal{H}$  with reasonable resolution, Haghtalab et al. [20] established a sample complexity upper bound of

$$[20] \quad \tilde{O}\left(\frac{d+k}{\epsilon^2} + \frac{dk}{\epsilon}\right). \quad (3a)$$

Nevertheless, the term  $dk/\epsilon$  in Equation (3a) fails to match the lower bound Equation (2); put another way, this term might result in a potentially large burn-in cost, as the optimality of this approach is only guaranteed (up to log factors) when the total sample size already exceeds a (potentially large) threshold on the order of  $\frac{d^2k^2}{d+k}$ . In an effort to alleviate this  $dk/\epsilon$  factor, Awasthi et al. [4] put forward an alternative algorithm — which utilizes an oracle to learn on a single distribution and

<sup>3</sup>Let  $\mathcal{X} = (k, d, \frac{1}{\epsilon}, \frac{1}{\delta})$ . Here and throughout, the notation  $f(\mathcal{X}) = O(g(\mathcal{X}))$  or  $f(\mathcal{X}) \lesssim g(\mathcal{X})$  (respectively  $f(\mathcal{X}) = \Omega(g(\mathcal{X}))$  or  $f(\mathcal{X}) \gtrsim g(\mathcal{X})$ ) mean that there exists some universal constant  $C_1 > 0$  (respectively  $C_2 > 0$ ) such that  $f(\mathcal{X}) \leq C_1 g(\mathcal{X})$  (respectively  $f(\mathcal{X}) \geq C_2 g(\mathcal{X})$ ); the notation  $f(\mathcal{X}) = \Theta(g(\mathcal{X}))$  or  $f(\mathcal{X}) \asymp g(\mathcal{X})$  mean  $f(\mathcal{X}) = O(g(\mathcal{X}))$  and  $f(\mathcal{X}) = \Omega(g(\mathcal{X}))$  hold simultaneously. The notation  $\tilde{O}(\cdot)$ ,  $\tilde{\Omega}(\cdot)$  and  $\tilde{\Omega}(\cdot)$  are defined analogously except that all log factors in  $(k, d, \frac{1}{\epsilon}, \frac{1}{\delta})$  are hidden.

Table 1. Sample Complexity Bounds of MDL with  $k$  Data Distributions and a Hypothesis Class of VC Dimension  $d$

Article	Sample complexity bound
[20]	$\frac{\log( \mathcal{H} )+k}{\varepsilon^2}$
[20]	$\frac{d+k}{\varepsilon^2} + \frac{dk}{\varepsilon}$
[4]	$\frac{d}{\varepsilon^4} + \frac{k}{\varepsilon^2}$
[32]	$\frac{d+k}{\varepsilon^2} \left(\frac{k}{\varepsilon}\right)^{o(1)}$
<b>our work (Theorem 1)</b>	$\frac{d+k}{\varepsilon^2}$
lower bound: [20]	$\frac{d+k}{\varepsilon^2}$

Here, we only report the polynomial dependency and hide all logarithmic dependency on  $(k, d, \frac{1}{\varepsilon}, \frac{1}{\delta})$

obliviates the need for computing an epsilon-net of  $\mathcal{H}$  — yielding a sample complexity of

$$[4] \quad \tilde{O}\left(\frac{d}{\varepsilon^4} + \frac{k}{\varepsilon^2}\right). \quad (3b)$$

However, this result Equation (3b) might fall short of optimality as well, given that the scaling  $d/\varepsilon^4$  is off by a factor of  $1/\varepsilon^2$  compared with the lower bound Equation (2). A more comprehensive list of past results can be found in Table 1.

Given the apparent gap between the state-of-the-art lower bound Equation (2) and achievability bounds Equation (3), a natural question arises:

**Question:** Is it plausible to design a multi-distribution learning algorithm with a sample complexity of  $\tilde{O}(\frac{d+k}{\varepsilon^2})$  for VC classes, thereby matching the established lower bound Equation (2)?

Notably, this question has been posed as an open problem during the Annual Conference on Learning Theory (COLT) 2023; see Awasthi et al. [4, Problem 1].

**A glimpse of our main contributions.** The present article delivers some encouraging news: we come up with a new MDL algorithm that successfully resolves the aforementioned open problem in the affirmative. Specifically, focusing on a hypothesis class with VC dimension  $d$  and a collection of  $k$  data distributions, our main findings can be summarized in the following theorem.<sup>4</sup> Note that the loss function  $\ell(\cdot)$  in this theorem is not restricted to take the form of a zero-one loss and can be fairly general.

**THEOREM 1.** *There exists an algorithm (see Algorithm 1 for details) such that: with probability exceeding  $1 - \delta$ , the randomized hypothesis  $h^{\text{final}}$  returned by this algorithm achieves*

$$\max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, h^{\text{final}}} [\ell(h^{\text{final}}, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon,$$

*provided that the total sample size exceeds*

$$\frac{d+k}{\varepsilon^2} \text{poly log}\left(k, d, \frac{1}{\varepsilon}, \frac{1}{\delta}\right). \quad (4)$$

<sup>4</sup>Following the definition of the VC dimension, the hypotheses in  $\mathcal{H}$  are assumed to be binary-valued for VC classes.

The polylog factor in Equation (4) will be specified momentarily. In a nutshell, we develop the first algorithm that provably achieves a sample complexity matching the lower bound Equation (2) modulo logarithmic factors. Following the game dynamics template adopted in previous methods – namely, viewing MDL as a game between the learner (who selects the best hypothesis) and the adversary (who chooses the most challenging mixture of distributions) – our algorithm is built upon a novel and meticulously designed sampling scheme that deviates significantly from previous methods. Further, we extend our algorithm and theory to accommodate Rademacher classes, establishing a similar sample complexity bound; see Section 7 for details. Our algorithm and theory can also be extended to accommodate the multi-objective setting, which we postpone to Section 8.

Additionally, we solve two other open problems posed by [4]:

- *Oracle-efficient solutions.* An algorithm is said to be oracle-efficient if it only accesses  $\mathcal{H}$  through an **empirical risk minimization (ERM)** oracle [15]. Awasthi et al. [4, Problem 4] then asked what the sample complexity of MDL is when confined to oracle-efficient paradigms. Encouragingly, our algorithm (i.e., Algorithm 1) adheres to the oracle-efficient criterion, thus uncovering that the sample complexity of MDL remains unchanged when restricted to oracle-efficient algorithms.
- *Necessity of improper learning.* Both our algorithm and the most sample-efficient methods preceding our work produce randomized hypotheses. As discussed around Awasthi et al. [4, Problem 3], a natural question concerns characterization of the sample complexity when restricting the final output to deterministic hypotheses from  $\mathcal{H}$ . Our result (see Theorem 2) delivers a negative message: under mild conditions, for any MDL algorithm, there exists a hard problem instance such that it requires at least  $\Omega(dk/\varepsilon^2)$  samples to find a deterministic hypothesis  $h \in \mathcal{H}$  that attains  $\varepsilon$ -accuracy. This constitutes an enormous sample complexity gap between what can be achieved under improper learning and proper learning.

**Concurrent work.** We shall mention that a concurrent work Peng [32], posted around the same time as our work, also studied the MDL problem and significantly improved upon the prior results. More specifically, Peng [32] established a sample complexity of  $O(\frac{(d+k)\log(d/\delta)}{\varepsilon^2}(\frac{k}{\varepsilon})^{o(1)})$ , which is optimal up to some sub-polynomial factor in  $k/\varepsilon$ ; in comparison, our sample complexity is optimal up to polylogarithmic factor. Additionally, it is worth noting that the algorithm therein relies upon a certain recursive structure to eliminate the non-optimal hypothesis, thus incurring exponential computational cost even when an ERM oracle is available.

**Notation.** Throughout this article, we denote  $[N] := \{1, \dots, N\}$  for any positive integer  $N$ . Let  $\text{conv}(\mathcal{A})$  represent the convex hull of a set  $\mathcal{A}$ . Denote by  $\Delta(n)$  the  $n$ -dimensional simplex for any positive integer  $n$ , and  $\Delta(\mathcal{A})$  the probability simplex over the set  $\mathcal{A}$ . For two vectors  $v = [v_i]_{1 \leq i \leq n}$  and  $v' = [v'_i]_{1 \leq i \leq n}$  with the same dimension, we overload the notation by using  $\max\{v, v'\} = [\max\{v_i, v'_i\}]_{1 \leq i \leq n}$  to denote the coordinate-wise maximum of  $v$  and  $v'$ . Also we say  $v \leq v'$  iff  $v_i \leq v'_i$  for all  $i \in [n]$ . For any random variable  $X$ , we use  $\text{Var}[X]$  to denote its variance, i.e.,  $\text{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2]$ . Let  $e_1^{\text{basis}}, \dots, e_k^{\text{basis}}$  represent the standard basis vectors in  $\mathbb{R}^k$ . For any two distributions  $P$  and  $Q$  supported on  $\mathcal{X}$ , the **Kullback-Leibler (KL)** divergence from  $Q$  to  $P$  is defined and denoted by

$$\text{KL}(P \| Q) := \mathbb{E}_Q \left[ \frac{dP}{dQ} \log \frac{dP}{dQ} \right]. \quad (5)$$

## 2 Problem Formulation

This section formulates the multi-distribution learning problem. We assume throughout that each datapoint takes the form of  $(x, y) \in \mathcal{X} \times \mathcal{Y}$ , with  $\mathcal{X}$  (respectively  $\mathcal{Y}$ ) the instance space (respectively label space).

**Learning from multiple distributions.** The problem setting encompasses several elements below.

- *Hypothesis class.* Suppose we are interested in a hypothesis class  $\mathcal{H}$ , comprising a set of candidate functions from the instance space  $\mathcal{X}$  to the label space  $\mathcal{Y}$ . Overloading the notation, we use  $h_\pi$  to represent a *randomized hypothesis* associated with a probability distribution  $\pi \in \Delta(\mathcal{H})$ , meaning that a hypothesis  $h$  from  $\mathcal{H}$  is randomly selected according to distribution  $\pi$ . When it comes to a VC class, we assume that the hypotheses in  $\mathcal{H}$  are binary-valued, and that the VC dimension [41] of  $\mathcal{H}$  is

$$\text{VC-dim}(\mathcal{H}) = d. \quad (6)$$

- *Loss function.* Suppose we are given a loss function  $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \rightarrow [-1, 1]$ , so that  $\ell(h, (x, y))$  quantifies the risk of using hypothesis  $h \in \mathcal{H}$  to make prediction on a datapoint  $(x, y) \in \mathcal{X} \times \mathcal{Y}$  (i.e., predicting  $y$  based on  $x$ ). One example is the 0-1 loss function  $\ell(h, (x, y)) = \mathbb{1}\{h(x) \neq y\}$ , which is often used to measure the misclassification error.

- *(Multiple) data distributions.* Suppose that there are  $k$  data distributions of interest supported on  $\mathcal{X} \times \mathcal{Y}$ , denoted by  $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_k\}$ . We are permitted to draw independent samples from each of these data distributions.

Given a target accuracy level  $\varepsilon \in (0, 1)$ , our objective is to identify a (randomized) hypothesis, represented by  $h_\pi$  with  $\pi \in \Delta(\mathcal{H})$ , such that

$$\max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, h \sim \pi} [\ell(h, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon. \quad (7)$$

**Sampling and learning processes.** In order to achieve the aforementioned goal Equation (7), we need to draw samples from the available data distributions in  $\mathcal{D}$ , and the current article focuses on sampling in an online fashion. More precisely, the learning process proceeds as follows: in each step  $\tau$ ,

- the learner selects  $i_\tau \in [k]$  based on the previous samples;
- the learner draws an *independent* sample  $(x_\tau, y_\tau)$  from the data distribution  $\mathcal{D}_{i_\tau}$ .

The sample complexity of a learning algorithm thus refers to the total number of samples drawn from  $\mathcal{D}$  throughout the learning process. A desirable learning algorithm would yield an  $\varepsilon$ -optimal (randomized) hypothesis (i.e., a hypothesis that achieves Equation (7)) using as few samples as possible.

## 3 Algorithm

In this section, we present our proposed algorithm for learning VC classes. Before proceeding, we find it convenient to introduce some notation concerning the loss under mixed distributions. Specifically, for any distribution  $w = [w_i]_{1 \leq i \leq k} \in \Delta(k)$  and any hypothesis  $h \in \mathcal{H}$ , the risk over the mixture  $\sum_{i \in [k]} w_i \mathcal{D}_i$  of data distributions is denoted by

$$L(h, w) := \sum_{i=1}^k w_i \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))]; \quad (8a)$$

similarly, the risk of a randomized hypothesis  $h_\pi$  (associated with  $\pi \in \Delta(\mathcal{H})$ ) over  $\sum_{i \in [k]} w_i \mathcal{D}_i$  is given by

$$L(h_\pi, w) := \sum_{i=1}^k w_i \mathbb{E}_{(x,y) \sim \mathcal{D}_i, h \sim \pi} [\ell(h_\pi, (x, y))] = \mathbb{E}_{h \sim \pi} [L(h, w)]. \quad (8b)$$

**ALGORITHM 1:** Hedge for multi-distribution learning on VC classes (MDL-Hedge-VC)

---

**1** **input:**  $k$  data distributions  $\{\mathcal{D}_i\}_{i=1}^k$ , hypothesis class  $\mathcal{H}$ , target accuracy level  $\varepsilon$ , target success rate  $1 - \delta$ .

**2** **hyper-parameters:** stepsize  $\eta = \frac{1}{100}\varepsilon$ , number of rounds  $T = \frac{20000\log(\frac{k}{\delta})}{\varepsilon^2}$ , auxiliary accuracy level  $\varepsilon_1 = \frac{1}{100}\varepsilon$ , auxiliary sub-sample-size  $T_1 := \frac{4000(k\log(k/\varepsilon_1) + d\log(kd/\varepsilon_1) + \log(1/\delta))}{\varepsilon_1^2}$ .

**3** **initialization:** for all  $i \in [k]$ , set  $W_i^1 = 1$ ,  $\widehat{w}_i^0 = 0$  and  $n_i^0 = 0$ ;  $\mathcal{S} = \emptyset$ .

**4** **for**  $t = 1, 2, \dots, T$  **do**

**5** set  $w^t = [w_i^t]_{1 \leq i \leq k}$  and  $\widehat{w}^t = [\widehat{w}_i^t]_{1 \leq i \leq k}$ , with  $w_i^t \leftarrow \frac{W_i^t}{\sum_j W_j^t}$  and  $\widehat{w}_i^t \leftarrow \widehat{w}_i^{t-1}$  for all  $i \in [k]$ .

*/\* recompute  $\widehat{w}^t$  & draw new samples for  $\mathcal{S}$  only if  $w^t$  changes sufficiently. \*/*

**6** **if** there exists  $j \in [k]$  such that  $w_j^t \geq 2\widehat{w}_j^{t-1}$  **then**

**7**  $\widehat{w}_i^t \leftarrow \max\{w_i^t, \widehat{w}_i^{t-1}\}$  for all  $i \in [k]$ ;

**8** **for**  $i = 1, \dots, k$  **do**

**9**  $n_i^t \leftarrow \lceil T_1 \widehat{w}_i^t \rceil$ ;

**10** draw  $n_i^t - n_i^{t-1}$  independent samples from  $\mathcal{D}_i$ , and add these samples to  $\mathcal{S}$ .

*/\* estimate the near-optimal hypothesis for weighted data distributions. \*/*

**11** compute  $h^t \leftarrow \arg \min_{h \in \mathcal{H}} \widehat{L}^t(h, w^t)$ , where

$$\widehat{L}^t(h, w^t) := \sum_{i=1}^k \frac{w_i^t}{n_i^t} \cdot \sum_{j=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})), \quad (9)$$

with  $(x_{i,j}, y_{i,j})$  being the  $j$ th datapoint from  $\mathcal{D}_i$  in  $\mathcal{S}$ .

*/\* estimate the loss vector and execute weighted updates. \*/*

**12**  $\overline{w}_i^t \leftarrow \max_{1 \leq r \leq t} w_i^r$  for all  $i \in [k]$ .

**13** **for**  $i = 1, \dots, k$  **do**

**14** draw  $\lceil k\overline{w}_i^t \rceil$  independent samples – denoted by  $\{(x_{i,j}^t, y_{i,j}^t)\}_{j=1}^{\lceil k\overline{w}_i^t \rceil}$  – from  $\mathcal{D}_i$ , and set

$$\widehat{r}_i^t = \frac{1}{\lceil k\overline{w}_i^t \rceil} \sum_{j=1}^{\lceil k\overline{w}_i^t \rceil} \ell(h^t, (x_{i,j}^t, y_{i,j}^t));$$

**15** update the weight as  $W_i^{t+1} = W_i^t \exp(\eta \widehat{r}_i^t)$ . // Hedge updates.

**16** **output:** a randomized hypothesis  $h^{\text{final}}$  uniformly distributed over  $\{h^t\}_{t=1}^T$ .

---

*Remark 1.* Note that in Algorithm 1, the quantities  $\{n_i^t\}_{1 \leq t \leq T}$  are designed to be non-decreasing in  $t$ .

*Remark 2.* In line 11 of Algorithm 1, we also allow  $h^t$  to be  $\varepsilon_1$ -best response that obeys  $\widehat{L}^t(h^t, w^t) \leq \min_{h \in \mathcal{H}} \widehat{L}^t(h, w^t) + \varepsilon_1$ ; our theory remains unchanged if we make this modification.

Following the game dynamics proposed in previous works [4, 20], our algorithm alternates between computing the most favorable hypothesis (performed by the learner) and estimating the most challenging mixture of data distributions (performed by the adversary), with the aid of no-regret learning algorithms [34, 37]. More specifically, in each round  $t$ , our algorithm performs the following two steps:

- (a) Given a mixture of data distributions  $\mathcal{D}^{(t)} = \sum_{i \in [k]} w_i^t \mathcal{D}_i$  (with  $w^t = [w_i^t]_{i \in [k]} \in \Delta(k)$ ), we construct a dataset to compute a hypothesis  $h^t$  that nearly minimizes the loss under  $\mathcal{D}^{(t)}$ , namely,

$$h^t \approx \arg \min_{h \in \mathcal{H}} L(h, w^t). \quad (10)$$

This is accomplished by calling an empirical risk minimization oracle.

- (b) Given hypothesis  $h^t$ , we compute an updated weight vector  $w^{t+1} \in \Delta(k)$  – and hence an updated mixed distribution  $\mathcal{D}^{(t+1)} = \sum_{i \in [k]} w_i^{t+1} \mathcal{D}_i$ . The weight updates are carried out using the celebrated Hedge algorithm [18] designed for online adversarial learning,<sup>5</sup> in an attempt to achieve low regret even when the loss vectors are adversarially chosen. More precisely, we run

$$w_i^{t+1} \propto w_i^t \exp(\eta \hat{r}_i^t), \quad i \in [k], \quad (11)$$

where the loss vector  $\hat{r}^t = [\hat{r}_i^t]_{i \in [k]}$  contains the empirical loss of  $h^t$  under each data distribution, i.e.,

$$\hat{r}_i^t \approx \underset{(x,y) \sim \mathcal{D}_i}{\mathbb{E}} [\ell(h^t, (x, y))], \quad i \in [k],$$

computed over another set of data samples.

In words, the min-player and the max-player in our algorithm follow the best-response dynamics and no-regret dynamics, respectively (note that the Hedge algorithm is known to be a no-regret algorithm [34]). At the end of the algorithm, we output a randomized hypothesis  $h^{\text{final}}$  that is uniformly distributed over the hypothesis iterates  $\{h^t\}_{1 \leq t \leq T}$  over all  $T$  rounds, following common practice in online adversarial learning.

While the above paradigm has been adopted in past works [4, 20], the resulting sample complexity depends heavily upon how data samples are collected and utilized throughout the learning process. For instance, Awasthi et al. [4, Algorithm 1] – which also alternates between best response and no-regret algorithm – draws *fresh data* at each step of every round, in order to ensure reliable estimation of the loss function of interest through elementary concentration inequalities. This strategy, however, becomes wasteful over time, constituting one of the main sources of its sample sub-optimality.

In order to make the best use of data, we propose the following key strategies.

- *Sample reuse in Step (a).* In stark contrast to Awasthi et al. [4, Algorithm 1] that draws new samples for estimating each  $h^t$ , we propose to reuse all samples collected in Step (a) up to the  $t$ th round to assist in computing  $h^t$ . As will be made precise in lines 6-11 of Algorithm 1, we shall maintain a *growing dataset*  $\mathcal{S}$  for conducting Step (a) throughout, ensuring that there are  $n_i^t$  samples drawn from distribution  $\mathcal{D}_i$  in the  $t$ th round. These datapoints are employed to construct an empirical loss estimator  $\hat{L}^t(h, w^t)$  for each  $h \in \mathcal{H}$  in each round  $t$ , with the aim of achieving uniform convergence  $|\hat{L}^t(h, w^t) - L(h, w^t)| \leq O(\varepsilon)$  over all  $h \in \mathcal{H}$ . More detailed explanations are provided in Section 4.1.
- *Weighted sampling for Step (b).* As shown in line 14 of Algorithm 1, in each round  $t$ , we sample each  $\mathcal{D}_i$  a couple of times to compute the empirical estimator for  $\mathbb{E}_{(x,y) \in \mathcal{D}_i} [\ell(h^t, (x, y))]$ , where the number of samples depends upon the running weights  $\{w_i^\tau\}$ . More precisely, we collect  $\lceil k \bar{w}_i^t \rceil$  fresh samples from each  $\mathcal{D}_i$ , where  $\bar{w}_i^t := \max_{1 \leq \tau \leq t} w_i^\tau$  is the maximum weight assigned to  $\mathcal{D}_i$  up to now. Informally speaking, this strategy is chosen carefully to ensure reduced variance of the estimators given a sample size budget. The interested reader is referred to Section 4.2 and Lemma 8 for more detailed explanations.

The whole procedure can be found in Algorithm 1.

---

<sup>5</sup>Note that the Hedge algorithm is closely related to Exponentiated Gradient Descent, Multiplicative Weights Update, Online Mirror Descent, and so on [2, 23, 37].

## 4 A Glimpse of Key Technical Novelties

In this section, we highlight two technical novelties that empower our analysis: (i) uniform convergence of the weighted sampling estimator that allows for sample reuse (see Section 4.1), and (ii) tight control of certain  $\|\cdot\|_{1,\infty}$  norm of the iterates  $\{w^t\}_{1 \leq t \leq T}$  that dictates the sample efficiency (see Section 4.2).

### 4.1 Toward Sample Reuse: Uniform Concentration and a Key Quantity

Recall that in Algorithm 1, we invoke the empirical risk estimator  $\hat{L}^t(h, w^t)$  as an estimate of the true risk of hypothesis  $h$  over the weighted distribution specified by  $w^t$  (cf. Equation (9)). In order to facilitate sample reuse when constructing such risk estimators across all iterations, it is desirable to establish uniform concentration results to control the errors of such risk estimators throughout the execution of the algorithm. Toward this end, our analysis strategy proceeds as follows:

**Step 1: concentration for any fixed set of parameters.** Consider any given set of integers  $n = \{n_i\}_{i=1}^k$  and any given vector  $w \in \Delta(k)$ . Suppose, for each  $i \in [k]$ , we have  $n_i$  i.i.d. samples drawn from  $\mathcal{D}_i$  – denoted by  $\{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i}$  – and let us look at the empirical risk estimator,

$$\hat{L}_n(h, w) := \sum_{i=1}^k w_i \cdot \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})), \quad (12)$$

which is a sum of independent random variables. Evidently, for a given hypothesis  $h$ , the variance of  $\hat{L}_n(h, w)$  is upper bounded by

$$\text{Var}(\hat{L}_n(h, w)) \leq \sum_{i=1}^k \frac{w_i^2}{n_i} \leq \left( \sum_{i=1}^k w_i \right) \frac{1}{\min_i n_i / w_i} = \frac{1}{\min_i n_i / w_i}.$$

Assuming that the central limit theorem is applicable, one can derive

$$\mathbb{P}\left\{ |\hat{L}_n(h, w) - L(h, w)| \geq \varepsilon \right\} \lesssim \exp\left(-\frac{\varepsilon^2}{2\text{Var}(\hat{L}_n(h, w))}\right) \lesssim \exp\left(-\frac{\varepsilon^2}{2} \min_i \frac{n_i}{w_i}\right).$$

Armed with this result, we can extend it to accommodate all  $h \in \mathcal{H}$  through the union bound. For a VC class with  $\text{VC-dim}(\mathcal{H}) = d$ , the celebrated Sauer–Shelah lemma [42, Proposition 4.18] tells us that the set of hypotheses can be effectively compressed into a subset with cardinality no larger than  $\exp(\tilde{O}(d))$ . Taking the union bound then yields

$$\mathbb{P}\left(\max_{h \in \mathcal{H}} |\hat{L}_n(h, w) - L(h, w)| \geq \varepsilon\right) \lesssim \exp\left(\tilde{O}(d) - \frac{\varepsilon^2}{2} \min_i \frac{n_i}{w_i}\right).$$

**Step 2: uniform concentration.** Next, we would like to extend the above result to establish uniform concentration over all  $n$  and  $w$  of interest. Toward this, we shall invoke the union bound as well as the standard epsilon-net arguments. Let the set  $\mathcal{X} \subseteq \Delta(k)$  be a proper discretization of  $\Delta(k)$ , with cardinality  $\exp(\tilde{O}(k))$ . In addition, given the trivial upper bound  $n_i \leq T_1$  for all  $i \in [k]$ , we know that there exist at most  $T_1^k = \exp(\tilde{O}(k))$  possible combinations of  $\{n_i\}_{i \in [k]}$ . We can then apply the union bound to show that

$$\mathbb{P}\left\{ \exists w \in \mathcal{X} \text{ and feasible } n \text{ s.t. } |\hat{L}_n(h, w) - L(h, w)| \geq \varepsilon \right\} \lesssim \exp\left(\tilde{O}(k) + \tilde{O}(d) - \frac{\varepsilon^2}{2} \min_i \frac{n_i}{w_i}\right). \quad (13)$$

When the discretized set  $\mathcal{X}$  is chosen to have sufficient resolution, we can straightforwardly employ the standard covering argument to extend the above inequality to accommodate all  $w \in \Delta(k)$  of interest.

**Key takeaways.** The above arguments reveal the following high-probability property: whenever we collect  $n = \{n_i\}_{i=1}^k$  samples in the learning process, we could obtain  $\varepsilon$ -approximation  $\widehat{L}_n(h, w)$  (see Equation (12)) of  $L(h, w)$  for all  $h \in \mathcal{H}$  and all  $w \in \Delta(k)$  with high probability, provided that

$$\min_i \frac{n_i}{w_i} \gtrsim \widetilde{O}\left(\frac{k+d}{\varepsilon^2}\right). \quad (14)$$

This makes apparent the pivotal role of the quantity  $\min_i n_i / w_i$ . In our algorithm, we design the update rule (cf. line 9 of Algorithm 1) to guaranteed that

$$\min_i \frac{n_i^t}{w_i^t} \gtrsim T_1 \geq \widetilde{\Omega}\left(\frac{k+d}{\varepsilon^2}\right), \quad (15)$$

for all  $1 \leq t \leq T$ . In fact, this explains our choice of  $T_1$  in Algorithm 1. Crucially, the aforementioned uniform concentration result allows us to reuse samples throughout the learning process instead of drawing fresh samples to estimate  $L(h, w^t)$  in each round  $t$  (note that the latter approach clearly falls short of data efficiency). To conclude, to guarantee  $\varepsilon$ -uniform convergence for all rounds, it suffices to choose  $T_1 = \widetilde{\Omega}\left(\frac{k+d}{\varepsilon^2}\right)$ .

Finally, recall that  $n_i^t \asymp T_1 \bar{w}_i^t$  for each  $i \in [k]$  and  $t \leq T$ , with  $\bar{w}_i^t := \max_{1 \leq \tau \leq t} w_i^\tau$ ; taking  $n_i^t \asymp T_1 \bar{w}_i^t$  (as opposed to  $n_i^t \asymp T_1 w_i^t$ ) ensures that the sample size  $n_i^t$  is monotonically non-decreasing in  $t$ . With Equation (15) in mind, the total number of samples collected within  $T$  rounds in Algorithm 1 obeys

$$\frac{1}{T_1} \sum_{i=1}^k n_i^T \asymp \sum_{i=1}^k \bar{w}_i^T =: \|\bar{w}^T\|_1. \quad (16)$$

This threshold  $\|\bar{w}^T\|_1$  – or equivalently, the  $\|\cdot\|_{1,\infty}$  norm of  $\{w_i^t\}_{1 \leq t \leq T}$  – is a critical quantity that we wish to control. In particular, in the desirable scenario where  $\|\bar{w}^T\|_1 \leq \widetilde{O}(1)$ , the total sample size obeys  $\sum_{i=1}^k n_i^T \asymp T_1 \|\bar{w}^T\|_1 = \widetilde{O}\left(\frac{k+d}{\varepsilon^2}\right)$ .

## 4.2 Bounding the Key Quantity $\|\bar{w}^T\|_1$ by Tracking the Hedge Trajectory

Perhaps the most innovative (and most challenging) part of our analysis lies in controlling the  $\|\cdot\|_{1,\infty}$  norm of  $\{w_i^t\}_{1 \leq t \leq T}$ , whose critical importance has been pointed out in Section 4.1.

Toward this end, the key lies in carefully tracking the dynamics of the Hedge algorithm. To elucidate the high-level idea, let us look at a simpler minimax optimization problem w.r.t. the set of loss vectors in the convex hull of a set  $\mathcal{Y} \subseteq \mathbb{R}^k$ :

$$\text{OPT} := \min_{y \in \text{conv}(\mathcal{Y})} \max_{w \in \Delta(k)} w^\top y \quad \left( \text{or equivalently, } \max_{w \in \Delta(k)} \min_{y \in \text{conv}(\mathcal{Y})} w^\top y \right), \quad (17)$$

where the equivalence arises from von Neumann's minimax theorem [39]. Consider the following algorithm (cf. Algorithm 2) tailored to this minimax problem, assuming perfect knowledge about the loss vectors.<sup>6</sup>

This algorithm is often referred to as the Hedge algorithm. In particular, if  $y^t$  is taken to be the best response  $y^t = \arg \min_{y \in \mathcal{Y}} \langle w^t, y \rangle$  w.r.t.  $w^t$  for all  $1 \leq t \leq T$ , then this Hedge algorithm is known to yield an  $\varepsilon$ -minimax solution within  $O\left(\frac{\log(k)}{\varepsilon^2}\right)$  iterations. A challenging question relevant to our analysis is:

**Question:** can we bound  $\|\bar{w}^T\|_1 := \sum_{i=1}^k \max_{1 \leq t \leq T} w_i^t$  in Algorithm 2 by poly-logarithmic terms?

<sup>6</sup>Note that in Algorithm 1, we can only estimate the loss vector using the collected samples. Additional efforts are needed to reduce the variability (see line 14 in Algorithm 1).

**ALGORITHM 2:** The Hedge algorithm for bilinear games.

---

```

1 Input:  $\mathcal{Y} \subseteq [-1, 1]^k$ , target accuracy level  $\varepsilon \in (0, 1)$ .
2 Initialization:  $T = \frac{100 \log(k)}{\varepsilon^2}$ ,  $\eta = \frac{1}{10}\varepsilon$ , and  $W_i^1 = 1$  for all  $1 \leq i \leq k$ .
3 for  $t = 1, 2, \dots, T$  do
4   compute  $w_i^t \leftarrow \frac{W_i^t}{\sum_j W_j^t}$  for every  $1 \leq i \leq k$ .
5   receive  $y^t = [y_1^t, y_2^t, \dots, y_k^t]^\top$ .
6   update  $W_i^{t+1} \leftarrow W_i^t \exp(\eta y_i^t)$  with for every  $1 \leq i \leq k$ .
7 Return:  $\frac{1}{T} \sum_{t=1}^T y^t$ .

```

---

As it turns out, we can answer this question affirmatively (see Lemma 3), and the key ideas will be elucidated in the remainder of this section.

**4.2.1 First Attempt: Bounding the Number of Distributions with  $\max_{1 \leq t \leq T} w_i^t = \Omega(1)$ .** Instead of bounding  $\|\bar{w}^T\|_1$  directly, our first attempt is to look at those  $i \in [k]$  with large  $\max_{1 \leq t \leq T} w_i^t$  (more specifically,  $\max_{1 \leq t \leq T} w_i^t \geq 1/4$ ) and show that:

- there exist at most  $\tilde{O}(1)$  coordinates  $i \in [k]$  obeying  $\max_{1 \leq t \leq T} w_i^t \geq 1/4$  (or some other universal constant).

In other words, we would like to demonstrate that the cardinality of the following set is small:

$$\mathcal{W}_{\text{large}} := \{i \in [k] \mid \max_{1 \leq t \leq T} w_i^t \geq 1/4\}. \quad (18)$$

To do so, note that some standard “continuity”-type argument tells us that: for a sufficiently small stepsize  $\eta$ , one can find, for each  $i \in \mathcal{W}_{\text{large}}$ , a time interval  $[s_i, e_i] \subseteq [0, T]$  obeying

$$1/16 \leq w_i^{s_i} \leq 1/8, \quad w_i^{e_i} \geq 1/4, \quad \text{and} \quad w_i^t \geq 1/8 \quad \forall t \in (s_i, e_i]. \quad (19)$$

In words,  $w_i^t$  at least doubles from  $t = s_i$  to  $t = e_i$ . We claim for the moment that

$$e_i - s_i \geq \Omega(1/\eta^2) = \Omega(1/\varepsilon^2) \quad \forall i \in \mathcal{W}_{\text{large}}. \quad (20)$$

Additionally, observe that for any  $t$ , there exist at most 8 coordinates  $i \in \mathcal{W}_{\text{large}}$  such that  $s_i \leq t \leq e_i$  (since  $w_i^t \geq 1/8$  for every  $t \in [s_i, e_i]$ ). This reveals that

$$8T \geq \sum_{i \in \mathcal{W}_{\text{large}}} (e_i - s_i) \geq |\mathcal{W}_{\text{large}}| \cdot \Omega(1/\varepsilon^2), \quad (21)$$

which combined with our choice of  $T = \tilde{O}(1/\varepsilon^2)$  (cf. line 2 of Algorithm 1) yields

$$|\mathcal{W}_{\text{large}}| \leq O(T\varepsilon^2) = \tilde{O}(1).$$

In words, the number of distributions with large  $\max_{1 \leq t \leq T} w_i^t$  (i.e.,  $\max_{1 \leq t \leq T} w_i^t \geq 1/4$ ) is fairly small.

**PROOF SKETCH FOR EQUATION (20).** Let us briefly discuss the high-level proof ideas. Following standard analysis for the Hedge algorithm (e.g., Lattimore and Szepesvári [27], Shalev-Shwartz [37]), we can often obtain (under certain mild conditions)

$$\text{KL}(w^{s_i} \| w^{e_i}) \leq O(\eta^2(e_i - s_i)).$$

Combine this relation with basic properties about the KL divergence (see, e.g., Lemmas 15 and 16 in Appendix A) and the choice  $\eta = \varepsilon/10$  to obtain

$$\begin{aligned} e_i - s_i &\geq \Omega(\eta^{-2} \text{KL}(w^{s_i} \| w^{e_i})) \geq \Omega\left(\eta^{-2} \text{KL}(\text{Ber}(w_i^{s_i}) \| \text{Ber}(w_i^{e_i}))\right) \\ &\geq \Omega\left(\eta^{-2} \frac{\frac{1}{16} \cdot 2^2}{4}\right) = \Omega(1/\eta^2) = \Omega(1/\varepsilon^2). \end{aligned}$$

□

**4.2.2 More General Cases: Issues and Solutions.** Naturally, one would hope to generalize the arguments in Section 4.2.1 to cope with more general cases. More specifically, let us look at the following set

$$\mathcal{W}(p) := \{i \in [k] \mid \max_{1 \leq t \leq T} w_i^t \in [2p, 4p]\}, \quad (22)$$

defined for each  $p \in [0, 1]$ . If one could show that

$$|\mathcal{W}(p)| = \tilde{O}(1/p) \quad \text{for each } p, \quad (23)$$

then a standard doubling argument would immediately lead to

$$\begin{aligned} \|\bar{w}^T\|_1 &= \sum_{i \in [k]} \max_{1 \leq t \leq T} w_i^t \approx \sum_{j=1}^{\log_2 k} \sum_{i \in \mathcal{W}(2^{-j})} \max_{1 \leq t \leq T} w_i^t \leq \sum_{j=1}^{\log_2 k} 4 \cdot 2^{-j} \cdot |\mathcal{W}(2^{-j})| \\ &\leq \sum_{j=1}^{\log_2 k} 4 \cdot 2^{-j} \cdot \tilde{O}\left(\frac{1}{2^{-j}}\right) = \tilde{O}(1). \end{aligned}$$

**A technical issue.** Nevertheless, simply repeating the arguments in Section 4.2.1 fails to deliver the desirable bound Equation (23) on  $|\mathcal{W}(p)|$  when  $p$  is small. Briefly speaking, for each  $i \in \mathcal{W}(p)$ , let  $[s_i, e_i]$  represent a time interval (akin to Equation (19)) such that

$$p/2 \leq w_i^{s_i} \leq p, \quad w_i^{e_i} \geq 2p \quad \text{and} \quad w_i^t \geq p \quad \text{for any } s_i < t \leq e_i. \quad (24)$$

Repeating the heuristic arguments in Section 4.2.1 leads to

$$e_i - s_i \geq \Omega(\eta^{-2} \text{KL}(w^{s_i} \| w^{e_i})) \geq \Omega\left(\eta^{-2} \text{KL}(\text{Ber}(w_i^{s_i}) \| \text{Ber}(w_i^{e_i}))\right) \geq \Omega(\eta^{-2} p) = \Omega(p/\varepsilon^2). \quad (25)$$

Given that each  $t$  is contained within at most  $1/(p/2)$  intervals associated with  $\mathcal{W}(p)$ , repeat the arguments for Equation (21) to derive

$$\frac{1}{p/2} \cdot T \geq \sum_{i \in \mathcal{W}(p)} (e_i - s_i) \geq |\mathcal{W}(p)| \cdot \Omega(p/\varepsilon^2) \implies |\mathcal{W}(p)| \leq \tilde{O}\left(\frac{1}{p^2}\right).$$

This bound, however, is clearly loose compared to the desirable one in Equation (23).

**Our solution.** To address this issue, we make two key observations below that inspire our approach:

- *Shared intervals.* For the interval  $[s_i, e_i]$  associated with  $i$  as defined above, if there exist other indices sharing the same interval  $[s_i, e_i]$  (in the sense that relations analogous to Equation (24) are satisfied for other indices), then it is plausible to improve the bound. For instance, if a set  $\mathcal{M}_i \subseteq [k]$  of indices share the same interval  $[s_i, e_i]$ , then one can follow the heuristic argument in Equation (25) to obtain

$$e_i - s_i \geq \Omega(\eta^{-2} \text{KL}(w^{s_i} \| w^{e_i})) \geq \Omega\left(\eta^{-2} \text{KL}\left(\text{Ber}\left(\sum_{j \in \mathcal{M}_i} w_j^{s_i}\right) \| \text{Ber}\left(\sum_{j \in \mathcal{M}_i} w_j^{e_i}\right)\right)\right) \geq \Omega(p|\mathcal{M}_i|/\varepsilon^2), \quad (26)$$

which clearly strengthens the original bound Equation (25) if  $|\mathcal{M}_i|$  is large.

– *Disjoint intervals.* Consider the special case where  $\mathcal{W}(p)$  can be divided into subsets  $\{\mathcal{V}_n\}_{n=1}^N$  obeying

(i) for each  $n \in [N]$ , all indices in  $\mathcal{V}_n$  share the same interval  $[s_n, e_n]$  (defined analogously as Equation (24));

(ii) the intervals  $\{[s_n, e_n]\}_{n=1}^N$  are *disjoint*.

Then one can derive the desired bound on  $|\mathcal{W}(p)|$ . More precisely, it follows from Equation (26) that

$$e_n - s_n \geq \Omega(p|\mathcal{V}_n|/\varepsilon^2), \quad (27)$$

which together with the disjointness property yields

$$|\mathcal{W}(p)| = \sum_{n=1}^N |\mathcal{V}_n| \leq \sum_{n=1}^N O\left(\frac{(e_n - s_n)\varepsilon^2}{p}\right) \leq O\left(\frac{T\varepsilon^2}{p}\right) = \tilde{O}\left(\frac{1}{p}\right). \quad (28)$$

In light of the above discussion, it is helpful to (a) merge those indices that share similar intervals, and (b) identify disjoint intervals whose associated indices can cover a good fraction of  $\mathcal{W}(p)$ .

Motivated by the aforementioned observation about “shared intervals,” we introduce the notion of “*segments*” to facilitate analysis.

**Definition 1 (Segment).** For any  $p, x > 0$  and  $i \in [k]$ , we say that  $(t_1, t_2)$  is a  $(p, q, x)$ -*segment* if there exists a subset  $\mathcal{J} \subseteq [k]$  such that

- (i)  $\sum_{i \in \mathcal{J}} w_i^{t_1} \in [p/2, p]$ ,
- (ii)  $\sum_{i \in \mathcal{J}} w_i^{t_2} \geq p \exp(x)$ ,
- (iii)  $\sum_{i \in \mathcal{J}} w_i^t \geq q$  for any  $t_1 \leq t \leq t_2$ .

We shall refer to  $t_1$  as the starting point and  $t_2$  as the end point, and call  $\mathcal{J}$  the associated index set. Moreover, two segments  $(s_1, e_1)$  and  $(s_2, e_2)$  are said to be disjoint if  $s_1 < e_1 \leq s_2 < e_2$  or  $s_2 < e_2 \leq s_1 < e_1$ .

This definition allows one to pool indices with similar intervals together.

In general, however, it is common for two segments to be overlapping (see Figure 2 in Appendix B.6), which precludes us from directly invoking our aforementioned observation about “disjoint intervals.” To address this issue, our strategy is to extract out shared sub-segments<sup>7</sup> of (a subset of) these segments in a meticulous manner. Encouragingly, it is possible to find such sub-segments that taken collectively cover a good fraction (i.e.,  $\frac{1}{\text{polylog}(k, T)}$ ) of all segments, meaning that we do not have to discard too many segments. The construction is built upon careful analysis of these segments, and will be elucidated in Appendix 5.3.

## 5 Analysis for VC Classes (Proof of Theorem 1)

### 5.1 Key Lemmas Underlying the Proof

The main steps for establishing Theorem 1 lie in proving three key lemmas, as stated below.

The first lemma is concerned with the hypothesis  $h^t = \arg \min_{h \in \mathcal{H}} \hat{L}^t(h, w^t)$  (cf. line 11 of Algorithm 1); in words,  $h^t$  is the minimizer of the empirical loss function  $\hat{L}^t(\cdot, w^t)$ , computed using samples obtained up to the  $t$ th round. The following lemma tells us that: even though  $h^t$  is an empirical minimizer, it almost optimizes the weighted population loss  $L(\cdot, w^t)$ . In other words, this lemma justifies that the adaptive sampling scheme proposed in Algorithm 1 ensures faithfulness of the empirical loss and its minimizer; here, we recall that  $\varepsilon_1 = \varepsilon/100$  (cf. line 2 of Algorithm 1).

<sup>7</sup>A sub-segment refers to a sub-interval of a segment, as illustrated in Figure 6.

LEMMA 1. *With probability at least  $1 - \delta/4$ ,*

$$L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1 \quad (29)$$

*holds for all  $1 \leq t \leq T$ , where  $h^t$  (respectively  $w^t$ ) is the hypothesis (respectively weight vector) computed in round  $t$  of Algorithm 1.*

PROOF. See Appendix B.1.  $\square$

Next, assuming that Equation (29) holds, we can resort to standard analysis for the Hedge algorithm to demonstrate the quality of the final output  $h^{\text{final}}$ .

LEMMA 2. *Suppose that lines 6-11 in Algorithm 1 are replaced with some oracle that returns a hypothesis  $h^t$  satisfying  $L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1$  in the  $t$ th round for each  $1 \leq t \leq T$ . With probability exceeding  $1 - \delta/4$ , the hypothesis  $h^{\text{final}}$  output by Algorithm 1 is  $\varepsilon$ -optimal in the sense that*

$$\max_{1 \leq i \leq k} L(h^{\text{final}}, e_i^{\text{basis}}) \leq \min_{\pi \in \Delta(\mathcal{H})} \max_{1 \leq i \leq k} L(h_\pi, e_i^{\text{basis}}) + \varepsilon \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} L(h, e_i^{\text{basis}}) + \varepsilon. \quad (30)$$

Here, we recall that  $e_i^{\text{basis}}$  indicates the  $i$ th standard basis vector.

PROOF. See Appendix B.2.  $\square$

Taking Lemma 1 and Lemma 2 together, one can readily see that Algorithm 1 returns an  $\varepsilon$ -optimal randomized hypothesis  $h^{\text{final}}$  with probability at least  $1 - \delta/2$ . The next step then lies in bounding the total number of samples that has been collected in Algorithm 1. Toward this end, recall that  $\bar{w}_i^T = \max_{1 \leq t \leq T} w_i^t$  for each  $i \in [k]$ . Recognizing that  $\hat{w}_i^t \leq \bar{w}_i^t$  for each  $t \in [T]$  and  $i \in [k]$ , we can bound the total sample size by

$$\begin{aligned} (\text{sample size}) \quad T_1 \sum_{i=1}^k \hat{w}_i^T + k + T \left( k \sum_{i=1}^k \bar{w}_i^T + k \right) &\leq \left( T_1 \|\bar{w}^T\|_1 + kT \|\bar{w}^T\|_1 \right) + k(T+1) \\ &\lesssim \frac{d \log \left( \frac{kd}{\varepsilon} \right) + k \log \left( \frac{k}{\delta\varepsilon} \right)}{\varepsilon^2} \cdot \|\bar{w}^T\|_1, \end{aligned} \quad (31)$$

where the last relation follows from our choices  $T \asymp \frac{\log(k/\delta)}{\varepsilon^2}$  and  $T_1 \asymp \frac{k \log(k/\varepsilon) + d \log(kd/\varepsilon) + \log(1/\delta)}{\varepsilon^2}$  (cf. line 2 of Algorithm 1) and the basic property that  $\|\bar{w}^T\|_1 \geq \sum_i w_i^1 = 1$ . Consequently, everything then comes down to bounding  $\|\bar{w}^T\|_1$ , for which we resort to the following lemma.

LEMMA 3. *Assume that lines 6-11 in Algorithm 1 are replaced with some oracle which returns a hypothesis  $h^t$  satisfies that  $L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1$  in the  $t$ th round for each  $1 \leq t \leq T$ . With probability at least  $1 - \delta/4$ , the quantity  $\|\bar{w}^T\|_1$  is bounded above by*

$$\|\bar{w}^T\|_1 \leq O \left( \log^8 \left( \frac{k}{\delta\varepsilon} \right) \right).$$

It is noteworthy that the proof of Lemma 3 is the most technically challenging part of the analysis, and we shall present this proof in Section 5.2.

Combining Lemma 3 with Equation (31) immediately reveals that, with probability at least  $1 - \delta$ , the sample complexity of Algorithm 1 is bounded by

$$O \left( \frac{d \log \left( \frac{kd}{\varepsilon} \right) + k \log \left( \frac{k}{\delta\varepsilon} \right)}{\varepsilon^2} \cdot \log^8 \left( \frac{k}{\delta\varepsilon} \right) \right),$$

as claimed in Theorem 1.

## 5.2 Proof of Lemma 3 (Controlling the Hedge Trajectory)

As alluded to previously, the proof of Lemma 3 forms the most technically challenging part of our analysis. Let us begin by introducing several convenient notation. Set  $\delta' = \delta/(32T^4k^2)$ , and let

$$\bar{j} = \left\lceil \log_2 \left( \frac{k \log^2(2)}{50(\log_2(1/\eta) + 1)^2 \log_2^2(k)} \right) \right\rceil - 2. \quad (32)$$

Define

$$\mathcal{W}_j := \{i \in [k] \mid \max_{1 \leq t \leq T} w_i^t \in (2^{-j}, 2^{-(j-1)}]\}, \quad 1 \leq j \leq \bar{j} \quad (33a)$$

$$\overline{\mathcal{W}} := [k] / \cup_j \mathcal{W}_j. \quad (33b)$$

In other words, we divide the  $k$  distributions into a logarithmic number of groups  $\{\mathcal{W}_j\}$ , where each  $\mathcal{W}_j$  consists of those distributions whose corresponding  $\max_t w_i^t$  are on the same order. The main step in establishing Lemma 3 lies in bounding the size of each  $\mathcal{W}_j$ , as summarized below.

**LEMMA 4.** *Suppose that the assumptions of Lemma 3 hold. Then with probability exceeding  $1 - 8T^4k\delta'$ ,*

$$|\mathcal{W}_j| \leq 8 \cdot 10^7 \cdot \left( (\log_2(1/\eta) + 1)^2 \log_2^2(k) (\log(k) + \log(1/\delta'))^3 (\log_2(T) + 1) \right) \cdot 2^j \quad (34)$$

holds all  $1 \leq j \leq \bar{j}$ , with  $\bar{j}$  defined in Equation (32).

In words, Lemma 4 asserts that the cardinality of each  $\mathcal{W}_j$  is upper bounded by

$$|\mathcal{W}_j| \leq \tilde{O}(2^j), \quad 1 \leq j \leq \bar{j}.$$

Importantly, this lemma tells us that, with probability at least  $1 - 8T^4k^2\delta' = 1 - \delta/4$ , one has

$$\begin{aligned} \|\bar{w}^T\|_1 &= \sum_{i=1}^k \max_{1 \leq t \leq T} w_i^t \leq k \cdot 2^{-\bar{j}-1} + \sum_{j=1}^{\bar{j}} |\mathcal{W}_j| 2^{-(j-1)} \\ &\leq k \cdot \frac{800(\log_2(1/\eta) + 1)^2 \log_2^2(k)}{k \log^2(2)} + \sum_{j=1}^{\bar{j}} |\mathcal{W}_j| 2^{-(j-1)} \\ &\leq 2 \cdot 10^8 \cdot \left( (\log_2(1/\eta) + 1)^2 \log_2^2(k) (\log(Tk) + \log(1/\delta))^3 (\log_2(T) + 1) \right), \end{aligned}$$

where the first inequality is valid since  $\max_{1 \leq t \leq T} w_i^t \leq 2^{-(j-1)}$  holds for any  $i \in \mathcal{W}_j$ . This immediately concludes the proof of Lemma 3, as long as Lemma 4 can be established.

Noteworthily, proving Lemma 4 is the most challenging part of our analysis, and we dedicate Section 5.3 to the proof of Lemma 4.

## 5.3 Proof of Lemma 4 (Bounding $|\mathcal{W}_j|$ for each $j$ )

The proof of Lemma 4 relies heavily on the concepts of “segments” introduced in Section 4.2. Throughout this section, we shall take  $\delta' = \delta/(32T^4k^2)$ , and focus on any  $j$  obeying

$$1 \leq j \leq \bar{j}. \quad (35)$$

Before continuing, we find it helpful to underscore a high-level idea: if  $|\mathcal{W}_j|$  is large, then there exist many disjoint segments, thereby requiring the total length  $T$  to be large enough in order to contain these segments. The key steps to construct such disjoint segments of interest are as follows:

- (1) Construct a suitable segment for each  $i \in \mathcal{W}_j$  (see Lemma 5);
- (2) identify sufficiently many disjoint blocks such that the segments within each block have nonempty intersection (see Lemma 6 and Figure 3);
- (3) from the above disjoint blocks, identify sufficiently many disjoint subsets such that the distributions associated with each subset can be linked with a common (sub)-segment, and that each of these (sub)-segments experiences sufficient changes between its starting and end points (see Lemma 7, Figures 4 and 6).

In the sequel, we shall present the details of our proof, which consist of multiple steps.

**5.3.1 Step 1: Showing Existence of a Segment for each Distribution in  $\mathcal{W}_j$ .** Recall that  $\mathcal{W}_j$  contains those distributions whose corresponding weight iterates obey  $\max_{1 \leq t \leq T} w_i^t \in (2^{-j}, 2^{-j+1}]$  (cf. Equation (33a)). As it turns out, for any  $i \in \mathcal{W}_j$ , one can find an  $(\frac{1}{2^{j+1}}, \frac{1}{2^{j+2}}, \log(2))$ -segment, as stated in the lemma below. This basic fact allows one to link each distribution in  $\mathcal{W}_j$  with a segment of suitable parameters.

**LEMMA 5.** *For each  $i \in \mathcal{W}_j$ , there exists  $1 \leq s_i < e_i \leq T$ , such that*

$$\frac{1}{2^{j+2}} < w_i^{s_i} \leq \frac{1}{2^{j+1}}, \quad w_i^{e_i} > \frac{1}{2^j}, \quad \text{and} \quad w_i^t > \frac{1}{2^{j+2}} \quad \forall t \in [s_i, e_i]. \quad (36)$$

*In other words, there exists a  $(\frac{1}{2^{j+1}}, \frac{1}{2^{j+2}}, \log(2))$ -segment  $(s_i, e_i)$  with the index set as  $\{i\}$  (see Definition 1).*

**PROOF.** From the definition Equation (33a) of  $\mathcal{W}_j$ , it is straightforward to find a time point  $e_i$  obeying  $w_i^{e_i} > \frac{1}{2^j}$ . It then remains to identify a valid point  $s_i$ . To this end, let us define

$$\tau = \max \{t \mid t \leq e_i, w_i^t \leq 2^{-(j+2)}\},$$

which is properly defined since  $w_i^1 = 1/k \leq 2^{-(j+2)}$  (see Equation (35)). With this choice in mind, we have

$$w_i^t > 2^{-(j+2)}, \quad \forall t \text{ obeying } \tau + 1 \leq t \leq e_i.$$

In addition, it follows from the update rule (cf. lines 5 and 15 of Algorithm 1) that

$$\begin{aligned} \log(w_i^{t+1}/w_i^t) &= \log(W_i^{t+1}/W_i^t) - \log\left(\sum_j W_j^{t+1} / \sum_j W_j^t\right) \\ &\leq \eta - \log\left(\sum_j W_j^{t+1} / \sum_j W_j^t\right) \leq 2\eta \leq 1/10, \end{aligned}$$

where the last inequality results from our choice of  $\eta$ . This in turn allows us to show that

$$w_i^{\tau+1} \leq w_i^\tau \exp(1/10) \leq \frac{1}{2^{j+2}} \cdot \exp(1/10) \leq \frac{1}{2^{j+1}}. \quad (37)$$

As a result, it suffices to choose  $s_i = \tau + 1$ , thus concluding the proof.  $\square$

**5.3.2 Step 2: Constructing Disjoint Segments with Good Coverage.** While Lemma 5 justifies the existence of suitable segments  $\{(s_i, e_i)\}$  associated with each distribution in  $\mathcal{W}_j$ , we need to divide (a nontrivial subset of) them into certain disjoint blocks, where the segments in each block have at least one common inner points. This is accomplished in the following lemma.

**LEMMA 6.** *Recall the definition of  $\mathcal{W}_j$  in Equation (33a). For each  $i \in \mathcal{W}_j$ , denote by  $(s_i, e_i)$  the segment identified in Lemma 5. Then there exist a group of disjoint subsets  $\{\mathcal{W}_j^p\}_{p=1}^P$  of  $\mathcal{W}_j$  obeying*

- (i)  $\mathcal{W}_j^p \subseteq \mathcal{W}_j$ ,  $\mathcal{W}_j^p \cap \mathcal{W}_j^{p'} = \emptyset$ ,  $\forall p \neq p'$ ;
- (ii)  $\sum_{p=1}^P |\mathcal{W}_j^p| \geq \frac{|\mathcal{W}_j|}{3(\log_2(T)+1)}$ ;
- (iii) Let  $\tilde{s}_p = \min_{i \in \mathcal{W}_j^p} s_i$  and  $\tilde{e}_p = \max_{i \in \mathcal{W}_j^p} e_i$  for each  $1 \leq p \leq P$ . One has  $1 \leq \tilde{s}_1 < \tilde{e}_1 \leq \tilde{s}_2 < \tilde{e}_2 \leq \dots \leq \tilde{s}_P < \tilde{e}_P \leq T$  and  $\max_{i \in \mathcal{W}_j^p} s_i \leq \min_{i \in \mathcal{W}_j^p} e_i$  for each  $1 \leq p \leq P$ .

PROOF. See Appendix B.3.  $\square$

In words, Lemma 6 reveals the existence a collection of *disjoint* subsets of  $\mathcal{W}_j$  such that (a) they account for a sufficiently large fraction of the indices contained in  $\mathcal{W}_j$ , and (b) the segments in each subset  $\mathcal{W}_j^p$  share at least one common inner point.

Thus far, each of the segments constructed above is associated with a single distribution in  $\mathcal{W}_j$ . Clearly, it is likely that many of these segments might have non-trivial overlap; in other words, many of them might have shared sub-segments. What we intend to do next is to further group the indices in  $\{\mathcal{W}_j^p\}$  into disjoint subgroups, and identify a common (sub)-segment for each of these subgroups. What remains unclear, however, is whether each of these (sub)-segments experiences sufficient weight changes between its starting and end points. We address these in the following lemma.

LEMMA 7. Recall the definition of  $\mathcal{W}_j$  in Equation (33a). Then there exists a group of subsets  $\{\mathcal{V}_j^n\}_{n=1}^N$  of  $\mathcal{W}_j$  satisfying the following properties:

- (i)  $\mathcal{V}_j^n \subseteq \mathcal{W}_j$ ,  $\mathcal{V}_j^n \cap \mathcal{V}_j^{n'} = \emptyset$ ,  $\forall n \neq n'$ ;
- (ii)  $\sum_{n=1}^N |\mathcal{V}_j^n| \geq \frac{|\mathcal{W}_j|}{24 \log_2(k) (\log_2(T)+1)}$ ;
- (iii) There exist  $1 \leq \hat{s}_1 < \hat{e}_1 \leq \hat{s}_2 < \hat{e}_2 \leq \dots \leq \hat{s}_N < \hat{e}_N \leq T$ , and  $\{g_n\}_{n=1}^N \in [1, \infty)^N$ , such that for each  $1 \leq n \leq N$ ,  $(\hat{s}_n, \hat{e}_n)$  is a  $(2^{-(j+1)} g_n |\mathcal{V}_j^n|, 2^{-(j+2)} |\mathcal{V}_j^n|, \frac{\log(2)}{2 \log_2(k)})$ -segment with index set as  $\mathcal{V}_j^n$ . That is, the following properties hold for each  $1 \leq n \leq N$ :
  - (a)  $\frac{g_n |\mathcal{V}_j^n|}{2^{j+2}} \leq \sum_{i \in \mathcal{V}_j^n} w_i^{\hat{s}_n} \leq \frac{g_n |\mathcal{V}_j^n|}{2^{j+1}}$ ;
  - (b)  $\frac{g_n |\mathcal{V}_j^n|}{2^j} \cdot \exp\left(\frac{\log(2)}{2 \log_2(k)}\right) \leq \sum_{i \in \mathcal{V}_j^n} w_i^{\hat{e}_n}$ ;
  - (c)  $\sum_{i \in \mathcal{V}_j^n} w_i^t \geq \frac{|\mathcal{V}_j^n|}{2^{j+2}}$  for any  $t$  obeying  $\hat{s}_n \leq t \leq \hat{e}_n$ .

PROOF. See Appendix B.4.  $\square$

*Remark 3.* The endpoints  $\{\hat{s}_n\}$  and  $\{\hat{e}_n\}$  mentioned above are carefully constructed in the proof. The quantities  $\{g_n\}$  are only implicitly defined here, since their precise values (except the property  $g_n \geq 1$ ) are not needed in the subsequent analysis; the interested reader can find more precise details about  $\{g_n\}$  in the proof.

In brief, Lemma 7 identifies a collection of subsets  $\{\mathcal{V}_j^n\}_{n=1}^N$  of  $\mathcal{W}_j$  enjoying the following useful properties:

- They are disjoint (see property (i) in the lemma); as explained in Section 4.2.2, having disjoint subsets that can cover a good fraction of  $\mathcal{W}_j$  facilitates analysis.
- While they might be unable to fully cover the set  $\mathcal{W}_j$ , these subsets taken collectively still cover a highly nontrivial fraction (i.e., at least  $\frac{1}{\text{polylog}(k,T)}$ ) of the elements in  $\mathcal{W}_j$ .
- Each subset  $\mathcal{V}_j^n$  is linked with a suitable segment, whose associated weights have increased sufficiently from its starting point to its end point.

We pause to point out some key ideas underlying the proof of Lemma 7. Our construction is built upon a collection of segments with common end points, as constructed in the proof of Lemma 6. Assume that these  $m$  segments have a shared starting point  $e_0$ , as well as end points  $e_0 < e_1 \leq e_2 \leq \dots \leq e_m$ . For simplicity, one can think of the case with  $e_i = i$  for  $0 \leq i \leq m$ . We then divide each segment into sub-segments with length of some power of 2, on the basis of the binary form of the associated index. Moreover, each segment contains at most one sub-segment of each length. This way, each segment provides at least one significant sub-segment, and the set of sub-segments with the same length is well aligned, thereby satisfying the key properties in the conditions stated in Lemma 7.

**5.3.3 Step 3: Bounding the Length of Segments.** In this step, we turn attention to the length of segments, unveiling the interplay between the segment length and certain sub-optimality gaps.

Define

$$v^t := L(h^t, w^t) - \text{OPT} \quad \text{with } \text{OPT} := \min_{\pi \in \Delta(\mathcal{H})} \max_{1 \leq i \leq k} L(h_\pi, e_i^{\text{basis}}), \quad (38)$$

with  $e_i^{\text{basis}}$  the  $i$ th standard basis vector. The following lemma assists in bounding the length of the segments defined in Definition 1.

**LEMMA 8.** *Let  $j_{\max} = \lfloor \log_2(1/\eta) \rfloor + 1$ . Assume the conditions in Lemma 3 hold. Suppose  $(t_1, t_2)$  is a  $(p, q, x)$ -segment satisfying  $p \geq 2q > 0$ . Then one has*

$$t_2 - t_1 \geq \frac{x}{2\eta}. \quad (39)$$

Moreover, if

$$\frac{qx^2}{50(\log_2(1/\eta) + 1)^2} \geq \frac{1}{k}, \quad (40)$$

holds, then with probability exceeding  $1 - 6T^4k\delta'$ , at least one of the following two claims holds:

(a) the length of the segment satisfies

$$t_2 - t_1 \geq \frac{qx^2}{200(\log_2(1/\eta) + 1)^2\eta^2}. \quad (41)$$

(b) the quantities  $\{v^t\}$  obey

$$4 \sum_{\tau=t_1}^{t_2-1} (-v^\tau + \varepsilon_1) \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2\eta}. \quad (42)$$

**PROOF.** See Appendix B.5. □

In words, Lemma 8 reveals that

- in general the length of a segment scales at least linearly in  $1/\eta$ ;
- if the parameter  $q$  of a segment (i.e., some lower bound on the weights of interest within this segment) is sufficiently large, then either the length of this segment scales at least *quadratically* in  $1/\eta$ , or the sum of certain sub-optimality gaps needs to be large enough.

While the proof of Lemma 8 – which is based on careful analysis of  $\text{KL}(w^{t_1} || w^{t_2})$  – is postponed to Appendix B.5, we take a moment to describe some high-level key ideas for an important case when  $-v^\tau = O(\varepsilon_1)$  for all  $\tau \geq 1$ .

- (1) First, the existence of the  $(p, q, x)$ -segments implies that  $\text{KL}(w^{t_1} \| w^{t_2}) \geq \Omega(qx^2)$ ;
- (2) In addition,  $\text{KL}(w^{t_1} \| w^{t_2})$  is bounded by  $O((t_2 - t_1)\varepsilon^2)$  because of the optimality condition that  $L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1$ .

It is also worth noting that our analysis only works for Hedge algorithm (i.e., online mirror descent with entropy regularization). It is unclear whether this result can be extended to accommodate other first-order optimization algorithms.

**5.3.4 Step 4: Putting all this Together.** With the above lemmas in place, we are positioned to establish Lemma 4. In what follows, we denote by  $\{\mathcal{V}_j^n\}_{n=1}^N$  and  $\{(\hat{s}_n, \hat{e}_n)\}_{n=1}^N$  the construction in Lemma 7.

To begin with, it is observed that: for any  $1 \leq j \leq \bar{j}$  (with  $\bar{j}$  defined in Equation (32)), one has

$$2^{-(j+2)}|\mathcal{V}_j^n| \cdot \frac{\log^2(2)}{50(\log_2(1/\eta) + 1)^2 \log_2^2(k)} \geq 2^{-\bar{j}+2} \cdot \frac{\log^2(2)}{50(\log_2(1/\eta) + 1)^2 \log_2^2(k)} \geq \frac{1}{k}. \quad (43)$$

Recall that  $v^\tau \leq \varepsilon_1$  (see Equation (80)). Combining this fact with Lemma 8 (by setting  $q = 2^{-(j+2)}|\mathcal{V}_j^n|$  and  $x = \frac{\log(2)}{\log_2(k)}$ ) reveals that, for each  $1 \leq n \leq N$ ,

$$T\eta + \left\{ 4T\varepsilon_1 + 4 \sum_{t=1}^T (-v^t) \right\} \geq \sum_{n=1}^N (\hat{e}_n - \hat{s}_n)\eta + 4 \sum_{n=1}^N \sum_{\tau=\hat{s}_n}^{\hat{e}_n-1} (-v^\tau + \varepsilon_1), \quad (44)$$

$$\geq \frac{2^{-(j+2)} \sum_{n=1}^N |\mathcal{V}_j^n| \log^2(2)}{800 \log_2^2(k) (\log_2(1/\eta) + 1)^2 \eta}, \quad (45)$$

where the first inequality results from the disjoint nature of the segments  $\{(\hat{s}_n, \hat{e}_n)\}_{1 \leq n \leq N}$ , and the second inequality comes from Lemma 8. Moreover, it follows from the convergence of Hedge algorithm (see Equation (85)) that

$$\sum_{t=1}^T (-v^t) \leq \frac{\log(k)}{\eta} + \eta T + 4\sqrt{T \log(1/\delta')}, \quad (46)$$

which taken together with Equation (45) and the choice  $\varepsilon_1 = \eta$  gives

$$\begin{aligned} \sum_{n=1}^N |\mathcal{V}_j^n| &\leq \frac{3200\eta(\log_2(1/\eta) + 1)^2 \log_2^2(k) \cdot 2^{j+2}}{\log^2(2)} \cdot \left( \frac{\log(k)}{\eta} + \eta T + 4\sqrt{T \log(1/\delta')} \right) \\ &\quad + \frac{4000T(\log_2(1/\eta) + 1)^2 \log_2^2(k) \cdot 2^{j+2}\eta^2}{\log^2(2)}. \end{aligned} \quad (47)$$

To finish up, it follows from Property (ii) of Lemma 7 that

$$\begin{aligned} |\mathcal{W}_j| &\leq 24 \log_2(k) (\log_2(T) + 1) \left( \sum_{n=1}^N |\mathcal{V}_j^n| \right) \\ &\leq 8 \cdot 10^7 \cdot \left( (\log_2(1/\eta) + 1)^2 \log_2^2(k) (\log_2(k) + \log(1/\delta'))^3 (\log_2(T) + 1) \right) \cdot 2^j \end{aligned}$$

for any  $1 \leq j \leq \bar{j}$ . The proof of Lemma 4 is thus completed by recalling that  $\delta' = \frac{\delta}{4(T+k+1)}$ .

## 6 Limitations of Proper Learning?

Given that the best-known sample complexities prior to our work were derived for algorithms that either output randomized hypotheses or invoke majority votes, Awasthi et al. [4] raised the question about how the sample complexity is impacted if only deterministic (or “proper”) hypotheses are permitted as the output of the learning algorithms. As it turns out, the restriction to proper learning algorithms substantially worsens the sample efficiency, as revealed by the following theorem.

**THEOREM 2.** *Assume that  $d \geq 2 \log(8k)$ . Consider any  $\varepsilon \in (0, 1/100)$ , and let  $N_0 = \frac{2^d - 1}{k}$ . One can find*

- a hypothesis class  $\mathcal{H}$  containing at most  $kN_0 + 1$  hypothesis,
- a collection of  $k$  distributions  $\mathcal{D} := \{\mathcal{D}_i\}_{i=1}^k$ ,
- a loss function  $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \rightarrow [-1, 1]$ ,

such that: for any algorithm  $\mathcal{G}$  that finds  $h \in \mathcal{H}$  obeying

$$\max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] \leq \min_{h' \in \mathcal{H}} \max_{i \in [k]} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h', (x, y))] + \varepsilon \quad (48)$$

with probability exceeding  $3/4$ , its sample size — denoted by  $M(\mathcal{G})$  — must exceed  $\mathbb{E}[M(\mathcal{G})] \geq \frac{dk}{240000\varepsilon^2}$ .

In words, the sample complexity when a deterministic output from  $\mathcal{H}$  is required scales at least with  $\Omega(\frac{dk}{\varepsilon^2})$ , which is considerably larger than the sample complexity  $\widetilde{O}(\frac{d+k}{\varepsilon^2})$  achievable via Algorithm 1 (which outputs a randomized hypothesis). This demonstrates the benefits of randomization, or more broadly, the necessity of improper learning. The proof of this theorem is deferred to Appendix C.

## 7 Extension: Learning Rademacher Classes

In this section, we adapt our algorithm and theory to accommodate MDL for Rademacher classes. Note that when learning Rademacher classes, the hypotheses in  $\mathcal{H}$  do not need to be binary-valued.

### 7.1 Preliminaries: Rademacher Complexity

Let us first introduce the formal definition of the Rademacher complexity; more detailed introduction can be found in, e.g., Shalev-Shwartz and Ben-David [38].

*Definition 2 (Rademacher complexity).* Given a distribution  $\mathcal{D}$  supported on  $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$  and a positive integer  $n$ , the Rademacher complexity is defined as

$$\text{Rad}_n(\mathcal{D}) := \mathbb{E}_{\{z_i\}_{i=1}^n} \left[ \mathbb{E}_{\{\sigma_i\}_{i=1}^n} \left[ \max_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n \sigma_i \ell(h, z_i) \right] \right], \quad (49)$$

where  $\{z_i\}_{i=1}^n$  are drawn independently from  $\mathcal{D}$ , and  $\{\sigma_i\}_{i=1}^n$  are i.i.d. Rademacher random variables obeying  $\mathbb{P}\{\sigma_i = 1\} = \mathbb{P}\{\sigma_i = -1\} = 1/2$  for each  $1 \leq i \leq n$ .

Next, we would like to make an assumption concerning the Rademacher complexity of mixtures of distributions. Denoting by  $\mathcal{D}(w)$  the mixed distribution

$$\mathcal{D}(w) := \sum_{i=1}^k w_i \mathcal{D}_i, \quad (50)$$

for any probability vector  $w = [w_i]_{1 \leq i \leq k} \in \Delta(k)$ , we can state our assumption as follows:

**ASSUMPTION 1.** *For each  $n \geq 1$ , there exists a quantity  $C_n > 0$  (known to the learner) such that*

$$C_n \geq \sup_{w \in \Delta(k)} \text{Rad}_n(\mathcal{D}(w)). \quad (51)$$

For instance, if the hypothesis class  $\mathcal{H}$  obeys  $\text{VC-dim}(\mathcal{H}) \leq d$ , then it is well-known that Assumption 1 holds with the choice (see, e.g., Mohri et al. [29])

$$C_n = \sqrt{\frac{2d\log(en/d)}{n}}.$$

*Remark 4.* One might raise a natural question about Assumption 1: can we use  $\tilde{C}_n := \max_{i \in [k]} \text{Rad}_n(\mathcal{D}_i)$  instead of  $C_n$  without incurring a worse sample complexity? The answer is, however, negative. In fact, the Rademacher complexity  $\text{Rad}_n(\mathcal{D}(w))$  is not convex in  $w$ , and hence we fail to use  $\max_i \text{Rad}_n(\mathcal{D}_i)$  to bound  $\max_{w \in \Delta(k)} \text{Rad}_n(\mathcal{D}(w))$ . The interested reader is referred to Appendix D.4 for more details.

To facilitate analysis, we find it helpful to introduce another notion called weighted Rademacher complexity.

*Definition 3 (Weighted Rademacher complexity).* Given a collection of distributions  $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^k$  and a set of positive integers  $\{n_i\}_{i=1}^k$ , the weighted (average) Rademacher complexity is defined as

$$\widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} := \mathbb{E}_{\{z_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \frac{1}{\sum_{i=1}^k n_i} \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{n_i} \sigma_i^j \ell(h, z_i^j) \right] \right], \quad (52)$$

where  $\{\{z_i^j\}_{j=1}^{n_i}\}_{i=1}^k$  are independently generated with each  $z_i^j$  drawn from  $\mathcal{D}_i$ , and  $\{\{\sigma_i^j\}_{j=1}^{n_i}\}_{i=1}^k$  are independent Rademacher random variables obeying  $\mathbb{P}\{\sigma_i^j = 1\} = \mathbb{P}\{\sigma_i^j = -1\} = 1/2$ . Throughout the rest of this article, we shall often abbreviate  $\widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} = \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k}(\mathcal{D})$ .

The following two lemmas provide useful properties about the weighted Rademacher complexity.

LEMMA 9. *For any two groups of positive integers  $\{n_i\}_{i=1}^k$  and  $\{m_i\}_{i=1}^k$ , it holds that*

$$\begin{aligned} \left( \sum_{i=1}^k n_i \right) \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} &\leq \left( \sum_{i=1}^k (m_i + n_i) \right) \widetilde{\text{Rad}}_{\{m_i + n_i\}_{i=1}^k} \\ &\leq \left( \sum_{i=1}^k n_i \right) \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + \left( \sum_{i=1}^k m_i \right) \widetilde{\text{Rad}}_{\{m_i\}_{i=1}^k}. \end{aligned} \quad (53)$$

PROOF. See Appendix D.2. □

LEMMA 10. *Consider any  $\{n_i\}_{i=1}^k$  obeying  $n_i \geq 12 \log(2k)$  for each  $i \in [k]$ . By taking  $w \in \Delta(k)$  with  $w_i = \frac{n_i}{\sum_{l=1}^k n_l}$ , one has*

$$\widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} \leq 72 \text{Rad}_{\sum_{i=1}^k n_i}(\mathcal{D}(w)).$$

PROOF. See Appendix D.3. □

## 7.2 Algorithm and Sample Complexity

We are now positioned to introduce our algorithm that learns a Rademacher class in the presence of multiple distributions, which is also based on a Hedge-type strategy to learn a convex-concave game; see Algorithm 3 for full details. Its main distinction from Algorithm 1 lies in the subroutine to learn  $h^t$  (see lines 7-12 in Algorithm 3) as well as the choice of  $T_1$  (see line 2 in Algorithm 3).

**ALGORITHM 3:** Hedge for multi-distribution learning on Rademacher Classes (MDL-Hedge-Rad)

---

**1** **input:**  $k$  data distributions  $\{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_k\}$ , hypothesis class  $\mathcal{H}$ , target accuracy level  $\varepsilon$ , target success rate  $1 - \delta$ , quantities  $\{C_n\}_{n \geq 1}$  as in Assumption 1.

**2** **hyper-parameters:** stepsize  $\eta = \frac{1}{100}\varepsilon$ , number of rounds  $T = \frac{20000 \log(\frac{k}{\delta})}{\varepsilon^2}$ , auxiliary accuracy level  $\varepsilon_1 = \frac{1}{100}\varepsilon$ , auxiliary sub-sample-size  $T_1 = \min\left\{t \geq \frac{4000(k \log(k/\varepsilon_1) + \log(1/\delta))}{\varepsilon_1^2} \mid C_t \leq \frac{\varepsilon_1}{4800}\right\}$ .

**3** **initialization:** for all  $i \in [k]$ , set  $W_i^1 = 1$ ,  $\widehat{w}_i^0 = 0$  and  $n_i^0 = 0$ ;  $\mathcal{S} = \emptyset$ .

**4** draw  $[12 \log(2k)]$  samples from  $\mathcal{D}_i$  for each  $i$ , and add these samples to  $\mathcal{S}$ .

**5** **for**  $t = 1, 2, \dots, T$  **do**

**6** set  $w^t = [w_i^t]_{1 \leq i \leq k}$  and  $\widehat{w}^t = [\widehat{w}_i^t]_{1 \leq i \leq k}$ , with  $w_i^t \leftarrow \frac{W_i^t}{\sum_j W_j^t}$  and  $\widehat{w}_i^t \leftarrow \widehat{w}_i^{t-1}$  for all  $i \in [k]$ .

*/\* recompute  $\widehat{w}^t$  & draw new samples for  $\mathcal{S}_w$  only if  $w^t$  changes sufficiently. \*/*

**7** **if** there exists  $j \in [k]$  such that  $w_j^t \geq 2\widehat{w}_j^{t-1}$  **then**

**8**  $\widehat{w}_i^t \leftarrow \max\{w_i^t, \widehat{w}_i^{t-1}\}$  for all  $i \in [k]$ ;

**9** **for**  $i = 1, \dots, k$  **do**

**10**  $n_i^t \leftarrow \lceil T_1 \widehat{w}_i^t \rceil$ ;

**11** draw  $n_i^t - n_i^{t-1}$  independent samples from  $\mathcal{D}_i$ , and add these samples to  $\mathcal{S}$ .

*/\* estimate the near-optimal hypothesis for weighted data distributions. \*/*

**12** compute  $h^t \leftarrow \arg \min_{h \in \mathcal{H}} \widehat{L}(h, w^t)$ , where

$$\widehat{L}^t(h, w^t) := \sum_{i=1}^k \frac{w_i^t}{n_i^{t,\text{rad}}} \cdot \sum_{j=1}^{n_i^{t,\text{rad}}} \ell(h, (x_{i,j}, y_{i,j})) \quad (55)$$

with  $n_i^{t,\text{rad}} = \min\{[T_1 w_i^t + 12 \log(2k)], T_1\}$  and  $(x_{i,j}, y_{i,j})$  being the  $j$ -th datapoint from  $\mathcal{D}_i$  in  $\mathcal{S}$ .

*/\* estimate the loss vector and execute weighted updates. \*/*

**13**  $\overline{w}_i^t \leftarrow \max_{1 \leq r \leq t} w_i^r$  for all  $i \in [k]$ .

**14** **for**  $i = 1, \dots, k$  **do**

**15** draw  $\lceil k\overline{w}_i^t \rceil$  independent samples – denoted by  $\{(x_{i,j}^t, y_{i,j}^t)\}_{j=1}^{\lceil k\overline{w}_i^t \rceil}$  – from  $\mathcal{D}_i$ , and set

$$\widehat{r}_i^t = \frac{1}{\lceil k\overline{w}_i^t \rceil} \sum_{j=1}^{\lceil k\overline{w}_i^t \rceil} \ell(h^t, (x_{i,j}^t, y_{i,j}^t)); \quad (55)$$

**16** update the weight as  $W_i^{t+1} = W_i^t \exp(\eta \widehat{r}_i^t)$ . // Hedge updates.

**17** **output:** a randomized hypothesis  $h^{\text{final}}$  as a uniform distribution over  $\{h_i^t\}_{i=1}^T$ .

---

More precisely, to compute the estimator  $\widehat{L}^t(h, w^t)$  for  $L(h, w^t)$ , instead of using the first  $n_i^t$  samples from  $\mathcal{D}_i$  for each  $i \in [k]$ , we choose to use the first

$$n_i^{t,\text{rad}} = \min\{[T_1 w_i^t + 12 \log(2k)], T_1\},$$

samples from  $\mathcal{D}_i$  for each  $i$ , where  $T_1$  is taken to be

$$T_1 = \min\left\{t \geq \frac{4000(k \log(k/\varepsilon_1) + \log(1/\delta))}{\varepsilon_1^2} \mid C_t \leq \frac{\varepsilon_1}{4800}\right\}. \quad (54)$$

Here,  $T_1$  needs to take advantage of the quantities  $\{C_n\}$  (cf. Assumption 1) that upper bound the associated Rademacher complexity.

Equipped with this algorithm, we are ready to establish the following theoretical guarantees.

**THEOREM 3.** Suppose Assumption 1 holds. With probability at least  $1 - \delta$ , the output  $h^{\text{final}}$  returned by Algorithm 3 satisfies

$$\max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, h^{\text{final}}} [\ell(h^{\text{final}}, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon. \quad (56)$$

In particular, the total number of samples collected by Algorithm 3 is bounded by

$$\left( \frac{k}{\varepsilon^2} + \min \{n \mid C_n \leq c_1 \varepsilon\} \right) \text{poly log} \left( k, \frac{1}{\varepsilon}, \frac{1}{\delta} \right),$$

with probability exceeding  $1 - \delta$ , where  $c_1 > 0$  is some sufficiently small constant, and  $T_1$  is defined in Equation (54).

In contrast to the VC classes, the sample complexity for learning Rademacher classes entails a term related to the Rademacher complexity — namely,  $\min\{n \mid C_n \leq c_1 \varepsilon\}$  — as opposed to the term  $d/\varepsilon^2$  concerning the VC-dimension. When it comes to the special case where  $\text{VC-dim}(\mathcal{H}) \leq d$ , taking  $C_n \leq \sqrt{\frac{2d \log(en/d)}{n}}$  in Equation (54) leads to  $T_1 = \tilde{O}(\frac{d+k}{\varepsilon^2})$ , which recovers the sample complexity bound of  $\tilde{O}(\frac{d+k}{\varepsilon^2})$  derived in Theorem 1 for VC classes.

**PROOF OF THEOREM 3.** In view of Lemma 2 and Lemma 3, it boils down to showing that running Algorithm 3 results in  $L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1$  for any  $1 \leq t \leq T$ , a property that holds with probability at least  $1 - \delta/4$ . To accomplish this, we have the lemma below.

**LEMMA 11.** Suppose Assumption 1 holds. With probability at least  $1 - \delta/4$ , the iterates of Algorithm 3 satisfy

$$L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1, \quad (57)$$

for any  $1 \leq t \leq T$ .

The proof of Lemma 11 is postponed to Appendix D. Combine this with Lemma 2 and Lemma 3 to show that: the total number of samples collected in Algorithm 3 is upper bounded by

$$T_1 \text{poly log} \left( k, \frac{1}{\varepsilon}, \frac{1}{\delta} \right) \leq \left( \frac{k}{\varepsilon^2} + \min \left\{ n \mid C_n \leq \frac{1}{4800} \varepsilon \right\} \right) \text{poly log} \left( k, \frac{1}{\varepsilon}, \frac{1}{\delta} \right),$$

as claimed.  $\square$

## 8 Extension: Multi-loss Multi-distribution Learning

In this section, we consider an extension of the MDL problem to the multi-loss (or multi-objective) setting.

**Problem setting.** Formally, consider again the hypothesis class  $\mathcal{H}$  and the  $k$  distributions of interest  $\{\mathcal{D}_i\}_{i=1}^k$ . Suppose that there are  $R \geq 2$  loss functions  $\mathcal{L} := \{\ell^j\}_{j=1}^R$  to consider, where  $\ell^j : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \rightarrow [-1, 1]$  denotes the  $j$ th loss function of interest. The goal is to find a (possibly randomized) hypothesis  $\hat{h} \in \mathcal{H}$  such that

$$\max_{1 \leq i \leq k} \max_{\ell \in \mathcal{L}} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, \hat{h}} [\ell(\hat{h}, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k, \ell \in \mathcal{L}} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon. \quad (58)$$

Evidently, this formulation is a natural extension of the single-loss MDL problem. It is particularly relevant in scenarios where multiple criteria need to be satisfied or optimized simultaneously, such as those involving fairness and robustness. It has also been shown that this extension could be helpful toward solving the multicalibration problem [21].

**Additional notation.** Let us introduce one more notation to be used throughout. For any  $i \in [k]$  and  $\ell \in \mathcal{L}$ , we denote by  $L_i^\ell(h)$  the expected loss of  $h$  w.r.t.  $\mathcal{D}_i$  and  $\ell$ , that is

$$L_i^\ell(h) := \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))]. \quad (59)$$

Then for any  $u \in \Delta(\mathcal{D} \times \mathcal{L})$ , define

$$L(h, u) := \sum_{i,\ell} u_{i,\ell} L_i^\ell(h). \quad (60)$$

**The proposed algorithm and main theorem.** As it turns out, Algorithm 1 can be extended to tackle the above multi-loss multi-distribution learning problem. The full algorithm is presented in Algorithm 4. As a key distinction from Algorithm 1, we need to maintain a Hedge-type algorithm over  $\mathcal{D} \times \mathcal{L}$  in an attempt to solve the following problem:

$$\max_{u \in \Delta(\mathcal{D} \times \mathcal{L})} \min_{h \in \mathcal{H}} L(h, u).$$

Akin to Algorithm 1, in each round  $t$ , we maintain a dataset  $\mathcal{S}$  to construct estimation of  $L(h, u^t)$  for all  $h \in \mathcal{H}$ , which help us to compute the optimal response  $h^t$  with respect to the current weight  $u^t$ . Then we query a small number (at most  $\tilde{O}(k)$ ) of new samples to assist in estimating each  $L_i^\ell(h^t)$ . Notably, we need to choose  $T_1$  to be larger than the case of Algorithm 1, in order to apply the union bound argument. Given that  $u^t$  is in the  $kR$ -dimensional simplex  $\Delta(\mathcal{D} \times \mathcal{L})$ , applying the union bound uniformly over all possible choices of  $u^t$  would lead to an undesirable factor of  $\tilde{O}(kR)$ . Instead of this analysis strategy, we consider controlling an upper bound on the error, which allows us to pay a smaller factor of  $O(k \log(R))$  instead.

The following theorem establishes the sample complexity of Algorithm 4.

**THEOREM 4.** *There exists an algorithm (see Algorithm 4 for details) such that: with probability exceeding  $1 - \delta$ , the randomized hypothesis  $h^{\text{final}}$  returned by this algorithm achieves*

$$\max_{1 \leq i \leq k} \max_{\ell \in \mathcal{L}} \mathbb{E}_{(x,y) \sim \mathcal{D}_i, h^{\text{final}}} [\ell(h^{\text{final}}, (x, y))] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k, \ell \in \mathcal{L}} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] + \varepsilon,$$

provided that the total sample size exceeds

$$\frac{(d + k \log(R)) \min\{\log(R), k\}}{\varepsilon^2} \text{poly log}\left(k, d, \frac{1}{\varepsilon}, \frac{1}{\delta}, \log(R)\right). \quad (61)$$

The full proof of Theorem 4 is provided in Appendix E. Most parts of the proof are identical to that of Theorem 1, except that: (a) the new method in estimating  $\{L(h, u^t)\}_{h \in \mathcal{H}}$  mentioned above (see also Lemma 22); (b) a more refined analysis of the trajectory bound to remove some  $\text{poly}(\log(R))$  factors (see Lemma 24).

In comparison to the single-loss case (see Theorem 1), the multi-loss version of our algorithm only incurs an additional price of  $\text{poly}(\log(R))$  factors, which is particularly appealing when the number  $R$  of loss functions is no greater than a polynomial function in  $d, k$ , and so on.

## 9 Discussion

In this article, we have settled the problem of achieving optimal sample complexity in multi-distribution learning, assuming the availability of adaptive (or on-demand) sampling. We have put forward a novel oracle-efficient algorithm that provably attains a sample complexity of  $\tilde{O}\left(\frac{d+k}{\varepsilon^2}\right)$  for VC classes, which matches the best-known lower bound up to some logarithmic factor. From the technical perspective, the key novelty of our analysis lies in carefully bounding the trajectory of the Hedge algorithm on a convex-concave optimization problem. We have further unveiled the

**ALGORITHM 4:** Hedge for multi-loss multi-distribution learning (MLMDL-Hedge-VC)

---

**1 input:**  $k$  data distributions  $\{\mathcal{D}_i\}_{i=1}^k$ , loss function class  $\mathcal{L} = \{\ell^j\}_{j=1}^R$ , hypothesis class  $\mathcal{H}$ , target accuracy level  $\epsilon$ , target success rate  $1 - \delta$ .

**2 hyper-parameters:** stepsize  $\eta = \frac{1}{100}\epsilon$ , number of rounds  $T = \frac{20000 \log(\frac{kR}{\delta\epsilon})}{\epsilon^2}$ , auxiliary accuracy level  $\epsilon_1 = \frac{1}{100}\epsilon$ , auxiliary sub-sample-size  $T_1 := \frac{40000 \left( k \log(\frac{kR}{\epsilon_1^2}) + d \log \left( \left( \frac{kd}{\epsilon_1^2} \right) + \log(\frac{1}{\delta}) \right) \right)}{\epsilon_1^2}$ .

**3 initialization:** for all  $i \in [k]$  and  $\ell \in \mathcal{L}$ , set  $U_{i,\ell}^1 = 1$ ,  $\hat{w}_i^0 = 0$  and  $n_i^0 = 0$ ;  $\mathcal{S} = \emptyset$ .

**4 for**  $t = 1, 2, \dots, T$  **do**

5 set  $u^t = [u_{i,\ell}^t]_{1 \leq i \leq k, \ell \in \mathcal{L}}$  with  $u_{i,\ell}^t \leftarrow \frac{U_{i,\ell}^t}{\sum_{i' \in [k], \ell' \in \mathcal{L}} U_{i',\ell'}^t}$ .

6 set  $\hat{w}^t = [\hat{w}_i^t]_{1 \leq i \leq k}$  with  $\hat{w}_i^t \leftarrow \hat{w}_i^{t-1}$  for all  $i \in [k]$ .  
/\* recompute  $\hat{w}^t$  & draw new samples for  $\mathcal{S}$  only if  $w^t$  changes sufficiently. \*/

7 **if** there exists  $j \in [k]$  such that  $\sum_{\ell \in \mathcal{L}} u_{j,\ell}^t \geq 2\hat{w}_j^{t-1}$  **then**

8     $\hat{w}_i^t \leftarrow \max \left\{ \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t, \hat{w}_i^{t-1} \right\}$  for all  $i \in [k]$ ;

9    **for**  $i = 1, \dots, k$  **do**

10      $n_i^t \leftarrow \lceil T_1 \hat{w}_i^t \rceil$ ;

11     draw  $n_i^t - n_i^{t-1}$  independent samples from  $\mathcal{D}_i$ , and add these samples to  $\mathcal{S}$ .

/\* estimate the near-optimal hypothesis for weighted data distributions. \*/

12 compute  $h^t \leftarrow \arg \min_{h \in \mathcal{H}} \hat{L}^t(h, u^t)$ , where

$$\hat{L}^t(h, u^t) := \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} \frac{u_{i,\ell}^t}{n_i^t} \cdot \sum_{j=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) \quad (62)$$

with  $(x_{i,j}, y_{i,j})$  being the  $j$ -th datapoint from  $\mathcal{D}_i$  in  $\mathcal{S}$ .

/\* estimate the loss vector and execute weighted updates. \*/

13  $\bar{w}_i^t \leftarrow \max_{1 \leq r \leq t} \sum_{\ell \in \mathcal{L}} u_{i,\ell}^r$  for all  $i \in [k]$ .

14 **for**  $i = 1, \dots, k$  **do**

15    draw  $\lceil k\bar{w}_i^t \rceil$  independent samples – denoted by  $\{(x_{i,j}^t, y_{i,j}^t)\}_{j=1}^{\lceil k\bar{w}_i^t \rceil}$  – from  $\mathcal{D}_i$ , and set  
$$\hat{r}_{i,\ell}^t = \frac{1}{\lceil k\bar{w}_i^t \rceil} \sum_{j=1}^{\lceil k\bar{w}_i^t \rceil} \ell(h^t, (x_{i,j}^t, y_{i,j}^t)) \quad \text{for each } \ell \in \mathcal{L};$$

16    update the weight as  $U_{i,\ell}^{t+1} = U_{i,\ell}^t \exp(\eta \hat{r}_{i,\ell}^t)$ . // Hedge updates.

**17 output:** a randomized hypothesis  $h^{\text{final}}$  as a uniform distribution over  $\{h^t\}_{t=1}^T$ .

---

necessity of improper learning, revealing that a considerably larger sample size is necessary if the learning algorithm is constrained to return deterministic hypotheses from  $\mathcal{H}$ . Notably, our work manages to solve three open problems presented in COLT 2023 (namely, Awasthi et al. [4, Problems 1, 3, and 4]).

Our work not only addresses existing challenges but also opens up several directions for future exploration. To begin with, note that our sample complexity results are only optimal up to logarithmic factors. The polylog factor, however, could be important for several practical scenarios (see, e.g., Deng and Qiao [13] for a setting where multiple hypotheses are allowed to be returned). Further studies are needed in order to sharpen the logarithmic dependency. Additionally, the current article assumes a flexible sampling protocol that allows the learner to take samples arbitrarily from any of the  $k$  distributions; how will the sample complexity be impacted under additional constraints imposed on the sampling process? Furthermore, can we extend our current analysis

(which bounds the dynamics of the Hedge algorithm) to control the trajectory of more general first-order/second-order algorithms, in the context of robust online learning? Another venue for exploration is the extension of our multi-distribution learning framework to tackle other related tasks like multi-calibration [21, 24]. We believe that our algorithmic and analysis framework can shed light on making progress in all of these directions.

## Acknowledgements

We thank Eric Zhao for answering numerous questions about the open problems. We would also like to thank Chicheng Zhang for pointing out an error in a preliminary version of this article.

## References

- [1] Jacob Abernethy, Pranjal Awasthi, Matthias Kleindessner, Jamie Morgenstern, Chris Russell, and Jie Zhang. 2022. Active sampling for min-max fairness. In *Proceedings of the 39th International Conference on Machine Learning*. 53–65.
- [2] Sanjeev Arora, Elad Hazan, and Satyen Kale. 2012. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing* 8, 1 (2012), 121–164.
- [3] Hilal Asi, Yair Carmon, Arun Jambulapati, Yujia Jin, and Aaron Sidford. 2021. Stochastic bias-reduced gradient methods. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*. 10810–10822.
- [4] Pranjal Awasthi, Nika Haghtalab, and Eric Zhao. 2023. Open problem: The sample complexity of multi-distribution learning for VC classes. In *Proceedings of the 36th Annual Conference on Learning Theory*. 5943–5949.
- [5] Avrim Blum, Nika Haghtalab, Richard Lanas Phillips, and Han Shao. 2021. One for one, or all for all: Equilibria and optimality of collaboration in federated learning. In *Proceedings of the 38th International Conference on Machine Learning*. 1005–1014.
- [6] Avrim Blum, Nika Haghtalab, Ariel D. Procaccia, and Mingda Qiao. 2017. Collaborative PAC learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*.
- [7] Avrim Blum, Shelby Heinecke, and Lev Reyzin. 2021. Communication-aware collaborative learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 6786–6793.
- [8] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1989. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM* 36, 4 (1989), 929–965.
- [9] Peter Bühlmann and Nicolai Meinshausen. 2015. Magging: Maximin aggregation for inhomogeneous large-scale data. *Proc. IEEE* 104, 1 (2015), 126–135.
- [10] Yair Carmon and Danielle Hause. 2022. Distributionally robust optimization via ball oracle acceleration. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*. 35866–35879.
- [11] Jiecao Chen, Qin Zhang, and Yuan Zhou. 2018. Tight bounds for collaborative PAC learning via multiplicative weights. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.
- [12] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. 2020. Distributionally robust federated averaging. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 15111–15122.
- [13] Yuyang Deng and Mingda Qiao. 2024. Collaborative Learning with Different Labeling Functions. In *Proceedings of the 41st International Conference on Machine Learning*. PMLR, 10530–10552.
- [14] Wei Du, Depeng Xu, Xintao Wu, and Hanghang Tong. 2021. Fairness-aware agnostic federated learning. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 181–189.
- [15] Miroslav Dudík, Nika Haghtalab, Haipeng Luo, Robert E Schapire, Vasilis Syrgkanis, and Jennifer Wortman Vaughan. 2020. Oracle-efficient online learning and auction design. *Journal of the ACM* 67, 5 (2020), 1–57.
- [16] Cynthia Dwork, Michael P. Kim, Omer Reingold, Guy N. Rothblum, and Gal Yona. 2021. Outcome indistinguishability. In *Proceedings of the ACM SIGACT Symposium on Theory of Computing*. 1095–1108.
- [17] David A. Freedman. 1975. On tail probabilities for martingales. *The Annals of Probability* 3, 1 (1975), 100–118.
- [18] Yoav Freund and Robert E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 1 (1997), 119–139.
- [19] Zijian Guo. 2023. Statistical inference for maximin effects: Identifying stable associations across multiple studies. *Journal of the American Statistical Association* (2023), 1–17.
- [20] Nika Haghtalab, Michael Jordan, and Eric Zhao. 2022. On-demand sampling: Learning optimally from multiple distributions. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*. 406–419.
- [21] Nika Haghtalab, Michael Jordan, and Eric Zhao. 2023. A unifying perspective on multi-calibration: Game dynamics for multi-objective learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*. 72464–72506.

- [22] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization. In *Proceedings of the 35th International Conference on Machine Learning*. 1929–1938.
- [23] Elad Hazan. 2022. *Introduction to Online Convex Optimization*. MIT Press.
- [24] Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. 2018. Multicalibration: Calibration for the (computationally-identifiable) masses. In *Proceedings of the 35th International Conference on Machine Learning*. 1939–1948.
- [25] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. 2018. Does distributionally robust supervised learning give robust classifiers?. In *Proceedings of the 35th International Conference on Machine Learning*. 2029–2037.
- [26] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. 2019. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4551–4560.
- [27] Tor Lattimore and Csaba Szepesvári. 2020. *Bandit Algorithms*. Cambridge University Press.
- [28] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2008. Domain adaptation with multiple sources. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*.
- [29] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of Machine Learning*. MIT press.
- [30] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *Proceedings of the 36th International Conference on Machine Learning*. 4615–4625.
- [31] Huy Nguyen and Lydia Zakynthinou. 2018. Improved algorithms for collaborative PAC learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.
- [32] Binghui Peng. 2024. The sample complexity of multi-distribution learning. In *Proceedings of the 37th Annual Conference on Learning Theory*. PMLR, 4185–4204.
- [33] Guy N. Rothblum and Gal Yona. 2021. Multi-group agnostic PAC learnability. In *Proceedings of the 38th International Conference on Machine Learning*. 9107–9115.
- [34] Tim Roughgarden. 2016. *Twenty Lectures on Algorithmic Game Theory*. Cambridge University Press.
- [35] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks. In *Proceedings of the International Conference on Learning Representations*.
- [36] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why overparameterization exacerbates spurious correlations. In *Proceedings of the 37th International Conference on Machine Learning*. 8346–8356.
- [37] Shai Shalev-Shwartz. 2012. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning* 4, 2 (2012), 107–194.
- [38] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.
- [39] J. v. Neumann. 1928. Zur theorie der gesellschaftsspiele. *Mathematische Annalen* 100, 1 (1928), 295–320.
- [40] Leslie G. Valiant. 1984. A theory of the learnable. *Communications of the ACM* 27, 11 (1984), 1134–1142.
- [41] Vladimir Vapnik, Esther Levin, and Yann Le Cun. 1994. Measuring the VC-dimension of a learning machine. *Neural computation* 6, 5 (1994), 851–876.
- [42] Martin J. Wainwright. 2019. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Vol. 48. Cambridge university press.
- [43] Zhenyu Wang, Peter Bühlmann, and Zijian Guo. 2023. Distributionally robust machine learning with multi-source data. arXiv:2309.02211. Retrieved from <https://arxiv.org/abs/2309.02211>
- [44] Xin Xiong, Zijian Guo, and Tianxi Cai. 2023. Distributionally robust transfer learning. arXiv:2309.06534. Retrieved from <https://arxiv.org/abs/2309.06534>
- [45] Jingzhao Zhang, Aditya Krishna Menon, Andreas Veit, Srinadh Bhojanapalli, Sanjiv Kumar, and Suvrit Sra. 2020. Coping with label shift via distributionally robust optimisation. In *Proceedings of the International Conference on Learning Representations*.
- [46] Zihan Zhang, Xiangyang Ji, and Simon Du. 2022. Horizon-free reinforcement learning in polynomial time: The power of stationary policies. In *Proceedings of the 35th Annual Conference on Learning Theory*. 3858–3904.
- [47] Sicheng Zhao, Bo Li, Pengfei Xu, and Kurt Keutzer. 2020. Multi-source domain adaptation in the deep learning era: A systematic survey. arXiv:2002.12169. Retrieved from <https://arxiv.org/abs/2002.12169>

## Appendices

### A Auxiliary lemmas

In this section, we introduce several technical lemmas that are used multiple times in our analysis.

We begin by introducing three handy concentrations inequalities. The first result is the well-renowned Freedman inequality [17], which assists in deriving variance-aware concentration inequalities for martingales.

LEMMA 12 (FREEDMAN'S INEQUALITY [17]). Let  $(M_n)_{n \geq 0}$  be a martingale obeying  $M_0 = 0$ . Define  $V_n := \sum_{k=1}^n \mathbb{E}[(M_k - M_{k-1})^2 | \mathcal{F}_{k-1}]$  for each  $n \geq 0$ , where  $\mathcal{F}_k$  denotes the  $\sigma$ -algebra generated by  $(M_1, M_2, \dots, M_k)$ . Suppose that  $M_k - M_{k-1} \leq 1$  for all  $k \geq 1$ . Then for any  $x > 0$  and  $y > 0$ , one has

$$\mathbb{P}(M_n \geq nx, V_n \leq ny) \leq \exp\left(-\frac{nx^2}{2(y + \frac{1}{3}x)}\right). \quad (63)$$

The second concentration result bounds the difference between the sum of a sequence of random variables and the sum of their respective conditional means (w.r.t. the associated  $\sigma$ -algebra).

LEMMA 13 (LEMMA 10 IN [46]). Let  $X_1, X_2, \dots$  be a sequence of random variables taking value in the interval  $[0, l]$ . For any  $k \geq 1$ , let  $\mathcal{F}_k$  be the  $\sigma$ -algebra generated by  $(X_1, X_2, \dots, X_k)$ , and define  $Y_k := \mathbb{E}[X_k | \mathcal{F}_{k-1}]$ . Then for any  $\delta > 0$ , we have

$$\begin{aligned} \mathbb{P}\left\{\exists n \in \mathbb{N}, \sum_{k=1}^n X_k \geq 3 \sum_{k=1}^n Y_k + l \log \frac{1}{\delta}\right\} &\leq \delta, \\ \mathbb{P}\left\{\exists n \in \mathbb{N}, \sum_{k=1}^n Y_k \geq 3 \sum_{k=1}^n X_k + l \log \frac{1}{\delta}\right\} &\leq \delta. \end{aligned}$$

The third concentration result is the Mcdiarmid inequality, a celebrated inequality widely used to control the flucutaion of multivariate functions when the input variables are independently generated.

LEMMA 14 (MCIDIARMID'S INEQUALITY). Let  $X_1, X_2, \dots, X_n$  be a sequence of independent random variables, with  $X_i$  supported on  $\mathcal{X}_i$ . Let  $f : \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_n \rightarrow \mathbb{R}$  be a function such that: for any  $i \in [n]$  and any  $\{x_1, \dots, x_n\} \in \mathcal{X}_1 \times \dots \times \mathcal{X}_n$ ,

$$\sup_{x'_i \in \mathcal{X}_i} |f(x_1, \dots, x_i, \dots, x_n) - f(x_1, \dots, x'_i, \dots, x_n)| \leq c$$

holds for some quantity  $c > 0$ . It then holds that

$$\mathbb{P}\left\{|f(X_1, X_2, \dots, X_n) - \mathbb{E}[f(X_1, X_2, \dots, X_n)]| \geq \varepsilon\right\} \leq 2 \exp\left(-\frac{2\varepsilon^2}{nc^2}\right).$$

Additionally, the following lemma presents a sort of the data processing inequality w.r.t. the **Kullback-Leibler (KL)** divergence, which is a classical result from information theory.

LEMMA 15. Let  $\mathcal{X}$  and  $\mathcal{Y}$  be two sets, and consider any function  $f : \mathcal{X} \rightarrow \mathcal{Y}$ . For any two random variables  $X_1$  and  $X_2$  supported on  $\mathcal{X}$ , it holds that

$$\text{KL}(\mu(X_1) \| \mu(X_2)) \geq \text{KL}(\mu(f(X_1)) \| \mu(f(X_2))), \quad (64)$$

where we use  $\mu(Z)$  to denote the distribution of a random variable  $Z$ .

Lastly, let us make note of an elementary bound regarding the KL divergence between two Bernoulli distributions.

LEMMA 16. Consider any  $q > 0$  and  $x \in [0, \log(2)]$ . Also, consider any  $y, y' \in (0, 1)$  obeying  $y \geq q$  and  $y' \geq \exp(x)y$ . It then holds that

$$\text{KL}(\text{Ber}(y) \| \text{Ber}(y')) \geq \frac{qx^2}{4},$$

where  $\text{Ber}(z)$  denotes the Bernoulli distribution with mean  $z$ .

PROOF. To begin with, the function defined below satisfies

$$f(a, b) := \text{KL}(\text{Ber}(a) \| \text{Ber}(b)) = a \log\left(\frac{a}{b}\right) + (1 - a) \log\left(\frac{1 - a}{1 - b}\right).$$

For any  $0 < a \leq b \leq 1$ , it is readily seen that

$$\frac{\partial f(a, b)}{\partial b} = -\frac{a}{b} + \frac{1-a}{1-b} = \frac{b-a}{b(1-b)} \geq 0.$$

It follows from our assumptions  $y \geq q$  and  $y' \geq \exp(x)y$  that

$$\begin{aligned} \text{KL}(\text{Ber}(y) \| \text{Ber}(y')) &= f(y, y') = f(y, y) + \int_y^{y'} \frac{\partial f(y, z)}{\partial z} dz = \int_y^{y'} \frac{z-y}{z(1-z)} dz \\ &\geq \frac{1}{y'} \int_y^{y'} (z-y) dz \geq \frac{(y'-y)^2}{2y'} \\ &\geq \frac{(y'-y)(1-\exp(-x))}{2} \\ &\geq \frac{y(\exp(x)-1)^2}{4} \geq \frac{qx^2}{4}, \end{aligned}$$

where the penultimate inequality uses  $x \in [0, \log(2)]$ , and the last inequality holds since  $y \geq q$ .  $\square$

Finally, let us present a basic property related to Rademacher random variables, which will play a useful role in understanding the Rademacher complexity.

LEMMA 17. *Let  $\mathcal{L}$  be a set of vectors in  $\mathbb{R}^n$ . Let  $w^1, w^2 \in \mathbb{R}^n$  be two vectors obeying  $|w_i^1| \leq |w_i^2|$  for all  $i \in [n]$ . Then it holds that*

$$\mathbb{E}_{\{\sigma_i\}_{i=1}^n} \left[ \max_{f \in \mathcal{L}} \sum_{i=1}^n \sigma_i w_i^1 f_i \right] \leq \mathbb{E}_{\{\sigma_i\}_{i=1}^n} \left[ \max_{f \in \mathcal{L}} \sum_{i=1}^n \sigma_i w_i^2 f_i \right], \quad (65)$$

where  $\{\sigma_i\}$  is a collection of independent Rademacher random variables obeying  $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = 1/2$ .

PROOF. Clearly, it suffices to prove Equation (65) for the special case where  $w_i^1 = w_i^2$  for  $1 \leq i \leq n-1$ , and  $|w_n^1| \leq |w_n^2|$ . Fixing  $\sigma_i$  for  $1 \leq i \leq n-1$ , we can deduce that

$$\begin{aligned} \mathbb{E}_{\sigma_n} \left[ \max_{f \in \mathcal{L}} \sum_{i=1}^n \sigma_i w_i f_i \right] &= \frac{1}{2} \max_{f \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i w_i f_i + w_n f_n \right) + \frac{1}{2} \max_{f \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i w_i f_i - w_n f_n \right) \\ &= \frac{1}{2} \max_{f, \tilde{f} \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i w_i (f_i + \tilde{f}_i) + w_n (f_n - \tilde{f}_n) \right) \\ &= \frac{1}{2} \max_{f, \tilde{f} \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i \tilde{w}_i (f_i + \tilde{f}_i) + w_n (f_n - \tilde{f}_n) \right) \\ &\leq \frac{1}{2} \max_{f, \tilde{f} \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i \tilde{w}_i (f_i + \tilde{f}_i) + |\tilde{w}_n (f_n - \tilde{f}_n)| \right) \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2} \max_{f, \tilde{f} \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i \tilde{w}_i (f_i + \tilde{f}_i) + \tilde{w}_n (f_n - \tilde{f}_n) \right) \\
&= \frac{1}{2} \max_{f \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i \tilde{w}_i f_i + \tilde{w}_n f_n \right) + \frac{1}{2} \max_{f \in \mathcal{L}} \left( \sum_{i=1}^{n-1} \sigma_i \tilde{w}_i f_i - \tilde{w}_n f_n \right) \\
&= \mathbb{E}_{\sigma_n} \left[ \max_{f \in \mathcal{L}} \sum_{i=1}^n \sigma_i \tilde{w}_i f_i \right].
\end{aligned}$$

The proof is thus completed by taking expectation over  $\{\sigma_i\}_{i=1}^{n-1}$ .  $\square$

## B Proofs of Auxiliary Lemmas for VC Classes

### B.1 Proof of Lemma 1

For ease of presentation, suppose there exists a dataset  $\tilde{\mathcal{S}}$  containing  $T_1$  independent samples drawn from each distribution  $\mathcal{D}_i$  ( $1 \leq i \leq k$ ), so that in total it contains  $kT_1$  samples. We find it helpful to introduce the following notation.

- For each  $i \in [k]$  and  $j \in [n_i]$ , denote by  $(x_{i,j}, y_{i,j})$  the  $j$ th sample in  $\tilde{\mathcal{S}}$  that is drawn from  $\mathcal{D}_i$ .
- For each set of integers  $n = \{n_i\}_{i=1}^k \in \mathbb{N}^k$ , we define  $\tilde{\mathcal{S}}(n)$  to be the dataset containing  $\{(x_{i,j}, y_{i,j})\}_{1 \leq j \leq n_i}$  for all  $i \in [k]$ ; namely, it comprises, for each  $i \in [k]$ , the first  $n_i$  samples in  $\tilde{\mathcal{S}}$  that are drawn from  $\mathcal{D}_i$ .
- We shall also let  $\tilde{\mathcal{S}}^+(n) = \{(x_{i,j}^+, y_{i,j}^+)\}_{j=1}^{n_i} \}_{i=1}^k$  be an *independent copy* of  $\tilde{\mathcal{S}}(n)$ , where for each  $i \in [k]$ ,  $\{(x_{i,j}^+, y_{i,j}^+)\}$  are independent samples drawn from  $\mathcal{D}_i$ .

Equipped with the above notation, we are ready to present our proof.

**Step 1: concentration bounds for any fixed  $n = \{n_i\}_{i=1}^k$  and  $w \in \Delta(k)$ .** Consider first any fixed  $n = \{n_i\}_{i=1}^k$  obeying  $0 \leq n_i \leq T_1$  for all  $i \in [k]$ , and any fixed  $w \in \Delta(k)$ . For any quantity  $\lambda \in [0, \min_{i \in [k]} \frac{n_i}{w_i}]$ , if we take

$$E(\lambda, n, w) := \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k w_i \frac{1}{n_i} \sum_{i=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w) \right\} \right) \right], \quad (66)$$

with the expectation taken over the randomness of  $\tilde{\mathcal{S}}(n)$ , then we can apply a standard “symmetrization” trick to bound  $E(\lambda, n, w)$  as follows:

$$\begin{aligned}
E(\lambda, n, w) &:= \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{i=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w) \right\} \right) \right] \\
&= \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{i=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \mathbb{E}_{\tilde{\mathcal{S}}^+(n)} \left[ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{i=1}^{n_i} \ell(h, (x_{i,j}^+, y_{i,j}^+)) \right] \right\} \right) \right] \\
&\leq \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \mathbb{E}_{\tilde{\mathcal{S}}^+(n)} \left[ \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{i=1}^{n_i} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \right] \right] \\
&\leq \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{i=1}^{n_i} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \right], \quad (67)
\end{aligned}$$

where the last two inequalities follow from Jensen’s inequality.

Next, let  $\sigma(n) := \{\{\sigma_{i,j}\}_{j=1}^{n_i}\}_{i=1}^k$  be a collection of i.i.d. Rademacher random variables obeying  $\mathbb{P}(\sigma_{i,j} = 1) = \mathbb{P}(\sigma_{i,j} = -1) = 1/2$ . Denoting  $\mathcal{C} = \{(x_{i,j}, y_{i,j})\} \cup \{(x_{i,j}^+, y_{i,j}^+)\}$ , we obtain

$$\begin{aligned} & \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \right] \\ &= \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \mathbb{E}_{\sigma(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \right]. \end{aligned} \quad (68)$$

Note that for any dataset  $\mathcal{C}$  with cardinality  $|\mathcal{C}|$ , the Sauer–Shelah lemma [42, Proposition 4.18] together with our assumption that  $\text{VC-dim}(\mathcal{H}) \leq d$  tells us that the cardinality of the following set obeys

$$|\mathcal{H}(\mathcal{C})| \leq (|\mathcal{C}| + 1)^d \leq (|\tilde{\mathcal{S}}| + |\tilde{\mathcal{S}}^+| + 1)^d \leq (2kT_1 + 1)^d, \quad (69)$$

where  $\mathcal{H}(\mathcal{C})$  denotes the set obtained by applying all  $h \in \mathcal{H}$  to the data points in  $\mathcal{C}$ , namely,

$$\mathcal{H}(\mathcal{C}) := \{(h(x_{1,1}), h(x_{1,1}^+), h(x_{1,2}), h(x_{1,2}^+), \dots) \mid h \in \mathcal{H}\}. \quad (70)$$

We shall also define  $\mathcal{H}_{\min, \mathcal{C}} \subseteq \mathcal{H}$  to be the *minimum-cardinality subset* of  $\mathcal{H}$  that results in the same outcome as  $\mathcal{H}$  when applied to  $\mathcal{C}$ , namely,

$$\mathcal{H}_{\min, \mathcal{C}}(\mathcal{C}) = \mathcal{H}(\mathcal{C}) \quad \text{and} \quad |\mathcal{H}_{\min, \mathcal{C}}| = |\mathcal{H}(\mathcal{C})|.$$

With these in place, we can demonstrate that

$$\begin{aligned} & \mathbb{E}_{\sigma(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \\ &= \mathbb{E}_{\sigma(n)} \left[ \max_{h \in \mathcal{H}_{\min, \mathcal{C}}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \\ &\leq \mathbb{E}_{\sigma(n)} \left[ \sum_{h \in \mathcal{H}_{\min, \mathcal{C}}} \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \\ &\leq |\mathcal{H}_{\min, \mathcal{C}}| \max_{h \in \mathcal{H}_{\min, \mathcal{C}}} \mathbb{E}_{\sigma(n)} \left[ \exp \left( \lambda \left\{ \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \\ &\leq (2kT_1 + 1)^d \max_{h \in \mathcal{H}} \prod_{i=1}^k \prod_{j=1}^{n_i} \mathbb{E}_{\sigma_{i,j}} \left[ \exp \left( \lambda \left\{ \frac{w_i}{n_i} \sigma_{i,j} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right\} \right) \mid \mathcal{C} \right] \\ &\leq (2kT_1 + 1)^d \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} \right). \end{aligned} \quad (71)$$

Here, the last inequality makes use of fact  $|\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))| \leq 2$  as well as the following elementary inequality

$$\mathbb{E}_{\sigma_{i,j}} [\exp(\sigma_{i,j}x)] = \frac{1}{2} (\exp(x) + \exp(-x)) \leq \exp(x^2).$$

Taking Equations (67), (68), and (71) together reveals that

$$E(\lambda) \leq (2kT_1 + 1)^d \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} \right). \quad (72)$$

Repeating the same arguments also yields an upper bound on the following quantity:

$$\begin{aligned}\bar{E}(\lambda) &:= \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \left\{ L(h, w) - \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) \right\} \right) \right] \\ &\leq (2kT_1 + 1)^d \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} \right),\end{aligned}$$

for any  $\lambda \in [0, \min_{i \in [k]} \frac{n_i}{w_i}]$ . Taking the above two inequalities and applying the Markov inequality reveal that, for any  $0 < \varepsilon' \leq 1$ ,

$$\begin{aligned}&\mathbb{P} \left( \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k w_i \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w) \right| \geq \varepsilon' \right) \\ &\leq \min_{0 \leq \lambda \leq \min_i \frac{n_i}{w_i}} \frac{E(\lambda) + \bar{E}(\lambda)}{\exp(\lambda\varepsilon')} \\ &\leq \min_{0 \leq \lambda \leq \min_i \frac{n_i}{w_i}} 2 \cdot (2kT_1 + 1)^d \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} - \lambda\varepsilon' \right).\end{aligned}\tag{73}$$

**Step 2: uniform concentration bounds over epsilon-nets w.r.t.  $n$  and  $w$ .** Next, we move on to extend the above result to uniform concentration bounds over all possible  $n$  and  $w$ . Toward this, let us first introduce a couple of notation.

- Let us use  $\Delta_{\varepsilon_2}(k) \subseteq \Delta(k)$  to denote an  $\varepsilon_2$ -net of  $\Delta(k)$  — namely, for any  $x \in \Delta(k)$ , there exists a vector  $x_0 \in \Delta_{\varepsilon_2}(k)$  obeying  $\|x - x_0\|_\infty \leq \varepsilon_2$ . We shall choose  $\Delta_{\varepsilon_2}(k)$  properly so that

$$|\Delta_{\varepsilon_2}(k)| \leq (1/\varepsilon_2)^k.$$

- Define the following set

$$\mathcal{B} = \left\{ n = \{n_i\}_{i=1}^k, w = \{w_i\}_{i=1}^k \mid \frac{n_i}{w_i} \geq \frac{T_1}{2}, 0 \leq n_i \leq T_1, \forall i \in [k], w \in \Delta_{\varepsilon_1/(8k)}(k) \right\},$$

which clearly satisfies

$$|\mathcal{B}| \leq T_1^k \cdot \left( \frac{8k}{\varepsilon_1} \right)^k.$$

Applying the union bound yields that, for any  $0 < \varepsilon' \leq 1$ ,

$$\begin{aligned}&\mathbb{P} \left( \exists (n, w) \in \mathcal{B}, \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k w_i \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w) \right| \geq \varepsilon' \right) \\ &\leq \sum_{(n,w) \in \mathcal{B}} \min_{0 \leq \lambda \leq \min_i \frac{n_i}{w_i}} 2 \cdot (2kT_1 + 1)^d \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} - \lambda\varepsilon' \right) \\ &\leq \sum_{(n,w) \in \mathcal{B}} \min_{0 \leq \lambda \leq \frac{T_1}{2}} 2 \cdot (2kT_1 + 1)^d \exp \left( 2\lambda^2 \cdot \frac{2}{T_1} - \lambda\varepsilon' \right) \\ &\leq \sum_{(n,w) \in \mathcal{B}} 2 \cdot (2kT_1 + 1)^d \exp \left( -\frac{T_1(\varepsilon')^2}{16} \right)\end{aligned}$$

$$\begin{aligned} &\leq |\mathcal{B}| \cdot 2 \cdot (2kT_1 + 1)^d \exp\left(-\frac{T_1(\varepsilon')^2}{16}\right) \\ &\leq 2 \cdot (8kT_1/\varepsilon_1)^k (2kT_1 + 1)^d \cdot \exp\left(-\frac{T_1(\varepsilon')^2}{16}\right), \end{aligned}$$

where the second inequality holds since  $\sum_{i=1}^k \frac{w_i^2}{n_i} \leq \frac{2}{T_1} \sum_{i=1}^k w_i = \frac{2}{T_1}$  (according to the definition of  $\mathcal{B}$ ).

**Step 3: concentration bounds w.r.t.  $n^t$  and  $w^t$ .** Let  $\mathcal{S}^t$  denote the value of  $\mathcal{S}$  after line 10 of Algorithm 1 in the  $t$ th round. Recall that  $n^t = [n_i^t]_{1 \leq i \leq k}$  denotes the number of samples for all  $k$  distributions in  $\mathcal{S}^t$ , and let  $w^t = [w_i^t]_{1 \leq i \leq k}$  represent the weight iterates in the  $t$ th round. It is easily seen from lines 6 and 9 of Algorithm 1 that  $n_i^t \leq T_1$  and  $n_i^t/w_i^t \geq n_i^t/(2\hat{w}_i^t) \geq T_1/2$ . For analysis purposes, it suffices to take  $\mathcal{S}^t = \tilde{\mathcal{S}}(n^t)$ .

In view of the update rule in Algorithm 1, one can always find  $(n^t, \tilde{w}^t) \in \mathcal{B}$  satisfying  $\|\tilde{w}^t - w^t\|_1 \leq k\|\tilde{w}^t - w^t\|_\infty \leq \varepsilon_1/8$ . As a result, for any  $0 < \varepsilon' \leq 1$ , we can deduce that

$$\begin{aligned} &\mathbb{P}\left(\exists t \in [T], \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k w_i^t \frac{1}{n_i^t} \sum_{i=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w^t) \right| \geq \varepsilon' + \frac{\varepsilon_1}{4} \right) \\ &\leq \mathbb{P}\left(\exists t \in [T], \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k \tilde{w}_i^t \frac{1}{n_i^t} \sum_{i=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) - L(h, \tilde{w}^t) \right| \geq \varepsilon' \right) \\ &\leq 2 \cdot (8kT_1/\varepsilon_1)^k (2kT_1 + 1)^d \cdot \exp\left(-\frac{T_1(\varepsilon')^2}{16}\right), \end{aligned} \tag{74}$$

where the second inequality arises from the fact that  $\frac{1}{n_i} \sum_{i=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) \in [-1, 1]$  and  $L(h, \tilde{w}^t) \in [-1, 1]$ . Taking  $\varepsilon' = \varepsilon_1/4$  and substituting  $T_1 = \frac{4000(k \log(k/\varepsilon_1) + d \log(kd/\varepsilon_1) + \log(1/\delta))}{\varepsilon_1^2}$  into Equation (74), we can obtain

$$\begin{aligned} &\mathbb{P}\left(\exists t \in [T], \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k w_i^t \cdot \frac{1}{n_i^t} \sum_{i=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) - L(h, w^t) \right| \geq \frac{\varepsilon_1}{2} \right) \\ &\leq 2 \cdot (8kT_1/\varepsilon_1)^k (2kT_1 + 1)^d \cdot \exp\left(-\frac{T_1 \varepsilon_1^2}{16}\right) \\ &\leq 2 \cdot (8kT_1/\varepsilon_1)^k (2kT_1 + 1)^d \cdot \exp\left(-10(k \log(k/\varepsilon_1) + d \log(kd/\varepsilon_1) + \log(1/\delta))\right) \\ &\leq 2 \cdot (8kT_1/\varepsilon_1)^k (2kT_1 + 1)^d \cdot (k/\varepsilon_1)^{-10k} \cdot (kd/\varepsilon_1)^{-10d} \cdot \delta \\ &\leq \delta/4. \end{aligned} \tag{75}$$

**Step 4: putting all this together.** Recalling that

$$\hat{L}^t(h, w^t) = \sum_{i=1}^k w_i^t \cdot \frac{1}{n_i^t} \sum_{i=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})),$$

one can see from Equation (75) that, with probability exceeding  $1 - \delta/4$ ,

$$\left| \hat{L}^t(h, w^t) - L(h, w^t) \right| \leq \frac{\varepsilon_1}{2}, \tag{76}$$

holds simultaneously for all  $t \in [T]$  and all  $h \in \mathcal{H}$ . Additionally, observing that

$$h^t = \arg \min_{h \in \mathcal{H}} \widehat{L}^t(h, w^t), \quad (77)$$

we can immediately deduce that

$$L(h^t, w^t) \leq \widehat{L}(h^t, w^t) + \frac{\varepsilon_1}{2} = \min_{h \in \mathcal{H}} \widehat{L}(h, w^t) + \frac{\varepsilon_1}{2} \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1. \quad (78)$$

This concludes the proof of Lemma 1.

## B.2 Proof of Lemma 2

Before proceeding, let us introduce some additional notation. Let  $\delta' := \frac{\delta}{4(T+k+1)}$ , and recall

$$\text{OPT} := \max_{w \in \Delta(k)} \min_{h \in \mathcal{H}} L(h, w) = \min_{\pi \in \Delta(\mathcal{H})} \max_{w \in \Delta(k)} [L(h_\pi, w)] \leq \min_{h \in \mathcal{H}} \max_{1 \leq i \leq k} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))],$$

to be the optimal objective value. Recall that

$$v^t = L(h^t, w^t) - \text{OPT}. \quad (79)$$

It follows from the assumption (i.e.,  $L(h^t, w^t) \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1$ ) that

$$v^t \leq \min_{h \in \mathcal{H}} L(h, w^t) - \text{OPT} + \varepsilon_1 \leq \varepsilon_1, \quad \forall 1 \leq t \leq T. \quad (80)$$

We now begin to present the proof. In view of the Azuma-Hoeffding inequality and the union bound, we see that with probability at least  $1 - (k+1)\delta'$ ,

$$\left| \sum_{t=1}^T \langle w^t, \widehat{r}^t \rangle - \sum_{t=1}^T L(h^t, w^t) \right| \leq 2\sqrt{T \log(1/\delta')}, \quad (81a)$$

$$\left| \sum_{t=1}^T \widehat{r}_i^t - \sum_{t=1}^T L(h^t, e_i^{\text{basis}}) \right| \leq 2\sqrt{T \log(1/\delta')}. \quad (81b)$$

These motivate us to look at  $\sum_{t=1}^T \langle w^t, \widehat{r}^t \rangle$  (respectively  $\sum_{t=1}^T \widehat{r}_i^t$ ) as a surrogate for  $\sum_{t=1}^T L(h^t, w^t)$  (respectively  $\sum_{t=1}^T L(h^t, e_i^{\text{basis}})$ ).

We then resort to standard analysis for the Hedge algorithm. Specifically, direct computation gives

$$\begin{aligned} \log \left( \frac{\sum_{i=1}^k W_i^{t+1}}{\sum_{i=1}^k W_i^t} \right) &\stackrel{(i)}{=} \log \left( \sum_{i=1}^k w_i^t \exp(\eta \widehat{r}_i^t) \right) \stackrel{(ii)}{\leq} \log \left( \sum_{i=1}^k w_i^t (1 + \eta \widehat{r}_i^t + \eta^2 (\widehat{r}_i^t)^2) \right) \\ &\leq \log \left( 1 + \eta \sum_{i=1}^k w_i^t \widehat{r}_i^t + \eta^2 \sum_{i=1}^k w_i^t (\widehat{r}_i^t)^2 \right) \leq \eta \sum_{i=1}^k w_i^t \widehat{r}_i^t + \eta^2. \end{aligned} \quad (82)$$

Here, (i) is valid since  $w_i^t = \frac{W_i^t}{\sum_j W_j^t}$  and  $W_i^{t+1} = W_i^t \exp(\eta \widehat{r}_i^t)$  (cf. lines 5 and 15 of Algorithm 1); (ii) arises from the elementary inequality  $e^x \leq 1 + x + x^2$  for  $x \in [0, 1]$  as well as the facts that  $\eta \leq 1$  and  $|\widehat{r}_i^t| \leq 1$ . Summing the inequality Equation (82) over all  $t$  and rearranging terms, we are

left with

$$\begin{aligned}
\eta \sum_{t=1}^T \langle w^t, \hat{r}^t \rangle &\geq \sum_{t=1}^T \left\{ \log \left( \frac{\sum_{i=1}^k W_i^{t+1}}{\sum_{i=1}^k W_i^t} \right) - \eta^2 \right\} \\
&= \log \left( \sum_{i=1}^k W_i^{T+1} \right) - \log \left( \sum_{i=1}^k W_i^1 \right) - T\eta^2 \\
&\geq \max_{1 \leq i \leq k} \log(W_i^{T+1}) - \log(k) - T\eta^2 \\
&\geq \eta \max_{1 \leq i \leq k} \sum_{t=1}^T \hat{r}_i^t - \log(k) - T\eta^2,
\end{aligned} \tag{83}$$

where the penultimate lines makes use of  $W_i^1 = 1$  for all  $i \in [k]$ , and the last line holds since  $\log(W_i^{T+1}) = \eta \sum_{t=1}^T \hat{r}_i^t$ . Dividing both sides by  $\eta$  yields

$$\sum_{t=1}^T \langle w^t, \hat{r}^t \rangle \geq \max_i \sum_{t=1}^T \hat{r}_i^t - \left( \frac{\log(k)}{\eta} + \eta T \right). \tag{84}$$

Combine the above inequality with Equation (81) to show that, with probability at least  $1 - (k+1)\delta'$ ,

$$\sum_{t=1}^T L(h^t, w^t) \geq \max_{1 \leq i \leq k} \sum_{t=1}^T L(h^t, e_i^{\text{basis}}) - \left( \frac{\log(k)}{\eta} + \eta T + 4\sqrt{T \log(1/\delta')} \right). \tag{85}$$

Recalling that  $\varepsilon_1 = \eta = \frac{1}{100}\varepsilon$  and  $T = \frac{20000 \log(\frac{k}{\delta'\varepsilon})}{\varepsilon^2}$ , we can derive

$$\begin{aligned}
\max_{1 \leq i \leq k} \sum_{t=1}^T y_i^t &\leq T \text{OPT} + \sum_{t=1}^T v^t + \left( \frac{\log(k)}{\eta} + \eta T + 4\sqrt{T \log(1/\delta')} \right) \\
&\leq T \text{OPT} + T\varepsilon_1 + \left( \frac{\log(k)}{\eta} + \eta T + 4\sqrt{T \log(1/\delta')} \right) \\
&\leq T \text{OPT} + T\varepsilon,
\end{aligned} \tag{86}$$

where the penultimate line results from Equation (80). Given that  $h^{\text{final}}$  is taken to be uniformly distributed over  $\{h^t\}_{1 \leq t \leq T}$ , we arrive at

$$\max_{1 \leq i \leq k} L(h^{\text{final}, e_i^{\text{basis}}}) \leq \text{OPT} + \varepsilon, \tag{87}$$

with probability at least  $1 - (k+1)\delta'$ . This concludes the proof by recalling that  $\delta' = \frac{\delta}{4(T+k+1)}$ .

### B.3 Proof of Lemma 6

For any integer  $1 \leq x \leq \log_2(T) + 1$ , define

$$\mathcal{W}_j(x) := \{i \in [k] \mid 2^{x-1} \leq e_i - s_i \leq 2^x\},$$

so that the length of each segment associated with  $\mathcal{W}_j(x)$  lies within  $[2^{x-1}, 2^x]$ . Let  $x^*$  indicate the one that maximizes the cardinality of  $\mathcal{W}_j(x)$ :

$$x^* = \arg \max_{1 \leq x \leq \log_2(T)+1} |\mathcal{W}_j(x)|.$$

Given that there are at most  $\log_2(T) + 1$  choices of  $x$ , the pigeonhole principle gives

$$|\mathcal{W}_j(x^*)| \geq \frac{|\mathcal{W}_j|}{\log_2(T) + 1}. \quad (88)$$

In the sequel, we intend to choose the subsets  $\{\mathcal{W}_j^m\}_{m=1}^M$  from  $\mathcal{W}_j(x^*)$ .

To proceed, let us set

$$\kappa_1 := \min_{i \in \mathcal{W}_j(x^*)} e_i, \quad \mathcal{U}_j^1 := \{i \in \mathcal{W}_j(x^*) \mid s_i \leq \kappa_1\}, \quad (89a)$$

and then for each  $o \geq 1$ , take

$$\kappa_{o+1} := \min_{i \in \mathcal{W}_j(x^*) / \cup_{o'=1}^o \mathcal{U}_j^{o'}} e_i, \quad (89b)$$

$$\mathcal{U}_j^{o+1} := \{i \in \mathcal{W}_j(x^*) / \cup_{o'=1}^o \mathcal{U}_j^{o'} \mid s_i \leq \kappa_{o+1}\}. \quad (89c)$$

We terminate such constructions until  $\cup_{o \geq 1} \mathcal{U}_j^o = \mathcal{W}_j(x^*)$ . By construction, for each  $o$ , we have

$$s_{i_2} \leq \kappa_o \leq e_{i_1}, \quad \forall i_1, i_2 \in \mathcal{U}_j^o \iff \max_{i \in \mathcal{U}_j^o} s_i \leq \min_{i \in \mathcal{U}_j^o} e_i. \quad (90)$$

Let us look at the three groups of subsets of  $\mathcal{W}_j(x^*)$ :  $\{\mathcal{U}_j^{3o-2}\}_{o \geq 1}$ ,  $\{\mathcal{U}_j^{3o-1}\}_{o \geq 1}$  and  $\{\mathcal{U}_j^{3o}\}_{o \geq 1}$ . Clearly, there exists  $l \in \{0, 1, 2\}$  such that  $\sum_{o \geq 1} |\mathcal{U}_j^{3o-l}| \geq \frac{1}{3} \sum_{o \geq 1} |\mathcal{U}_j^o|$ ; without loss of generality, assume that

$$\sum_{o \geq 1} |\mathcal{U}_j^{3o-2}| \geq \frac{1}{3} \sum_{o \geq 1} |\mathcal{U}_j^o| = \frac{1}{3} |\mathcal{W}_j(x^*)|. \quad (91)$$

With the above construction in place, we would like to verify that  $\{\mathcal{U}_j^{3o-2}\}_{o \geq 1}$  forms the desired group of subsets. First of all, Condition (i) holds directly from the definition of  $\{\mathcal{U}_j^o\}_{o \geq 1}$ . When it comes to Condition (ii), it follows from Equation (91) and Equation (88) that

$$\sum_{o \geq 1} |\mathcal{U}_j^{3o-2}| \geq \frac{1}{3} |\mathcal{W}_j(x^*)| \geq \frac{|\mathcal{W}_j|}{3(\log_2(T) + 1)}.$$

Regarding Condition (iii), it suffices to verify that

$$\max_{i \in \mathcal{U}_j^{3o-2}} e_i \leq \min_{i \in \mathcal{U}_j^{3o+1}} s_i, \quad (92)$$

for any  $o$ . To do so, note that for each  $o \geq 1$ , there exists  $i \in \mathcal{W}_j(x^*)$  such that  $s_i \geq \kappa_o$  and  $\kappa_{o+1} = e_i$ . We can then deduce that

$$\kappa_{o+1} = e_i \geq s_i + 2^{x^*-1} \geq \kappa_o + 2^{x^*-1}. \quad (93)$$

It then follows that, for any  $i \in \mathcal{U}_j^{3o+1}$ , one has

$$s_i \geq \kappa_{3o} \geq \kappa_{3o-1} + 2^{x^*-1} \geq \kappa_{3o-2} + 2^{x^*}.$$

In addition, for any  $l \in \mathcal{U}_j^{3o-2}$ , it is seen that

$$e_l \leq s_l + 2^{x^*} \leq \kappa_{3o-2} + 2^{x^*}.$$

Putting all this together yields

$$\max_{i \in \mathcal{U}_j^{3o-2}} e_i \leq \kappa_{3o-2} + 2^{x^*} \leq \min_{i \in \mathcal{U}_j^{3o+1}} s_i.$$

The proof is thus complete.

#### B.4 Proof of Lemma 7

We shall begin by presenting our construction of the subsets, followed by justification of the advertised properties. In what follows, we set  $x = \log(2)$ .

**Our construction.** Let  $\{\mathcal{W}_j^p\}_{p=1}^P$  and  $\{(\tilde{s}_p, \tilde{e}_p)\}_{p=1}^P$  be the construction in Lemma 6.

*Step (a): constructing  $\widehat{\mathcal{W}}_j^p$ .* Consider any  $1 \leq p \leq P$ . Set

$$t_{\text{mid}}^p := \min_{i \in \mathcal{W}_j^p} e_i,$$

which, in view of Lemma 6, is a common inner point of the segments in  $\mathcal{W}_j^p$ . We can derive, for each  $i \in \mathcal{W}_j^p$ ,

$$\max \left\{ \log \left( \frac{w_i^{e_i}}{w_i^{t_{\text{mid}}^p}} \right), \log \left( \frac{w_i^{t_{\text{mid}}^p}}{w_i^{s_i}} \right) \right\} \geq \frac{1}{2} \log \left( \frac{w_i^{e_i}}{w_i^{t_{\text{mid}}^p}} \right) + \frac{1}{2} \log \left( \frac{w_i^{t_{\text{mid}}^p}}{w_i^{s_i}} \right) = \frac{1}{2} \log \left( \frac{w_i^{e_i}}{w_i^{s_i}} \right) \geq \frac{x}{2},$$

where the last inequality holds since  $(s_i, e_i)$  is constructed to be a  $(\frac{1}{2^{j+1}}, \frac{1}{2^{j+2}}, x)$ -segment (see Lemma 5). It then follows that

$$\sum_{i \in \mathcal{W}_j^p} \left( \mathbb{1} \left\{ \log \left( \frac{w_i^{e_i}}{w_i^{t_{\text{mid}}^p}} \right) \geq \frac{x}{2} \right\} + \mathbb{1} \left\{ \log \left( \frac{w_i^{e_i}}{w_i^{t_{\text{mid}}^p}} \right) \geq \frac{x}{2} \right\} \right) \geq |\mathcal{W}_j^p|.$$

Without loss of generality, we assume that

$$\sum_{i \in \mathcal{W}_j^p} \mathbb{1} \left\{ \log \left( \frac{w_i^{e_i}}{w_i^{t_{\text{mid}}^p}} \right) \geq \frac{x}{2} \right\} \geq \frac{|\mathcal{W}_j^p|}{2}. \quad (94)$$

This means that the set define below

$$\widehat{\mathcal{W}}_j^p := \left\{ i \in \mathcal{W}_j^p \mid \log \left( w_i^{e_i} / w_i^{t_{\text{mid}}^p} \right) \geq \frac{x}{2} \right\}, \quad (95)$$

satisfies

$$|\widehat{\mathcal{W}}_j^p| \geq \frac{|\mathcal{W}_j^p|}{2}. \quad (96)$$

In what follows, we take<sup>8</sup>

$$Q(p) := |\widehat{\mathcal{W}}_j^p|, \quad \tilde{l}(p) := \max \{l \geq 0 \mid 2^l \leq Q(p)\} \quad \text{and} \quad \widetilde{Q}(p) := 2^{\tilde{l}(p)}.$$

Without loss of generality, we assume

$$\widehat{\mathcal{W}}_j^p = \{1, 2, \dots, Q(p)\} \quad \text{and} \quad e_1 \leq e_2 \leq \dots \leq e_{Q(p)}. \quad (97)$$

In the sequel, we shall often abbreviate  $Q(p)$ ,  $\tilde{l}(p)$  and  $\widetilde{Q}(p)$  as  $Q$ ,  $\tilde{l}$  and  $\widetilde{Q}$ , respectively, as long as it is clear from the context.

---

<sup>8</sup>We assume  $\widetilde{Q} \geq 2$  without loss of generality. In the case  $\widetilde{Q} = 1$ , we simply choose an arbitrary element in  $\widehat{\mathcal{W}}_j^p$  as a single subset. In this way, we can collect at least  $\frac{1}{4}|\widehat{\mathcal{W}}_j^p|$  segments.

*Step (b): constructing  $\mathcal{K}_l$  and  $\tilde{\mathcal{W}}_j^p(l)$ .* Let us take  $e_0 = t_{\text{mid}}^p$ , and employ  $[e_0, e_k] \oplus a$  as a shorthand notation for  $[e_a, e_{k+a}]$ . We can then define a group of disjoint intervals of  $[T]$  as follows:

$$\mathcal{K}_1 = \{[e_0, e_{2^{\tilde{l}-1}}]\}; \quad (98a)$$

$$\mathcal{K}_2 = \{[e_0, e_{2^{\tilde{l}-2}}], [e_0, e_{2^{\tilde{l}-2}}] \oplus 2^{\tilde{l}-1}\}; \quad (98b)$$

$$\mathcal{K}_3 = \{[e_0, e_{2^{\tilde{l}-3}}], [e_0, e_{2^{\tilde{l}-3}}] \oplus 2^{\tilde{l}-2}, [e_0, e_{2^{\tilde{l}-3}}] \oplus 2 \cdot 2^{\tilde{l}-2}, [e_0, e_{2^{\tilde{l}-3}}] \oplus 3 \cdot 2^{\tilde{l}-2}\}; \quad (98c)$$

...

$$\mathcal{K}_l = \{[e_0, e_{2^{\tilde{l}-l}}], [e_0, e_{2^{\tilde{l}-l}}] \oplus 2^{\tilde{l}-l+1}, [e_0, e_{2^{\tilde{l}-l}}] \oplus 2 \cdot 2^{\tilde{l}-l+1}, \dots, [e_0, e_{2^{\tilde{l}-l}}] \oplus (2^{l-1} - 1)2^{\tilde{l}-l+1}\}; \quad (98d)$$

...

$$\mathcal{K}_{\tilde{l}} = \{[e_{2i}, e_{2i+1}] \mid i = 0, 1, 2, \dots, 2^{\tilde{l}-1} - 1\}; \quad (98e)$$

$$\mathcal{K}_{\tilde{l}+1} = \{[e_{2i+1}, e_{2i+2}] \mid i = 0, 1, 2, \dots, 2^{\tilde{l}-1} - 1\}. \quad (98f)$$

For each  $i \in [\tilde{Q} - 1]$  with binary form  $\{i_l\}_{l=1}^{\tilde{l}}$  and  $0 \leq l \leq \tilde{l}$ , we define  $\text{trunc}(i, l)$  to be the number with binary form  $\{i_1, i_2, \dots, i_l, 0, 0, \dots, 0\}$ . For example,  $\text{trunc}(i, 0) = 0$  and  $\text{trunc}(i, \tilde{l}) = i$ .

From the definition Equation (95) of  $\tilde{\mathcal{W}}_j^p$ , we know that for each  $i \in [\tilde{Q} - 1]$ ,

$$\frac{x}{2} \leq \log \left( \frac{w_i^{e_i}}{w_i^{e_0}} \right) = \sum_{l=1}^{\tilde{l}} \log \left( \frac{w_i^{e_{\text{trunc}(i,l)}}}{w_i^{e_{\text{trunc}(i,l-1)}}} \right) = \sum_{l=1}^{\tilde{l}} \log \left( \frac{w_i^{e_{\text{trunc}(i,l)}}}{w_i^{e_{\text{trunc}(i,l-1)}}} \right) \mathbb{1} \{e_{\text{trunc}(i,l)} \neq e_{\text{trunc}(i,l-1)}\}, \quad (99)$$

which in turn implies that

$$\max_{1 \leq l \leq \tilde{l}} \log \left( \frac{w_i^{e_{\text{trunc}(i,l)}}}{w_i^{e_{\text{trunc}(i,l-1)}}} \right) \geq \frac{x}{2\tilde{l}}. \quad (100)$$

By defining

$$\tilde{\mathcal{W}}_j^p(l) := \left\{ i \in \tilde{\mathcal{W}}_j^p : \arg \max_{1 \leq l' \leq \tilde{l}} \log \left( \frac{w_i^{e_{\text{trunc}(i,l')}}}{w_i^{e_{\text{trunc}(i,l'-1)}}} \right) = l \right\},$$

for each<sup>9</sup>  $1 \leq l \leq \tilde{l}$ , we can demonstrate that

$$\sum_{l=1}^{\tilde{l}} |\tilde{\mathcal{W}}_j^p(l)| \geq \tilde{Q} - 1, \quad (101)$$

thus implying the existence of some  $1 \leq l^* \leq \tilde{l}$  obeying

$$|\tilde{\mathcal{W}}_j^p(l^*)| \geq \frac{\tilde{Q} - 1}{\tilde{l}} \geq \frac{\tilde{Q}}{2\tilde{l}}. \quad (102)$$

*Step (c): constructing  $\tilde{\mathcal{W}}_j^p(l, o)$ ,  $\hat{s}(p, o)$  and  $\hat{e}(p, o)$ .* By definition, for any  $i$ , if  $\text{trunc}(i, l^*) \neq \text{trunc}(i, l^* - 1)$ , then one has

$$[e_{\text{trunc}(i,l^*-1)}, e_{\text{trunc}(i,l^*)}] \in \mathcal{K}_{l^*},$$

<sup>9</sup>Without loss of generality, we assume the arg max function is a single-valued function.

where the set  $\mathcal{K}_l$  has been defined in Equation (98). In addition, from the construction of  $\tilde{\mathcal{W}}_j^p(l^*)$  (see Equation (102)), we know that  $\text{trunc}(i, l^*) \neq \text{trunc}(i, l^* - 1)$  for any  $i \in \tilde{\mathcal{W}}_j^p(l^*)$ . For each  $1 \leq o \leq 2^{l^*-1}$ , define

$$\tilde{\mathcal{W}}_j^p(l^*, o) := \left\{ i \in \tilde{\mathcal{W}}_j^p(l^*) \mid [e_{\text{trunc}(i, l^*-1)}, e_{\text{trunc}(i, l^*)}] = [e_0, e_{2^{\tilde{l}}-l^*}] \oplus (o-1)2^{\tilde{l}-l^*+1} \right\}, \quad (103)$$

where we employ the notation  $l^*$  and  $\tilde{l}$  to abbreviate  $l^*(p)$  and  $\tilde{l}(p)$ , respectively.

In addition, for any  $1 \leq p \leq P$  and  $1 \leq o \leq 2^{l^*(p)-1}$ , we set

$$\hat{s}(p, o) = e_{(o-1)2^{\tilde{l}}(p)-l^*(p)+1}, \quad (104a)$$

$$\hat{e}(p, o) = e_{2^{\tilde{l}}(p)-l^*(p)+(o-1)2^{\tilde{l}}(p)-l^*(p)+1}. \quad (104b)$$

In words,  $[\hat{s}(p, o), \hat{e}(p, o)]$  can be understood as the  $o$ th interval in the set  $\mathcal{K}_{l^*(p)}$ .

*Step (d): construction output.* With the above construction in mind, we would like to select

$$\left\{ \left\{ \tilde{\mathcal{W}}_j^p(l^*(p), o) \right\}_{o=1}^{2^{l^*(p)-1}} \right\}_{p=1}^P \quad \text{with intervals} \quad \left\{ \left\{ \hat{s}(p, o), \hat{e}(p, o) \right\}_{o=1}^{2^{l^*(p)-1}} \right\}_{p=1}^P$$

as the group of subsets we construct. With slight abuse of notation, we use  $(p, o)$  as the index of the segments instead of  $n$ . In what follows, we validate this construction.

**Verification of the advertised properties.** We now proceed to justify the claimed properties.

**Property (i).** By construction, it is clearly seen that

$$\tilde{\mathcal{W}}_j^p(l^*(p), o) \subseteq \tilde{\mathcal{W}}_j^p(l^*(p)) \subseteq \widehat{\mathcal{W}}_j^p \subseteq \mathcal{W}_j^p \subseteq \mathcal{W}_j.$$

In addition, if

$$\tilde{\mathcal{W}}_j^{p_1}(l^*(p_1), o_1) \cap \tilde{\mathcal{W}}_j^{p_2}(l^*(p_2), o_2) \neq \emptyset,$$

then one has  $\mathcal{W}_j^{p_2} \cap \mathcal{W}_j^{p_2} \neq \emptyset$ , and as a result,  $p_1 = p_2$  (otherwise it violates the condition that  $\mathcal{W}_j^{p_2} \cap \mathcal{W}_j^{p_2} = \emptyset$  for  $p_1 \neq p_2$ ). It also follows from the definition in Equation (103) that  $o_1 = o_2$ . Therefore, for any  $(p_1, o_1)$  that does not equal  $(p_2, o_2)$ , we have  $\tilde{\mathcal{W}}_j^{p_1}(l^*(p_1), o_1) \cap \tilde{\mathcal{W}}_j^{p_2}(l^*(p_2), o_2) = \emptyset$ .

**Property (ii).** By construction, we have

$$\sum_{o=1}^{2^{l^*(p)-1}} |\tilde{\mathcal{W}}_j^p(l^*(p), o)| = |\tilde{\mathcal{W}}_j^p(l^*(p))| \geq \frac{|\widehat{\mathcal{W}}_j^p|}{4 \log_2(|\widehat{\mathcal{W}}_j^p|)} \geq \frac{|\mathcal{W}_j^p|}{8 \log_2(|\widehat{\mathcal{W}}_j^p|)}, \quad (105)$$

where we have made use of Equation (102) and Equation (96). Summing over  $p$  and applying Lemma 6 yield

$$\sum_{p=1}^P \sum_{o=1}^{2^{l^*(p)-1}} |\tilde{\mathcal{W}}_j^p(l^*(p), o)| \geq \sum_{p=1}^P \frac{|\mathcal{W}_j^p|}{8 \log_2(k)} \geq \frac{|\mathcal{W}_j|}{24 \log_2(k)(\log_2(T) + 1)}. \quad (106)$$

**Property (iii)(a).** Let us set the parameters  $\left\{ \left\{ g(p, o) \right\}_{o=1}^{2^{l^*(p)}} \right\}_{p=1}^P$  as follows:

$$g(p, o) = \frac{\sum_{i \in \tilde{\mathcal{W}}_j(l^*(p), o)} \hat{s}(p, o)}{2^{-(j+2)} \cdot |\tilde{\mathcal{W}}_j^p(l^*(p), o)|} \geq 1,$$

where the last inequality holds since, by construction,  $w_i^{\hat{s}(p,o)} \geq 2^{-(j+2)}$  (see Lemma 5). Then Property (iii)(a) is satisfied since

$$\sum_{i \in \tilde{\mathcal{W}}_j(l^*(p), o)} w_i^{\hat{s}(p,o)} = \frac{g(p, o) |\tilde{\mathcal{W}}_j(l^*(p), o)|}{2^{j+2}}.$$

**Property (iii)(b).** For any  $i \in \tilde{\mathcal{W}}_j^p \subseteq \mathcal{W}_j^p$ , we have

$$s_i \leq e_{\text{trunc}(i, l-1)} \leq e_i \quad \text{for any } 1 \leq l \leq \tilde{l}(p),$$

which is valid since  $\max_{i \in \mathcal{W}_j^p} s_i \leq \min_{i \in \mathcal{W}_j^p} e_i$  (see Lemma 6) and Equation (97). It then holds that

$$s_i \leq \hat{s}(p, o) \leq e_i \quad \text{for any } i \in \tilde{\mathcal{W}}_j^p.$$

Also, the construction of  $(s_i, e_i)$  (see Lemma 5) tells us that  $w_i^{\hat{s}(p,o)} \geq 2^{-(j+2)}$ .

Moreover, by construction, we know that for any  $i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)$ ,

$$\log\left(\frac{w_i^{\hat{e}(p,o)}}{w_i^{\hat{s}(p,o)}}\right) \geq \frac{x}{2\tilde{l}(p)} \quad \text{and} \quad w_i^{\hat{s}(p,o)} \geq 2^{-(j+2)}.$$

Recalling that  $x = \log(2)$ , one can further derive

$$\begin{aligned} \sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^{\hat{s}(p,o)} &\geq 2^{-(j+2)} \cdot |\tilde{\mathcal{W}}_j^p(l^*(p), o)| \\ \log\left(\frac{\sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^{\hat{e}(p,o)}}{\sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^{\hat{s}(p,o)}}\right) &\geq \log\left(\frac{\sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^{\hat{s}(p,o)} \cdot \exp\left(\frac{x}{2\tilde{l}(p)}\right)}{\sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^{\hat{s}(p,o)}}\right) = \frac{x}{2\tilde{l}(p)} \geq \frac{\log(2)}{2\log_2(k)}. \end{aligned}$$

**Property (iii)(c).** Note that for any  $t$  obeying  $\hat{s}(p, o) \leq t \leq \hat{e}(p, o)$ , and any  $i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)$ , it holds that  $s_i \leq \hat{s}(p, o) \leq t \leq \hat{e}(p, o) \leq e_i$ . Recall that  $w_i^t \geq 2^{-(j+2)}$  for any  $t \in [s_i, e_i]$  (see Lemma 5). As a result,

$$\sum_{i \in \tilde{\mathcal{W}}_j^p(l^*(p), o)} w_i^t \geq |\tilde{\mathcal{W}}_j^p(l^*(p), o)| \cdot 2^{-(j+2)}.$$

**Proper ordering.** To finish up, it remains to verify that the intersection of  $[\hat{s}(p_1, o_1), \hat{e}(p_1, o_1)]$  and  $[\hat{s}(p_2, o_2), \hat{e}(p_2, o_2)]$  is either empty or contains only the boundary points, unless  $(p_1, o_1) = (p_2, o_2)$ . To show this, note that in the case where  $p_1 \neq p_2$  (assuming  $p_1 < p_2$ ), we have

$$\tilde{s}_{p_1} \leq \hat{s}(p_1, o_1) < \hat{e}(p_1, o_1) \leq \tilde{e}_{p_1} \leq \tilde{s}_{p_2} \leq \hat{s}(p_2, o_2) < \hat{e}(p_2, o_2),$$

which arises from Lemma 6. Also, in the case where  $p_1 = p_2 = p$  and  $o_1 \neq o_2$  (assuming  $o_1 < o_2$ ), we have

$$\hat{s}(p, o_1) < \hat{e}(p, o_1) < \hat{s}(p, o_2) < \hat{e}(p, o_2),$$

which comes from the construction Equation (104).

We have thus completed the proof of this lemma.

## B.5 Proof of Lemma 8

Throughout this proof, we find it convenient to denote  $Z^t = \sum_{i=1}^k W_i^t$ .

**B.5.1 Proof of the First Claim Equation (39).** We start by proving the first claim Equation (39). Recall that  $[t_1, t_2]$  is assumed to be a  $(p, q, x)$ -segment. From the definition of the segment (see Definition 1), there exists  $i \in [k]$  such that

$$\log\left(\frac{w_i^{t_2}}{w_i^{t_1}}\right) \geq x.$$

Given that  $W_i^{t_2} = W_i^{t_1} \exp(\eta \sum_{\tau=t_1}^{t_2-1} \hat{r}_i^\tau)$  and  $w_t = W_t/Z_t$  (see lines 15 and 5 of Algorithm 1), the above inequality can be equivalently expressed as

$$\eta \sum_{\tau=t_1}^{t_2-1} \hat{r}_i^\tau - \log(Z^{t_2}/Z^{t_1}) \geq x. \quad (107)$$

Moreover, recognizing that

$$\log(Z^{t_2}/Z^{t_1}) = \log\left(\frac{\sum_{i \in [k]} W_i^{t_1} \exp(\eta \sum_{\tau=t_1}^{t_2-1} \hat{r}_i^\tau)}{\sum_{i \in [k]} W_i^{t_1}}\right) \geq -\eta(t_2 - t_1),$$

and  $\hat{r}_i^\tau \leq 1$  for any  $1 \leq \tau \leq T$ , we can invoke Equation (107) to show that

$$x \leq 2(t_2 - t_1)\eta, \quad (108)$$

from which the claimed inequality Equation (39) follows.

**B.5.2 Proof of the Remaining Claims.** We now turn to the remaining claims of Lemma 8. For each hypothesis  $h \in \mathcal{H}$ , let us introduce the following vector  $\lambda_h \in \mathbb{R}^k$ :

$$\lambda_h = [\lambda_{h,i}]_{i \in [k]} \quad \text{with } \lambda_{h,i} = L(h, e_i^{\text{basis}}) - \text{OPT}. \quad (109)$$

Given the  $\varepsilon_1$ -optimality of  $h^t$  (see Lemma 3), we have the following property that holds for any  $1 \leq \tau, t \leq T$ :

$$\langle \lambda_{h^\tau}, w^t \rangle \geq \min_{h \in \mathcal{H}} \langle \lambda_h, w^t \rangle \geq \langle \lambda_{h^t}, w^t \rangle - \varepsilon_1 = v^t - \varepsilon_1, \quad (110)$$

where we recall the definition of  $v^t$  in Equation (38). In the sequel, we divide the proof into a couple of steps.

**Step 1: decomposing the KL divergence between  $w^t$  and  $w^{t_2}$ .** Let us write

$$W_i^t = \exp\left(\eta \sum_{\tau=1}^t \hat{r}_i^\tau\right) = \exp\left(\eta \sum_{\tau=1}^t (\lambda_{h^\tau, i} + \text{OPT} + \xi_i^\tau)\right) \quad \text{with } \xi_i^\tau = \hat{r}_i^\tau - \lambda_{h^\tau, i} - \text{OPT},$$

where  $\xi_i^\tau$  is clearly a zero-mean random variable. Define

$$\Delta_{t_1, t_2} = \sum_{\tau=t_1}^{t_2-1} \xi_i^\tau \quad \text{with } \xi^\tau = [\xi_i^\tau]_{i \in [k]}.$$

Taking  $W^t = [W_i^t]_{i \in [k]}$  and denoting by  $\log(x/y)$  the vector  $\{\log(x_i/y_i)\}_{i \in [k]}$  for two  $k$ -dimensional vectors  $(x, y)$ , one can then deduce that

$$\left\langle \frac{1}{\eta} \log\left(\frac{W^{t_2}}{W^{t_1}}\right) - \Delta_{t_1, t_2}, w^t \right\rangle - (t_2 - t_1)\text{OPT} = \sum_{\tau=t_1}^{t_2-1} \langle \lambda_{h^\tau}, w^t \rangle \geq (t_2 - t_1)(v^t - \varepsilon_1), \quad (111)$$

where the last inequality results from Equation (110).

Recall that  $Z^t = \sum_{i=1}^k W_i^t$  and  $w_i^t = \frac{W_i^t}{Z^t}$ . By taking  $t_1 = t$ , we can derive from Equation (111) that

$$\left\langle \log\left(\frac{w^{t_2}}{w^t}\right) - \eta\Delta_{t,t_2}, w^t \right\rangle + \log\left(\frac{Z^{t_2}}{Z^t}\right) - \eta(t_2 - t)\text{OPT} \geq \eta(t_2 - t)(v^t - \varepsilon_1). \quad (112)$$

As it turns out, this inequality allows us to bound the KL divergence between  $w^t$  and  $w^{t_2}$  as follows:

$$\begin{aligned} \text{KL}(w^t \| w^{t_2}) &:= \left\langle w^t, \log\left(\frac{w^t}{w^{t_2}}\right) \right\rangle \\ &\leq \log(Z^{t_2}/Z^t) - \eta(t_2 - t)\text{OPT} - \eta\langle w^t, \Delta_{t,t_2} \rangle + \eta(t_2 - t)(\varepsilon_1 - v^t). \end{aligned} \quad (113)$$

In what follows, we shall cope with the right-hand side of Equation (113).

**Step 2: bounding the term  $\log(Z^{t_2}/Z^t)$ .** With probability exceeding  $1 - 2T^2k\delta'$ , it holds that

$$\begin{aligned} \log(Z^{t_2}/Z^t) &= \sum_{\tau=t}^{t_2-1} \log(Z^{\tau+1}/Z^\tau) = \sum_{\tau=t}^{t_2-1} \log\left(\sum_{i \in [k]} \frac{W_i^\tau \exp(\eta \hat{r}_i^\tau)}{\sum_{j \in [k]} W_j^\tau}\right) \\ &\stackrel{(i)}{=} \sum_{\tau=t}^{t_2-1} \log\left(\sum_{i=1}^k w_i^\tau \exp(\eta \hat{r}_i^\tau)\right) \stackrel{(ii)}{\leq} \sum_{\tau=t}^{t_2-1} \log\left(\sum_{i=1}^k w_i^\tau + \sum_{i=1}^k w_i^\tau (\eta \hat{r}_i^\tau) + 2 \sum_{i=1}^k w_i^\tau \eta^2 (\hat{r}_i^\tau)^2\right) \\ &\stackrel{(iii)}{\leq} \sum_{\tau=t}^{t_2-1} \log\left(1 + \eta \sum_{i=1}^k w_i^\tau \hat{r}_i^\tau + 2\eta^2\right) \leq \sum_{\tau=t}^{t_2-1} \left(\eta \sum_{i=1}^k w_i^\tau \hat{r}_i^\tau + 2\eta^2\right) \\ &\stackrel{(iv)}{=} \eta \sum_{\tau=t}^{t_2-1} v^\tau + \eta(t_2 - t)\text{OPT} + \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \cdot (\hat{r}_i^\tau - \lambda_{h^\tau,i} - \text{OPT}) + 2(t_2 - t)\eta^2 \\ &\stackrel{(v)}{\leq} \eta(t_2 - t)\varepsilon_1 + \eta(t_2 - t)\text{OPT} + \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \cdot (\hat{r}_i^\tau - \lambda_{h^\tau,i} - \text{OPT}) + 2(t_2 - t)\eta^2. \end{aligned} \quad (114)$$

Here, (i) comes from line 5 of Algorithm 1, (ii) follows from the elementary inequality  $\exp(x) \leq 1 + x + 2x^2$  for any  $x \leq 1$ , (iii) is valid since  $\sum_i w_i^\tau = 1$  and  $|\hat{r}_i^\tau| \leq 1$ , (iv) holds due to the fact that  $v^\tau = \langle w^\tau, \lambda_{h^\tau} \rangle$ , and (v) arises from the fact that  $v^\tau \leq \varepsilon_1$  (see Equation (80)).

**Step 3: bounding the weighted sum of  $\{\xi_i^\tau\}$ .** Next, we intend to control the two random terms below:

$$\eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \cdot (\hat{r}_i^\tau - \lambda_{h^\tau,i} - \text{OPT}) = \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \xi_i^\tau, \quad (115a)$$

$$\eta \langle w^\tau, \Delta_{t,t_2} \rangle = \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \xi_i^\tau. \quad (115b)$$

Let  $\mathcal{F}^\tau$  denote what happens before the  $\tau$ th round in Algorithm 1. Two properties are worth noting.

- The variance of  $\xi_i^\tau$  is at most  $O(\frac{1}{k\bar{w}_i^\tau})$ , according to the update rule (see line 14 in Algorithm 1);
- $\{\xi_i^\tau\}_{i \in [k]}$  are independent conditioned on  $\mathcal{F}^\tau$ .

Let us develop bounds on the two quantities in Equation (115) below.

- Letting  $q^\tau = \sum_{i=1}^k w_i^\tau \xi_i^\tau$ , one sees that

$$|q^\tau| \leq 1, \quad \mathbb{E}[q^\tau | \mathcal{F}^\tau] = 0 \quad \text{and} \quad \text{Var}[q^\tau | \mathcal{F}^\tau] \leq \sum_{i=1}^k \frac{(w_i^\tau)^2}{k\bar{w}_i^\tau} \leq \sum_{i=1}^k \frac{w_i^\tau}{k} = \frac{1}{k}. \quad (116)$$

By virtue of Freedman's inequality (cf. Lemma 12), with probability at least  $1 - \delta'$  one has

$$\left| \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \xi_i^\tau \right| \leq 2 \sqrt{\frac{t_2-t}{k} \log(2/\delta')} + 2 \log(2/\delta'); \quad (117)$$

– Regarding the other term, by letting  $\hat{q}^\tau = \sum_{i=1}^k w_i^\tau \xi_i^\tau$ , we have

$$|\hat{q}^\tau| \leq 1, \quad \mathbb{E}[\hat{q}^\tau | \mathcal{F}^\tau] = 0 \quad \text{and} \quad \text{Var}[\hat{q}^\tau | \mathcal{F}^\tau] \leq \sum_{i=1}^k \frac{(w_i^\tau)^2}{k w_i^\tau} \leq \sum_{i=1}^k \frac{w_i^\tau}{k} = \frac{1}{k}.$$

Invoke Freedman's inequality (cf. Lemma 12) once again to show that, with probability exceeding  $1 - \delta'$ ,

$$\left| \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k w_i^\tau \xi_i^\tau \right| \leq 2 \sqrt{\frac{t_2-t}{k} \log(2/\delta')} + 2 \log(2/\delta'). \quad (118)$$

**Step 4: bounding the KL divergence between  $w^t$  and  $w^{t_2}$ .** Combining Equation (113), Equation (114), Equation (117) and Equation (118), and applying the union bound over  $(t, t_2)$ , we can demonstrate that with probability at least  $1 - 6T^4 k \delta'$ ,

$$\begin{aligned} \text{KL}(w^t \| w^{t_2}) &\leq 2(t_2 - t)\eta\varepsilon_1 - (t_2 - t)\eta v^t \\ &\quad + 4\eta \sqrt{\frac{(t_2 - t)\log(2/\delta')}{k}} + 2(t_2 - t)\eta^2 + 4\eta \log(2/\delta') \end{aligned} \quad (119)$$

holds for any  $1 \leq t < t_2 \leq T$ . The analysis below then operates under the condition that Equation (119) holds for any  $1 \leq t < t_2 \leq T$ .

**Step 5: connecting the KL divergence with the advertised properties.** Set

$$\tau_{\hat{j}} := \min\{\tau \mid t_1 \leq \tau \leq t_2 - 1, -v^\tau \leq 2^{-(\hat{j}-1)}\}, \quad 1 \leq \hat{j} \leq j_{\max} := \left\lceil \log_2(1/\eta) + 1 \right\rceil; \quad (120a)$$

$$\tau_{j_{\max}+1} := t_2. \quad (120b)$$

By definition, we know that  $\tau_1 = t_1$  and  $\tau_{j_2} \geq \tau_{j_1}$  for  $j_2 \geq j_1$ . Let  $\mathcal{J}$  be the index set associated with this segment  $[t_1, t_2]$ , and set  $y_j := \sum_{i \in \mathcal{J}} w_i^j$ . We then claim that there exists  $1 \leq \tilde{j} \leq j_{\max}$  such that

$$\log\left(\frac{y_{\tilde{j}+1}}{y_{\tilde{j}}}\right) \geq \frac{x}{\log_2(1/\eta) + 1}. \quad (121)$$

**PROOF OF EQUATION (121).** Suppose that none of  $1 \leq \tilde{j} \leq j_{\max}$  satisfies Equation (121). Then for any  $1 \leq \hat{j} \leq j_{\max}$ , it holds that  $\log\left(\frac{y_{\hat{j}+1}}{y_{\hat{j}}}\right) < \frac{x}{\log_2(1/\eta) + 1}$ , which implies that  $y_{\hat{j}} > y_{\hat{j}+1} \exp(-\frac{x}{\log_2(1/\eta) + 1})$ . As a result, we have

$$\begin{aligned} y_1 &> y_{j_{\max}+1} \cdot \exp\left(-j_{\max} \cdot \frac{x}{\log_2(1/\eta) + 1}\right) = \left(\sum_{i \in \mathcal{J}} w_i^{t_2}\right) \cdot \exp\left(-j_{\max} \cdot \frac{x}{\log_2(1/\eta) + 1}\right) \\ &\geq p \exp(x) \cdot \exp(-x) = p, \end{aligned}$$

thus leading to contradiction (as according to the definition of the  $(p, q, x)$ -segment, one has  $y_1 \leq p$ ).  $\square$

Now, assume that  $\tilde{j}$  satisfies Equation (121). From the definition of the  $(p, q, x)$ -segment, we have  $y_{\tilde{j}} \geq q$ . It follows from Equation (119) that

$$\begin{aligned} \text{KL}(w^{\tau_{\tilde{j}}} \| w^{\tau_{\tilde{j}+1}}) &\leq 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta\varepsilon_1 + (\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta 2^{-(\tilde{j}-1)} \\ &\quad + 4\eta\sqrt{\frac{(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\log(2/\delta')}{k}} + 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta^2 + 4\eta\log(2/\delta'). \end{aligned} \quad (122)$$

Since  $\log(\frac{y_{\tilde{j}+1}}{y_{\tilde{j}}}) \geq \frac{x}{\log_2(1/\eta)+1}$  and  $y_{\tilde{j}} \geq q$ , we can invoke Lemma 15 and Lemma 16 to show that

$$\text{KL}(w^{\tau_{\tilde{j}}} \| w^{\tau_{\tilde{j}+1}}) \geq \text{KL}\left(\text{Ber}(y_{\tilde{j}}) \| \text{Ber}(y_{\tilde{j}+1})\right) \geq \frac{qx^2}{4(\log_2(1/\eta)+1)^2},$$

where  $\text{Ber}(x)$  denote the Bernoulli distribution with mean  $x \in [0, 1]$ . As a result, we can obtain

$$\begin{aligned} \frac{qx^2}{4(\log_2(1/\eta)+1)^2} &\leq 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta\varepsilon_1 + (\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta 2^{-(\tilde{j}-1)} \\ &\quad + 4\eta\sqrt{\frac{(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\log(2/\delta')}{k}} + 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta^2 + 4\eta\log(2/\delta'), \end{aligned} \quad (123)$$

which in turn results in

$$\begin{aligned} \tau_{\tilde{j}+1} - \tau_{\tilde{j}} &\geq \min\left\{\frac{qx^2}{100(\log_2(1/\eta)+1)^2} \min\left\{\frac{1}{\eta\varepsilon_1}, \frac{2^{\tilde{j}-1}}{\eta}, \frac{1}{\eta^2}\right\}, \frac{kq^2x^4}{10000\eta^2\log(1/\delta')(\log_2(1/\eta)+1)^4}\right\} \\ &= \frac{qx^2}{100(\log_2(1/\eta)+1)^2} \min\left\{\frac{2^{\tilde{j}-1}}{\eta}, \frac{1}{\eta^2}\right\} \end{aligned} \quad (124)$$

$$= \frac{qx^2}{100(\log_2(1/\eta)+1)^2} \frac{2^{\tilde{j}-1}}{\eta}. \quad (125)$$

Here, to see why Equation (124) and Equation (125) hold, it suffices to note that

$$\frac{qx^2}{100} \cdot \frac{2^{\tilde{j}-1}}{\eta} \leq \frac{qx^2}{100} \cdot \frac{1}{\eta^2} \leq \frac{kq^2x^4}{10000\eta^2\log(1/\delta')(\log_2(1/\eta)+1)^2},$$

a property that arises from the fact that  $2^{\tilde{j}-1} \leq 1/\eta$  and the assumption that  $\frac{qx^2}{50(\log_2(1/\eta)+1)^2} \geq \frac{1}{k}$ .

With Equation (125) in mind, we are ready to finish the proof by dividing into two cases.

– Case 1:  $\tilde{j} = j_{\max}$ . It follows from Equation (125) that

$$t_2 - t_1 \geq \tau_{\tilde{j}+1} - \tau_{\tilde{j}} \geq \frac{qx^2}{100(\log_2(1/\eta)+1)^2} \frac{2^{j_{\max}-1}}{\eta} \geq \frac{qx^2}{200(\log_2(1/\eta)+1)^2 \eta^2}.$$

– Case 2:  $1 \leq \tilde{j} \leq j_{\max} - 1$ . It comes from the definition Equation (120) that

$$\sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\hat{j}}\} \geq \sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau > 2^{-\hat{j}}\} \geq \tau_{\tilde{j}+1} - t_1 \geq \tau_{\tilde{j}+1} - \tau_{\tilde{j}} \quad \text{for any } 1 \leq \hat{j} \leq j_{\max} - 1.$$

When  $1 \leq \tilde{j} \leq j_{\max} - 1$ , the above display taken collectively with Equation (125) gives

$$\sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\tilde{j}}\} \geq \tau_{\tilde{j}+1} - \tau_{\tilde{j}} \geq \frac{qx^2 \cdot 2^{\tilde{j}-1}}{100(\log_2(1/\eta) + 1)^2 \eta}, \quad (126)$$

and as a result,

$$\sum_{\tau=t_1}^{t_2-1} \sum_{\hat{j}=1}^{j_{\max}-1} \mathbb{1}\{\max\{-v^\tau, 0\} \geq 2^{-\hat{j}}\} 2^{-(\hat{j}-1)} \geq \sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\tilde{j}}\} 2^{-(\tilde{j}-1)} \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}. \quad (127)$$

By observing that

$$\sum_{\hat{j}=1}^{\infty} \mathbb{1}\{z \geq 2^{-\hat{j}}\} \cdot 2^{-(\hat{j}-1)} \leq 4z$$

holds for any  $z \geq 0$ , we can combine this fact with Equation (127) to derive

$$4 \sum_{\tau=t_1}^{t_2-1} \max\{-v^\tau, 0\} \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}. \quad (128)$$

Furthermore, recalling that  $v^\tau \leq \varepsilon_1$  (cf. Equation (80)), one can deduce that

$$\sum_{\tau=t_1}^{t_2-1} \max\{-v^\tau, 0\} = \sum_{\tau=t_1}^{t_2-1} (-v^\tau) - \sum_{\tau=t_1}^{t_2-1} \min\{-v^\tau, 0\} \leq \sum_{\tau=t_1}^{t_2-1} (-v^\tau) + (t_2 - t_1)\varepsilon_1,$$

which combined with Equation (128) yields

$$4(t_2 - t_1)\varepsilon_1 + 4 \sum_{\tau=t_1}^{t_2-1} (-v^\tau) \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}.$$

The proof is thus complete.

## B.6 Additional Illustrative Figures for Segment Construction

In this section, we provide several illustrative figures to help the readers understand the concept of “segments” and certain key properties, which play an important role in the segment construction described in Section 4.2.2 and the proof of Lemma 7.

- Figure 1 illustrates an example in which any two segments either coincide or are disjoint. In this case, our heuristic arguments in Section 4.2.2 about “shared intervals” and “disjoint intervals” become applicable.
- Figure 2 gives an example where two non-identical segments might overlap. Due to the presence of non-disjoint segments, our heuristic arguments in Section 4.2.2 about “shared intervals” and “disjoint intervals” are not readily applicable.
- In Figure 3, we provide an example of the partition of blocks (as in the proof of Lemma 6), whereas in Figure 4, we illustrate how to align one side of the segments using a common inner point (as in the proof of Lemma 7).
- In Figures 5 and 6, we illustrate how to construct disjoint segments from a group of segments with common starting points in the case with  $k = 8$ . In this particular example, we have in total 5 groups of disjoint segments, marked with different colors.

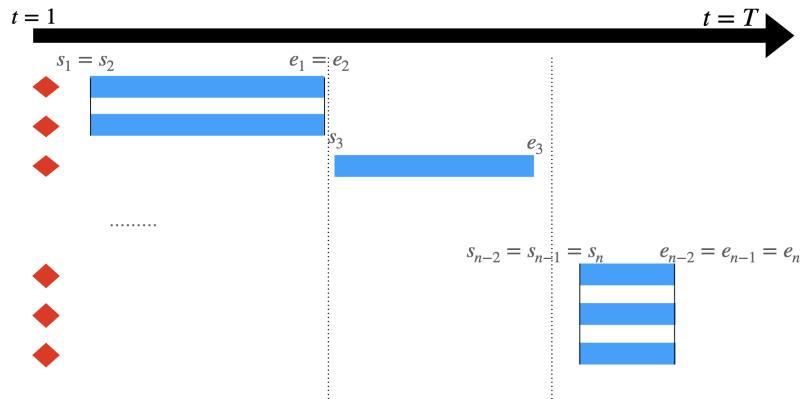


Fig. 1. An example where any two segments either coincide or are disjoint.

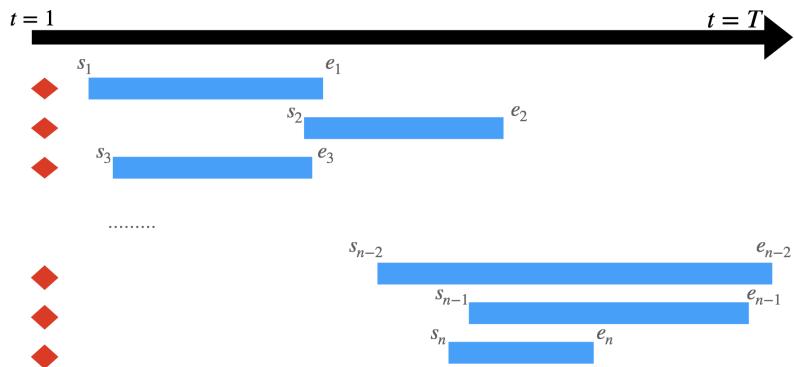


Fig. 2. An example where two non-identical segments might overlap.

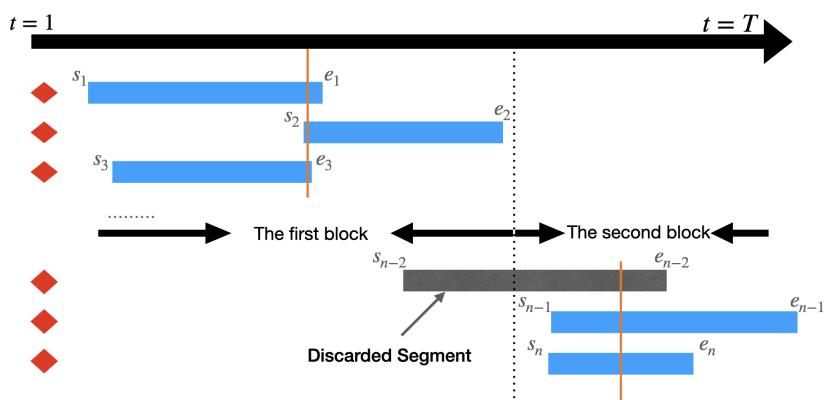


Fig. 3. Partition of blocks as in the proof of Lemma 7.

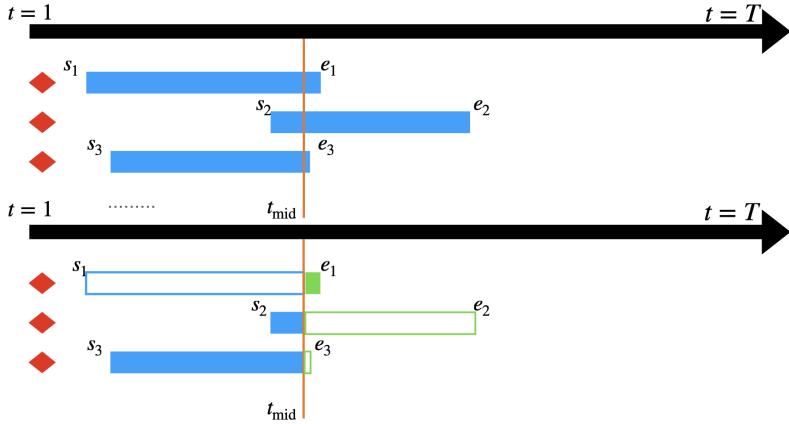


Fig. 4. An illustration of how we use a common inner point (i.e.,  $t_{mid}$  to align one side of the segments (as in the proof of Lemma 7). The unfilled part of the segments means that the weight changes from  $w_i^{s_i}$  to  $w_i^{t_{mid}}$  (i.e.,  $\log(w_i^{t_{mid}} / w_i^{s_i})$ ) are not significant enough.

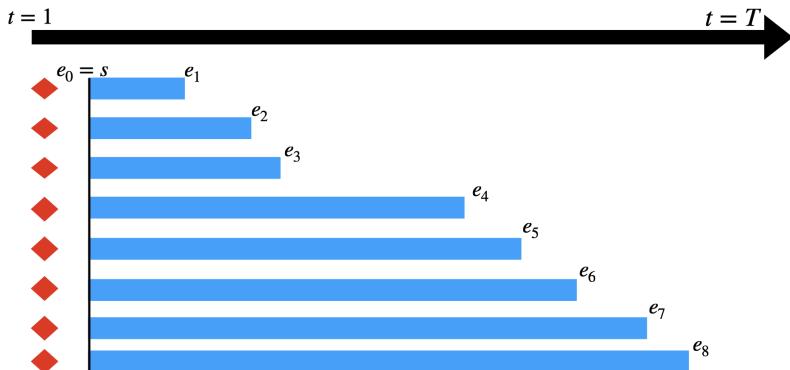


Fig. 5. A group of segments with common starting points.

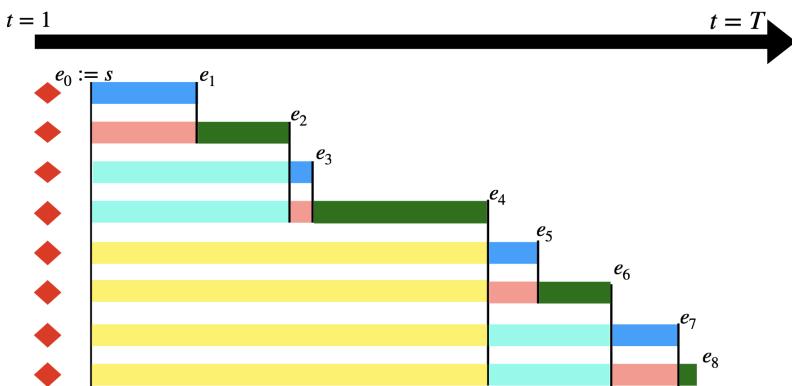


Fig. 6. Construction of groups of disjoint segments, where each of the original segments is divided into at most  $\log_2(k) + 1$  sub-segments marked with different colors.

## C Proofs of the Lower Bound in Theorem 2

### C.1 Proof of Theorem 2

Note that  $N_0 = \frac{2^d - 1}{k}$ . Set  $N = kN_0 + 1 = 2^d$ . Set  $\mathcal{X} = \{-1, 0, 1\}^{kN}$ . We set  $\mathcal{Y} = \{1\}$  to be a set with only a single element. Without loss of generality, we write  $\ell(h, (x, y)) = \ell(h, x)$ .

**Our construction.** We now introduce our construction, with the key components described below.

— *Hypothesis class and loss function.* There are  $N$  hypotheses in  $\mathcal{H}$ , where each hypothesis is assigned  $k$  unique dimensions (recall that  $\mathcal{X}$  consists of  $(KN)$ -dimensional vectors). For each  $h \in \mathcal{H}$ , we let  $\mathcal{J}_h = \{j_{h,i}\}_{i=1}^k$  denote the  $k$  dimensions assigned to  $h$ , so that  $\mathcal{J}_h \cap \mathcal{J}_{h'} = \emptyset$  for  $h \neq h'$ . We then define  $h(x)$  and  $\ell(h, x)$  as follows:

$$h(x) = \ell(h, x) = \begin{cases} 0, & \text{if } x_i = 0 \text{ for all } i \in \mathcal{J}_h; \\ x_{i'} \text{ with } i' = \arg \min_{i \in \mathcal{J}_h, x_i \neq 0} x_i, & \text{else.} \end{cases}$$

Next, divide  $\mathcal{H}$  arbitrarily into  $(k+1)$  disjoint subsets as

$$\mathcal{H} = (\cup_{i=1}^k \mathcal{H}_i) \cup \{h^*\},$$

where each  $\mathcal{H}_i$  contains  $N_0$  hypotheses. In our construction, we intend to make  $h^*$  the unique optimal policy, and will design properly so that the hypothesis  $h \in \mathcal{H}_i$  performs poorly on the  $i$ th distribution  $\mathcal{D}_i$  ( $i \in [k]$ ).

— *Data distributions.* We design the  $k$  data distributions  $\{\mathcal{D}_i\}_{i=1}^k$ . For any given  $i \in [k]$  and any  $x \in \mathcal{X}$ , we let

$$\mathbb{P}_{\mathcal{D}_i}\{x\} = \prod_{l=1}^{kN} \mathbb{P}_{\mathcal{D}_{i,l}}\{x_l\},$$

be a product distribution, where

$$\begin{aligned} \mathbb{P}_{\mathcal{D}_{i,l}}\{x_l\} &= \mathbb{1}\{x_l = 0\}, & l \notin \mathcal{J}_{h,i} \mid h \in \mathcal{H}, \\ \mathbb{P}_{\mathcal{D}_{i,l}}\{x_l\} &= \frac{1}{2}\mathbb{1}\{x_l = 1\} + \frac{1}{2}\mathbb{1}\{x_l = -1\}, & l \in \mathcal{J}_{h,i} \mid h \notin \mathcal{H}_i, \\ \mathbb{P}_{\mathcal{D}_{i,l}}\{x_l\} &= \left(\frac{1}{2} + 4\varepsilon\right)\mathbb{1}\{x_l = 1\} + \left(\frac{1}{2} - 4\varepsilon\right)\mathbb{1}\{x_l = -1\}, & l \in \mathcal{J}_{h,i} \mid h \in \mathcal{H}_i. \end{aligned}$$

The above construction enjoys the following properties:

- (i)  $\ell(h, x) \in [-1, 1]$  for any  $h \in \mathcal{H}$  and  $x \in \mathcal{X}$ ;
- (ii)  $\mathbb{E}_{x \sim \mathcal{D}_i}[\ell(h, x)] = \mathbb{E}_{x \sim \mathcal{D}_{i,j_{h,i}}} [x_{j_{h,i}}] = 8\varepsilon \cdot \mathbb{1}\{h \in \mathcal{H}_i\}$  for any  $i \in [k]$  and  $h \in \mathcal{H}$ ;
- (iii) the only  $\varepsilon$ -optimal hypothesis is  $h^*$ , because for any  $h \neq h^*$ , there exists some  $i$  such that  $h \in \mathcal{H}_i$ ;
- (iv)  $h(x) \in \{-1, 1\}$  for  $x \in \cup_{i=1}^k \text{supp}(\mathcal{D}_i)$ ,<sup>10</sup> and  $|\mathcal{H}| = N = kN_0 + 1 = 2^d$ , which imply that

$$\text{VC-dim}(\mathcal{H}) \leq \log_2(N) \leq d$$

over  $\cup_{i=1}^k \text{supp}(\mathcal{D}_i)$ ;

- (v)  $\ell(h, x)$  could be regarded as a function of  $h(x)$  because  $\ell(h, x) = h(x)$ .

**Sample complexity lower bound.** Before proceeding, let us introduce the notation  $\text{Query}(\mathcal{D}_i)$  such that: for each call to  $\text{Query}(\mathcal{D}_i)$ , we can obtain independent observations  $\{x_{j_{h,i}}\}_{h \in \mathcal{H}}$  where  $x_{j_{h,i}} \sim \mathcal{D}_{i,j_{h,i}}$  for each  $h \in \mathcal{H}$ . Now for  $i \in [k]$ , denote by  $M_i$  the number of calls to  $\text{Query}(\mathcal{D}_i)$ .

<sup>10</sup>We denote by  $\text{supp}(\mathcal{D})$  the support of the distribution  $\mathcal{D}$ .

Our aim is at showing that: in order to distinguish  $h^*$  from  $\mathcal{H}_i$ , the quantity  $M_i$  has to be at least  $\Omega(d/\varepsilon^2)$ .

Suppose now that there is an algorithm  $\mathcal{G}$  with numbers of samples  $\{M_{ij}\}_{j=1}^k$  such that the output is  $h^*$  with probability at least 3/4. Let  $\mathbb{P}_{\mathcal{G}}\{\cdot\}$  and  $\mathbb{E}_{\mathcal{G}}[\cdot]$  denote respectively the probability and expectation when Algorithm  $\mathcal{G}$  is executed, and let  $h_{\text{out}}$  be the output hypothesis. It then holds that

$$\mathbb{P}_{\mathcal{G}}\{h_{\text{out}} = h^*\} \geq \frac{3}{4}.$$

Let  $\Pi_{\mathcal{H}}$  be the set of permutations over  $\mathcal{H}$ , and let  $\text{Unif}(\Pi_{\mathcal{H}})$  be the uniform distribution over  $\Pi(\mathcal{H})$ . With slight abuse of notation, for  $x \in \{-1, 0, 1\}^{kN}$  and  $\sigma \in \Pi_{\mathcal{H}}$ , we define  $\sigma(x)$  to be the vector  $y$  such that  $y_{j_{h,i}} = x_{j_{\sigma(h),i}}$  for all  $h \in \mathcal{H}$  and  $i \in [k]$ . Let  $\mathcal{G}'$  be the algorithm with  $\mathcal{H}$  replaced by  $\sigma(\mathcal{H})$  in the input where  $\sigma \sim \text{Unif}(\Pi_{\mathcal{H}})$ . Recognizing that  $\mathcal{G}$  returns the optimal hypothesis with probability at least 3/4 for all problem instances, we can see that

$$\mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*\} \geq \frac{3}{4}.$$

The lemma below then assists in bounding the probability of returning a sub-optimal hypothesis.

**LEMMA 18.** Consider  $\tilde{i} \in [k]$ . If  $\mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} \geq 1/2$  for some  $m \geq 0$ , then for any  $h \in \mathcal{H}_{\tilde{i}}$ , one has

$$\mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h, M_{\tilde{i}} \leq m\} \geq \frac{1}{2} \mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} \exp(-80\sqrt{m\varepsilon} - 40m\varepsilon^2),$$

and moreover,  $m$  necessarily exceeds  $m \geq \frac{\log(N_0/4)}{30000\varepsilon^2}$ .

**PROOF.** See Appendix C.2. □

In view of Lemma 18, we have

$$\mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} < \frac{\log(N_0/4)}{30000\varepsilon^2}\right\} < \frac{1}{2}. \quad (129)$$

Observing that  $\mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*\} \geq 3/4$ , we can derive

$$\begin{aligned} \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \geq \frac{\log(N_0/4)}{30000\varepsilon^2}\right\} &= \mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*\} - \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} < \frac{\log(N_0/4)}{30000\varepsilon^2}\right\} \\ &> \frac{3}{4} - \frac{1}{2} = \frac{1}{4}, \end{aligned}$$

which implies that

$$\begin{aligned} \mathbb{E}_{\mathcal{G}'}[M_{\tilde{i}}] &\geq \frac{\log(N_0/4)}{30000\varepsilon^2} \cdot \mathbb{P}_{\mathcal{G}'}\left\{M_{\tilde{i}} \geq \frac{\log(N_0/4)}{30000\varepsilon^2}\right\} \geq \frac{\log(N_0/4)}{30000\varepsilon^2} \cdot \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \geq \frac{\log(N_0/4)}{30000\varepsilon^2}\right\} \\ &\geq \frac{\log(N_0/4)}{120000\varepsilon^2} \geq \frac{d - \log_2(8k)}{120000\varepsilon^2} \geq \frac{d}{240000\varepsilon^2}. \end{aligned}$$

Summing over  $i \in [k]$  gives

$$\mathbb{E}_{\mathcal{G}'}\left[\sum_{i=1}^k M_i\right] \geq \frac{dk}{240000\varepsilon^2}, \quad (130)$$

thereby concluding the proof.

## C.2 Proof of Lemma 18

Consider any  $h \in \mathcal{H}_{\tilde{i}}$  (and hence  $h \neq h^*$ ). For  $v = [v_p]_{1 \leq p \leq m} \in \{-1, 1\}^m$ , we let

$$n^+(v) = \sum_{p=1}^m \mathbb{1}\{v_p = 1\},$$

denote the number of 1's in the coordinates of  $v$ . Let  $\mathcal{V}$  be a subset of  $\{-1, 1\}^{2m}$  defined as

$$\mathcal{V} := \{v^1, v^2 \in \{-1, 1\}^m \mid n^+(v^1) - n^+(v^2) \leq 4\sqrt{m} + 2m\varepsilon\}.$$

Let  $\mathbb{P}_{\mathcal{C}}\{\cdot\}$  denote the probability distribution of  $\{\{x_{j_{h,i}}^l(\tilde{i})\}_{l=1}^m, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m\}$ , and  $\mathbb{P}_{\mathcal{C}'}\{\cdot\}$  the probability distribution of  $\{\{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m\}$ . Hoeffding's inequality then tells us that

$$\mathbb{P}_{\mathcal{C}'}\{\mathcal{V}\} \geq \frac{3}{4}.$$

Also, observe that

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m, \{\{x_{j_{h,i}}^l(\tilde{i})\}_{l=1}^m, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m\} \in \mathcal{V}\right\} \\ & \geq \mathbb{P}_{\mathcal{G}'}\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} - (1 - \mathbb{P}_{\mathcal{C}'}\{\mathcal{V}\}) \geq \frac{3}{4} - \left(1 - \frac{3}{4}\right) = \frac{1}{2}, \end{aligned} \quad (131)$$

namely,

$$\sum_{v=\{v^1, v^2\} \in \mathcal{V}} \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \geq \frac{1}{2}.$$

In addition, for any  $v = \{v^1, v^2\} \in \mathcal{V}$ , it is readily seen that

$$\begin{aligned} \mathbb{P}_{\mathcal{C}'}\{v\} &= \mathbb{P}_{\mathcal{C}}\{v\} \cdot (1 - 8\varepsilon)^{n^+(v^1) - n^+(v^2)} (1 + 8\varepsilon)^{n^+(v^2) - n^+(v^1)} \\ &= \mathbb{P}_{\mathcal{C}}\{v\} \left(\frac{1 - 8\varepsilon}{1 + 8\varepsilon}\right)^{n^+(v^1) - n^+(v^2)} \\ &\geq \mathbb{P}_{\mathcal{C}}\{v\} \exp\left(-20(n^+(v^1) - n^+(v^2))\varepsilon\right) \\ &\geq \mathbb{P}_{\mathcal{C}}\{v\} \exp(-80\sqrt{m}\varepsilon - 40m\varepsilon^2), \end{aligned}$$

where the last line follows from the definition of  $\mathcal{V}$ . As a result, we can demonstrate that, for any  $v = \{v^1, v^2\} \in \mathcal{V}$ ,

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h, M_{\tilde{i}} \leq m, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \\ &= \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h, M_{\tilde{i}} \leq m \mid \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \mathbb{P}_{\mathcal{C}'}\{v\} \\ &\geq \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h, M_{\tilde{i}} \leq m \mid \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \mathbb{P}_{\mathcal{C}}\{v\} \exp(-80\sqrt{m}\varepsilon - 40m\varepsilon^2) \\ &= \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \mathbb{P}_{\mathcal{C}}\{v\} \exp(-80\sqrt{m}\varepsilon - 40m\varepsilon^2) \\ &= \mathbb{P}_{\mathcal{G}'}\left\{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2\right\} \exp(-80\sqrt{m}\varepsilon - 40m\varepsilon^2). \end{aligned} \quad (132)$$

To see why Equation (132) holds, observe that (here, for any  $1 \leq l \leq m$ , let  $v_l^1$  (respectively  $v_l^2$ ) be the  $l$ th coordinate of  $v^1$  (respectively  $v^2$ ):

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h, M_{\tilde{i}} \leq m \mid \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \\ &= \sum_{m'=1}^m \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h, M_{\tilde{i}} = m' \mid \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \\ &= \sum_{m'=1}^m \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h, M_{\tilde{i}} = m' \mid \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^{m'} = \{v_l^1\}_{l=1}^{m'}, \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^{m'} = \{v_l^2\}_{l=1}^{m'} \right\} \end{aligned} \quad (133)$$

$$= \sum_{m'=1}^m \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} = m' \mid \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^{m'} = \{v_l^1\}_{l=1}^{m'}, \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^{m'} = \{v_l^2\}_{l=1}^{m'} \right\} \quad (134)$$

$$= \sum_{m'=1}^m \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} = m' \mid \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \quad (135)$$

$$= \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} \leq m \mid \{x_{j_{h, \tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*}, \tilde{i}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\}.$$

where Equation (134) results from Lemma 19 in Appendix C.3, and Equation (133) and Equation (135) hold since for  $h' = h, h^*$ , the event  $\{h_{\text{out}} = h', M_{\tilde{i}} = m'\}$  is independent of  $\{x^l(\tilde{i})\}_{l \geq m'+1}$ . Taking the sum over  $v = \{v^1, v^2\} \in \mathcal{V}$ , we obtain

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}'} \{h_{\text{out}} = h, M_{\tilde{i}} \leq m\} \\ & \geq \sum_{v \in \mathcal{V}} \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h, M_{\tilde{i}} \leq m, \{x_{j_{h^*}, \tilde{i}}^\ell(\tilde{i})\}_{\ell=1}^m = v^1, \{x_{j_{h, \tilde{i}}}^\ell(\tilde{i})\}_{\ell=1}^m = v^2 \right\} \\ & \geq \sum_{v \in \mathcal{V}} \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} \leq m, \{x_{j_{h, \tilde{i}}}^\ell(\tilde{i})\}_{\ell=1}^m = v^1, \{x_{j_{h^*}, \tilde{i}}^\ell(\tilde{i})\}_{\ell=1}^m = v^2 \right\} \exp(-80\sqrt{m}\epsilon - 40m\epsilon^2) \\ &= \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} \leq m, \left\{ \{x_{j_{h, \tilde{i}}}^\ell(\tilde{i})\}_{\ell=1}^m, \{x_{j_{h^*}, \tilde{i}}^\ell(\tilde{i})\}_{\ell=1}^m \right\} \in \mathcal{V} \right\} \exp(-80\sqrt{m}\epsilon - 40m\epsilon^2) \\ &\geq \frac{1}{2} \mathbb{P}_{\mathcal{G}'} \{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} \exp(-80\sqrt{m}\epsilon - 40m\epsilon^2), \end{aligned}$$

where the last line arises from Equation (131).

Summing over all  $h \in \mathcal{H}_{\tilde{i}}$ , we reach

$$1 \geq \mathbb{P}_{\mathcal{G}'} \{h_{\text{out}} \in \mathcal{H}_{\tilde{i}}, M_{\tilde{i}} \leq m\} \geq \frac{N_0}{2} \mathbb{P}_{\mathcal{G}'} \{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} \exp(-40m\epsilon^2 - 80\sqrt{m}\epsilon), \quad (136)$$

given that each  $\mathcal{H}_{\tilde{i}}$  contains  $N_0$  hypotheses. This in turn reveals that

$$\frac{1}{2} \leq \mathbb{P}_{\mathcal{G}'} \{h_{\text{out}} = h^*, M_{\tilde{i}} \leq m\} \leq \frac{2}{N_0} \exp(40m\epsilon^2 + 80\sqrt{m}\epsilon). \quad (137)$$

Consequently, we arrive at

$$40m\epsilon^2 + 80\sqrt{m}\epsilon \geq \log(N_0/4),$$

which implies that

$$m \geq \min \left\{ \frac{\log(N_0/4)}{80\epsilon^2}, \frac{\log^2(N_0/4)}{30000\epsilon^2} \right\} \geq \frac{\log(N_0/4)}{30000\epsilon^2}.$$

### C.3 Statement and Proof of Lemma 19

LEMMA 19. For any  $i \in [k]$  and  $l \geq 1$ , let  $x^l(i)$  denote the  $l$ th sample from  $\mathcal{D}_i$ . For any  $\tilde{i} \in [k]$ ,  $h \in \mathcal{H}_{\tilde{i}}$ ,  $m > 0$ , and  $v^1, v^2 \in \{-1, 1\}^{2m}$ , one has

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h^*, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \\ &= \mathbb{P}_{\mathcal{G}'} \left\{ h_{\text{out}} = h, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1 \right\}. \end{aligned}$$

PROOF. Let  $\bar{\sigma}$  be the permutation over  $\mathcal{H}$  with  $\bar{\sigma}(h^*) = h$ ,  $\bar{\sigma}(h) = h^*$  and  $\bar{\sigma}(h') = h'$  for all  $h' \notin \{h, h^*\}$ . It then holds that  $\bar{\sigma}^{-1} = \bar{\sigma}$ .

Consider a given sequence  $\{m_i\}_{i=1}^k$ . Let  $X(i) = \{X^l(i)\}_{l=1}^{m_i} \in \{-1, 0, 1\}^{kNm_i}$  for  $i \in [k]$ , and let  $x(i) = \{x^l(i)\}_{l=1}^{m_i}$  be the datapoints of the first  $m_i$  calls to  $\text{Query}(D_i)$ . With slight abuse of notation, we take  $\sigma(x(i)) = \{\sigma(x^l(i))\}_{l=1}^{m_i}$  for each  $i \in [k]$ . It then follows from Lemma 20 in Appendix C.4 that

$$\begin{aligned} & \mathbb{P}_{\mathcal{G},\mathcal{H}} \left\{ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid x(i) = X(i), \forall i \in [k] \right\} \\ &= \mathbb{P}_{\mathcal{G},\bar{\sigma}(\mathcal{H})} \left\{ h_{\text{out}} = \sigma^{-1}(h), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \sigma^{-1}(x(i)) = X(i), \forall i \in [k] \right\}, \end{aligned}$$

which implies that

$$\begin{aligned} & \mathbb{P}_{\mathcal{G},\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h^*, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \sigma(x(i)) = X(i), \forall i \in [k] \right\} \\ &= \mathbb{P}_{\mathcal{G},\bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \bar{\sigma}\sigma(x(i)) = X(i), \forall i \in [k] \right\} \cdot \frac{\mathbb{P}\{\sigma(x(i)) = X(i), \forall i \in [k]\}}{\mathbb{P}\{\bar{\sigma}\sigma(x(i)) = X(i), \forall i \in [k]\}} \\ &= \mathbb{P}_{\mathcal{G},\bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \bar{\sigma}\sigma(x(i)) = X(i), \forall i \in [k] \right\} \cdot \frac{\mathbb{P}\{\sigma(x(\tilde{i})) = X(\tilde{i})\}}{\mathbb{P}\{\bar{\sigma}\sigma(x(\tilde{i})) = X(\tilde{i})\}} \\ &= \mathbb{P}_{\mathcal{G},\bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \bar{\sigma}\sigma(x(i)) = X(i), \forall i \in [k] \right\} \\ &\quad \cdot \frac{\mathbb{P}\left\{\{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}\right\}}{\mathbb{P}\left\{\{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}\right\}}. \end{aligned} \tag{138}$$

Rearrange the equation to arrive at

$$\begin{aligned} & \mathbb{P}_{\mathcal{G},\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h^*, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \sigma(x(i)) = X(i), \forall i \in [k] \right. \\ & \quad \left| \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} \right\} \\ &= \mathbb{P}_{\mathcal{G},\bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k, \bar{\sigma}\sigma(x(i)) = X(i), \forall i \in [k] \right. \\ & \quad \left| \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i}, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_i} \right\}. \end{aligned}$$

Taking the sum over all possible choices of  $\{X(i)\}_{i \neq \tilde{i}}, \{X_{j_{h',i'}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}}, \{X_{j_{h',i'}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} \}_{i' \in [k], h' \notin \{h, h^*\}}, \{X_{j_{h',i'}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} \}_{h' \in \{h, h^*\}, i' \neq \tilde{i}}$  and  $\{m_i\}_{i \neq \tilde{i}}$ , we reach

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = h^*, M_{\tilde{i}} = m_{\tilde{i}} \right. \\ & \quad \left| \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}}, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} \right\} \\ & = \mathbb{P}_{\mathcal{G}, \bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, M_{\tilde{i}} = m_{\tilde{i}} \right. \\ & \quad \left| \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = \{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}}, \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} \right\} \end{aligned}$$

for any  $X(\tilde{i}) \in \{-1, 0, 1\}^{kN m_{\tilde{i}}}$ .

Fix  $m_{\tilde{i}} = m$ , and choose  $\{X_{j_{\sigma(h),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = v_1, \{X_{j_{\sigma(h^*),\tilde{i}}}^l(\tilde{i})\}_{l=1}^{m_{\tilde{i}}} = v_2$ . We then have

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}', \mathcal{H}} \left\{ h_{\text{out}} = h^*, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \\ & = \frac{1}{|\Pi_{\mathcal{H}}|} \sum_{\sigma \in \Pi_{\mathcal{H}}} \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = h^*, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2 \right\} \\ & = \frac{1}{|\Pi_{\mathcal{H}}|} \sum_{\sigma \in \Pi_{\mathcal{H}}} \mathbb{P}_{\mathcal{G}, \bar{\sigma}\sigma(\mathcal{H})} \left\{ h_{\text{out}} = \bar{\sigma}^{-1}(h^*), M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1 \right\} \\ & = \frac{1}{|\Pi_{\mathcal{H}}|} \sum_{\sigma \in \Pi_{\mathcal{H}}} \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = h, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1 \right\} \\ & = \mathbb{P}_{\mathcal{G}', \mathcal{H}} \left\{ h_{\text{out}} = h, M_i = m \mid \{x_{j_{h,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^2, \{x_{j_{h^*,\tilde{i}}}^l(\tilde{i})\}_{l=1}^m = v^1 \right\}. \end{aligned}$$

This completes the proof.  $\square$

#### C.4 Statement and Proof of Lemma 20

LEMMA 20. Consider any  $\{m_i\}_{i \in [k]}, \sigma \in \Pi_{\mathcal{H}}$  and  $X \in \{-1, 0, 1\}^{kN \sum_{i=1}^k m_i}$ . Let  $\{X(i)\}_{i \in [k]}$  and  $\{x(i)\}$  be defined as in Lemma 19. It then holds that

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}, \mathcal{H}} \left[ h_{\text{out}} = h, \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid x(i) = X(i), \forall i \in [k] \right] \\ & = \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = \sigma^{-1}(h), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \sigma^{-1}(x(i)) = X(i), \forall i \in [k] \right\}. \end{aligned} \quad (139)$$

PROOF. Let  $\mathcal{H}' = \sigma(\mathcal{H})$ . Let  $h_p(\cdot)$  denote the  $p$ th hypothesis in the hypothesis set. Then one has

$$\begin{aligned} & \mathbb{P}_{\mathcal{G}, \mathcal{H}} \left\{ h_{\text{out}} = h_p(\mathcal{H}), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid x(i) = X(i), \forall i \in [k] \right\} \\ & = \mathbb{P}_{\mathcal{G}, \mathcal{H}} \left\{ h_{\text{out}} = h_p(\mathcal{H}), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \{\{x_{j_{h,p(\mathcal{H}),i}}^l(i')\}_{p'=1,i=1}^{|\mathcal{H}|,k}\}_{l=1}^{m_{i'}} \}_{i'=1}^k = X \right\} \\ & = \mathbb{P}_{\mathcal{G}, \mathcal{H}'} \left\{ h_{\text{out}} = h_p(\mathcal{H}'), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \{\{x_{j_{h,p'(\mathcal{H}'),i}}^l(i')\}_{p'=1,i=1}^{|\mathcal{H}'|,k}\}_{l=1}^{m_{i'}} \}_{i'=1}^k = X \right\} \end{aligned} \quad (140)$$

$$\begin{aligned} & = \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = h_p(\sigma(\mathcal{H})), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \sigma^{-1}(x(i)) = X(i), \forall i \in [k] \right\} \\ & = \mathbb{P}_{\mathcal{G}, \sigma(\mathcal{H})} \left\{ h_{\text{out}} = \sigma^{-1}(h_p(\mathcal{H})), \{M_i\}_{i=1}^k = \{m_i\}_{i=1}^k \mid \sigma^{-1}(x(i)) = X(i), \forall i \in [k] \right\}. \end{aligned} \quad (141)$$

Here, *Equation (140)* holds since the algorithm  $\mathcal{G}$  cannot distinguish  $\mathcal{H}$  from  $\mathcal{H}'$  using its own randomness.  $\square$

## D Proofs of Auxiliary Lemmas for Rademacher Classes

### D.1 Proof of Lemma 11

In this section, we shall follow the notation adopted in the proof of Lemma 1 (e.g., the dataset  $\tilde{\mathcal{S}}$ , the data subset  $\tilde{\mathcal{S}}(n)$  and its independent copy  $\tilde{\mathcal{S}}^+(n)$ , the Rademacher random variables  $\{\sigma_i^j\}$ ).

Consider any given  $n = \{n_i\}_{i=1}^k$  obeying  $n_i \geq 12 \log(2k)$  for all  $i \in [k]$ , and any given  $w \in \Delta(k)$ . Let  $\kappa = \min_i \frac{n_i}{w_i}$ . Recall that  $(x_{i,j}, y_{i,j})$  is the  $j$ th sample from  $\mathcal{D}_i$ , and  $\{(x_{i,j}^+, y_{i,j}^+)\}$  are independent copies. Define

$$F(n, w) := \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \sum_{i=1}^k w_i L(h, e_i^{\text{basis}}) \right) \right],$$

which can be upper bounded by

$$\begin{aligned} F(n, w) &= \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \left( \ell(h, (x_{i,j}, y_{i,j})) - \mathbb{E}_{\tilde{\mathcal{S}}^+(n)} [\ell(h, (x_{i,j}^+, y_{i,j}^+))] \right) \right) \right] \\ &\leq \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} (\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))) \right) \right] \\ &= \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n), \{\{\sigma_i^j\}_{j=1}^{n_i}\}_{i=1}^k} \left[ \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_i^j \{\ell(h, (x_{i,j}, y_{i,j})) - \ell(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \right] \\ &\leq 2 \mathbb{E}_{\tilde{\mathcal{S}}(n), \{\{\sigma_i^j\}_{j=1}^{n_i}\}_{i=1}^k} \left[ \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{1}{\kappa} \sum_{j=1}^{n_i} \sigma_i^j \ell(h, (x_{i,j}, y_{i,j})) \right) \right] \\ &= 2 \frac{\sum_{i=1}^k n_i}{\kappa} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k}. \end{aligned} \tag{142}$$

Here, the first inequality arises from Jensen's inequality, whereas the penultimate line applies Lemma 17.

Invoking the Mcdiarmid inequality (see Lemma 14) with the choice  $c = 1/\kappa$ , we obtain

$$\mathbb{P} \left\{ \left| \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \sum_{i=1}^k w_i L(h, e_i^{\text{basis}}) \right) - F(n, w) \right| \geq \varepsilon \right\} \leq 2 \exp \left( - \frac{2\kappa^2 \varepsilon^2}{\sum_{i=1}^k n_i} \right). \tag{143}$$

This taken together with Equation (142) reveals that: for any  $\delta' \in (0, 1]$ , with probability at least  $1 - \delta'$  we have

$$\max_{h \in \mathcal{H}} \left( \sum_{i=1}^k \frac{w_i}{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \sum_{i=1}^k w_i L(h, e_i^{\text{basis}}) \right) \leq 2 \frac{\sum_{i=1}^k n_i}{\kappa} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + \frac{\sum_{i=1}^k n_i}{\kappa} \sqrt{\frac{\log(2/\delta')}{2 \sum_{i=1}^k n_i}}. \tag{144}$$

Evidently, this inequality continues to hold if we replace  $(\ell, L)$  with  $(-\ell, -L)$ . As a consequence, with probability at least  $1 - 2\delta'$  one has

$$\max_{h \in \mathcal{H}} \left| \sum_{i=1}^k \frac{w_i}{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \sum_{i=1}^k w_i L(h, e_i^{\text{basis}}) \right| \leq 2 \frac{\sum_{i=1}^k n_i}{\kappa} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + \frac{\sum_{i=1}^k n_i}{\kappa} \sqrt{\frac{\log(2/\delta')}{2 \sum_{i=1}^k n_i}}. \quad (145)$$

Now, fix  $\kappa \geq 0$ , and define

$$\tilde{\mathcal{L}} = \left\{ \mathbf{n} = \{n_i\}_{i=1}^k, \mathbf{w} = \{w_i\}_{i=1}^k \in \Delta_{\varepsilon_1/(8k)}(k) \mid T_1 w_i \leq 2n_i, 12 \log(2k) \leq n_i \leq T_1, \forall i \in [k], \sum_{i=1}^k n_i \leq 2T_1 \right\},$$

where  $\Delta_{\varepsilon_1/(8k)}(k)$  (i.e., an  $\varepsilon_1/(8k)$ -net of  $\Delta(k)$ ) has been defined in Appendix B.1. Inequality Equation (145) combined with the union bound tells us that: for any  $\delta' > 0$ , with probability at least  $1 - \delta'$

$$\begin{aligned} & \max_{h \in \mathcal{H}} \left| \sum_{i=1}^k \sum_{j=1}^{n_i} \frac{w_i}{n_i} \ell(h, (x_{i,j}, y_{i,j})) - \sum_{i=1}^k w_i L(h, e_i^{\text{basis}}) \right| \\ & \leq \frac{\sum_{i=1}^k n_i}{T_1/2} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + \frac{\sum_{i=1}^k n_i}{T_1/2} \sqrt{\frac{\log(|\tilde{\mathcal{L}}|) + \log(2/\delta')}{2 \sum_{i=1}^k n_i}} \\ & \leq 4 \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + 4 \sqrt{\frac{\log(2|\tilde{\mathcal{L}}|) + \log(2/\delta')}{T_1}} \\ & \leq 4 \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + 4 \sqrt{\frac{2k \log(16kT_1/\varepsilon_1) + \log(2/\delta')}{T_1}} \\ & \leq 600C_{T_1} + 4 \sqrt{\frac{2k \log(16kT_1/\varepsilon_1) + \log(2/\delta')}{T_1}} \end{aligned}$$

holds simultaneously for all  $\{\mathbf{n}, \mathbf{w}\} \in \tilde{\mathcal{L}}$ ; see the use of the union bound in Appendix B.1 too. Here, we have made use of Assumption 1, Lemma 9, Lemma 10 and the fact that  $\sum_{i=1}^k n_i \geq T_1/2$ .

Note that in Algorithm 3, we take

$$\hat{L}^t(h, w^t) = \sum_{i=1}^k \frac{w_i^t}{n_i^{t,\text{rad}}} \cdot \sum_{j=1}^{n_i^{t,\text{rad}}} \ell(h, (x_{i,j}, y_{i,j})).$$

Given our choice that  $n_i^{t,\text{rad}} = \min\{[T_1 w_i^t + 12 \log(2k)], T_1\}$  for  $i \in [k]$ , we see that

$$T_1 w_i^t \leq n_i^{t,\text{rad}} - 1 \quad \text{and} \quad 12 \log(2k) \leq n_i^{t,\text{rad}} \leq T_1$$

for all  $i \in [k]$ . In addition, it is seen from our choice of  $n_i^{t,\text{rad}}$  and  $T_1$  that

$$\sum_{i=1}^k n_i^{t,\text{rad}} \leq \sum_{i=1}^k [T_1 w_i^t + 12 \log(2k)] \leq T_1 + k + 12k \log(2k) \leq 2T_1 - 2.$$

Therefore, there exists some  $\tilde{w}^t \in \Delta(k)$  satisfying

$$\{\{n_i^{t,\text{rad}}\}_{i=1}^k, \tilde{w}^t\} \in \tilde{\mathcal{L}} \quad \text{and} \quad \|w^t - \tilde{w}^t\|_1 \leq \frac{\varepsilon_1}{8k}$$

for each  $1 \leq t \leq T$ . Taking  $\delta' = \delta/4$ , we obtain that with probability at least  $1 - \delta/4$ ,

$$\begin{aligned} \max_{h \in \mathcal{H}} |\hat{L}^t(h, w^t) - L(h, w^t)| &\leq \max_{h \in \mathcal{H}} |\hat{L}^t(h, \tilde{w}^t) - L(h, \tilde{w}^t)| + \max_{h \in \mathcal{H}} |\hat{L}^t(h, \tilde{w}^t) - \hat{L}^t(h, w^t)| \\ &\quad + \max_{h \in \mathcal{H}} |L(h, \tilde{w}^t) - L(h, w^t)| \\ &\leq 600C_{T_1} + 4\sqrt{\frac{2k \log(16kT_1/\varepsilon_1) + \log(2/\delta')}{T_1}} + \frac{\varepsilon_1}{8k} + \frac{\varepsilon_1}{8k} \\ &\leq \frac{\varepsilon_1}{2} \end{aligned}$$

for any  $1 \leq t \leq T$ , where the last inequality results from the definition of  $T_1$ .

Finally, the fact that  $h^t = \arg \min_{h \in \mathcal{H}} \hat{L}^t(h, w^t)$  allows one to derive

$$L(h^t, w^t) \leq \hat{L}^t(h^t, w^t) + \frac{\varepsilon_1}{2} = \min_{h \in \mathcal{H}} \hat{L}^t(h, w^t) + \frac{\varepsilon_1}{2} \leq \min_{h \in \mathcal{H}} L(h, w^t) + \varepsilon_1,$$

which concludes the proof.

## D.2 Proof of Lemma 9

In what follows, assume that each  $z_i^j$  obeys  $z_i^j \sim \mathcal{D}_i$ , and each  $\sigma_i^j$  is a zero-mean Rademacher random variable. Direct computation then gives

$$\begin{aligned} \left( \sum_{i=1}^k n_i \right) \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} &= \mathbb{E}_{\{z_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{n_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \\ &\stackrel{(i)}{=} \mathbb{E}_{\{z_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \max_{h \in \mathcal{H}} \sum_{\{z_i^j\}_{j=n_i+1}^{n_i+m_i}, \{\sigma_i^j\}_{j=n_i+1}^{n_i+m_i}, \forall i \in [k]} \mathbb{E}_{\{\sigma_i^j\}_{j=n_i+1}^{n_i+m_i}, \forall i \in [k]} \left[ \sum_{i=1}^k \sum_{j=1}^{n_i+m_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \right] \\ &\stackrel{(ii)}{\leq} \mathbb{E}_{\{z_i^j\}_{j=1}^{n_i+m_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i+m_i}, \forall i \in [k]} \left[ \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{n_i+m_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \\ &= \left( \sum_{i=1}^k (n_i + m_i) \right) \widetilde{\text{Rad}}_{\{n_i + m_i\}_{i=1}^k} \\ &\stackrel{(iii)}{\leq} \mathbb{E}_{\{z_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{n_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \\ &\quad + \mathbb{E}_{\{z_i^j\}_{j=n_i+1}^{n_i+m_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=n_i+1}^{n_i+m_i}, \forall i \in [k]} \left[ \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=n_i+1}^{n_i+m_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \\ &= \left( \sum_{i=1}^k n_i \right) \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} + \left( \sum_{i=1}^k m_i \right) \widetilde{\text{Rad}}_{\{m_i\}_{i=1}^k}. \end{aligned}$$

Here, (i) is valid due to the zero-mean property of  $\{\sigma_i^j\}$ , (ii) comes from Jensen's inequality, and (iii) follows since  $\max_x (f_1(x) + f_2(x)) \leq \max_x f_1(x) + \max_x f_2(x)$ .

### D.3 Proof of Lemma 10

Set  $n = \sum_{i=1}^k n_i$ . Let  $\{X_j\}_{j=1}^n$  be  $n$  i.i.d. multinomial random variables with parameter  $\{w_i\}_{i=1}^k$ , and take

$$\hat{n}_i = \sum_{j=1}^n \mathbb{1}\{X_j = i\},$$

for each  $i \in [k]$ . From Equation (50) and Definition 2, it is easily seen that

$$\text{Rad}_n(D(w)) = \mathbb{E}_{\{X_i\}_{i=1}^n} \left[ \mathbb{E}_{\{z_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \frac{1}{n} \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{\hat{n}_i} \sigma_i^j \ell(h, z_i^j) \right] \right] \right],$$

where each  $z_i^j$  is independently drawn from  $\mathcal{D}_i$ , and each  $\sigma_i^j$  is an independent Rademacher random variable.

In addition, Lemma 13 tells us that: for any  $i \in [k]$ , one has

$$\hat{n}_i \geq \frac{1}{3} n_i - 2 \log(2k) \geq \frac{1}{6} n_i \quad \implies \quad \hat{n}_i \geq \left\lceil \frac{1}{6} n_i \right\rceil =: \tilde{n}_i, \quad (146)$$

with probability exceeding  $1 - 1/(2k)$ . Defining  $\mathcal{E}$  to be the event that  $\hat{n}_i \geq n_i/6$  holds for all  $i \in [k]$ , we can invoke the union bound to see that

$$\mathbb{P}(\mathcal{E}) \geq 1/2.$$

Consequently, we can derive

$$\begin{aligned} \text{Rad}_n(D(w)) &\geq \mathbb{P}(\mathcal{E}) \cdot \mathbb{E}_{\{X_i\}_{i=1}^n} \left[ \mathbb{E}_{\{z_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \frac{1}{n} \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{\hat{n}_i} \sigma_i^j \ell(h, z_i^j) \mid \mathcal{E} \right] \right] \right] \\ &\geq \frac{1}{2} \cdot \mathbb{E}_{\{z_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{\hat{n}_i}, \forall i \in [k]} \left[ \frac{1}{n} \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{\hat{n}_i} \sigma_i^j \ell(h, z_i^j) \mid \mathcal{E} \right] \right] \\ &= \frac{1}{2} \cdot \frac{\sum_{i=1}^k \tilde{n}_i}{n} \widetilde{\text{Rad}}_{\{\tilde{n}_i\}_{i=1}^k} \\ &\geq \frac{1}{12} \cdot \frac{1}{6} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k}, \end{aligned} \quad (147)$$

thus concluding the proof.

### D.4 Necessity of Assumption 1

In this subsection, we study whether Assumption 1 can be replaced by the following weaker assumption, the latter of which only assumes that the Rademacher complexity on each  $\mathcal{D}_i$  is well-bounded.

**ASSUMPTION 2.** For each  $n \geq 1$ , there exists a quantity  $\tilde{C}_n > 0$  (known to the learner) such that

$$\tilde{C}_n \geq \max_{1 \leq i \leq k} \text{Rad}_n(\mathcal{D}_i). \quad (148)$$

Formally, we have the following results.

**LEMMA 21.** Let  $w^0 = [1/k, 1/k, \dots, 1/k]^\top$ . There exist a group of distributions  $\{\mathcal{D}_i\}_{i=1}^k$  and a hypothesis set  $\mathcal{H}$  such that

$$\text{Rad}_n(D(w^0)) \geq \Omega\left(\frac{1}{k} \sum_{i=1}^k \text{Rad}_{n/k}(\mathcal{D}_i)\right), \quad (149)$$

for  $n \geq 12k \log(k)$ .

PROOF. Without loss of generality, consider the case where  $\mathcal{Y} = \{0\}$  and  $\ell(h, (x, y)) = h(x) - y = h(x)$ . We can then view  $\mathcal{D}_i$  as a distribution over  $\mathcal{X}_i$ , as there is only one element in  $\mathcal{Y}$ .

Pick  $k$  subsets of  $\mathcal{X}$  as  $\{\mathcal{X}_i\}_{i=1}^k$ . For each  $i \in [k]$ , we choose the distribution  $\mathcal{D}_i$  to be an arbitrary distribution supported on  $\mathcal{X}_i$ . In addition, we define  $\mathcal{H}_i$  to be a set of hypothesis obeying  $h(x) = 0$  for all  $x \notin \mathcal{X}_i$  for each  $i \in [k]$ . For a collection of hypothesis  $\{h_i\}_{i=1}^k$  such that  $h_i \in \mathcal{H}_i$ , we define  $\text{joint}(\{h_i\}_{i=1}^k)$  to be the hypothesis  $h$  such that

$$h(x) = \begin{cases} h_i(x) & \text{if } x \in \mathcal{X}_i, i \in [k]; \\ 0 & \text{if } x \notin \cup_i \mathcal{X}_i. \end{cases} \quad (150)$$

The hypothesis set  $\mathcal{H}$  is then constructed as

$$\mathcal{H} = \{\text{joint}(\{h_i\}_{i=1}^k) \mid h_i \in \mathcal{H}_i, \forall i \in [k]\}. \quad (151)$$

Recalling the definition of  $\widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k}$ , we see that

$$\begin{aligned} \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} &= \mathbb{E}_{\{x_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \frac{1}{\sum_{i=1}^k n_i} \max_{h \in \mathcal{H}} \sum_{i=1}^k \sum_{j=1}^{n_i} \sigma_i^j h(x_i^j) \right] \right] \\ &= \mathbb{E}_{\{x_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \frac{1}{\sum_{i=1}^k n_i} \sum_{i=1}^k \max_{h_i \in \mathcal{H}_i} \sum_{j=1}^{n_i} \sigma_i^j h_i(x_i^j) \right] \right] \\ &= \frac{1}{\sum_{i=1}^k n_i} \sum_{i=1}^k n_i \mathbb{E}_{\{x_i^j\}_{j=1}^{n_i}, \forall i \in [k]} \left[ \mathbb{E}_{\{\sigma_i^j\}_{j=1}^{n_i}} \left[ \frac{1}{n_i} \max_{h_i \in \mathcal{H}_i} \sum_{j=1}^{n_i} \sigma_i^j h_i(x_i^j) \right] \right] \\ &= \frac{1}{\sum_{i=1}^k n_i} \sum_{i=1}^k n_i \text{Rad}_{n_i}(\mathcal{D}_i), \end{aligned} \quad (152)$$

where Equation (152) results from the definition of  $\mathcal{H}$ . By taking  $n_i = \frac{n}{k}$  for all  $i \in [k]$  and applying Lemma 10, we reach

$$\frac{1}{k} \sum_{i=1}^k \text{Rad}_{n/k}(\mathcal{D}_i) = \widetilde{\text{Rad}}_{\{n_i\}_{i=1}^k} \leq 72 \text{Rad}_n(\mathcal{D}(w^0)), \quad (153)$$

as claimed.  $\square$

By virtue of Lemma 21, if we set  $\tilde{C}_n = \tilde{\Theta}(\sqrt{d/n})$  in Assumption 2, then the best possible upper bound on  $\text{Rad}_n(\mathcal{D}(w^0))$  is  $\text{Rad}_n(\mathcal{D}(w^0)) = \tilde{\Theta}(\sqrt{dk/n})$ , which implies that more samples are needed to learn the mixed distribution  $\mathcal{D}(w^0)$  than learning each individual distribution. Moreover, under the construction in Lemma 21, if we further assume that  $\min_{h_i \in \mathcal{H}_i} \mathbb{E}_{x \sim \mathcal{D}_i}[h_i(x)] = 1/2$  for all  $i \in [k]$ , then to find  $h$  such that

$$\max_{1 \leq i \leq k} \mathbb{E}_{x \sim \mathcal{D}_i}[h(x)] \leq \frac{1}{2} + \varepsilon, \quad (154)$$

we need to find, for each  $i \in [k]$ , a hypothesis  $h_i \in \mathcal{H}_i$  such that

$$\mathbb{E}_{x \sim \mathcal{D}_i}[h_i(x)] \leq \frac{1}{2} + \varepsilon. \quad (155)$$

Following this intuition, we can construct a counter example under Assumption 2, with a formal theorem stated as follows:

**THEOREM 5.** *There exist a group of distributions  $\{\mathcal{D}_i\}_{i=1}^k$  and a hypothesis set  $\mathcal{H}$  such that Assumption 2 holds with  $\tilde{C}_n = \tilde{O}(\sqrt{\frac{d}{n}})$ , and it takes at least  $\tilde{\Omega}(\frac{dk}{\varepsilon^2})$  samples to find some  $h \in \mathcal{H}$  obeying*

$$\max_{i \in [k]} L(h, e_i^{\text{basis}}) \leq \min_{h' \in \mathcal{H}} \max_{i \in [k]} L(h', e_i^{\text{basis}}) + \varepsilon.$$

**PROOF.** With the construction in Lemma 21, it suffices to find some  $\mathcal{H}'$  and  $\mathcal{D}'$  such that the following three conditions hold:

(1) The following inequality holds:

$$\text{Rad}_n(\mathcal{D}', \mathcal{H}') := \frac{1}{n} \mathbb{E}_{\{x^j\}_{j=1}^n \stackrel{\text{i.i.d.}}{\sim} \mathcal{D}', \{\sigma^j\}_{j=1}^n \stackrel{\text{i.i.d.}}{\sim} \{\pm 1\}} \left[ \max_{h' \in \mathcal{H}'} \sum_{j=1}^n \sigma^j h'(x^j) \right] \leq \tilde{C}_n; \quad (156)$$

(2)  $\min_{h' \in \mathcal{H}'} \mathbb{E}_{x \sim \mathcal{D}'}[h(x)] = \frac{1}{2}$ ;

(3) It takes at least  $\tilde{\Omega}(\frac{d}{\varepsilon^2})$  samples to find some  $h$  such that  $\mathbb{E}_{x \sim \mathcal{D}'}[h(x)] \leq \frac{1}{2} + \varepsilon$ .

This construction is also straightforward. Set  $N = 2^d$  and  $\mathcal{X}' = \{0, 1\}^N$ . Let  $\mathcal{D}'$  be the distribution

$$\mathbb{P}_{\mathcal{D}'}\{x\} = \prod_{n=1}^N \mathbb{P}_{\mathcal{D}'_n}\{x_n\},$$

where

$$\mathbb{P}_{\mathcal{D}'_{n^*}}\{x_{n^*}\} = \frac{1}{2} \mathbb{1}\{x_{n^*} = 1\} + \frac{1}{2} \mathbb{1}\{x_{n^*} = 0\} \quad \text{for some } n^*; \quad (157)$$

$$\mathbb{P}_{\mathcal{D}'_n}\{x_n\} = \left( \frac{1}{2} + 2\varepsilon \right) \mathbb{1}\{x_n = 1\} + \left( \frac{1}{2} - 2\varepsilon \right) \mathbb{1}\{x_n = 0\} \quad \text{for all } n \neq n^*. \quad (158)$$

We then choose  $\mathcal{H}' = \{h^n\}_{n=1}^N$  with  $h^n(x) = x_n$  for each  $n \in [N]$ . It is then easy to verify that the first two conditions hold. Regarding the third condition, following the arguments in Theorem 2, we need at least  $\tilde{\Omega}(d/\varepsilon^2)$  i.i.d. samples from  $\mathcal{D}'$  to identify  $n^*$ . The proof is thus completed.  $\square$

## E Proof for Multi-loss Multi-distribution Learning in Theorem 4

The proof of Theorem 4 is similar as that of Theorem 1, except that we need to establish uniform convergence for the new loss estimators in Algorithm 4.

In Algorithm 4, we re-define the total stepsize and auxiliary step size as

$$T := \frac{20000 \log\left(\frac{kR}{\delta\varepsilon}\right)}{\varepsilon^2}, \quad T_1 := \frac{40000 \left(k \log\left(\frac{kR}{\varepsilon_1}\right) + d \log\left(\left(\frac{kd}{\varepsilon_1} + \log\left(\frac{1}{\delta}\right)\right)\right)\right)}{\varepsilon_1^2}.$$

### E.1 Main Steps of the Proof

Let us begin by presenting the key lemmas needed to establish Theorem 4. Recall that  $u^t = \{u_{i,\ell}\}_{\ell \in \mathcal{L}, i \in [k]}$ . For any  $u \in \Delta([k] \times \mathcal{L})$ ,  $h \in \mathcal{H}$  and  $\ell \in \mathcal{L}$ , recall that

$$L_i^\ell(h) = \mathbb{E}_{(x,y) \sim \mathcal{D}_i} [\ell(h, (x, y))] \quad \text{and} \quad L(h, u) = \sum_{i,\ell} u_{i,\ell} L_i^\ell(h). \quad (159)$$

For  $\pi \in \Delta(\mathcal{H})$ , we define  $L_i^\ell(h_\pi) = \mathbb{E}_{h \sim \pi}[L_i^\ell(h)]$  and  $L(h_\pi, u) = \sum_{i,\ell} u_{i,\ell} L_i^\ell(h_\pi)$ .

The first step is to establish the goodness of the empirical minimizer  $h^t$ , akin to Lemma 1.

**LEMMA 22.** *With probability at least  $1 - \delta/4$ , it holds that*

$$L(h^t, u^t) \leq \min_{h \in \mathcal{H}} L(h, u^t) + \varepsilon_1, \quad (160)$$

for any  $1 \leq t \leq T$ , where  $h^t$  (respectively  $u^t$ ) is the hypothesis (respectively weight vector) computed in round  $t$  of Algorithm 4.

Similar to Lemma 2, the next step is to prove that the output hypothesis  $h^{\text{final}}$  is  $\varepsilon$ -optimal.

LEMMA 23. *With probability at least  $1 - \delta/2$ , the output policy  $h^{\text{final}}$  is  $\varepsilon$ -optimal in the sense that*

$$\max_{i \in [k], \ell \in \mathcal{L}} \frac{1}{T} \sum_{t=1}^T L_i^\ell(h^t) \leq \min_{\pi \in \Delta(\mathcal{H})} \max_{i \in [k], \ell \in \mathcal{L}} L_i^\ell(h_\pi) + \varepsilon \leq \min_{h \in \mathcal{H}} \max_{i \in [k], \ell \in \mathcal{L}} L_i^\ell(h) + \varepsilon.$$

Furthermore, the following lemma upper bounds the sample complexity of the proposed algorithm.

LEMMA 24. *With probability at least  $1 - \delta/2$ , the sample complexity of Algorithm 4 is bounded by*

$$O\left(\frac{(d \log\left(\frac{d}{\varepsilon}\right) + k \log\left(\frac{kR}{\delta\varepsilon}\right)) \min\{\log(R), k\}}{\varepsilon^2} \cdot \log^5(k) \log^3\left(\frac{k \log(R)}{\delta\varepsilon}\right)\right). \quad (161)$$

Armed with the above lemmas, we can readily establish Theorem 4. From Lemma 22 and Lemma 23, we know that with probability exceeding  $1 - \delta$ , the output hypothesis  $h^{\text{final}}$  is  $\varepsilon$ -optimal. Then by virtue of Lemma 24, the sample complexity is no greater than Equation (161), as advertised in the theorem. The rest of this section is thus dedicated to proving the above lemmas.

## E.2 Proof of Lemma 22

By definition of  $h^t$ , it suffices to show that

$$\left| \widehat{L}^t(h, u^t) - L(h, u^t) \right| \leq \frac{1}{2} \varepsilon_1, \quad (162)$$

holds simultaneously for every  $h \in \mathcal{H}$  and  $1 \leq t \leq T$ . Take

$$\widehat{L}_i^\ell(h) = \frac{1}{n_i^t} \sum_{j=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})). \quad (163)$$

By definition, we have

$$\begin{aligned} \widehat{L}^t(h, u^t) - L(h, u^t) &= \sum_{i, \ell} u_{i, \ell}^t \widehat{L}_i^\ell(h) - \sum_{i, \ell} u_{i, \ell}^t L_i^\ell(h) = \sum_{i, \ell} u_{i, \ell}^t (\widehat{L}_i^\ell(h) - L_i^\ell(h)) \\ &\leq \sum_{i=1}^k \left( \sum_{\ell \in \mathcal{L}} u_{i, \ell}^t \right) \cdot \max_{\ell \in \mathcal{L}} (\widehat{L}_i^\ell(h) - L_i^\ell(h)), \end{aligned} \quad (164)$$

and similarly, we can also lower bound

$$\widehat{L}^t(h, u^t) - L(h, u^t) \geq \sum_{i=1}^k \left( \sum_{\ell \in \mathcal{L}} u_{i, \ell}^t \right) \cdot \min_{\ell \in \mathcal{L}} (\widehat{L}_i^\ell(h) - L_i^\ell(h)). \quad (165)$$

Consequently, it boils down to developing a uniform upper (respectively lower) bound on the right-hand side of Equation (164) (respectively Equation (165)).

To begin with, let us fix any  $n = \{n_i\}_{i=1}^k \in \mathbb{N}^k$ ,  $\{w_i\}_{i=1}^k \in \Delta(k)$  and  $\{m_i\}_{i=1}^k \in [R]^k$ . Recall the definitions of  $(x_{i,j}, y_{i,j})$ ,  $(x_{i,j}^+, y_{i,j}^+)$ ,  $\widetilde{\mathcal{S}}(n)$ ,  $\widetilde{\mathcal{S}}^+(n)$ ,  $\mathcal{C}$ , and  $\sigma(n) = \{\{\sigma_{i,j}\}_{j=1}^{n_i}\}_{i=1}^k$  in the proof of Lemma 1 (see Appendix B). For any  $\lambda \in [0, \min_i \frac{n_i}{w_i}]$ , define

$$E(\lambda, \{n_i\}_{i=1}^k, \{w_i\}_{i=1}^k, \{m_i\}_{i=1}^k) := \mathbb{E}_{\widetilde{\mathcal{S}}(n)} \left[ \exp \left( \lambda \cdot \max_{h \in \mathcal{H}} \left( \sum_{i=1}^k w_i \cdot \left( \frac{1}{n_i} \sum_{j=1}^{n_i} \ell^{m_i}(h, (x_{i,j}, y_{i,j})) - L_i^{\ell^{m_i}}(h) \right) \right) \right) \right].$$

Repeating the same symmetrization arguments in the proof of Lemma 1, we can derive

$$\begin{aligned}
& E\left(\lambda, \{n_i\}_{i=1}^k, \{w_i\}_{i=1}^k, \{m_i\}_{i=1}^k\right) \\
&= \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \left( \ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \mathbb{E}_{(x_{i,j}^+, y_{i,j}^+)} [\ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))] \right) \right) \right] \\
&\leq \mathbb{E}_{\tilde{\mathcal{S}}(n)} \left[ \max_{h \in \mathcal{H}} \mathbb{E}_{\tilde{\mathcal{S}}^+(n)} \left[ \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \right] \right] \\
&\leq \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \max_{h \in \mathcal{H}} \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \right] \\
&= \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \mathbb{E}_{\sigma(n)} \left[ \max_{h \in \mathcal{H}_{\min, \mathcal{C}}} \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \middle| \mathcal{C} \right] \right] \\
&\leq \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ |\mathcal{H}_{\min, \mathcal{C}}| \max_{h \in \mathcal{H}_{\min, \mathcal{C}}} \mathbb{E}_{\sigma(n)} \left[ \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \middle| \mathcal{C} \right] \right] \\
&\leq (2kT_1 + 1)^d \mathbb{E}_{\tilde{\mathcal{S}}(n), \tilde{\mathcal{S}}^+(n)} \left[ \max_{h \in \mathcal{H}} \mathbb{E}_{\sigma(n)} \left[ \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \middle| \mathcal{C} \right] \right]. \tag{166}
\end{aligned}$$

For any given  $\mathcal{C}$  and  $h \in \mathcal{H}$ , by observing that  $\lambda \frac{w_i}{n_i} \leq 1$  we have (similar to the arguments for Equation (71))

$$\begin{aligned}
& \mathbb{E}_{\sigma(n)} \left[ \exp \left( \lambda \sum_{i=1}^k \frac{w_i}{n_i} \sum_{j=1}^{n_i} \sigma_{i,j} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \right) \middle| \mathcal{C} \right] \\
&= \prod_{i=1}^k \prod_{j=1}^{n_i} \mathbb{E}_{\sigma_{i,j}} \left[ \lambda \sigma_{i,j} \cdot \frac{w_i}{n_i} \{\ell^{m_i}(h, (x_{i,j}, y_{i,j})) - \ell^{m_i}(h, (x_{i,j}^+, y_{i,j}^+))\} \middle| \mathcal{C} \right] \\
&\leq \exp \left( 2\lambda^2 \cdot \sum_{i=1}^k \frac{w_i^2}{n_i} \right). \tag{167}
\end{aligned}$$

Substituting Equation (166) into Equation (167) gives

$$E\left(\lambda, \{n_i\}_{i=1}^k, \{w_i\}_{i=1}^k, \{m_i\}_{i=1}^k\right) \leq (2kT_1 + 1)^d \exp \left( 2\lambda^2 \cdot \sum_{i=1}^k \frac{(w_i)^2}{n_i} \right). \tag{168}$$

It then follows that, for any  $0 < \varepsilon' < 1$ ,

$$\begin{aligned}
& \mathbb{P} \left( \max_{h \in \mathcal{H}} \sum_{i=1}^k w_i \left\{ \frac{1}{n_i} \sum_{j=1}^{n_i} \ell^{m_i}(h, (x_{i,j}, y_{i,j})) - L_i^{\ell^{m_i}}(h) \right\} \geq \varepsilon' \right) \\
&\leq (2kT_1 + 1)^d \min_{\lambda \in [0, \min_i \frac{n_i}{m_i}]} \exp \left( 2\lambda^2 \sum_{i=1}^k \frac{(w_i)^2}{n_i} - \lambda \varepsilon' \right), \tag{169}
\end{aligned}$$

where we have repeated the arguments for Equation (73).

Next, for any  $\kappa > 0$ , we define  $\mathcal{B}(\kappa)$  to be the set of tuples as follows:

$$\mathcal{B}(\kappa) := \left\{ n = \{n_i\}_{i=1}^k, w = \{w_i\}_{i=1}^k, m = \{m_i\}_{i=1}^k \mid n_i \geq \kappa w_i, n_i \in [T_1], m_i \in [R], \forall i \in [k], w \in \Delta_{\varepsilon_1/(8k)}(k) \right\}.$$

Then it can be easily verified that

$$|\mathcal{B}(\kappa)| \leq (T_1 R)^k \cdot |\Delta_{\varepsilon_1/(8k)}(k)| \leq \left(\frac{8kT_1R}{\varepsilon_1}\right)^k,$$

for any  $\kappa > 0$ . Taking this together with Equation (169), we can reach

$$\begin{aligned} & \sum_{n,w,m \in \mathcal{B}(\kappa)} \mathbb{P} \left( \max_{h \in \mathcal{H}} \sum_{i=1}^k w_i \left( \frac{1}{n_i} \sum_{j=1}^{n_i} \ell^{m_i}(h, (x_{i,j}, y_{i,j})) - L_i^{\ell^{m_i}}(h, w) \right) \geq \varepsilon' \right) \\ & \leq (2kT_1 + 1)^d \cdot \left(\frac{8kT_1R}{\varepsilon_1}\right)^k \cdot \min_{\lambda \in [0, \kappa]} \exp(2\lambda^2/\kappa - \lambda\varepsilon') \\ & \leq (2kT_1 + 1)^d \cdot \left(\frac{8kT_1R}{\varepsilon_1}\right)^k \cdot \exp(-\kappa(\varepsilon')^2/8). \end{aligned} \quad (170)$$

By setting  $\varepsilon' = \varepsilon_1/8$  and  $\kappa_0 = \frac{d\log(2kT_1) + k\log(8kT_1R/\varepsilon_1) + \log(\delta/8)}{8(\varepsilon')^2} \leq T_1/2$ , we can show that: with probability exceeding  $1 - \delta/8$ ,

$$\max_{h \in \mathcal{H}} \sum_{i=1}^k w_i \left( \frac{1}{n_i} \sum_{j=1}^{n_i} \ell^{m_i}(h, (x_{i,j}, y_{i,j})) - L_i^{\ell^{m_i}}(h, w) \right) < \varepsilon', \quad (171)$$

holds simultaneously for all  $(n, w, m) \in \mathcal{B}(\kappa_0)$ . Moreover, observing that  $n_i^t \geq 2\kappa_0 \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t$ , we see that: with probability at least  $1 - \delta/8$ , for any  $1 \leq t \leq T$  one has

$$\max_{h \in \mathcal{H}} \sum_{i=1}^k \left( \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t \right) \max_{\ell \in \mathcal{L}} \left( \frac{1}{n_i^t} \sum_{j=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) - L_i^\ell(h) \right) < \varepsilon' + \frac{\varepsilon_1}{4} \leq \frac{\varepsilon_1}{2}. \quad (172)$$

Applying the same arguments, we can also show that: with probability exceeding  $1 - \delta/8$ , for any  $1 \leq t \leq T$  one has

$$\max_{h \in \mathcal{H}} \sum_{i=1}^k \left( \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t \right) \max_{\ell \in \mathcal{L}} \left( L_i^\ell(h) - \frac{1}{n_i^t} \sum_{j=1}^{n_i^t} \ell(h, (x_{i,j}, y_{i,j})) \right) \leq \frac{\varepsilon_1}{2}. \quad (173)$$

Combining Equation (165), Equation (164), Equation (172), and Equation (173) allows one to conclude that: with probability at least  $1 - \delta/4$ ,

$$|\hat{L}(h, u^t) - L(h, u^t)| \leq \frac{\varepsilon_1}{2},$$

holds simultaneously for every  $1 \leq t \leq T$  and every  $h \in \mathcal{H}$ . The proof is thus completed.

### E.3 Proof of Lemma 23

Take

$$\text{OPT} := \min_{\pi \in \Delta(\mathcal{H})} \max_{i \in [k], \ell \in \mathcal{L}} L_i^\ell(h_\pi) \leq \min_{h \in \mathcal{H}} \max_{i \in [k], \ell \in \mathcal{L}} L_i^\ell(h).$$

Let

$$v^t = L(h^t, u^t) - \text{OPT} \quad \text{and} \quad f^t = \min_{h \in \mathcal{H}} L(h, u^t) - \text{OPT} \leq 0.$$

By virtue of Lemma 22, we know that with probability at least  $1 - \delta/4$ ,

$$L(h^t, u^t) \leq \min_{h \in \mathcal{H}} L(h, u^t) + \varepsilon_1,$$

and we have  $v^t \leq f^t + \varepsilon_1 \leq \varepsilon_1$  for any  $1 \leq t \leq T$ .

Let  $\delta' = \frac{\delta}{4(T+kR+1)}$ . Note that  $\eta \leq 1$  and  $|\hat{r}_{i,\ell}^t| \leq 1$  for any proper  $(i, \ell, t)$ . Direct computation gives

$$\begin{aligned} \log\left(\frac{\sum_{i=1}^k \sum_{\ell \in \mathcal{L}} U_{i,\ell}^{t+1}}{\sum_{i=1}^k \sum_{\ell \in \mathcal{L}} U_{i,\ell}^t}\right) &= \log\left(\sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t \exp(\eta \hat{r}_{i,\ell}^t)\right) \leq \log\left(\sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t (1 + \eta \hat{r}_{i,\ell}^t + \eta^2 (\hat{r}_{i,\ell}^t)^2)\right) \\ &\leq \eta \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t \hat{r}_{i,\ell}^t + \eta^2. \end{aligned} \quad (174)$$

Let  $\hat{r}^t = [\hat{r}_{i,\ell}^t]_{i \in [k], \ell \in \mathcal{L}}$ . As a result, we can obtain

$$\begin{aligned} \eta \sum_{t=1}^T \langle u^t, \hat{r}^t \rangle &\geq \log\left(\sum_{i=1}^k \sum_{\ell \in \mathcal{L}} U_{i,\ell}^{T+1}\right) - T\eta^2 - \log(kR) \\ &\geq \max_{i,\ell} \log(U_{i,\ell}^T) - T\eta^2 - \log(kR) \\ &\geq \eta \max_{i,\ell} \sum_{t=1}^T \hat{r}_{i,\ell}^t - T\eta^2 - \log(kR). \end{aligned} \quad (175)$$

Dividing both side with  $\eta$  yields

$$\sum_{t=1}^T \langle u^t, \hat{r}^t \rangle \geq \max_{i,\ell} \sum_{t=1}^T \hat{r}_{i,\ell}^t - \left(\frac{\log(kR)}{\eta} + \eta T\right). \quad (176)$$

In addition, Azuma's inequality tells us that: with probability at least  $1 - (kR+1)\delta'$ , for any  $i \in [k], \ell \in \mathcal{L}$  one has

$$\begin{aligned} \left| \sum_{t=1}^T \langle u^t, \hat{r}^t \rangle - \sum_{t=1}^T L(h^t, u^t) \right| &\leq 2\sqrt{T \log(1/\delta')}; \\ \left| \sum_{t=1}^T \hat{r}_{i,\ell}^t - \sum_{t=1}^T L_i^\ell(h^t) \right| &\leq 2\sqrt{T \log(1/\delta')}. \end{aligned}$$

Thus, with probability exceeding  $1 - (k+1)\delta'$ , it holds that

$$\begin{aligned} \sum_{t=1}^T L(h^t, u^t) &\geq \max_{i,\ell} \sum_{t=1}^T L_i^\ell(h^t) - \left(\frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')}\right) \\ &\geq TOPT - \left(\frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')}\right). \end{aligned} \quad (177)$$

Let us overload the notation by letting  $e_i^{\text{basis}}$  be the  $i$ th standard basis in  $\mathbb{R}^{kR}$ . In view of the fact that  $\varepsilon_1 = \eta = \frac{1}{100}\varepsilon$  and  $T = \frac{20000 \log(\frac{kR}{\delta'\varepsilon})}{\varepsilon^2}$ , we can demonstrate that

$$\begin{aligned} \max_i \sum_{t=1}^T L(h^t, e_i^{\text{basis}}) &\leq TOPT + \sum_{t=1}^T v^t + \left(\frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')}\right) \\ &\leq TOPT + T\varepsilon_1 + \left(\frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')}\right) \\ &\leq TOPT + T\varepsilon. \end{aligned} \quad (178)$$

As a result, we arrive at

$$\max_{i \in [k], \ell \in \mathcal{L}} L_i^\ell(h^{\text{final}}) = \max_{i \in [k], \ell \in \mathcal{L}} \frac{1}{T} \sum_{t=1}^T L_i^\ell(h^t) = \max_i \frac{1}{T} \sum_{t=1}^T L(h^t, e_i^{\text{basis}}) \leq \text{OPT} + \varepsilon. \quad (179)$$

We can then conclude the proof by observing that  $\delta' = \frac{\delta}{4(T+kR+1)}$ .

#### E.4 Proof of Lemma 24

It is easily seen that the sample complexity of Algorithm 4 is bounded by

$$O\left(T_1 \sum_{i=1}^k \bar{w}_i^T + Tk \left( \sum_{i=1}^k \bar{w}_i^T + 1 \right)\right) = O\left(\frac{d \log\left(\frac{d}{\varepsilon}\right) + k \log\left(\frac{Rk}{\delta\varepsilon}\right)}{\varepsilon^2} \cdot \sum_{i=1}^k \bar{w}_i^T\right). \quad (180)$$

As a result, it comes down to bounding  $\sum_{i=1}^k \bar{w}_i^T$ , which we accomplish below.

With slight abuse of notation, define

$$\mathcal{W}_j = \{i \in [k] \mid \bar{w}_i^T \in (2^{-j}, 2^{-(j-1)}]\}$$

for any  $1 \leq j \leq \lfloor \log(1/k) \rfloor + 1$ , and we would like to bound the size of each  $\mathcal{W}_j$  separately. This is the focus of the following lemma. Recall that  $\tilde{j} = \lfloor \log_2\left(\frac{k \log^2(2)}{50(\log_2(1/\eta)+1)^2 \log_2^2(k)}\right) \rfloor - 2$ .

**LEMMA 25.** *Assume the conditions in Lemma 22 and Lemma 23 hold. Let  $\delta' = \frac{\delta}{32T^4k^2}$ . For any  $1 \leq j \leq \tilde{j}$ , with probability at least  $1 - 8T^4k\delta'$ , it holds that*

$$|\mathcal{W}_j| \leq 8 \cdot 10^7 \cdot ((\log_2(k) + 1)^5 (\log(kR) + \log(1/\delta')) (\log_2(T) + 1)) \cdot 2^j.$$

Armed with Lemma 25, we can demonstrate that: with probability exceeding  $1 - 8T^4k^2\delta' = 1 - \delta/4$ , one has

$$\begin{aligned} \sum_{i=1}^k \bar{w}_i^T &\leq k \cdot 2^{-\tilde{j}-1} + \sum_{j=1}^{\tilde{j}} |\mathcal{W}_j| \cdot 2^{-(j-1)} \\ &\leq 2 \cdot 10^8 \cdot ((\log_2(k) + 1)^6 (\log(kR) + \log(1/\delta)) (\log_2(T) + 1)). \end{aligned}$$

The proof of Lemma 24 is thus complete by noting that  $T = O\left(\frac{d \log(d/\varepsilon) + k \log(Rk/(\varepsilon\delta))}{\varepsilon^2}\right)$ .

*Remark 5.* The proof of Lemma 25 is similar to that of Lemma 4 by regarding  $\sum_{\ell \in \mathcal{L}} u_{i,\ell}^t$  as  $w_i^t$ .

#### E.5 Proof of Lemma 25

With slight abuse of notation, we re-define

$$w_i^t = \sum_{\ell \in \mathcal{L}} u_{i,\ell}^t$$

by running Algorithm 4 for all  $i \in [k]$  and  $1 \leq t \leq T$ . Recall the definition of segments in Definition 1. Similar to Lemma 5, for each  $i \in \mathcal{W}_j$ , there exists a  $(\frac{1}{2^{j+1}}, \frac{1}{2^{j+2}}, \log(2))$ -segment with the index set  $\{i\}$ . We can then present the following counterpart of Lemma 5.

**LEMMA 26.** *For each  $i \in \mathcal{W}_j$ , there exist  $1 \leq s_i < e_i \leq T$  satisfying  $\frac{1}{2^{j+2}} < w_i^{s_i} \leq \frac{1}{2^{j+1}}$ ,  $\frac{1}{2^j} < w_i^{e_i}$ , and  $w_i^t > 2^{-(j+2)}$  for any  $s_i \leq t \leq e_i$ .*

PROOF. Repeating the arguments in proof of Lemma 5, we can readily complete the proof by noting that

$$\log_2 \left( \frac{w_i^{t+1}}{w_i^t} \right) = \log_2 \left( \frac{\sum_{\ell} u_{i,\ell}^{t+1}}{\sum_{\ell} u_{i,\ell}^t} \right) \leq \frac{\eta}{\log(2)} < \frac{1}{4}.$$

□

Armed with Lemma 26, we continue to present the counterpart of Lemma 7 in the following lemma. Note that the proof of Lemma 27 is exactly the same as that of Lemma 7 with the new definitions of  $\mathcal{W}_j$  and  $\{w_i^t\}_{i \in [k], t \in [T]}$ .

LEMMA 27. Given  $\mathcal{W}_j$  and  $(s_i, e_i)$  for  $i \in \mathcal{W}_j$  defined above, there exists a group of subsets  $\{\mathcal{V}_j^n\}_{n=1}^N$  such that the conditions below hold

- (i)  $\mathcal{V}_j^n \subset \mathcal{W}_j$ ,  $\mathcal{V}_j^n \cap \mathcal{V}_j^{n'} = \emptyset$ ,  $\forall n \neq n'$ ;
- (ii)  $\sum_{n=1}^N |\mathcal{V}_j^n| \geq \frac{|\mathcal{W}_j|}{24 \log_2(k)(\log_2(T)+1)}$ ;
- (iii) There exists  $1 \leq \hat{s}_1 < \hat{e}_1 \leq \hat{s}_2 < \hat{e}_2 \leq \dots \leq \hat{s}_N < \hat{e}_N \leq T$ , and  $\{g_n\}_{n=1}^N \in [1, \infty)^N$  such that for each  $1 \leq n \leq N$ ,  $(\hat{s}_n, \hat{e}_n)$  is a  $(2^{-(j+1)} g_n |\mathcal{V}_j^n|, 2^{-(j+2)} |\mathcal{V}_j^n|, \frac{\log(2)}{2 \log_2(k)})$ -segment with index set as  $\mathcal{V}_j^n$ . That is, the following hold for each  $1 \leq n \leq N$ :
  - $-\frac{g_n |\mathcal{V}_j^n|}{2^{j+2}} < \sum_{i \in \mathcal{V}_j^n} w_i^{\hat{s}_n} \leq \frac{g_n |\mathcal{V}_j^n|}{2^{j+1}}$ ;
  - $-\frac{g_n |\mathcal{V}_j^n|}{2^j} \cdot \exp(\frac{\log(2)}{2 \log_2(k)}) < \sum_{i \in \mathcal{V}_j^n} w_i^{\hat{e}_n}$ ;
  - $-\sum_{i \in \mathcal{V}_j^n} w_i^t \geq \frac{|\mathcal{V}_j^n|}{2^{j+2}}$  for any  $\hat{s}_n \leq t \leq \hat{e}_n$ .

The next step is to introduce the counterpart of Lemma 8 as follows: Note that most analysis arguments for establishing Lemma 28 are the same as the ones used in the proof of Lemma 8, except that the noise term is a slightly different. The proof is provided in Appendix E.6.

LEMMA 28. Assume the conditions in Lemma 22 and Lemma 23 hold. Recall the definition of  $h^t$  and  $u^t$  in Algorithm 4, and the definition that  $\text{OPT} = \min_{h \in \mathcal{H}} \max_{i \in [k], \ell \in \mathcal{L}} L_i^{\ell}(h)$ . Also recall that  $v^t = L(h^t, u^t) - \text{OPT}$ . Suppose  $(t_1, t_2)$  is a  $(p, q, x)$ -segment such that  $p \geq 2q$ . Then one has

$$t_2 - t_1 \geq \frac{x}{2\eta}. \quad (181)$$

Moreover, if we assume that  $\frac{qx^2}{50(\log_2(k)+1)^2} \geq \frac{1}{k}$ , it then holds that: with probability at least  $1 - 6T^4 k \delta'$ , at least one of the two claims holds:

- (a) the length of the segment satisfies

$$t_2 - t_1 \geq \frac{qx^2}{200(\log_2(1/\eta) + 1)^2 \eta^2}; \quad (182)$$

- (b) the quantities  $\{v^t\}$  obey

$$4 \sum_{\tau=t_1}^{t_2-1} (-v^{\tau} + \epsilon_1) \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}. \quad (183)$$

With the preceding lemmas in place, we are now ready to prove Lemma 25. In what follows, we denote by  $\{\mathcal{V}_j^n\}_{n=1}^N$  and  $\{(\hat{s}_n, \hat{e}_n)\}_{n=1}^N$  the construction in Lemma 27.

First of all, we make the observation that: for any  $1 \leq j \leq \tilde{j}$ , one has

$$2^{-(j+2)}|\mathcal{V}_j^n| \cdot \frac{\log^2(2)}{50(\log_2(1/\eta) + 1)^2 \log_2^2(k)} \geq 2^{-(\tilde{j}+2)} \cdot \frac{\log^2(2)}{50(\log_2(1/\eta) + 1)^2 \log_2^2(k)} \geq \frac{1}{k}.$$

Recall that  $v^\tau \leq \varepsilon_1$ . Combining this fact with Lemma 28 (by setting  $q = 2^{-(j+2)}|\mathcal{V}_j^n|$  and  $x = \frac{\log(2)}{\log_2(k)}$ ) allows us to show that, for each  $1 \leq n \leq N$ ,

$$\begin{aligned} T\eta + \left\{ 4T\varepsilon_1 + 4 \sum_{t=1}^T (-v^t) \right\} &\geq \sum_{n=1}^N (\hat{e}_n - \hat{s}_n) \eta + 4 \sum_{n=1}^N \sum_{\tau=\hat{s}_n}^{\hat{e}_n-1} (-v^\tau + \varepsilon_1) \\ &\geq \frac{2^{-(j+2)} \sum_{n=1}^N |\mathcal{V}_j^n| \log^2(2)}{800 \log_2^2(k) (\log_2(1/\eta) + 1)^2 \eta}, \end{aligned} \quad (184)$$

where the first inequality makes use of the disjoint nature of the segments  $\{(\hat{s}_n, \hat{e}_n)\}_{1 \leq n \leq N}$ , and the second inequality follows from Lemma 28. In addition, Equation (177) in Lemma 23 tells us that

$$\sum_{t=1}^T (-v^t) \leq 100 \left( \frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')} \right). \quad (185)$$

Taking this collectively with Equation (184) gives

$$\begin{aligned} \sum_{n=1}^N |\mathcal{V}_j^n| &\leq \frac{3200(\log_2(k) + 1)^4 \eta \cdot 2^{j+2}}{\log^2(2)} \cdot \left( 100 \left( \frac{\log(kR)}{\eta} + \eta T + 2\sqrt{T \log(1/\delta')} \right) \right) + \\ &\quad + \frac{4000T(\log_2(k) + 1)^4 \cdot 2^{j+2} \eta^2}{\log^2(2)}. \end{aligned} \quad (186)$$

Finally, it follows from Property (ii) of Lemma 27 that

$$\begin{aligned} |\mathcal{W}_j| &\leq 24 \log_2(k) (\log_2(T) + 1) \left( \sum_{n=1}^N |\mathcal{V}_j^n| \right) \\ &\leq 8 \cdot 10^7 \cdot ((\log_2(k) + 1)^5 (\log(kR) + \log(1/\delta')) (\log_2(T) + 1)) \cdot 2^j, \end{aligned}$$

thus concluding the proof.

## E.6 Proof of Lemma 28

Throughout this proof, let us take

$$Z^t = \sum_{i \in [k], t \in \mathcal{L}} U_{i,t}^t.$$

We begin by establishing the first claim (181). By definition, there exists  $i \in [k]$  such that

$$\log \left( \frac{w_i^{t_2}}{w_i^{t_1}} \right) \geq x,$$

which corresponds to

$$\eta \sum_{\tau=t_1}^{t_2-1} \max_{\ell \in \mathcal{L}} \hat{r}_{i,\ell}^\tau - \log \left( \frac{Z^{t_2}}{Z^{t_1}} \right) \geq x.$$

Given that  $\log(Z^{t_2}/Z^{t_1}) \geq -\eta(t_2 - t_1)$  and  $\widehat{r}_{i,\ell}^\tau \leq 1$  for any  $1 \leq \tau \leq T$ , we can demonstrate that

$$x \leq 2(t_2 - t_1)\eta,$$

from which the claim (181) follows.

We now turn to the remaining claims of Lemma 28. For each hypothesis  $h \in \mathcal{H}$ , let us introduce the following vector  $\bar{\lambda}_h \in \mathbb{R}^{kR}$ :

$$\bar{\lambda}_h = [\bar{\lambda}_{h,i,\ell}]_{i \in [k], \ell \in \mathcal{L}} \quad \text{with } \bar{\lambda}_{h,i,\ell} = L_i^\ell(h) - \text{OPT}. \quad (187)$$

Give the  $\varepsilon$ -optimality of  $h^t$  (see Lemma 22), we have the following property that holds for any  $1 \leq \tau, t \leq T$ ,

$$\langle \bar{\lambda}_{h^\tau}, u^t \rangle \geq \langle \bar{\lambda}_{h^t}, u^t \rangle - \varepsilon_1 = v^t - \varepsilon_1. \quad (188)$$

Summing over  $\tau$  leads to

$$\sum_{\tau=t_1}^{t_2-1} \langle \bar{\lambda}_{h^\tau}, u^t \rangle \geq (t_2 - t_1)(v^t - \varepsilon_1). \quad (189)$$

In the sequel, we divide the remaining proof into several steps.

**Step 1: decomposing the KL divergence between  $u^t$  and  $u^{t_2}$ .** Write

$$U_{i,\ell}^t = \exp \left( \eta \sum_{\tau=1}^t \widehat{r}_{i,\ell}^\tau \right) = \exp \left( \eta \sum_{\tau=1}^t (\bar{\lambda}_{h^\tau,i,\ell} + \text{OPT} + \xi_{i,\ell}^\tau) \right) \quad \text{with } \xi_{i,\ell}^\tau = \widehat{r}_{i,\ell}^\tau - \bar{h}_{i,\ell}^\tau - \text{OPT}, \quad (190)$$

where  $\xi_{i,\ell}^\tau$  is a zero-mean random variable. Re-define

$$\Delta_{t_1,t_2} = \sum_{\tau=t_1}^{t_2-1} \xi^\tau \quad \text{with } \xi^\tau = [\xi_{i,\ell}^\tau]_{i \in [k], \ell \in \mathcal{L}} \quad (191)$$

Taking  $U^t = [U_{i,\ell}^t]_{i \in [k], \ell \in \mathcal{L}}$  and denoting  $\log(x/y)$  the vector  $\{\log(x_{i,\ell}/y_{i,\ell})\}_{i \in [k], \ell \in \mathcal{L}}$  for two  $kR$ -dimensional vectors  $(x, y)$ , one can then deduce that

$$\left\langle \frac{1}{\eta} \log \left( \frac{W^{t_2}}{W^{t_1}} \right) - \Delta_{t_1,t_2}, u^t \right\rangle - (t_2 - t_1)\text{OPT} = \sum_{\tau=t_1}^{t_2-1} \langle \bar{\lambda}_{h^\tau}, u^t \rangle \geq (t_2 - t_1)(v^t - \varepsilon_1), \quad (192)$$

where the last inequality results from Equation (189). Recall that  $Z^t = \sum_{i \in [k], \ell \in \mathcal{L}} U_{i,\ell}^t$  and  $u_{i,\ell}^t = \frac{U_{i,\ell}^t}{Z^t}$ . By taking  $t_1 = t$ , we can derive from Equation (192) that

$$\left\langle \log \left( \frac{w^{t_2}}{w^t} \right) - \eta \Delta_{t_1,t_2}, u^t \right\rangle + \log \left( \frac{Z^{t_2}}{Z^t} \right) - \eta(t_2 - t)\text{OPT} \geq \eta(t_2 - t)(v^t - \varepsilon_1). \quad (193)$$

This result in turn assists in bounding the KL divergence between  $u^t$  and  $u^{t_2}$ :

$$\begin{aligned} \text{KL}(u^t \| u^{t_2}) &:= u^t \cdot \left( \log \left( \frac{u^t}{u^{t_2}} \right) \right) \\ &\leq \log(Z^{t_2}/Z^t) - \eta(t_2 - t)\text{OPT} - \eta u^t \cdot \Delta_{t_1,t_2} + (t_2 - t)\eta(\varepsilon_1 - v^t). \end{aligned} \quad (194)$$

In the following, we shall cope with the right-hand side of Equation (194)

**Step 2: bounding the term  $\log(Z^{t_2}/Z^t)$ .** With probability at least  $1 - 2T^2k\delta'$ , it holds that

$$\begin{aligned} \log(Z^{t_2}/Z^t) &:= \sum_{\tau=t}^{t_2-1} \log(Z^{\tau+1}/Z^\tau) \\ &= \sum_{\tau=t}^{t_2-1} \log \left( \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^\tau \exp(\eta \hat{r}_{i,\ell}^\tau) \right) \\ &\leq \sum_{\tau=t}^{t_2-1} \left( \eta \sum_{i=1}^k u_{i,\ell}^\tau \hat{r}_{i,\ell}^\tau + 2\eta^2 \right) \\ &\leq \eta \sum_{\tau=t}^{t_2-1} v^\tau + \eta(t_2-t)\text{OPT} + \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} (u_{i,\ell}^\tau (\hat{r}_{i,\ell}^\tau - \bar{\lambda}_{h^\tau,i,\ell} - \text{OPT})) + 2(t_2-t)\eta^2 \end{aligned} \quad (195)$$

$$\leq \eta(t_2-t)\varepsilon_1 + \eta(t_2-t)\text{OPT} + \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} (u_{i,\ell}^\tau (\hat{r}_{i,\ell}^\tau - \bar{\lambda}_{h^\tau,i,\ell} - \text{OPT})) + 2(t_2-t)\eta^2. \quad (196)$$

Here, Equation (195) holds true due to the fact that  $v^\ell = \langle u^\ell, \bar{\lambda}_h \rangle$ , and Equation (196) is comes from the fact that  $v^\tau \leq \varepsilon_1$ .

**Step 3: bounding the weighted sum of  $\{\xi_{i,\ell}^\tau\}$ .** Next, we intend to control the two random terms below

$$\eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^\tau \cdot (\hat{r}_{i,\ell}^\tau - \bar{\lambda}_{h^\tau,i,\ell} - \text{OPT}) = \eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^\tau \xi_{i,\ell}^\tau, \quad (197)$$

$$-\eta \langle u^\ell, \Delta_{t,t_2} \rangle = -\eta \sum_{\tau=t}^{t_2-1} \sum_{i=1}^k \sum_{\ell \in \mathcal{L}} u_{i,\ell}^\tau \xi_{i,\ell}^\tau. \quad (198)$$

Let  $\overline{\mathcal{F}}^\tau$  denote all events happening before the  $\tau$ -th round in Algorithm 4. Direct computation yields

$$\begin{aligned} &\sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \sum_{\ell} u_{i,\ell}^{t_1} (\hat{r}_{i,\ell}^\tau - \bar{\lambda}_{h,i,\ell}^\tau - \text{OPT}) \\ &= \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{1}{\lceil k\bar{w}_i^\tau \rceil} \sum_{j=1}^{\lceil k\bar{w}_i^\tau \rceil} \left( \sum_{\ell} u_{i,\ell}^{t_1} \ell(h^\tau, (x_{i,j}^\tau, y_{i,j}^\tau)) - \sum_{\ell} u_{i,\ell}^{t_1} L_i^\ell(h^\tau) \right). \end{aligned} \quad (199)$$

Note that  $\{(x_{i,j}, y_{i,j})\}_{j=1}^{\lceil k\bar{w}_i^\tau \rceil}$  are independently sampled conditioned on  $\overline{\mathcal{F}}^\tau$ . Recalling the fact that  $|\sum_{\ell} u_{i,\ell}^{t_1} \ell(h^\tau, (x_{i,j}^\tau, y_{i,j}^\tau))| \leq w_i^{t_1} \leq \bar{w}_i^\tau$  for any  $t_1 \leq \tau \leq t_2 - 1$ , one can apply Freedman's inequality (cf. Lemma 12) to see that

$$\begin{aligned} &\left| \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{1}{\lceil k\bar{w}_i^\tau \rceil} \sum_{j=1}^{\lceil k\bar{w}_i^\tau \rceil} \left( \sum_{\ell} u_{i,\ell}^{t_1} \ell(h^\tau, (x_{i,j}^\tau, y_{i,j}^\tau)) - \sum_{\ell} u_{i,\ell}^{t_1} L_i^\ell(h^\tau) \right) \right| \\ &\leq 2 \sqrt{\log(2/\delta')} \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \sum_{j=1}^{\lceil k\bar{w}_i^\tau \rceil} \frac{(w_i^{t_1})^2}{(\lceil k\bar{w}_i^\tau \rceil)^2} + 2 \log(2/\delta') \end{aligned}$$

$$\begin{aligned}
&= 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{(w_i^{\tau})^2}{|kw_i^\tau|}} + 2 \log(2/\delta') \\
&\leq 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{w_i^{\tau}}{k}} + 2 \log(2/\delta') \\
&\leq 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \frac{1}{k}} + 2 \log(2/\delta')
\end{aligned} \tag{200}$$

holds with probability exceeding  $1 - \delta'$ . The bound of Equation (197) is thus finished. We then use similar arguments to show that, with probability exceeding  $1 - \delta'$ ,

$$\begin{aligned}
&\left| \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{1}{|kw_i^\tau|} \sum_{j=1}^{|kw_i^\tau|} \left( \sum_{\ell} u_{i,\ell}^\tau \ell(h^\tau, (x_{i,j}^\tau, y_{i,j}^\tau)) - \sum_l u_{i,l}^\tau L_i^\ell(h^\tau) \right) \right| \\
&\leq 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \sum_{j=1}^{|kw_i^\tau|} \frac{(w_i^\tau)^2}{(|kw_i^\tau|)^2}} + 2 \log(2/\delta') \\
&= 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{(w_i^\tau)^2}{|kw_i^\tau|}} + 2 \log(2/\delta') \\
&\leq 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \sum_{i=1}^k \frac{w_i^\tau}{k}} + 2 \log(2/\delta') \\
&\leq 2 \sqrt{\log(2/\delta') \sum_{\tau=t_1}^{t_2-1} \frac{1}{k}} + 2 \log(2/\delta').
\end{aligned} \tag{201}$$

**Step 4: bounding the KL divergence between  $u^t$  and  $u^{t_2}$ .** Combining Equation (194), Equation (196), Equation (200) and Equation (201), and applying the union bounds over  $(t, t_2)$ , we see that with probability exceeding  $1 - 6T^4 k \delta'$ ,

$$\begin{aligned}
\text{KL}(w^t \| w^{t_2}) &\leq \text{KL}(u^t \| u^{t_2}) \leq 2(t_2 - t)\eta\varepsilon_1 - (t_2 - t)\eta\nu^t \\
&\quad + 4\eta\sqrt{\frac{(t_2 - t)\log(2/\delta')}{k}} + 2(t_2 - t)\eta^2 + 4\log(2/\delta')\eta,
\end{aligned} \tag{202}$$

holds for any  $1 \leq t < t_2 \leq T$ . The analysis below then operates under the condition that Equation (202) holds for any  $1 \leq t < t_2 \leq T$ .

**Step 5: connecting the KL divergence with the advertised properties.** Recall that  $j_{\max} = \lfloor \log_2(1/\eta) + 1 \rfloor$ . Set

$$\begin{aligned}
\tau_{\hat{j}} &:= \min \{ \tau \mid t_1 \leq \tau \leq t_2 - 1, -v^\tau \leq 2^{-(\hat{j}-1)} \}, \quad 1 \leq \hat{j} \leq j_{\max} \\
\tau_{j_{\max+1}} &:= t_2.
\end{aligned} \tag{203}$$

By definition, we know that  $\tau_1 = t_1$  and  $\tau_{j_2} \geq \tau_{j_1}$  for  $j_2 \geq j_1$ . Let  $\mathcal{J}$  be the index set of this segment. Let  $y_j := \sum_{i \in \mathcal{J}} w_i^{\tau_j}$ .

We then claim that there exists  $1 \leq \tilde{j} \leq j_{\max}$  such that

$$\log\left(\frac{y_{\tilde{j}+1}}{y_{\tilde{j}}}\right) \geq \frac{x}{\log_2(1/\eta) + 1}. \quad (204)$$

**PROOF OF EQUATION (204).** Suppose that there exists no  $1 \leq \tilde{j} \leq j_{\max}$  satisfying Equation (204). Then for any  $j$ , it holds that  $\log\left(\frac{y_{j+1}}{y_j}\right) \leq \frac{x}{\log_2(1/\eta) + 1}$ , which implies that  $y_j \geq y_{j+1} \exp\left(\frac{x}{\log_2(1/\eta) + 1}\right)$ . As a result, we have

$$y_1 \geq y_{j_{\max}+1} \cdot \exp\left(-j_{\max} \cdot \frac{x}{\log_2(1/\eta) + 1}\right) \geq p,$$

which leads to contradiction (since according to the definition of the  $(p, q, x)$ -segment, one has  $y_1 \leq p$ ).  $\square$

Now, assume that  $\tilde{j}$  satisfies Equation (204). From the definition of the  $(p, q, x)$ -segment, we have  $y_{\tilde{j}} \geq q$ . It then follows from Equation (202) that

$$\begin{aligned} \text{KL}(w^{\tilde{j}} \| w^{\tau_{\tilde{j}+1}}) &\leq 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta\varepsilon_1 + (\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta 2^{-(\tilde{j}-1)} \\ &\quad + 4\eta\sqrt{\frac{(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\log(2/\delta')}{k}} + 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta^2 + 4\log(2/\delta')\eta. \end{aligned}$$

Since  $\log\left(\frac{y_{\tilde{j}+1}}{y_{\tilde{j}}}\right) \geq \frac{x}{\log_2(1/\eta) + 1}$  and  $y_{\tilde{j}} \geq q$ , we can invoke Lemma 15 and Lemma 16 to show that

$$\text{KL}(w^{\tau_{\tilde{j}}} \| w^{\tau_{\tilde{j}+1}}) \geq \text{KL}\left(\text{Ber}(y_{\tilde{j}}) \| \text{Ber}(y_{\tilde{j}+1})\right) \geq \frac{qx^2}{4(\log_2(1/\eta) + 1)^2},$$

where  $\text{Ber}(x)$  denote the Bernoulli distribution with parameter  $x \in [0, 1]$ . As a result, we obtain

$$\begin{aligned} \frac{qx^2}{4(\log_2(1/\eta) + 1)^2} &\leq 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta\varepsilon_1 + (\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta 2^{-(\tilde{j}-1)} \\ &\quad + 4\eta\sqrt{\frac{(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\log(2/\delta')}{k}} + 2(\tau_{\tilde{j}+1} - \tau_{\tilde{j}})\eta^2 + 4\log(2/\delta')\eta. \end{aligned} \quad (205)$$

Combining Equation (205) with the facts that: (i)  $\frac{qx^2}{50(\log_2(1/\eta) + 1)^2} \geq \frac{1}{k}$ ; (ii)  $\frac{2^{j_{\max}}}{\eta} \geq \frac{1}{\eta^2}$ , we can demonstrate that

$$\begin{aligned} \tau_{\tilde{j}+1} - \tau_{\tilde{j}} &\geq \min\left\{\frac{qx^2}{100(2\log_2(k) + 1)^2} \min\left\{\frac{1}{\eta\varepsilon_1}, \frac{2^{\tilde{j}-1}}{\eta}, \frac{1}{\eta^2}\right\}, \frac{kx^2 2^{-2l_1}}{10000\eta^2 \log(1/\delta') (\log_2(k) + 1)^4}\right\} \\ &\geq \frac{qx^2 \cdot 2^{\tilde{j}-1}}{100(\log_2(1/\eta) + 1)^2 \eta}. \end{aligned} \quad (206)$$

With Equation (206) in place, we are ready to finish the proof by dividing into two cases.

– Case 1:  $\tilde{j} = j_{\max}$ . It follows from Equation (206) that

$$t_2 - t_1 \geq \tau_{\tilde{j}+1} - \tau_{\tilde{j}} \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)} \frac{2^{j_{\max}-1}}{\eta} \geq \frac{qx^2}{200(\log_2(1/\eta) + 1)^2 \eta^2}.$$

– Case 2:  $1 \leq \tilde{j} \leq j_{\max} - 1$ . It comes from the definition Equation (203) that

$$\sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\hat{j}}\} \geq \sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau > 2^{-\hat{j}}\} \geq \tau_{\hat{j}+1} - t_1 \geq \tau_{\hat{j}+1} - \tau_{\hat{j}}, \quad \text{for any } 1 \leq \hat{j} \leq j_{\max} - 1.$$

When  $1 \leq \tilde{j} \leq j_{\max} - 1$ , the above display taken collectively with Equation (202) gives

$$\sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\tilde{j}}\} \geq \tau_{\tilde{j}+1} - \tau_{\tilde{j}} \geq \frac{qx^2 \cdot 2^{\tilde{j}-1}}{100(\log_2(1/\eta) + 1)^2 \eta}, \quad (207)$$

and as a result,

$$\sum_{\tau=t_1}^{t_2-1} \sum_{\hat{j}=1}^{j_{\max}-1} \mathbb{1}\{\max\{-v^\tau, 0\} \geq 2^{-\hat{j}}\} 2^{-(\hat{j}-1)} \geq \sum_{\tau=t_1}^{t_2-1} \mathbb{1}\{-v^\tau \geq 2^{-\tilde{j}}\} 2^{-(\tilde{j}-1)} \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}. \quad (208)$$

By observing that

$$\sum_{\hat{j}=1}^{\infty} \mathbb{1}\{x \geq 2^{-\hat{j}}\} \cdot 2^{-(\hat{j}-1)} \leq 4x$$

holds for any  $x \geq 0$ , one can combine this fact with Equation (208) to show that

$$4 \sum_{\tau=t_1}^{t_2-1} \max\{-v^\tau, 0\} \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}. \quad (209)$$

Additionally, recalling that  $v^\tau \leq \varepsilon_1$ , one can deduce that

$$\sum_{\tau=t_1}^{t_2-1} \max\{-v^\tau, 0\} = \sum_{\tau=t_1}^{t_2-1} (-v^\tau) - \sum_{\tau=t_1}^{t_2-1} \min\{-v^\tau, 0\} \leq \sum_{\tau=t_1}^{t_2-1} (-v^\tau) + (t_2 - t_1)\varepsilon_1,$$

which combined with Equation (209) yields

$$4(t_2 - t_1)\varepsilon_1 + 4 \sum_{\tau=t_1}^{t_2-1} (-v^\tau) \geq \frac{qx^2}{100(\log_2(1/\eta) + 1)^2 \eta}.$$

The proof is thus complete.

Received 13 July 2024; revised 15 March 2025; accepted 27 July 2025



# Bayesian Design Principles for Frequentist Sequential Learning

**YUNBEI XU**, Department of Industrial Systems Engineering & Management, National University of Singapore, Singapore, Singapore

**ASSAF ZEEVI**, Graduate School of Business, Columbia University, New York, United States

---

We develop a general theory to optimize the frequentist regret for sequential learning problems, from which efficient bandit and reinforcement learning algorithms can be derived via unified Bayesian principles. Building on the recent Decision-Estimation Coefficient (DEC) framework, we propose a novel optimization approach to generate “algorithmic beliefs” at each round and use Bayesian posteriors for decision-making. The optimization objective, termed “Algorithmic Information Ratio” (AIR), represents an intrinsic complexity measure that effectively characterizes the frequentist regret of any algorithm. Although AIR’s minimax regret aligns with that provided by DEC, it additionally offers an algorithm-dependent perspective—distinct from a minimax complexity-facilitating algorithm design and analysis. Specifically, AIR enables deriving explicit algorithms via belief parameterization and provides clear approximation guidelines with provable guarantees. Moreover, the resulting algorithms have a simple structure and are computationally efficient for several representative problems. We illustrate our framework with a novel algorithm for multi-armed bandits that performs strongly across stochastic, adversarial, and non-stationary environments, and demonstrate applicability to linear bandits, convex bandits, and reinforcement learning.

CCS Concepts: • Theory of computation → Sequential decision making; Online learning theory;

Additional Key Words and Phrases: Bandit algorithms, reinforcement learning, algorithmic beliefs, algorithmic information ratio, adaptive posterior sampling, frequentist regret

**ACM Reference Format:**

Yunbei Xu and Assaf Zeevi. 2025. Bayesian Design Principles for Frequentist Sequential Learning. *J. ACM* 72, 5, Article 37 (October 2025), 65 pages. <https://doi.org/10.1145/3766898>

---

## 1 Introduction

### 1.1 Background

We address a broad class of sequential learning problems in the presence of *partial feedback*, which arise in numerous application areas including personalized recommendation [37], game playing [46], and control [40]. An agent sequentially chooses among a set of possible decisions to maximize the cumulative reward. By “partial feedback” we mean the agent is only able to observe the feedback of her chosen decision, but does not generally observe what the feedback would be if she had chosen a different decision. For example, in **multi-armed bandits (MAB)**, the agent can only observe the

---

A preliminary version of this article appeared at ICML 2023.

Authors’ Contact Information: Yunbei Xu (corresponding author), Department of Industrial Systems Engineering & Management, National University of Singapore, Singapore, Singapore; e-mail: yunbeixu.xyb@gmail.com; Assaf Zeevi, Graduate School of Business, Columbia University, New York, New York, United States; e-mail: assaf@gsb.columbia.edu.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 0004-5411/2025/10-ART37

<https://doi.org/10.1145/3766898>

reward of her chosen action, but does not observe the rewards of other actions. In **reinforcement learning (RL)**, the agent is only able to observe her state insofar as the chosen action is concerned, while other possible outcomes are not observed and the underlying state transition dynamics are unknown. In this article, we present a unified approach that applies to bandit problems, RL, and beyond.

The central challenge for sequential learning with partial feedback is to determine the optimal tradeoff between *exploration* and *exploitation*. That is, the agent needs to try different decisions to learn the environment; at the same time, she wants to focus on good” decisions that maximize her payoff. There are two basic approaches to study such exploration-exploitation tradeoff: frequentist and Bayesian. One of the most celebrated examples of the frequentist approach is the family of **Upper Confidence Bound (UCB)** algorithms [6, 31]. Here, the agent typically uses sample average or regression to estimate the mean rewards; and she optimizes the UCBs of the mean rewards to make decisions. Another widely used frequentist algorithm is EXP3 [7] which was designed for adversarial bandits; it uses **inverse probability weighting (IPW)** to estimate the rewards, and then applies exponential weighting to construct decisions. One of the most celebrated examples of the Bayesian approach is **Thompson Sampling (TS)** with a pre-specified, fixed prior [51]. Here, the agent updates the Bayesian posterior at each round to learn the environment, and she uses draws from that posterior to optimize decisions.

The advantage of the frequentist approach is that it does not require a priori knowledge of the environment. However, it heavily depends on a case-by-case analysis exploiting special structure of a particular problem. For example, regression-based approaches can not be easily extended to adversarial problems; and IPW-type estimators are only known for simple rewards/losses such as discrete and linear. The advantage of the Bayesian approach is that Bayesian posterior is a generic and often optimal estimator if the prior is known. However, the Bayesian approach requires knowing the prior at the inception, which may not be accessible in complex or adversarial environments. Moreover, maintaining posteriors is computationally expensive for most priors.

In essence, the frequentist approach requires less information, but is less principled, or more bottom-up. On the other hand, the Bayesian approach is more principled, or top-down, but requires stronger assumptions. In this article we focus on the following research question:

*Can we design principled Bayesian-type algorithms, that are prior-free, computationally efficient, and work well in both stochastic and adversarial/non-stationary environments?*

Our work can be viewed as a direct follow-up to recent influential results, particularly the **Decision-Estimation Coefficient (DEC)** framework introduced by [23, 24]. While we do not surpass these works in terms of statistical complexity or general-purpose algorithms, we address key questions left open by prior research, specifically: (i) how the general optimization problems inherent in the DEC framework can be addressed via appropriate parameterization and gradient-based approximation; and (ii) how this general theory can help analyze and advance existing, widely-used methods—particularly posterior-based (Bayesian-type) and IPW-based algorithms—whose strengths and limitations we outlined above.

## 1.2 Contributions

In this article, we synergize frequentist and Bayesian approaches to successfully answer the above question, through a novel idea that creates “algorithmic beliefs” that are generated sequentially in each round, and uses Bayesian posteriors to make decisions. Our contributions encompass comprehensive theory, novel methodology, and applications. We summarize the main contributions as follows.

*Making Bayesian-type algorithms prior-free and applicable to adversarial settings.* We provide a new approach that allows Bayesian-type algorithms to operate without prior assumptions, is applicable even in adversarial (and non-stationary) settings. This approach is general and often computationally tractable, and it achieves regret bounds matching the best known results in the literature (in particular, it attains the same order of regret as the existing DEC-based guarantees). In addition to its applicability in adversarial/non-stationary environments, our approach offers the advantages of being prior-free and often computationally manageable, which are typically not achievable by traditional Bayesian algorithms, except for simple model classes like discrete [4] and linear [5] rewards/losses.

**Algorithmic Information Ratio (AIR)**—*a variant of the DEC.* We introduce intrinsic complexity measures called the AIR and its stochastic counterpart, **Model-index AIR (MAIR)**, which serve as objective functions for generating “algorithmic beliefs” using round-dependent information. Our approach selects algorithmic beliefs by approximately maximizing AIR, and we establish that AIR provides a general bound on the frequentist regret of any algorithm. Importantly, we demonstrate that in the worst-case setting, AIR and MAIR matches (but does not surpass) the complexity of the DEC; and it is always bounded by the Bayesian **Information Ratio (IR)**. Thus, AIR recovers the tight regret guarantees previously established by DEC and Bayesian IR. While rooted in the foundational work on DEC, our approach further extends these ideas by providing a principled way to construct practical frequentist algorithms via belief parameterization, gradient-based optimization, and provable approximation methods, broadening their algorithmic applicability.

*Novel algorithm for MAB with adaptive performance across regimes.* As a major illustration, we propose a novel algorithm for Bernoulli MAB that adapts to stochastic, adversarial, and non-stationary regimes, providing rigorous guarantees and achieving state-of-the-art empirical performance in each case. This algorithm is quite different from, and significantly outperforms the traditional EXP3 algorithm (the default choice for adversarial MAB for decades) in adversarial settings. At the same time, the algorithm outperforms UCB and is comparable to TS in the stochastic environment. Moreover, it outperforms strong benchmarks specially designed for non-stationary environments, including EXP3.S and “clairvoyant” restarted algorithms.

*Applications to linear bandits, bandit convex optimization, and reinforcement learning.* Our theory can be applied to various settings, including linear and convex bandits and RL, by the principle of approximately maximizing AIR. Specifically, for linear bandits, we derive a modified version of EXP2 based on our framework, which establishes a novel connection between IPW and Bayesian posteriors. For bandit convex optimization, we propose the first algorithm that attains the best-known  $\tilde{O}(d^{2.5}\sqrt{T})$  regret of [32] with a finite  $\text{poly}(e^d \cdot T)$  running time. Lastly, in RL, we build upon the DEC framework of [23] to develop Bayesian-type algorithms whose frequentist regret bounds align with existing results.

### 1.3 Related Literature

The literature in the broad area of our research is vast. To maintain a streamlined and concise presentation, we focus solely on the most relevant works.

References [44, 45] propose the concept of IR to analyze and design Bayesian bandit algorithms. Their work studies Bayesian regret with a known prior rather than the frequentist regret. Reference [33] proposes an algorithm called **Exploration by Optimization (EBO)**, which is the first general frequentist algorithm that optimally bounds the frequentist regret of bandit problems and

partial monitoring using IR. However, the EBO algorithm is more of a conceptual construct as it requires intractable optimization over the complete class of functional estimators,” and hence may not be implementable in most settings of interest. Our algorithms are inspired by EBO, but are simpler in structure and run in decision and model spaces (rather than intractable functional spaces). In particular, our approach advances EBO by employing explicit construction of estimators, offering flexibility in selecting updating rules, and providing thorough design and computation guidelines with provable guarantees.

Our work also builds upon the line of research initiated by [23], which introduces the concept of the DEC as a general complexity measure for bandit and RL problems. Remarkably, DEC not only provides sufficient conditions but also offers the first necessary characterization (lower bounds) for interactive decision making, akin to the VC dimension and the Rademacher complexity in statistical learning. It is important to note that while DEC is tight for one part of the regret (as demonstrated in [21, 23]), the remaining “estimation complexity” term in the regret bound can be sub-optimal when the model class is large. The proposed E2D algorithm in [23] separates the black-box estimation method from the decision-making rule. For this reason, E2D is often unable to achieve optimal regret, even for MAB, when the size of the model class causes redundant estimation complexity. Moreover, E2D only works for stochastic environments. Reference [23] also provides an adaptation of EBO from [33] to improve estimation complexity, and the subsequent work [24] extends the theory of DEC to adversarial environments by adapting EBO. However, as adaptations of EBO, these algorithms present computational challenges, as discussed earlier.

Our contributions build directly upon the foundational works of [23, 24]. In particular, our AIR can be viewed as a reformulation of DEC, with added flexibility through an algorithm-dependent perspective and explicit posterior interpretation. Under this reformulation, optimal decision rules naturally emerge as posterior distributions. Consequently, by approximately solving for “algorithmic beliefs,” we derive concrete algorithms whose guarantees closely align with those identified within the DEC framework. Thus, AIR *operationalizes* the insights of DEC, providing a constructive perspective and approximation procedure that complements DEC’s statistical and algorithmic foundation.

## 2 Preliminaries and Definition of AIR

### 2.1 Problem Formulation

To state our results in the broadest manner, we adopt the general formulation of **Adversarial Decision Making with Structured Observation (Adversarial DMSO)** introduced in [24]. The setting covers broad problems including bandit problems, RL, partial monitoring, and beyond. For a locally compact metric space we denote by  $\Delta(\cdot)$  the set of Borel probability measures on that space. Let  $\Pi$  be a compact decision space. Let  $\mathcal{M}$  be a compact model class where each model  $M : \Pi \rightarrow \Delta(\mathcal{O})$  is a mapping from the decision space to a distribution over the observation space  $\mathcal{O}$ . A problem instance in this protocol can be described by the decision space  $\Pi$  and the model class  $\mathcal{M}$ . We define the mean reward function associated with model  $M$  by  $f_M$ .

Consider a  $T$ -round game played by a randomized player in an adversarial environment. At each round  $t = 1, \dots, T$ , the agent determines a probability  $p_t$  over the decisions, and the environment selects a model  $M_t \in \mathcal{M}$  (which can depends on past history and  $p_t$ ). Then the decision  $\pi_t \sim p_t$  is sampled and an observation  $o_t \sim M_t(\pi_t)$  is revealed to the agent. An *admissible algorithm* ALG can be described by a sequence of mappings where the  $t$ -th mapping maps the past decision and observation sequence  $\{\pi_i, o_i\}_{i=1}^{t-1}$  to a probability  $p_t$  over decisions. Throughout the article we denote  $\{\mathfrak{F}_t\}_{t=1}^T$  as the filtration where  $\mathfrak{F}_t$  is the  $\sigma$ -algebra generated by random variables  $\{\pi_s, o_s\}_{s=1}^t$ , and use the notations  $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot | \mathfrak{F}_t]$  for conditional expectation starting round  $t + 1$ .

The *frequentist regret* of the algorithm ALG against the usual target of single best decision in hindsight is defined as

$$\mathfrak{R}_T = \sup_{\pi^* \in \Pi} \mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\pi^*) - \sum_{t=1}^T f_{M_t}(\pi_t) \right],$$

where the expectation is taken with respect to the randomness in decisions and observations. There is a large literature that focuses on the so-called stochastic environment, where  $M_t = M^* \in \mathcal{M}$  for all rounds, and the single best decision  $\pi^* \in \arg \min f_{M^*}(\pi)$  is the natural oracle. Regret bounds for adversarial sequential learning problems naturally apply to stochastic problems. We illustrate how the general formulation covers bandit problems, and leave the discussion of RL to Section 7.

*Example 2.1 (Bernoulli Multi-armed Bandits (MAB)).* We illustrate how the general formulation reduces to the basic MAB problem with Bernoulli reward. Let  $\Pi = [K] = \{1, \dots, K\}$  be a finite set of  $K$  actions, and  $\mathcal{F}$  be the set of all possible mappings from  $[K]$  to  $[0, 1]$ . Take  $\mathcal{M} = \{M_f : f \in \mathcal{F}\}$  as the induced model class, where each  $M_f$  maps  $\pi$  into the Bernoulli distribution  $\text{Bern}(f(\pi))$ . The mean reward function for model  $M_f$  is  $f$  itself. At each round  $t$ , the environment selects a mean reward function  $f_t$ , and the observation  $o_t$  is the incurred reward  $r_t(\pi_t) \sim \text{Bern}(f_t(\pi_t))$ .

*Example 2.2 (Structured Bandits).* We consider bandit problems with general structure of the mean reward function. Let  $\Pi$  be a  $d$ -dimensional action set, and  $\mathcal{F} \subseteq \{f : \Pi \rightarrow [0, 1]\}$  be a function class that encodes the structure of the mean reward function. Take  $\mathcal{M} = \{M_f : f \in \mathcal{F}\}$  as the induced model class, where each  $M_f$  maps  $\pi$  to the Bernoulli distribution  $\text{Bern}(f(\pi))$ . The mean reward function for model  $M_f$  is  $f$  itself. For example, in  $d$ -dimensional linear bandits, the mean reward function  $f$  is parametrized by some  $\theta \in \Theta \subseteq \mathbb{R}^d$  such that  $f(\pi) = \theta^T \pi, \forall \pi \in \Pi$ . And in bandit convex optimization, the mean reward (or loss) function class  $\mathcal{F}$  is the set of all concave (or convex) mappings from  $\Pi$  to  $[0, 1]$ .

## 2.2 Algorithmic Information Ratio

Let  $v$  be a probability measure of the joint random variable  $(M, \pi^*) \in \mathcal{M} \times \Pi$ , and  $p$  be a distribution of another independent random variable  $\pi \in \Pi$ . Given a probability measure  $v$ , let

$$v_{\pi^*}(\cdot) = \int_{\mathcal{M}} v(M, \cdot) dM$$

be the marginal distribution of  $\pi^* \in \Pi$ . Viewing  $v$  as a prior belief over  $(M, \pi^*)$ , we define  $v(\cdot | \pi, o)$  as the Bayesian posterior belief conditioned on the decision being  $\pi$  and the observation (generated from the distribution  $M(\pi)$ ) being  $o$ ; and we define the marginal posterior belief of  $\pi^*$  conditioned on  $\pi$  and  $o$  as

$$v_{\pi^* | \pi, o}(\cdot) = \int_{\mathcal{M}} v(M, \cdot | \pi, o) dM,$$

where the subscript  $\pi^*$  in  $v_{\pi^* | \pi, o}$  is an index notation, whereas  $\pi$  and  $o$  in  $v_{\pi^* | \pi, o}$  are random variables. For two probability measures  $\mathbb{P}$  and  $\mathbb{Q}$  on a measurable space, the **Kullback–Leibler (KL)** divergence between  $\mathbb{P}$  and  $\mathbb{Q}$  is defined by  $\text{KL}(\mathbb{P}, \mathbb{Q}) = \int \log \frac{d\mathbb{P}}{d\mathbb{Q}} d\mathbb{P}$  if the measure  $\mathbb{P}$  is absolutely continuous with respect to the measure  $\mathbb{Q}$ , and  $+\infty$  otherwise.

Now we introduce a central definition in this article—AIR.

*Definition 2.3 (Algorithmic Information Ratio).* Given a reference probability  $q \in \text{int}(\Delta(\Pi))$ <sup>1</sup> and learning rate  $\eta > 0$ , we define the AIR for probability  $p$  of  $\pi$  and belief  $v$  of  $(M, \pi^*)$  as

$$\text{AIR}_{q,\eta}(p, v) = \mathbb{E}_{p,v} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{\pi^*|\pi,o}, q) \right],$$

where the expectation is taken with  $\pi \sim p, (M, \pi^*) \sim v, o \sim M(\pi)$ .

The term “Algorithmic Information Ratio” was used to highlight the key difference between AIR and classical IR measures (to be presented shortly in Equation (1)). Firstly, AIR incorporates a reference probability  $q$  in its definition, while classical IR does not. This additional flexibility makes AIR useful for algorithm design and analysis. Secondly, AIR is defined in an offset form, whereas IR is defined in a ratio form. We choose the word “algorithmic” because AIR is particularly suited to designing constructive and efficient frequentist algorithms; we retain the term “ratio” as it is consistent with previous literature on the topic. The formulation of AIR is directly inspired by recent optimization frameworks, including EBO [33] and DEC [23, 24]. Additionally, a closely related maximin concept (see Equation (4)) has been introduced as the “parametric information ratio” in [24]. There are various equivalences between different variants of AIR, DEC, EBO, and IR when considering minimax algorithms, worst-case environments, and choosing the appropriate index, divergence, and formulation (see the next two subsections for details). However, AIR crucially focuses on “algorithmic belief”  $v$  rather than maximizing with respect to the worst-case deterministic model, providing algorithmic unity, interpretability, and flexibility.

Note that AIR is linear with respect to  $p$  and concave with respect to  $v$ , as conditional entropy is always concave with respect to the joint probability measure (see Lemma 4.1). By the generalized Pythagorean theorem (Lemma 4.7) for the KL divergence and the fact about posterior  $\mathbb{E}_{p,v}[v_{\pi^*|\pi,o}] = v_{\pi^*}$ , we have the equality  $\mathbb{E}_{p,v}[\text{KL}(v_{\pi^*|\pi,o}, q)] = \mathbb{E}_{p,v}[\text{KL}(v_{\pi^*|\pi,o}, v_{\pi^*})] + \text{KL}(v_{\pi^*}, q)$ . Thus it will be illustrative to write AIR as the sum of three items:

$$\begin{aligned} \text{AIR}_{q,\eta}(p, v) &= \underbrace{\mathbb{E}_{\pi \sim p, (M, \pi^*) \sim v} [f_M(\pi^*) - f_M(\pi)]}_{\text{expected regret}} \\ &\quad - \underbrace{\frac{1}{\eta} \mathbb{E}_{\pi \sim p, (M, \pi^*) \sim v, o \sim M(\pi)} [\text{KL}(v_{\pi^*|\pi,o}, v_{\pi^*})]}_{\text{information gain}} - \underbrace{\frac{1}{\eta} \text{KL}(v_{\pi^*}, q)}_{\text{regularization by } q}, \end{aligned}$$

where: the “expected regret” measures the difficulty of exploitation; the “information gain” is the amount of information gained about the marginal distribution of  $\pi^*$  by observing the random variables  $\pi$  and  $o$ , and this in fact measures the degree of exploration; and the last “regularization” term forces the marginal distribution of  $\pi^*$  to be “close” to the reference probability distribution  $q$ . By maximizing AIR, we generate an “algorithmic belief” that simulates the worst-case environment. This algorithmic belief will automatically balance exploration and exploitation, as well as being close to the chosen reference belief (e.g., a standard reference is the posterior from previous round, as used in traditional TS).

### 2.3 Bounding AIR by IR and DEC

Notably, our framework allows for the utilization of essential all existing upper bounds for IR and DEC in practical applications, enabling the derivation of the sharpest regret bounds known, along

<sup>1</sup>AIR can be defined as  $-\infty$  when  $q$  resides on the boundary of  $\Delta(\Pi)$  and  $v_{\pi^*}$  is not absolutely continuous with respect to  $q$ . Such simple extension can provide an alternative solution to address any concerns related to boundary issues that may arise in the article, with a slightly more careful reference to minimax theorem (Lemma 4.2) remaining valid even when the value domain is extended to  $[-\infty, \infty]$ .

with the development of constructive algorithms. In this subsection we demonstrate that AIR can be upper bounded by IR and DEC.

We present here the traditional definition of Bayesian IR [45]. See [26, 34, 35, 44, 45] for upper bounds of IR in bandit problems and structured RL problems.

*Definition 2.4 (Information Ratio).* Given belief  $v$  of  $(M, \pi^*)$  and decision probability  $p$  of  $\pi$ , the IR is defined as

$$\text{IR}(v, p) = \frac{\mathbb{E}_{v,p} [f_M(\pi^*) - f_M(\pi)]^2}{\mathbb{E}_{v,p} [\text{KL}(v_{\pi^*|\pi,o}, v_{\pi^*})]} \quad (1)$$

Note that the traditional IR (Equation (1)) does not involve any reference probability distribution  $q$  (unlike AIR). By completing the square, it is easy to show that AIR can always be bounded by IR as follows.

**LEMMA 2.5 (BOUNDING AIR BY IR).** *For any  $q \in \text{int}(\Delta(\Pi))$ ,  $p \in \Delta(\Pi)$ , belief  $v \in \Delta(\mathcal{M} \times \Pi)$ , and  $\eta > 0$ , we have*

$$\text{AIR}_{q,\eta}(p, v) \leq \frac{\eta}{4} \cdot \text{IR}(v, p).$$

The recent paper [23] introduced the complexity measure DEC, with the aim of unifying bandits and many RL problems of interest.

*Definition 2.6 (Decision-estimation Coefficient).* Given a model class  $\mathcal{M}$ , a nominal model  $\bar{M}$  and  $\eta > 0$ , we define the DEC by

$$\text{DEC}_\eta(\mathcal{M}, \bar{M}) = \inf_{p \in \Delta(\Pi)} \sup_{M \in \mathcal{M}} \mathbb{E}_p \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} D_H^2(M(\pi), \bar{M}(\pi)) \right],$$

where  $\pi_M \in \arg \min_{\Pi} f_M(\pi)$  is the optimal decision induced by model  $M$ , and  $D_H^2(\mathbb{P}, \mathbb{Q}) = \int (\sqrt{d\mathbb{P}} - \sqrt{d\mathbb{Q}})^2$  is the squared Hellinger distance between two probability measures. And we define

$$\text{DEC}_\eta(\mathcal{M}) = \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \text{DEC}_\eta(\mathcal{M}, \bar{M}). \quad (2)$$

DEC offers a unifying perspective on various existing structural conditions in the literature concerning RL. For comprehensive explanations on how DEC relates to and subsumes these structural conditions, such as Bellman rank [28], Witness rank [50], (Bellman-) Eluder dimension [29, 43, 53], bilinear classes [18], and linear function approximation [17, 30, 58], we encourage readers to consult Section 1 and Section 7 in [23] and the references therein. Moreover, a slightly strengthened version of DEC, defined through the KL divergence instead of the squared Hellinger divergence,

$$\text{DEC}_\eta^{\text{KL}}(\mathcal{M}) = \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \inf_{p \in \Delta(\Pi)} \sup_{M \in \mathcal{M}} \mathbb{E}_p \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \bar{M}(\pi)) \right],$$

can be upper bounded by the traditional IR. This result follows from Proposition 9.1 in [23]. Note that the convex hull feasible region for the reference model  $\bar{M}$  in the notation Equation (2) is fundamental because the convex hull also appears in the lower bound, see the discussions in [21] and Section 3.5.1 in [23] for details.

We demonstrate in the following lemma that the worst-case value of AIR, when employing a “maximin” strategy for selecting  $p$ , is equivalent to the KL version of DEC applied to the convex hull of the model class (see below). Consequently, it is bounded by the standard version of DEC using the squared Hellinger distance.

LEMMA 2.7 (EQUIVALENCE OF MAXIMIN AIR AND CONVEXIFIED DEC). *Given model class  $\mathcal{M}$  and  $\eta > 0$ , we have*

$$\sup_{q \in \text{int}(\Delta(\Pi))} \sup_v \inf_p \text{AIR}_{q,\eta}(p, v) = \text{DEC}_\eta^{\text{KL}}(\text{conv}(\mathcal{M})) \leq \text{DEC}_\eta(\text{conv}(\mathcal{M})). \quad (3)$$

To prove Lemma 2.7, we can start by noting that the left-hand side of Equation (3) is equivalent to the “parametric information ratio,” defined as

$$\max_v \min_p \mathbb{E}_{v,p} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{\pi|\pi,o}, v_{\pi^*}) \right], \quad (4)$$

which was introduced in [24]. This equivalence can be shown by using the concavity of AIR in  $q$  to exchange sup over  $q$  and min over  $p$ . Furthermore, the equivalence between Equation (4) and  $\text{DEC}_\eta^{\text{KL}}(\text{conv}(\mathcal{M}))$  has been established by Theorem 3.1 in [24]. Therefore, we obtain a proof of Lemma 2.7.

We conclude that AIR is as tight as the best-known complexity measures in the adversarial setting. In particular, its worst-case (maximin) value is equivalent to the KL-based convexified DEC of [24], the EBO objective in [33], and the “parametric information ratio” (Equation (4)) in [24]. Such tightness is established by Lemma 2.7 and the lower bounds in [24]. However, for RL problems in the stochastic setting, it is often desirable to remove the convex hull on the right-hand side of Equation (3). To this end, we introduce a tighter version of AIR, called MAIR, which is upper bounded by the original version of DEC using model class  $\mathcal{M}$  rather than its convex hull (see Lemma 2.10 in the next subsection), and allows us to apply essentially all existing regret upper bounds using DEC to our framework.

## 2.4 Model-Index AIR (MAIR) and DEC

Recall that decision  $\pi_M \in \arg \min_{\Pi} f_M(\pi)$  is the induced optimal decision of model  $M$ . In the stochastic environment, where  $M_t = M^* \in \mathcal{M}$  for all rounds, the benchmark policy in the definition of regret is the natural oracle  $\pi_{M^*}$ . Unlike the adversarial setting, where algorithmic beliefs are formed over pairs of models and optimal decisions, in the stochastic setting, we only need to search for algorithmic beliefs regarding the underlying model. This distinction allows us to develop a strengthened version of AIR, which we call MAIR, particularly suited for studying RL problems.

*Definition 2.8 (Model-index AIR).* Denote  $\rho \in \text{int}(\Delta(\mathcal{M}))$  be a reference distribution of models, and  $\mu \in \Delta(\mathcal{M})$  be a prior belief of models, we define the MAIR as

$$\text{MAIR}_{\rho,\eta}(p, \mu) = \mathbb{E}_{\mu,p} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(\mu(\cdot|\pi, o), \rho) \right],$$

where  $\mu(\cdot|\pi, o)$  is the Bayesian posterior belief of models induced by the prior belief  $\mu$ , and the expectation is taken over  $M \sim \mu, \pi \sim p$ .

By the data processing inequality (Lemma 4.9), KL divergence between two model distributions will be no smaller than KL divergence between the two induced decision distributions. Thus we have the following Lemma.

LEMMA 2.9 (RELATIONSHIP OF MAIR AND AIR). *When  $q$  is the decision distribution of  $\pi_M$  induced by the model distribution  $\rho$ , and  $v$  is the distribution of  $(M, \pi_M)$  induced by the model distribution  $\mu$ , we have*

$$\text{MAIR}_{\rho,\eta}(p, \mu) \leq \text{AIR}_{q,\eta}(p, v).$$

**ALGORITHM 1:** Maximizing AIR to create algorithmic beliefs

**Require:** algorithm ALG and learning rate  $\eta > 0$ . Initialize  $q_1$  to be the uniform distribution over  $\Pi$ .

1: **for** round  $t = 1, 2, \dots, T$  **do**

2:   Obtain  $p_t$  from ALG. Find a distribution  $v_t$  of  $(M, \pi^*)$  that solves

$$\sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{q_t, \eta}(p_t, v).$$

3:   The algorithm ALG samples decision  $\pi_t \sim p_t$  and observes the feedback  $o_t \sim M_t(\pi_t)$ .

4:   Update  $q_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}}$ .

Lemma 2.7 has shown that the worst-case value of AIR under the “maximin” strategy is smaller than DEC of the convex hull of  $\mathcal{M}$ . Now we demonstrate that the worst-case value of MAIR under a “maximin” strategy is smaller than DEC of the original model class  $\mathcal{M}$ , which does not use the convex hull in its argument. In fact, there is an exact equivalence between the worst-case value of MAIR and the KL version of DEC, as is the case in Lemma 2.7 for AIR.

LEMMA 2.10 (BOUNDING MAIR BY DEC). *Given model class  $\mathcal{M}$  and  $\eta > 0$ , we have*

$$\sup_{\rho \in \text{int}(\Delta(\mathcal{M}))} \sup_{\mu} \inf_p \text{MAIR}_{\rho, \eta}(p, \mu) = \text{DEC}_\eta^{\text{KL}}(\mathcal{M}) \leq \text{DEC}_\eta(\mathcal{M}).$$

We conclude that lemma 2.10 enables our approach to match the tightest known regret upper bounds using DEC, and the tightness of MAIR follows from the lower bounds of DEC in [23]. In Section 7, we discuss how to derive principled algorithms using MAIR and application to RL. In Section 7.1, we also discuss how the pursuit of MAIR comes at the cost of larger estimation complexity, explaining the tradeoff between AIR and MAIR.

### 3 Algorithms

In this section, we focus on the adversarial setting and leverage AIR to analyze regret and design algorithms. All the results are extended to leverage MAIR in the stochastic setting in Section 7. For the comparison between AIR and MAIR, we refer to the ending paragraphs of Section 3.3 and Section 7.1 for detailed discussion.

#### 3.1 A Generic Regret Bound Leveraging AIR

Given an arbitrary admissible algorithm ALG (defined in Section 2.1), we can generate a sequence of *algorithmic beliefs*  $\{v_t\}_{t=1}^T$  and a corresponding sequence of *reference probabilities*  $\{q_t\}_{t=1}^T$  in a sequential manner as shown in Algorithm 1. Maximizing AIR to create algorithmic beliefs is an alternative approach to traditional estimation procedures, as the resulting algorithmic beliefs will simulate the true or worst-case environment. In particular, this approach only stores a single distribution  $q_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}}$  at round  $t$ , which is the Bayesian posterior obtained from belief  $v_t$  and observations  $\pi_t, o_t$ , and it is made to forget all the rest information from the past. We should note that all values in the sequence  $\{q_t\}_{t=1}^T$  will reside within the interior of  $\Delta(\Pi)$ . This is due to the fact that the (negative) AIR is a Legendre function concerning the marginal distribution  $v_{\pi^*}$ .<sup>2</sup>

Based on these algorithmic beliefs, we can provide regret bound for an arbitrary algorithm. Here we assume  $\Pi$  to be finite (but potentially large) for simplicity; this assumption can be relaxed using standard discretization and covering arguments.

<sup>2</sup>We refer to Definition 4.4 and Lemma 4.5 for more details regarding the property that  $(v_t)_{\pi^*} \in \text{int}(\Delta(\Pi))$  for all  $t$ .

**THEOREM 3.1 (GENERIC REGRET BOUND FOR ARBITRARY LEARNING ALGORITHM).** *Given a finite decision space  $\Pi$ , a compact model class  $\mathcal{M}$ , the regret of an arbitrary learning algorithm  $\text{ALG}$  is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right], \quad (5)$$

where  $p_t$  are returned by the learning algorithm  $\text{ALG}$ , and  $v_t$  and  $q_t$  are generated according to Algorithm 1.

Let us provide some intuition to illustrate (through a two-line proof sketch) why the concept of algorithmic belief is natural and tight as a generic regret analysis framework; a more detailed analysis is deferred to Section 5. By the definition of AIR, we have

$$\begin{aligned} \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{\eta, q_t}(p_t, v_t) \right] &= \mathbb{E} \left[ \sum_{t=1}^T \underbrace{\mathbb{E}_{v_t, p_t} [f_M(\pi^*) - f_M(\pi)]}_{\text{regret under } v_t} - \frac{1}{\eta} \sum_{t=1}^T \underbrace{\mathbb{E}_{v_t, p_t} \left[ \log \frac{q_{t+1}(\pi^*)}{q_t(\pi^*)} \right]}_{\text{KL part}} \right] \\ &\stackrel{?}{\geq} \mathbb{E} \left[ \sum_{t=1}^T \underbrace{\mathbb{E}_{p_t} [f_{M_t}(\pi^*) - f_{M_t}(\pi)]}_{\text{actual regret}} - \frac{1}{\eta} \log |\Pi| \right], \end{aligned} \quad (6)$$

where the first equality follows directly from the choice of  $q_{t+1} = (v_t)_{\pi^* | \pi_t, o_t}$  as the posterior in the KL term within AIR. At first glance, the second inequality appears plausible, as  $v_t$  maximizes AIR at every round, and substituting the actual realizations  $(M_t, \pi^*)$  should only decrease this value. However, there is a subtle technical issue: *the posterior distribution  $q_{t+1}$  depends explicitly on the belief  $v_t$  itself*. Thus, rigorously justifying this inequality requires carefully examining whether optimizing  $v_t$  while treating  $q_{t+1}$  as a function of  $v_t$  yields the same result as fixing  $q_{t+1}$  and then optimizing over  $v_t$ . We resolve this subtlety through a duality argument developed in Section 5, which formally ensures that such swapping of optimization steps is valid.

Note that Theorem 3.1 provides a powerful tool to study the regret of an arbitrary algorithm using the concept of AIR. Furthermore, it suggests that the algorithm should choose decision with probability  $p_{t+1}$  according to the posterior  $(v_t)_{\pi^* | \pi_t, o_t}$ . Building on this principle to generate algorithmic beliefs, we provide two concrete algorithms: “**Adaptive Posterior Sampling**” (APS) and “**Adaptive Minimax Sampling**” (AMS). Surprisingly, their regret bounds are as sharp as the best known regret bounds of existing Bayesian algorithms that *require* knowledge of a well-specified prior. An analogy of the theorem leveraging MAIR in the stochastic setting is presented as Theorem 7.1.

### 3.2 Adaptive Posterior Sampling (APS)

When the agent always selects  $p_{t+1}$  to be equal to the posterior  $q_{t+1} = (v_t)_{\pi^* | \pi_t, o_t}$ , and optimizes for algorithmic beliefs as in Algorithm 1, we call the resulting algorithm APS.

At round  $t$ , APS inputs  $p_t$  to the objective  $\text{AIR}_{p_t, \eta}(p_t, v)$  to optimize for the algorithmic belief  $v_t$ ; and it sets  $p_{t+1}$  to be the Bayesian posterior obtained from belief  $v_t$  and observations  $\pi_t, o_t$ . Unlike traditional TS, APS does not require knowing the prior or stochastic environment; instead, APS creates algorithmic beliefs “on the fly” to simulate the worst-case environment. We can prove the following theorem using the regret bound (Equation (5)) in Theorem 3.1, and the upper bounds of AIR by IR and DEC established in Section 2.3.

**ALGORITHM 2:** Adaptive Posterior Sampling (APS)

**Require:** Input learning rate  $\eta > 0$ . Initialize  $p_1 = \text{Unif}(\Pi)$ .

1: **for** round  $t = 1, 2, \dots, T$  **do**

2:   Find a distribution  $v_t$  of  $(M, \pi^*)$  that solves

$$\sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{p_t, \eta}(p_t, v).$$

3:   Sample decision  $\pi_t \sim p_t$  and observe  $o_t \sim M_t(\pi_t)$ .

4:   Update  $p_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}}$ .

**THEOREM 3.2 (REGRET OF APS).** *Assume that  $f_M(\pi) \in [0, 1]$  for all  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . The regret of Algorithm 2 with  $\eta = \sqrt{2 \log |\Pi| / ((\text{IR}_H(\text{TS}) + 4) \cdot T)}$  and all  $T \geq 5 \log |\Pi|$  is bounded by*

$$\mathfrak{R}_T \leq \sqrt{2 \log |\Pi| (\text{IR}_H(\text{TS}) + 4) T},$$

where  $\text{IR}_H(\text{TS}) := \sup_v \text{IR}_H(v, v_{\pi^*})$  is the maximal value of  $\text{IR}^3$  for TS. Moreover, the regret of Algorithm 2 with any  $\eta \in (0, 1/3]$  is bounded as follows, for all  $T \in \mathbb{N}_+$

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + T \cdot (\text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) + 2\eta),$$

where  $\text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) := \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \sup_{\mu \in \Delta(\text{conv}(\mathcal{M}))} \mathbb{E}_{\mu, p^{\text{TS}}} [f_M(\pi_M) - f_M(\pi) - \frac{1}{2\eta} D_H^2(M(\pi), \bar{M}(\pi))]$  is DEC of  $\text{conv}(\mathcal{M})$  for the TS strategy  $p^{\text{TS}}(\pi) = \mu(\{M : \pi_M = \pi\})$ .

For  $K$ -armed bandits, APS achieves the near-optimal regret  $O(\sqrt{KT \log K})$  because  $\text{IR}_H(\text{TS}) \leq 4K$ ; for  $d$ -dimensional linear bandits, APS recovers the optimal regret  $O(\sqrt{d^2 T})$  because  $\text{IR}_H(\text{TS}) \leq 4d$ . See Appendix 1.9 for the proof of these IR bounds.

The main messages about APS and Theorem 3.2 are: (1) the regret bound of APS is no worse than the standard regret bound of TS [45], but in contrast to the latter, does not rely on any knowledge needed to specify a prior! (2) Because APS only keeps the marginal beliefs of  $\pi^*$  but forgets beliefs of the models, it is robust to adversarial and non-stationary environments. And (3) Experimental results in Section 4 show that APS empirically performs extremely well across stochastic, adversarial, and non-stationary environments.

To the best of our knowledge, Theorem 3.2 is the first generic result to make TS prior-free and applicable to adversarial environment. To that end, we note that Corollary 19 in [33] only applies to  $K$ -armed bandits because of their truncation procedure.

### 3.3 Adaptive Minimax Sampling (AMS)

When the agent selects decision  $p_t$  by solving the minimax problem

$$\inf_{p_t} \sup_v \text{AIR}_{q_t, \eta}(p, v),$$

and optimizes for algorithmic beliefs as in Algorithm 1, we call the resulting algorithm AMS. By the regret bound (Equation (5)) in Theorem 3.1 and the upper bounds of AIR by DEC and IR established in Section 2.3, it is straightforward to prove the following theorem.

<sup>3</sup>For technical reason we use the squared Hellinger distance to define  $\text{IR}_H$  (instead of KL as in the definition (Equation (1)) of IR). There is no essential difference between the definitions of IR and  $\text{IR}_H$  in practical applications, since all currently known bounds on the IR hold for the stronger definition  $\text{IR}_H$  with added absolute constants. See Appendix 1.9 for details.

**ALGORITHM 3:** Adaptive Minimax Sampling (AMS-EBO)

**Require:** Input learning rate  $\eta > 0$ . Initialize  $q_1 = \text{Unif}(\Pi)$ .

1: **for** round  $t = 1, 2, \dots, T$  **do**

2: Find a distribution  $p$  of  $\pi$  and a distribution  $v_t$  of  $(M, \pi^*)$  that solves the saddle point of

$$\inf_{p \in \Delta(\Pi)} \sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{q_t, \eta}(p, v).$$

3: Sample decision  $\pi_t \sim p_t$  and observe  $o_t \sim M_t(\pi_t)$ .

4: Update  $q_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}}$ .

**THEOREM 3.3 (REGRET OF AMS).** *For a finite decision space  $\Pi$  and a compact model class  $\mathcal{M}$ , the regret of Algorithm 3 with any  $\eta > 0$  is always bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \text{DEC}_\eta^{\text{KL}}(\text{conv}(\mathcal{M})) \cdot T. \quad (7)$$

In particular, the regret of Algorithm 3 with  $\eta = 2\sqrt{\log |\Pi| / (\text{IR}(\text{IDS}) \cdot T)}$  and all  $T \in \mathbb{N}_+$  is bounded by

$$\mathfrak{R}_T \leq \sqrt{\log |\Pi| \cdot \text{IR}(\text{IDS}) \cdot T},$$

where  $\text{IR}(\text{IDS}) := \sup_v \inf_p \text{IR}(v, p)$  is the maximal IR of **Information-Directed Sampling (IDS)**.

Theorem 3.3 shows that the regret bound of AMS is always no worse than that of IDS [44]. Algorithm 3 is implicitly equivalent with a much simplified implementation of the EBO algorithm from [33], but this implementation runs in computationally tractable spaces (rather than intractable functional spaces) and does not require limit analysis. We propose the term ‘‘AMS’’ with the intention of describing the algorithmic idea broadly, encompassing its various variants, including the important one we will discuss shortly, **Model-index AMS (MAMS)**. Alternatively, the EBO perspective is convenient and essential when one wishes to establish connections with mirror descent and analyze general Bregman divergences, as we do in Appendix 1.7.

However, it is important to note that solving the above minimax optimization exactly can be computationally intractable in general, especially for large decision spaces. This challenge is not unique to AMS – for example, the inner optimization of the related EBO algorithm faces a similar difficulty. In practice, tractable implementations of AMS require either restricting to special cases where the optimization simplifies (e.g., finite-tabular or linear bandits that admit known polynomial-time solutions) or using approximation techniques (such as sampling-based methods) to estimate the optimal distribution. Thus, while AMS provides a conceptually unified ‘‘one-shot’’ formulation, it does not by itself overcome the fundamental computational hardness of planning over a large action space.

In Section 7, we develop a model-index version of AMS in the stochastic setting, which we term MAMS and introduce as Algorithm 6. MAMS leverage MAIR as the optimization objective in the algorithmic belief generation. The regret of MAMS is always bounded by

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_\eta^{\text{KL}}(\mathcal{M}) \cdot T. \quad (8)$$

Compared with the regret bound of AMS in Equation (7), the regret bound Equation (8) of MAMS uses DEC of the original model class  $\mathcal{M}$  rather than its convex hull  $\text{conv}(\mathcal{M})$ ; on the other hand, the estimation complexity term in Equation (8) is  $\log |\mathcal{M}|/\eta$ , which is larger than the  $\log |\Pi|/\eta$  term in Equation (7). Furthermore, MAMS is only applicable to the stochastic environment, while AMS is applicable to the adversarial environment. We refer to Section 7.1 for a more detailed comparison between AIR and MAIR.

### 3.4 Using Approximate Maximizers

In Algorithm 1, we ask for the algorithmic beliefs to maximize AIR. In order to give computationally efficient algorithms in practical applications (MAB, linear bandits, RL, ...), we will require the algorithmic beliefs to approximately maximize AIR. This argument is made rigorous in the following theorem, which uses the first-order optimization error of AIR to represent the regret bound.

**THEOREM 3.4 (GENERIC REGRET BOUND USING APPROXIMATE MAXIMIZERS).** *Let  $\Pi$  be a finite decision space and  $\mathcal{M}$  a compact model class. Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and maintain a sequence of beliefs  $v_1, \dots, v_T$  where  $q_{t+1} := (v_t)_{\pi^*|_{\pi_t, o_t}} \in \text{int}(\Delta(\Pi))$  for all rounds. Then for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{AIR}_{q_t, \eta}(p_t, v_t) + \sup_{v^*} \left( \left. \frac{\partial \text{AIR}_{q_t, \eta}(p_t, v)}{\partial v} \right|_{v=v_t}, v^* - v_t \right) \right) \right].$$

Thus we give a concrete approach toward systematical algorithms with rigorous guarantees—by minimizing the gradient of AIR, we aim to approximately maximize AIR. This is an important factor upon which we advance the existing literature in EBO and DEC. We not only provide regret guarantees for “minimax algorithm” and “worst-case environment,” but also offer a comprehensive path for designing algorithms by leveraging the key idea of “algorithmic beliefs.” An analogy of this theorem leveraging MAIR in the stochastic setting is presented as Theorem 7.1 in Section 7.

### 3.5 Extensions to Dynamic Regret

This subsection presents a theoretical guarantee that our algorithmic-belief optimization method automatically adapts to certain non-stationary environments, particularly when the optimal decision changes only a finite number of times. We consider a stronger notion of regret called dynamic regret [6, 10], which compares the cumulative reward of an algorithm to the cumulative reward of the best sequence of decisions (rather than a fixed single decision) in hindsight. Formally, we define:

$$\mathfrak{R}_T^D (\{\pi_t^*\}_{t=1}^T) = \mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\pi_t^*) - \sum_{t=1}^T f_{M_t}(\pi_t) \right], \quad (9)$$

where  $\{\pi_t^*\}_{t=1}^T$  is a sequence of comparator decisions, chosen after round  $T$ .

We extend Theorem 3.1 to non-stationary settings and dynamic regret. The only modification is the reference probabilities  $\{q_t\}_{t=1}^T$ : instead of the exact posteriors used in Algorithm 1, we adopt the forced-exploration mixture  $q_{t+1} := (1 - \gamma)(v_t)_{\pi^*|_{\pi_t, o_t}} + \gamma \text{Unif}(\Pi)$  where  $\gamma \in (0, 1)$  is the exploration rate.

**THEOREM 3.5 (GENERIC DYNAMIC REGRET BOUND).** *Given a finite decision space  $\Pi$  and a compact model class  $\mathcal{M}$ . Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and  $\gamma \in (0, 1)$ , generate a sequence of beliefs  $v_1, \dots, v_t, \dots$  where  $v_t$  solves  $\sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{q_t, \eta}(p_t, v)$ , and set  $q_{t+1} := (1 - \gamma)(v_t)_{\pi^*|_{\pi_t, o_t}} + \gamma \text{Unif}(\Pi)$ . Then for all  $T \in \mathbb{N}_+$ , the dynamic regret of ALG is bounded by*

$$\mathfrak{R}_T^D (\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi| + \frac{T\gamma}{1-\gamma} + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right] \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right], \quad (10)$$

where  $\sum_{t=2}^T \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)}$  measures the total shift in the comparator sequence  $\{\pi_t^*\}_{t=1}^T$  up to round  $T$ . Specifically, denote

$$D_T = \#\{t : \pi_t^* \neq \pi_{t+1}^*, 1 \leq t < T\}$$

as the number of changes in the optimal decision before round  $T$ , and set  $\gamma = 1/T$ , then the dynamic regret is bounded by

$$\mathfrak{R}_T^D(\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi| (D_T \log T + 1) + \frac{T}{T-1} \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right]. \quad (11)$$

The concise proof is presented in Appendix 1.8, closely following the two-line proof sketch Equation (6). The primary modification involves recognizing that the cumulative sum of the log-ratio term  $\log \frac{q_{t+1}(\pi_t^*)}{q_t(\pi_t^*)}$  telescopes into the shift term appearing in Theorem 3.5. This naturally reveals the connection between AIR—particularly the use of optimized algorithmic beliefs and sequential posterior updates—and dynamic regret, providing some validation for our algorithms’ robust convergent performance across different regimes. A novelty of this theorem is its applicability to very general interactive decision-making problems while explicitly incorporating dynamic regret.

It is easy to derive Equation (11) from Equation (10) because the shift term  $\sum_{t=2}^T \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)}$  in Equation (10) can be trivially upper-bounded by

$$\sum_{t=2}^T \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \leq D_T \cdot \log \frac{|\Pi|}{\gamma},$$

where  $D_T = \#\{t : \pi_t^* \neq \pi_{t+1}^*, 1 \leq t < T\}$ , which denotes the number of changes in the optimal decision before round  $T$ , is a commonly studied practical measure of non-stationarity since [6]. Exploring the applicability of our method to more general non-stationary settings is an intriguing direction that we defer to future work.

## 4 Application to Bernoulli MAB

Our Bayesian design principles give rise to a novel algorithm for the Bernoulli MAB problem. It is well-known that every bounded-reward MAB problem can equivalently be reduced to the Bernoulli MAB problem, so our algorithm and experimental results actually apply to all bounded-reward MAB problems. The reduction is very simple: assuming the rewards are always bounded in  $[a, b]$ , then after receiving  $r_t(\pi_t)$  at each round, the agent re-samples a binary reward  $\tilde{r}_t(\pi_t) \sim \text{Bern}((r_t(\pi_t) - a)/b - a)$  so that  $\tilde{r}_t(\pi_t) \in \{0, 1\}$ .

### 4.1 Simplified APS for Bernoulli MAB

In Example 2.1,  $\Pi = [K] = \{1, \dots, K\}$  is a set of  $K$  actions, and each model  $\mathcal{M}$  is a mapping from actions to Bernoulli distributions. Given belief  $v \in \Delta(\mathcal{M} \times [K])$ , we introduce the following parameterization:  $\forall i, j \in [K]$ ,

$$\theta_i(j) := \mathbb{E}[r(j)|\pi^* = i], \quad (\text{conditional mean reward})$$

$$\alpha(i) := v_{\pi^*}(i), \quad (\text{marginal belief})$$

$$\beta_i(j) := \alpha(i) \cdot \theta_i(j). \quad (\text{guarantees concavity})$$

Then we have a concave parameterization of AIR by the  $K(K+1)$ -dimensional vector  $(\alpha, \beta) = (\alpha, \beta_1, \dots, \beta_K)$ :

$$\begin{aligned} \text{AIR}_{q, \eta}(p, v) &= \sum_{i \in [K]} \beta_i(i) - \sum_{i, j \in [K]} p(j) \beta_i(j) \\ &\quad - \frac{1}{\eta} \sum_{i, j \in [K]} p(j) \alpha(i) \text{kl} \left( \frac{\beta_i(j)}{\alpha(i)}, \sum_{i \in [K]} \beta_i(j) \right) - \frac{1}{\eta} \text{KL}(\alpha, q), \end{aligned}$$

**ALGORITHM 4:** Simplified APS for Bernoulli MAB

**Require:** Input learning rate  $\eta > 0$ . Initialize  $p_1 = \text{Unif}(\Pi)$ .

- 1: **for** round  $t = 1, 2, \dots, T$  **do**
- 2:   Sample action  $\pi_t \sim p_t$  and receives  $r_t(\pi_t)$ .
- 3:   Update  $p_{t+1}$  by

$$p_{t+1}(\pi_t) = \begin{cases} \frac{1-\exp(-\eta)}{1-\exp(-\eta/p_t(\pi_t))}, & \text{if } r_t(\pi_t) = 1 \\ \frac{1-\exp(\eta)}{1-\exp(\eta/p_t(\pi_t))}, & \text{if } r_t(\pi_t) = 0 \end{cases}, \text{ and}$$

$$p_{t+1}(\pi) = p_t(\pi) \cdot \frac{1 - p_{t+1}(\pi_t)}{1 - p_t(\pi_t)}, \quad \forall \pi \neq \pi_t.$$

where  $\text{kl}(x, y) := x \log \frac{x}{y} + (1-x) \log \frac{1-x}{1-y}$  for all  $x, y \in (0, 1)$ . By setting the gradients of AIR with respect to all  $K^2$  coordinates in  $\beta$  to be exactly zero, and choosing  $\alpha = p$  (which results in the gradient of AIR with respect to  $\alpha$  being suitably bounded), we are able to write down a simplified APS algorithm in closed form (see Algorithm 4). We apply Theorem 3.4 to show that the algorithm achieves optimal  $O(\sqrt{KT \log K})$  regret in the general adversarial setting.

**THEOREM 4.1 (REGRET OF SIMPLIFIED APS FOR BERNOUlli MAB).** *The regret of Algorithm 4 with  $\eta = \sqrt{\log K / (2KT + 4T)}$  and  $T \geq 3$  is bounded by*

$$\mathfrak{R}_T \leq 2\sqrt{2(K+2)T \log K}.$$

We leave the detailed derivation and analysis of the Algorithm 4 to Appendix 2.2.

At each round, Algorithm 4 increases the weight of the selected action  $\pi_t$  if  $r_t(\pi_t) = 1$ , and decreases the weight if  $r_t(\pi_t) = 0$ . The algorithm also maintains the “relative weight” between all unchosen actions  $\pi \neq \pi_t$ , allocating probabilities to these actions proportionally to  $p_t$ . Algorithm 4 is clearly very different from the well-known EXP3 algorithm [7], which samples action by the forced-exploration mixture ( $\gamma \in (0, 1)$  is the exploration rate)

$$\pi_t \sim \tilde{p}_t := (1 - \gamma)p_t + \gamma \text{Unif}([K]),$$

and updates  $p_{t+1}$  by the exponential-weight formula

$$p_{t+1}(\pi) = p_t(\pi) \exp\left(\eta \cdot \frac{r_t(\pi_t) \mathbb{1}\{\pi = \pi_t\}}{p_t(\pi_t)}\right), \quad \forall \pi \in [K].$$

In Section 6.2, we recover a modified version of EXP3 by Bayesian principle assuming the reward distribution is exponential or Gaussian. We conclude that Algorithm 2 uses a precise posterior for Bernoulli reward, while EXP3 estimates worst-case exponential or Gaussian reward. This may explain why Algorithm 4 performs much better in all of our experiments.

*Reflection symmetry and mirror map:* We would like to highlight a notable property of our new algorithm: Algorithm 4 exhibits *reflection symmetry* (or *reflection invariance*), signifying that when we interchange the roles of reward and loss, the algorithm remains unchanged. The theoretical convergence of our algorithm is not contingent on any truncation. On the contrary, the traditional IPW estimator does not exhibit this property: the reward-based IPW and the loss-based IPW yield distinct algorithms. While the expected regret convergence of EXP3 with the loss-based estimator does not necessitate truncation, EXP3 with the reward-based estimator mandates forced uniform exploration for convergence. (The original EXP3 algorithm from [7] is reward-based; we refer to Section 11.4 in [35] for loss-based EXP3.)

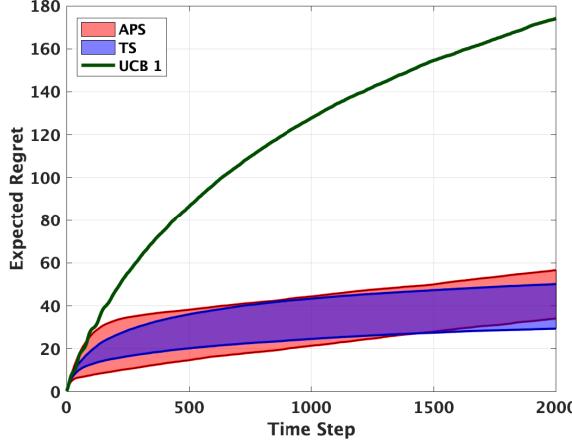


Fig. 1. A stochastic bandit problem. APS outperforms UCB and rival TS (Section 4.2.1).

We would also like to comment on the relationship between Algorithm 4 and **online mirror descent (OMD)**. Algorithm 4 is equivalent to running mirror descent (specifically exponential weight here) using the reward estimator:

$$\begin{aligned}\hat{r}_t(\pi) &= \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t}, r_t(\pi_t)}(\pi)}{p_t(\pi)} \\ &= \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t}, 0}(\pi)}{p_t(\pi)} + \frac{\mathbb{1}(\pi = \pi_t)r_t(\pi_t)}{p_t(\pi)} - r_t(\pi_t).\end{aligned}\quad (12)$$

From this perspective, Algorithm 4 can be understood as addressing a long-standing question: how to fix the issue of exploding variance in the IPW estimator and automatically find the optimal bias-variance tradeoff. It is worth noting that the derivation of our algorithms reveals a deep connection between frequentist estimators and the natural parameters of exponential family distributions (see Appendix 2.6 for details). We hope this framework can be applied to challenging problems where a more robust estimator than IPW is required, such as bandit convex optimization [13] and efficient linear bandits [1].

## 4.2 Numerical Experiments

We implement Algorithm 4 (with the legend ‘‘APS’’ in the figures) and compare it with other algorithms across the stochastic, adversarial, and non-stationary environments (code available [here](#)). We plot expected regret (average of 100 runs) for different choices of  $\eta$ , and set a forced exploration rate  $\gamma = 0.001$  in all experiments. We find APS (1) outperforms UCB and matches TS in the stochastic environment; (2) outperforms EXP3 in the adversarial environment; and (3) outperforms EXP3 and is comparable to the ‘‘clairvoyant’’ benchmarks (that have prior knowledge of the changes) in the non-stationary environment. For this reason we say Algorithm 4 (APS) achieves the ‘‘best-of-all-worlds’’ performance. We note that the optimized choice of  $\eta$  in APS differ instance by instance, but by an initial tuning we typically see good results, whether we tune  $\eta$  optimally or not optimally.

**4.2.1 Stochastic Bernoulli MAB.** In Figure 1 we report the expected regret for APS with different choices of  $\eta$ , TS with different Beta priors, and the UCB 1 algorithm, in a stochastic 16-armed



Fig. 2. Jackson Pollock, *Mural* (1943) [Oil on canvas]. University of Iowa Museum of Art (Section 4.2.2).

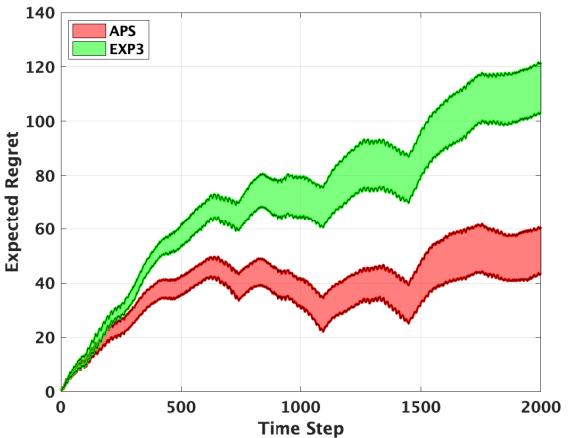


Fig. 3. An adversarial bandit problem generated from the art work. APS outperforms EXP3 (Section 4.2.2).

Bernoulli bandit problem. We refer to this as “sensitivity analysis” because the red, semi-transparent, area reports the regret of APS when learning rates  $\eta$  are chosen across a range of values drawn from the interval  $[0.05, 0.5]$  (the interval is specified by an initial tuning); and the priors of TS are chosen from  $\text{Beta}(c, 1)$  where  $c \in [0.5, 5]$ . In particular, the bottom curve of the red (or blue) area is the regret curve of APS (or TS) using optimally tuned  $\eta$  (respectively, prior); The shaded region is formed by the regret curves of the best-tuned parameter (lower boundary) and worst-tuned parameter (upper boundary). The conclusion is that APS outperforms UCB 1, and is comparable to TS in this stochastic environment.

Intuitively, APS outperforms UCB because it leverages posterior-like distributions as optimal randomized estimators, whereas UCB relies on less informative point estimates; although formal proofs remain challenging, this Bayesian–frequentist contrast explains APS’s empirical edge.

**4.2.2 Adversarial Bernoulli MAB.** We equidistantly take 16 horizontal lines from an abstract art piece by Jackson Pollock to simulate the rewards (pre-specified) in an adversarial environment, and study this via a 16-armed bandit problem. To be more specific, we transform these 16 horizontal lines into grayscale values, resulting in each line becoming a sequence of 2000 values in the range  $[0, 1]$ . As depicted in Figure 2, it’s evident that the environment is inherently adversarial and lacks any statistical regularity. This is further confirmed by empirical findings that the best-in-hindsight arm changes over time. Figure 3 shows the sensitivity analysis for APS and EXP3 when both the learning rates are chosen from  $[0.1, 5]$  (the interval is specified by an initial tuning). In particular, the red and green lower curves compare the optimally tuned versions of APS and EXP3. The conclusion is that APS outperforms EXP3 whether  $\eta$  is tuned optimally or not.

**4.2.3 Non-Stationary Bernoulli MAB (with Change Points).** We consider the stronger notion of dynamic regret defined in Equation (9), which compares the cumulative reward of an algorithm to the cumulative reward of the best non-stationary policy (rather than a single arm) in hindsight. In the MAB setting, the benchmark is to select the best arm in every rounds. We present two compelling experiments: one comparing APS against algorithms specifically designed for change-point environments, and another comparing APS with “clairvoyant” algorithms that have prior knowledge of environment “restarts.”

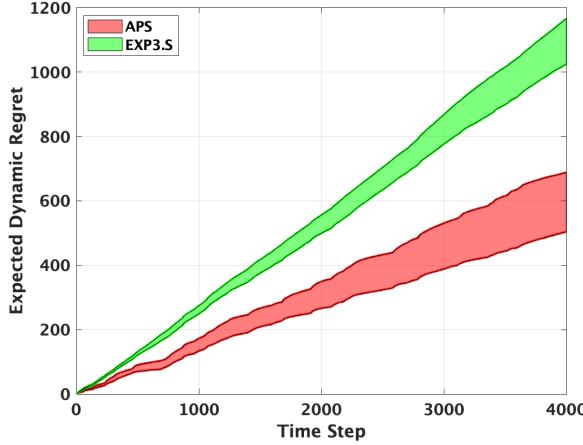


Fig. 4. A non-stationary bandit problem with many random change points, evaluated by dynamic regret. APS outperforms EXP3.S, a benchmark algorithm specially designed for change-point environments. (The first experiment in Section 4.2.3)

In Figure 4, we study a 16-armed Bernoulli bandit problem in a non-stationary environment. We generate 40 batches of i.i.d. sequences, where the change points are random across rounds 1 to 4000. To benchmark against prior work, we compare APS with EXP3.S [7], a notable variant of EXP3 specifically designed for change-point environments. We perform sensitivity analysis for APS and EXP3.S, where the learning rates are chosen across [1, 20]. Since the agent will not know when and how the adversarial environment changes in general, it is most reasonable to compare APS with EXP3.S without any knowledge of the environment as in Figure 4. We observe that APS dramatically improves the dynamic regret by several times.

In Figure 5, we compare APS to three “clairvoyant” restarted algorithms, which require knowing that the environment consists of 4 batches of i.i.d. sequences, as well as knowing the exact change points. We tune the parameters in these algorithms optimally. Without knowledge of the environment, APS performs better than restarted EXP3 and restarted UCB 1, and is comparable to restarted TS. (It is important to emphasize again that the latter algorithms are restarted based on foreknowledge of the change points.)

As an immediate corollary of Theorem 3.5, we show that a mildly modified version of Algorithm 4 achieves a regret bound

$$\tilde{O}(\sqrt{KT(1 + D_T)}), \quad D_T = \#\{t : \pi_t^* \neq \pi_{t+1}^*, 1 \leq t < T\}, \quad (13)$$

where  $D_T$  is the number of change in optimal decision before round  $T$ , assuming  $D_T$  is known in advance (the algorithm need not know the locations of the change points; note that removing any prior knowledge of the total change budget  $D_T$  is information-theoretically impossible [39]). See Appendix 2.7 for details. This result confirms that our method automatically adapts to non-stationary environments with a bounded number of optimal-decision shifts. Although this dynamic-regret bound guarantees eventual convergence, it does not by itself explain why APS empirically reacts much faster than EXP3.S in Figure 4, since EXP3.S has a comparable bound.

**4.2.4 Non-Stationary Bernoulli MAB (with “Sine Curve” Reward Sequences).** We generate a 4-armed bandit problem with the mean-reward structure shown in Figure 6. The four sine curves (with different colors) in Figure 6 represent the mean reward sequences of the 4 arms. We tune the

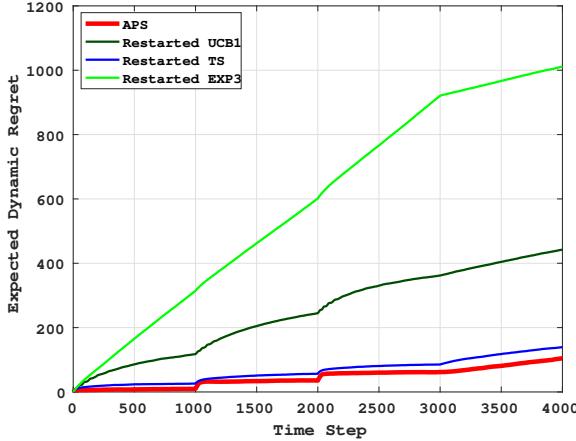


Fig. 5. Another strong benchmark that compares APS to “clairvoyant” restarted algorithms in an environment with few change points, evaluated by dynamic regret. APS (without knowing restart points) outperforms “clairvoyant” restarted EXP3 and UCB, and rivals “clairvoyant” restarted TS. (The second experiment in Section 4.2.3)

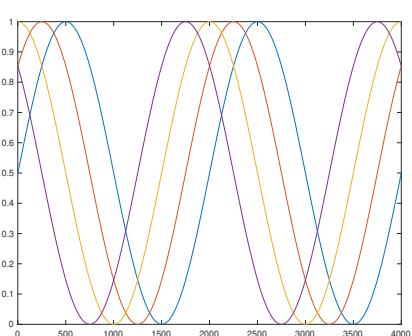


Fig. 6. “Sine curve” reward sequences for 4 arms. (Section 4.2.4)

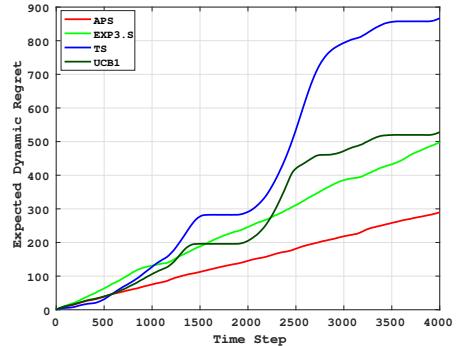


Fig. 7. Dynamic regret curves in the “sine curve” environment. APS performs the best, while TS performs the worst. (Section 4.2.4)

parameters in all the algorithms to optimal and report their regret curves in Figure 7. As shown in Figure 7, APS achieves the best performance, while TS fails in this non-stationary environment. This experiment shows the vulnerability of TS if the environment is not stationary, such as the sine curve structure shown here.

To better illustrate the smartness of APS compared with TS in the non-stationary environment, we track the selected arms and the best arms throughout the process. In Figures 8 and 9, the horizontal line represents the 4000 rounds, and the vertical lines represent the 4 arms (indexed as 1, 2, 3, and 4). In Figure 8, the red points show the selected arms of APS, and the black points represent the best arms at each round in this “sine curve” non-stationary environment. In Figure 9, the blue points show the selected arms of TS. The more consistent the selected arms are with the best arms (black points), the better choices an algorithm makes. Comparing Figures 8 and 9, we can see

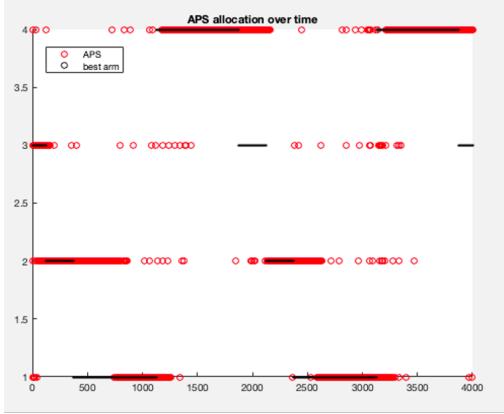


Fig. 8. Tracking selected arms of APS in the “sine curve” environment. APS is fast to respond to the change of optimal arms. (Section 4.2.4)

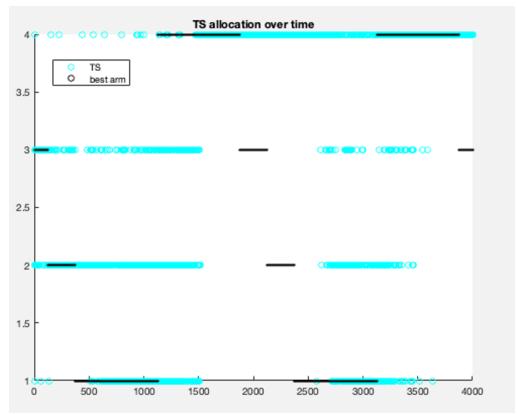


Fig. 9. Tracking selected arms of TS in a “sine curve” environment. TS is sluggish to the changes to the optimal arms. (Section 4.2.4)

that APS is highly responsive to changes in the best arm, whereas TS is relatively sluggish in this regard. The implication of this experiment is that creating a new algorithmic belief at each round has the potential to significantly improve performance and be a game changer in many problem settings.

We conclude that these experiments provide some numerical evidence indicating that APS achieves the “best-of-all-worlds” across stochastic, adversarial, and non-stationary environments.

## 5 Key Intuition and Underlying Theory

It is worth noting that the proofs of Theorem 3.1 and its generalizations are quite insightful and parsimonious. The two major steps in the proofs may be interesting on its own. The first step is a succinct analysis to bound the cumulative regret by sum of AIR (see Section 5.1); and the second step is to extend the classical minimax theory of “exchanging values” into a constructive approach to design minimax decisions (estimators, algorithms, etc.), which will be presented in Section 5.2. At the end, we also illustrate how such analysis generalizes to approximate algorithmic beliefs, as presented in Theorems 3.4 and 7.1.

### 5.1 Bounding Regret by Sum of AIR

For every  $\bar{\pi} \in \Pi$ , we have

$$\sum_{t=1}^T \left[ \log \frac{q_{t+1}(\bar{\pi})}{q_t(\bar{\pi})} \right] = \log \frac{q_{T+1}(\bar{\pi})}{q_1(\bar{\pi})} \leq \log |\Pi|. \quad (14)$$

By subtracting the additive elements on the left-hand side of Equation (14) (divided by  $\eta$ ) from the per-round regrets against  $\bar{\pi}$ , we obtain

$$\begin{aligned} & \mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\bar{\pi}) - \sum_{t=1}^T f_{M_t}(\pi_t) - \frac{1}{\eta} \cdot \sum_{t=1}^T \log \frac{q_{t+1}(\bar{\pi})}{q_t(\bar{\pi})} \right] \\ &= \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ f_{M_t}(\bar{\pi}) - f_{M_t}(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|\pi_t, o_t}(\bar{\pi})}{q_t(\bar{\pi})} \right] \right] \end{aligned}$$

$$\begin{aligned}
& \stackrel{\diamond}{\leq} \mathbb{E} \left[ \sum_{t=1}^T \sup_{M, \bar{\pi}} \mathbb{E}_{t-1} \left[ f_M(\bar{\pi}) - f_M(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*| \pi_t, o_t}(\bar{\pi})}{q_t(\bar{\pi})} \right] \right] \\
& \stackrel{(*)}{=} \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right], \tag{15}
\end{aligned}$$

where the first equality is by taking  $q_{t+1} = (v_t)_{\pi^*| \pi_t, o_t}$  as in Algorithm 1, and the conditional expectation notation is introduced in Section 2.1; the inequality  $(\diamond)$  is by taking supremum at each rounds; and the last equality  $(*)$  in Equation (15) is by Lemma 5.2, an important identity to be explained in Section 5.2, which is derived from the fact that the pair of maximizer  $v_t$  and posterior functional is a Nash equilibrium of a convex-concave function. We note that the second-to-last formula  $(\diamond)$  is true without imposing any restrictions on  $v_t$ ; only the last formula  $(*)$  relies on the fact that  $v_t$  maximizes AIR.

Combining Equations (15) and (14), we obtain the following inequality for every  $\bar{\pi} \in \Pi$ :

$$\mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\bar{\pi}) - \sum_{t=1}^T f_{M_t}(\pi_t) \right] - \frac{\log |\Pi|}{\eta} \leq \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right].$$

By taking the supremum over  $\bar{\pi} \in \Pi$  on the left-hand side of the inequality, we are able to prove Theorem 3.1:

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right].$$

## 5.2 Duality Theory: From Value to Construction

Consider a decision space  $\mathcal{X}$ , a space  $\mathcal{Y}$  of the adversary's outcome, and a convex-concave function  $\psi(x, y)$  defined in  $\mathcal{X} \times \mathcal{Y}$ . The classical minimax theorem [48] says that, under regularity conditions, the minimax and maximin values of  $\psi(x, y)$  are equal:

$$\min_{\mathcal{X}} \max_{\mathcal{Y}} \psi(x, y) = \max_{\mathcal{Y}} \min_{\mathcal{X}} \psi(x, y).$$

We refer to  $\arg \min_{\mathcal{X}} \max_{\mathcal{Y}} \psi(x, y)$  as the set of "minimax decisions," as they are optimal in the worst-case scenario. And we say  $\tilde{x} \in \arg \min_{\mathcal{X}} \psi(x, \bar{y})$  is "maximin decision" if  $\bar{y} \in \arg \max_{\mathcal{Y}} \min_{\mathcal{X}} \psi(x, y)$  is "maximin adversary's outcome." One natural and important question is, *when will the "maximin decision"  $\tilde{x}$  also be a "minimax decision?"* Making use of strong convexity, we extends the classical minimax theorem for values into the following theorem for constructive minimax solutions:

**LEMMA 5.1 (A CONSTRUCTIVE MINIMAX THEOREM FOR MINIMAX SOLUTIONS).** *Let  $\mathcal{X}$  and  $\mathcal{Y}$  be convex and compact sets, and  $\psi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  a function which for all  $y$  is strongly convex and continuous in  $x$  and for all  $x$  is concave and continuous in  $y$ . For each  $y \in \mathcal{Y}$ , let  $x_y = \min_{x \in \mathcal{X}} \psi(x, y)$  be the corresponding unique minimizer. Then, by maximizing the concave objective*

$$\bar{y} \in \arg \max_{y \in \mathcal{Y}} \psi(x_{\bar{y}}, y),$$

*we can establish that  $x_{\bar{y}}$  is a minimax solution of  $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y)$  and  $(x_{\bar{y}}, \bar{y})$  is a Nash equilibrium.*

The above lemma, although straightforward to prove through the classical minimax theorem, is conceptually interesting because it emphasizes the significance of strong convexity (often achieved

through regularization). This strong convexity allows us to obtain “minimax solutions” by considering a “worst-case adversary” scenario, such as the least favorable Bayesian prior. This approach simplifies the construction of minimax estimators in classical decision theory, moving away from the need to consider “least favorable prior sequence” and specific losses by imposing regularization.

Applying Lemma 5.1 to our framework, we can show: (1) Bayesian posterior  $v_{\pi^*|\pi,o}$  is the optimal functional to make decision under belief  $v$ ; and (2) by choosing worst-case belief  $\bar{v}$ , we construct a Nash equilibrium. As a result, we can establish the following per-round identity, which leads to the key identity denoted as  $(*)$  in Equation (15). This identity plays a crucial role in the proof of Theorem 3.1.

**LEMMA 5.2 (IDENTITY BY NASH EQUILIBRIUM).** *Given  $q \in \text{int}(\Delta(\Pi))$ ,  $\eta > 0$  and  $p \in \Delta(\Pi)$ , denote  $\bar{v}$  as the unique maximizer of  $\text{AIR}_{q,\eta}(p, \bar{v})$ . Then we have*

$$\sup_{M, \bar{\pi}} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ f_M(\bar{\pi}) - f_M(\pi) - \frac{1}{\eta} \log \frac{\bar{v}_{\pi^*|\pi,o}(\bar{\pi})}{q(\bar{\pi})} \right] = \text{AIR}_{q,\eta}(p, \bar{v}).$$

The proof intuition of the above lemma is straightforward: the left-hand side represents the scenario in which we treat  $\bar{v}_{\pi^*|\pi,o}$  as a fixed probability distribution and then optimize over  $(M, \bar{\pi})$ . In contrast, the right-hand side corresponds to optimizing over all distributions  $v$  of  $(M, \bar{\pi})$ , viewing  $v_{\pi^*|\pi,o}$  explicitly as a function of  $v$  rather than fixed. The validity of interchanging these two optimization orders is rigorously established by our constructive minimax theorem (Lemma 5.1), which fully justifies the final step of the concise proof sketch presented under Theorem 3.1.

Furthermore, we establish the following lemma, demonstrating that the role of AIR and its derivatives is generic for any belief selected by the agent and any environment specified by nature. Recall that the second-to-last formula  $(\diamond)$  in Equation (15) holds without imposing any restrictions on  $v_t$ . Thus, Lemma 5.3 leads us to Theorem 3.4 and Theorem 7.1. This fact is proven through Danskin’s theorem [8] (Lemma 4.3), which establishes equivalence between the gradient of the optimized objective and the partial derivative at the optimizer. The “identity” nature of Lemma 5.3 indicates that regret analysis using derivatives of AIR can be principled, precise, and reliable for arbitrary algorithmic beliefs (in addition to maximizers) and any type of environment.

**LEMMA 5.3 (IDENTITY FOR ARBITRARY ALGORITHMIC BELIEF AND ANY ENVIRONMENT).** *Given  $q \in \text{int}(\Delta(\Pi))$ ,  $\eta > 0$   $p \in \Delta(\Pi)$ , and an arbitrary algorithmic belief  $\hat{v} \in \Delta(\mathcal{M} \times \Pi)$  selected by the agent and an arbitrary environment  $(M, \bar{\pi})$  specified by the nature. Then we have*

$$\mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ f_M(\bar{\pi}) - f_M(\pi) - \frac{1}{\eta} \log \frac{\hat{v}_{\pi^*|\pi,o}(\bar{\pi})}{q(\bar{\pi})} \right] = \text{AIR}_{q,\eta}(p, \hat{v}) + \left\langle \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial v} \Big|_{v=\hat{v}}, \mathbb{1}(M, \bar{\pi}) - \hat{v} \right\rangle,$$

where  $\mathbb{1}(M, \bar{\pi})$  is a vector in simplex  $\Delta(\mathcal{M} \times \Pi)$  where all the weights are given to  $(M, \bar{\pi})$ .

In addition to their utility in upper bound analyses, we hope that the duality methodology presented in this section may also offer values for lower bound analyses. Furthermore, the derivation of our bandit and RL algorithms unveils a rich mathematical structure, which we will demonstrate in Appendices 2 and 3 for more details.

## 6 Applications to Infinite-Armed Bandits

Our design principles can be applied in many sequential learning and decision making environments. In order to maximize AIR in practical applications, we parameterize the belief  $v$ , and make the gradient of AIR with respect to such parameter small. We will present our results for linear bandits and bandit convex optimization in this section and present our results for RL in Section 7. We give a high-level overview of the applications to linear bandits and bandit convex optimization here.

*Application to linear bandits.* A classical algorithm for adversarial linear bandits (described in Example 2.2) is the EXP2 algorithm [16], which uses IPW for linear reward as a black-box estimation method, and combines it with continuous exponential weight. We derive a modified version of EXP2 from our framework, establishing interesting connection between IPW and Bayesian posteriors.

*Application to bandit convex optimization.* Bandit convex optimization (described in Example 2.2) is a notoriously challenging problem, and much effort has been put to understanding its minimax regret and algorithm design [12, 13, 32]. The best known result, which is of order  $\tilde{O}(d^{2.5}\sqrt{T})$ , is derived through the non-constructive IR analysis in [32]. As a corollary of Theorem 3.3, AMS recovers the best known regret bound with a simple constructive algorithm, which can be computed in  $\text{poly}(e^d \cdot T)$  time. To the best of our knowledge, this is the first finite-running-time algorithm that attains the best known  $\tilde{O}(d^{2.5}\sqrt{T})$  regret.

## 6.1 Maximization of AIR for Structured Bandits

Consider the structured bandit problems described in Example 2.2. We consider the computation complexity of the optimization problem

$$\sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{q,\eta}(p, v). \quad (16)$$

The computational complexity of Equation (16) may be  $O(\text{poly}(\exp(\exp(d))))$  in the worst case as the size of  $\mathcal{M} \times \Pi$ . However, when the mean reward function class  $\mathcal{F}$  is a convex function class, the computational complexity will be  $O(\text{poly}(|\Pi|))$  which is efficient for  $K$ -armed bandits and is no more than  $O(\text{poly}(e^d))$  in general (by standard discretization and covering arguments, we may assume  $\Pi \subset \mathbb{R}^d$  to have finite cardinality  $O(e^d)$  for the simplicity of theoretical analysis). Moreover, we also give efficient algorithm for linear bandits with exponential-many actions. We refer to Appendix 2.1 for the detailed discussion on the parameterization method and computational complexity.

## 6.2 Application to Gaussian Linear Bandits

We consider the adversarial linear bandit problem with Gaussian reward. In such a problem,  $\Pi = \mathcal{A} \subseteq \mathbb{R}^d$  is a convex action set with dimension  $d$ . The model class  $\mathcal{M}$  can be parameterized by a  $d$ -dimensional vector  $\theta \in \mathbb{R}^d$  that satisfies  $\theta^\top a \in [-1, 1]$  for all  $a \in \mathcal{A}$ . Here we use the notations  $\mathcal{A}$  (as action set),  $a$  (as action), and  $a^*$  (as optimal action) to follow the tradition of literature about linear bandits. The reward  $r(a)$  for each action  $a \in \mathcal{A}$  is drawn from a Gaussian distribution that has mean  $\theta^\top a$  and variance 1. To facilitate the handling of mixture distributions, we introduce the “homogeneous noise” assumption as follows: for all actions  $a \in \mathcal{A}$ , the reward of  $a$  is denoted as  $r(a) = \theta^\top a + \epsilon$ , where  $\epsilon \sim N(0, 1)$  is Gaussian noise that is identical across all actions (meaning all actions share the same randomness within the same rounds). This “homogeneous noise” simplifies our expression of AIR, and the resulting algorithms remain applicable to independent noise models and the broader sub-Gaussian setting.

As discussed in Section 6.1, we restrict our attention to sparse  $v$  where for each  $a^* \in \mathcal{A}$  there is only one model  $M$ , which corresponds to the Gaussian distribution  $r(a) \sim N(\theta_{a^*}(a), 1)$ . We parameterize the prior  $v$  by vectors  $\{\beta_{a^*}\}_{a^* \in \mathcal{A}}$  and  $\alpha \in \Delta(\mathcal{A})$ , where  $\alpha = \mathbb{P}_v(a^*)$  and  $\beta_{a^*} = \alpha(a^*) \cdot \theta_{a^*}$ . Note that AIR in this “homogeneous noise” setting can be expressed as

$$\begin{aligned} \text{AIR}_{q,\eta}(p, v) &= \int_{\mathcal{A}} \beta_{a^*}^\top a^* da^* - \int_{\mathcal{A}} \int_{\mathcal{A}} p(a) \beta_{a^*}^\top a da^* da \\ &\quad - \frac{1}{2\eta} \int_{\mathcal{A}} \int_{\mathcal{A}} p(a) \alpha(a^*) \left( \frac{\beta_{a^*}^\top a}{\alpha(a^*)} - \int_{\mathcal{A}} \beta_{a^*}^\top a da^* \right)^2 da - \frac{1}{\eta} \text{KL}(\alpha, q). \end{aligned} \quad (17)$$

**ALGORITHM 5:** Simplified APS for Gaussian linear bandits

**Require:** Input learning rate  $\eta > 0$ , forced exploration rate  $\gamma$ , and action set  $\mathcal{A}$ . Initialize  $p_1 = \text{Unif}(\mathcal{A})$ .

- 1: **for** round  $t = 1, 2, \dots, T$  **do**
- 2:   Let  $S'_t$  be a  $(p_t, \exp(-(4\sqrt{d} + \log(2T))))$ -exp-volumetric spanner of  $\mathcal{A}$ ,  
      Let  $S''_t$  be a  $2\sqrt{d}$ -ratio-volumetric spanner of  $\mathcal{A}$ .  
      Set  $S_t$  as the union of  $S'_t$  and  $S''_t$ .
- 3:   Update  $\tilde{p}_t$  by  $\tilde{p}_t(a) = (1 - \gamma)p_t(a) + \frac{\gamma}{|S_t|}\mathbb{1}\{a \in S_t\}$
- 4:   Sample action  $a_t \sim \tilde{p}_t$  and receives  $r_t(a_t)$ .
- 5:   Calculate  $p_{t+1}$  by

$$p_{t+1}(a) \propto p_t(a) \exp(\eta \hat{r}_t(a)),$$

where  $\hat{r}_t$  is the modified IPW estimator for linear loss,

$$\hat{r}_t(a) = a^\top (\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a_t r_t(a_t) + \frac{\eta}{2} \left( a(\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a - (a^\top (\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a_t)^2 \right).$$

By setting the gradients of Equation (17) with respect to  $\alpha$  to zero and the gradients with respect to  $\{\beta_{a^*}\}_{a^* \in \mathcal{A}}$  to nearly zero, we obtain an approximate maximizer of AIR in Equation (17). We calculate the Bayesian posterior, and find that the resulting algorithm is an exponential weight algorithm with a modified IPW estimator: at each round  $t$ , the agent update  $p_{t+1}$  by

$$p_{t+1}(a) \propto p_t(a) \exp(\eta \hat{r}_t(a)),$$

where  $\hat{r}_t$  is the modified IPW estimator for linear reward,

$$\hat{r}_t(a) = \underbrace{a^\top (\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a_t r_t(a_t)}_{\text{IPW estimator}} + \underbrace{\frac{\eta}{2} \left( a(\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a - (a^\top (\mathbb{E}_{a \sim p_t}[aa^\top])^{-1} a_t)^2 \right)}_{\text{mean zero regularizer}}. \quad (18)$$

Note that in order to avoid boundary conditions in our derivation, we require forced exploration to ensure  $\lambda_{\min}(\mathbb{E}_{a \sim p_t}[aa^\top]) \geq \gamma$ . This can be done with the help of the volumetric spanners constructed in [27]. The use of volumetric spanner makes our final proposed algorithm (Algorithm 5) to be slightly more involved, but we only use the volumetric spanner in a “black-box” manner. We note that the additional “mean zero regularizer” term in Equation (18) does not change the standard regret analysis of exponential weight algorithms to establish the optimal  $\tilde{O}(d\sqrt{T})$  regret bound for Algorithm 5.<sup>4</sup> One may also analyze Algorithm 5 within our algorithmic belief framework through Theorem 3.4, as we did for Algorithm 4 in Section 2.2; we omit the analysis here. Finally, we note that the algorithm reduces to a modified version of EXP3 for finite armed bandits, a connection we mentioned at the end of Section 4.1.

### 6.3 Application to Bandit Convex Optimization

We consider the bandit convex optimization problem described in Example 2.2. In bandit convex optimization,  $\Pi \subseteq \mathbb{R}^d$  is a  $d$ -dimensional action set whose diameter is bounded by  $\text{diam}(\Pi)$ , and the mean reward (or loss) function is required to be concave (respectively, convex) with respect to actions:

$$\mathcal{F} = \{f : \Pi \rightarrow [0, 1] : f \text{ is concave w.r.t. } \pi \in \Pi\}.$$

<sup>4</sup>Specifically, the mean of the additional “regularizer” term in Equation (18) is zero, so Lemma 6.3 (mean-zero property) in [27] holds true; the variance of this “regularizer” term is bounded by  $O(d)$ , so Lemma 6.4 (bounded variance) in [27] holds true with increased numerical constant. Carefully examine all the proofs to Theorem 6.1 in [27] with minor changes in numerical constants validates  $\tilde{O}(d\sqrt{T})$  regret bound.

The problem is often formed with finite (but exponentially large) action set by standard discretization arguments [32]. Bandit convex optimization is a notoriously challenging problem, and much effort has been put to understanding its minimax regret and algorithm design. The best known result, which is of order  $\tilde{O}(d^{2.5}\sqrt{T})$ , is derived through the non-constructive IR analysis in [32]. By the IR upper bound for the non-constructive Bayesian IDS algorithm in [32], Lemma 2.5 that bounds AIR by IR, and Theorem 3.3 (regret of AMS), we immediately have that Algorithm 3 (AMS) with optimally tuned  $\eta$  achieves

$$\mathfrak{R}_T \leq O\left(d^{2.5}\sqrt{T} \cdot \text{polylog}(d, \text{diam}(\mathcal{A}), T)\right).$$

As a result, AMS recovers the best known  $\tilde{O}(d^{2.5}\sqrt{T})$  regret with a constructive algorithm. By our discussion on the computational complexity in Appendix 2.1, AMS solves convex optimization in a  $\text{poly}(|\Pi|)$ -dimensional space, so it can be computed in  $\text{poly}(e^d \cdot T)$  time for bandit convex optimization. To the best of our knowledge, this is the first algorithm with a finite running time that attains the best known  $\tilde{O}(d^{2.5}\sqrt{T})$  regret. We note that the EBO algorithm in [33] has given a constructive algorithm that achieves the same  $\tilde{O}(d^{2.5}\sqrt{T})$  regret derived by Bayesian non-constructive analysis. However, EBO operates in an abstract functional space, so it is less clear how to execute the computation. However, it is important to note that AMS still demands solving a hard minimax problem and has computational complexity exponential in  $d$ . Addressing this computational challenge—through heuristics, relaxations, or leveraging problem-specific simplifications—remains crucial for deploying our approach in broader domains.

## 7 MAIR and Application to RL

In the stochastic environment, where  $M_t = M^* \in \mathcal{M}$  for all rounds, we only need to search for algorithmic beliefs regarding the underlying model to determine the best decision  $\pi_{M^*}$ . This distinction allows us to introduce a strengthened version of AIR in Section 2.4, which we term MAIR, particularly suited for studying RL problems in the stochastic setting.

Crucially, we introduce a generic and closed-form sequence of algorithmic beliefs designed to approximate the maximization of MAIR at each round. By leveraging these beliefs, we develop MAMS and **Model-index APS (MAPS)** algorithms that re-derive the existing state-of-the-art regret bounds for RL problems within the bilinear class [18, 23]. While our statistical guarantees do not exceed those of [23] in the minimax sense, we believe our algorithmic derivations offer some complementary technical perspectives. For instance, several of our algorithms feature a general closed-form updating rule based directly on original Bayesian posteriors, rather than employing the scaled exponential-of-log-likelihood approach ( $\exp(1/2 \log\text{-likelihood})$ ) previously used in the RL [3, 15, 22, 60] as well as density estimation [25, 59] literature.

### 7.1 Generic Analysis Leveraging MAIR

In the stochastic setting, by leveraging MAIR, we can prove the following generic regret bound for arbitrary learning algorithm, and with arbitrarily chosen algorithmic belief sequence  $\{\mu_t\}_{t=1}^T$ . This generic regret bound is an analogy of Theorem 3.4, which is not only applicable to any algorithm but also allows the algorithmic beliefs to be flexibly selected.

**THEOREM 7.1 (GENERIC REGRET BOUND FOR ARBITRARY LEARNING ALGORITHM LEVERAGING MAIR).** *In the stochastic setting, let  $\mathcal{M}$  be a finite model class, and  $M^* \in \mathcal{M}$  be the ground truth model. Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $\rho_1 = \text{Unif}(\mathcal{M})$  and maintain a sequence of beliefs  $\mu_1, \dots, \mu_t, \dots$  where  $\rho_{t+1} := (\mu_t)_{\pi^*|\pi_t, o_t} \in \text{int}(\Delta(\mathcal{M}))$  for all rounds.*

**ALGORITHM 6:** Model-index Adaptive Minimax Sampling

**Require:** Input learning rate  $\eta > 0$ . Initialize  $\rho_1 = \text{Unif}(\mathcal{M})$ .

1: **for** round  $t = 1, 2, \dots, T$  **do**

2:   Find a distribution  $p$  of  $\pi$  and a distribution  $\mu_t$  of  $M$  that solves the saddle point of

$$\inf_{p \in \Delta(\Pi)} \sup_{\mu \in \Delta(\mathcal{M})} \text{MAIR}_{\rho_t, \eta}(p, \mu).$$

3:   Sample decision  $\pi_t \sim p_t$  and observe  $o_t \sim M^*(\pi_t)$ .

4:   Update  $\rho_{t+1} = (\mu_t)_{M|\pi_t, o_t}$ .

Then for all  $T \in \mathbb{N}_+$

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{MAIR}_{\rho_t, \eta}(p_t, \mu_t) + \left\langle \frac{\partial \text{MAIR}_{\rho_t, \eta}(p_t, \mu)}{\partial \mu} \Big|_{\mu=\mu_t}, \mathbb{1}(M^*) - \mu_t \right\rangle \right) \right],$$

where  $\mathbb{1}(M^*)$  is the vector whose  $M^*$ -coordinate is 1 but all other coordinates are 0.

An appealing aspect of Theorem 7.1 is that we only need to bound the gradient-based optimization error with respect to the same  $M^*$  for all rounds. This property arises due to the stochastic environment.

Specifically, when the algorithmic beliefs exactly maximizes MAIR at each round (optimization errors equal to 0), and when applying the minimax strategy in algorithm design, we propose MAMS, see Algorithm 6 below. Because the minimax value of MAIR is always upper bounded by DEC, as illustrated in Lemma 2.10, it is straightforward to prove the following theorem.

**THEOREM 7.2 (REGRET OF MAMS).** *For a finite and compact model class  $\mathcal{M}$ , the regret of Algorithm 6 with any  $\eta > 0$  is always bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{MAIR}_{\rho_t, \eta}(p_t, \mu_t) \right] \leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_{\eta}^{\text{KL}}(\mathcal{M}) \cdot T.$$

This shows that the regret bound of MAMS is always no worse than the regret bound of the **Estimation-to-Decision (E2D)** algorithm in [23].

**Comparing AIR and MAIR.** We have seen from Lemmas 2.7, 2.10, and 2.9 that (1) Maximin AIR can be bounded by DEC of the convex hull  $\text{conv}(\mathcal{M})$ ; (2) Maximin MAIR can be bounded by DEC of the original class  $\mathcal{M}$ ; and (3) MAIR is “smaller” than AIR. However, as we have shown in Theorem 3.1 and Theorem 7.1, the regret bound using AIR scales with a  $\log |\Pi|$  term (estimation complexity of decision space), while the regret bound using MAIR scales with a bigger  $\log |\mathcal{M}|$  term (estimation complexity of model class). We explain their difference as follows.

**When to use AIR versus MAIR?** First, AIR is applicable to both stochastic and adversarial environments, whereas MAIR may only be applicable to stochastic environments. Second, when using AIR, an estimation complexity term of  $\log |\Pi|$  is introduced, whereas MAIR results in a larger estimation complexity term of  $\log |\mathcal{M}|$ . Therefore, AIR often yields tighter regret bounds for bandit problems. For instance, AIR provides optimal regret bounds for MAB and achieves  $\sqrt{T}$ -type regret bounds for the challenging problem of bandit convex optimization, whereas MAIR may not attain these results. On the other hand, MAIR does achieve optimal regret bounds for stochastic linear bandits. Notably, MAIR also achieves optimal regret bounds for stochastic contextual bandits with general realizable function classes [19, 47], including scenarios with potentially infinite actions [20, 57, 61]. Moreover, MAIR is better suited than AIR for RL problems in which taking the convex hull to the model class may significantly enhance its expressiveness. For instance, in RL problems, the model

class, especially the state transition dynamics, often does not adhere to convexity assumptions. In general, AIR is more suitable for “infinite divisible” problems where taking the convex hull does not substantially increase the complexity of the model class. Conversely, MAIR is better suited for stochastic model-based bandit and RL problems in which avoiding convex hull operations is preferred.

## 7.2 Near-Optimal Algorithmic Beliefs in Closed Form

For any fixed decision probability  $p$ , it is illustrative to write MAIR as

$$\begin{aligned} \text{MAIR}_{\rho, \eta}(p, \mu) &= \mathbb{E}_{p, \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(\mu(\cdot|\pi, o), \rho) \right] \\ &= \mathbb{E}_{p, \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(\mu(\cdot|\pi, o), \mu) - \frac{1}{\eta} \text{KL}(\mu, \rho) \right] \\ &= \mathbb{E}_{p, \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \mu_{o|\pi}) - \frac{1}{\eta} \text{KL}(\mu, \rho) \right], \end{aligned} \quad (19)$$

where  $\mu_{o|\pi} = \mathbb{E}_{M \sim \mu}[M(\pi)]$  is the induced distribution of  $o$  conditioned on  $\pi$ , and the third equality is by property of mutual information. We would like to give a sequence of algorithmic beliefs that approximately maximize MAIR at each rounds, as well as having closed-form expression.

We consider the following algorithmic priors at each round:

$$\mu_t(M) \propto \rho_t(M) \cdot \exp \left( \underbrace{\eta(f_M(\pi_M) - \mathbb{E}_{\pi \sim p_t}[f_M(\pi)]) - \frac{1}{3} \mathbb{E}_{\pi \sim p_t}[D_H^2(M(\pi), (\rho_t)_{o|\pi})]}_{\text{adaptive algorithmic belief}} \right), \quad (20)$$

where the adaptive algorithmic belief in Equation (20) attempts to optimize the MAIR objective in Equation (19), with  $D_H^2(M(\pi), (\rho_t)_{o|\pi})$  approximates  $\text{KL}(M(\pi), (\mu_t)_{o|\pi})$ . The factor 1/3 before the squared Hellinger distance is a technical consequence of the triangle-type inequality of the squared Hellinger distance, which has a factor 2 rather than 1 (see Lemma 4.12). And we use their corresponding posteriors to update the sequence of reference probabilities:

$$\rho_{t+1}(M) = \mu_t(M|\pi_t, o_t) \propto \mu_t(M) M[\pi_t](o_t).$$

This results in the following update of  $\rho$ :

$$\rho_{t+1}(M) \propto \exp \left( \sum_{s=1}^t \left( \underbrace{\log[M(\pi_s)](o_s)}_{\text{log likelihood}} + \underbrace{\eta(f_M(\pi_M) - \mathbb{E}_{\pi \sim p_s}[f_M(\pi)]) - \frac{1}{3} \mathbb{E}_{\pi \sim p_s}[D_H^2(M(\pi), (\rho_s)_{o|\pi})]}_{\text{adaptive algorithmic belief}} \right) \right). \quad (21)$$

Using such algorithmic beliefs, the regret of an arbitrary algorithm can be bounded as follows.

**THEOREM 7.3 (REGRET FOR ARBITRARY ALGORITHM WITH CLOSED-FORM BELIEFS).** *Given a finite model class  $\mathcal{M}$  where the underlying true model is  $M^* \in \mathcal{M}$ , and  $f_M(\pi) \in [0, 1]$  for every  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . For an arbitrary algorithm  $\text{ALG}$  and any  $\eta > 0$ , the regret of algorithm  $\text{ALG}$  is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{\mu_t, p_t} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), (\rho_t)_{o|\pi}) - \frac{1}{3\eta} \text{KL}(\mu_t, \rho_t) \right] \right].$$

where  $\rho_t$  and  $\mu_t$  are closed-form beliefs generated according the Algorithm 7.

---

**ALGORITHM 7:** Closed-form approximate maximizers of MAIR

---

**Require:** Input algorithm ALG and learning rate  $\eta > 0$ . Initialize  $\rho_1$  to be the uniform distribution over  $\mathcal{M}$ .

- 1: **for** round  $t = 1, 2, \dots, T$  **do**
- 2:   Obtain  $p_t$  from ALG. The algorithm ALG samples  $\pi_t \sim p_t$  and observe the feedback  $o_t \sim M_t(\pi_t)$ .
- 3:   Update

$$\begin{aligned} \mu_t(M) &\propto \rho_t(M) \cdot \exp\left(\eta(f_M(\pi_M) - \mathbb{E}_{\pi \sim p_t}[f_M(\pi)]) - \frac{1}{3}\mathbb{E}_{\pi \sim p_t}\left[D_H^2(M(\pi), \rho_{t|o})\right]\right), \\ \rho_{t+1}(M) &= \mu_t(M|\pi_t, o_t). \end{aligned}$$


---

---

**ALGORITHM 8:** Model-index Adaptive Posterior Sampling

---

**Require:** Input learning rate  $\eta$ . Initialize  $\mu_1$  to be the uniform distribution over  $\mathcal{M}$ .

- 1: **for** round  $t = 1, 2, \dots, T$  **do**
  - 2:   Sample  $\pi_t \sim p_t$  where  $p_t(\pi) = \sum_{M: \pi_M=\pi} \rho_t(M)$ , and observe the feedback  $o_t \sim M_t(\pi_t)$ .
  - 3:   Update  $\mu_t$  and  $\rho_{t+1}$  according to Algorithm 7.
- 

By combining the belief generation process in Algorithm 7 as an estimation oracle to E2D [23], we can recover its regret bound using DEC, as we have demonstrated with MAMS in Theorem 7.2. Now, we have closed-form expressions for the near-optimal beliefs. Note that our reference probabilities (Equation (21)) update both the log-likelihood term and an adaptive algorithmic belief term at each iteration. In contrast, most existing algorithms, whether optimistic or not, typically update only the log-likelihood term and rely on a fixed prior term. Furthermore, it's worth highlighting that existing "optimistic" or "model-free" algorithms often employ a scaled version of the Bayesian posterior formula: e.g.,  $\exp(1/2 \log\text{-likelihood})$  [3, 15, 22, 60]. In contrast, we approach the problem using the original Bayesian posterior formula by incorporating adaptive beliefs. This approach may also have implications for fundamental problems like density estimation with proper estimators, as earlier methods have typically relied on the scaled version of posterior formulas [25, 59], rather than directly utilizing the original posterior formula as we do here.

In our applications, we often use a simple posterior sampling strategy for which we always induce the distribution of optimal decisions from the posterior distribution of models. We refer to the resulting algorithm, Algorithm 8, as "Model-index Adaptive Posterior Sampling" (MAPS).

MAPS draws inspiration from the optimistic posterior sampling algorithm proposed in [3] (also referred to as feel-good TS in [60]). However, our approach incorporates adaptive algorithmic beliefs and the original Bayesian posterior formula, rather than using the fixed prior and the scaled posterior update formulas as in [3, 60].

For model class  $\mathcal{M}$ , a nominal model  $\bar{M}$ , and the posterior sampling strategy  $p^{\text{TS}}(\pi) = \mu(\{M : \pi_M = \pi\})$ , we can define the Bayesian DEC of TS by

$$\text{DEC}_\eta^{\text{TS}}(\mathcal{M}) = \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \sup_{\mu \in \Delta(\mathcal{M})} \mathbb{E}_{\mu, p^{\text{TS}}} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} D_H^2(M(\pi), \bar{M}(\pi)) \right]. \quad (22)$$

This value is bigger than the minimax DEC in Definition 2.6, but often easier to use in RL problems.

**THEOREM 7.4 (REGRET OF MODEL-INDEX ADAPTIVE POSTERIOR SAMPLING).** *Given a finite model class  $\mathcal{M}$  where  $f_M(\pi) \in [0, 1]$  for every  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . The regret of Algorithm 8 with  $\eta \in (0, 1/3]$  is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_{6\eta}^{\text{TS}}(\mathcal{M}, \bar{M}) \cdot T + 6\eta T.$$

### 7.3 Application to Reinforcement Learning

We mention in passing that by using MAMS (Algorithm 6), MAPS (Algorithm 8), and Algorithm 7 (closed-form algorithmic belief generation), we are able to recover several results in [23] that bound the regret of RL by DEC and the estimation complexity  $\log |\mathcal{M}|$  of the model class. See Appendix 3.1 for details.

Our results in this section apply to RL problems where the DEC is easy to upper bound, but bounding the IR may be more challenging, particularly for complex RL problems where the model class  $\mathcal{M}$  may not be convex and the average of two MDPs may not belong to the model class. Specifically, we propose MAIR and provide a generic algorithm that uses DEC and the estimation complexity of the model class ( $\log |\mathcal{M}|$ ) to bound the regret. Another promising research direction is to extend our general results for AIR and the tools from Section 6.2 to RL problems with suitably bounded IRs, such as tabular MDPs and linear MDPs, as suggested in [26]. We hope our framework can facilitate the development of constructive algorithms whose regret bounds scale exclusively with the estimation complexity of the value function class—a complexity typically lower than that of the model class. Indeed, subsequent work by [38] has already started to explore this promising direction using our approach.

## 8 Conclusion and Future Directions

In this work, we propose a novel approach to solve sequential learning problems by generating “algorithmic beliefs.” We optimize the AIR to generate these beliefs. Surprisingly, our algorithms achieve regret bounds that are as good as those assuming prior knowledge, even in the absence of such knowledge, which is often the case in adversarial or complex environments. Our approach yields a set of conceptually simple algorithms applicable to a wide range of problems (MAB, linear and convex bandits, RL). In certain specific cases, these algorithms can be made computationally efficient – for example, in multi-armed and linear bandits. However, we do not claim general efficiency for all settings; rather, the contribution is in providing a unifying formulation and analysis that can guide the design of efficient algorithms in well-structured special cases.

Our work provides a new perspective on designing and analyzing bandit and RL algorithms. Our theory applies to any algorithm through the notions of AIR and algorithmic beliefs, and it provides a simple and constructive understanding of the duality between frequentist regret and Bayesian regret in sequential learning. Optimizing AIR is a key principle to design effective and efficient bandit and RL algorithms. We demonstrate the effectiveness of our framework empirically via experiments on Bernoulli MAB and show that our derived algorithm achieves “best-of-all-worlds” empirical performance. Specifically, our algorithm outperforms UCB and is comparable to TS in stochastic bandits, outperforms EXP3 in adversarial bandits, and outperforms TS as well as clairvoyant restarted algorithms in non-stationary bandits.

Our study suggests several potential research directions, and we hope to see progress made by utilizing the methodology developed in this work. Firstly, an important task is to provide computational and representational guidelines for optimizing algorithmic beliefs, such as techniques for selecting belief parameterization and index representation (function class approximation). Notably, optimizing AMS and APS exactly in a large-scale problem is generally intractable, so a key research direction is to develop approximation strategies and identify special structures to make this approach practical. Secondly, a major goal is to achieve near-optimal regrets with efficient algorithms for challenging problems in infinite-armed bandits, contextual bandits, and RL. An initial step involves exploring the Bayesian interpretation of existing frequentist approaches, including gaining a deeper understanding of IPW-type estimators and related computationally-efficient algorithms [1, 2, 13]. Moreover, it is worth investigating whether our approach can facilitate the development of more precise regret bounds and principled algorithm design for RL problems

involving function approximation. Thirdly, an important direction is to leverage algorithmic beliefs to study adaptive [14, 54] and dynamic regrets [55], and explore instance optimality [52]. Fourthly, our article introduces a novel framework for analyzing regret through AIR and offers a rich mathematical structure to explore and uncover, including the geometry of natural parameterization for maximizing AIR and its correspondence to the mirror space (see Appendix 2 for details). Fifthly, our aim is to comprehend alternative formulations of AIR, including the constrained formulation (drawing inspiration from the recent investigation of the constrained formulation of DEC in [21]); connection may be made between regularization and the notion of localization [56]. Finally, we hope that the duality identities, which can accommodate arbitrary beliefs, algorithms, and any type of environment (as illustrated in Lemma 5.3), may offer values for lower bound analyses and information theory. We encourage exploration in these and all other relevant directions.

## Acknowledgments

The first author gratefully acknowledges that part of this work was initiated at Columbia University. We thank Yunzong Xu for valuable discussions, and Adam Wierman as well as the anonymous reviewers for their helpful comments and careful reading.

## References

- [1] Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. 2008. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the 21st Annual Conference on Learning Theory, COLT 2008*.
- [2] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. 2014. Taming the monster: A fast and simple algorithm for contextual bandits. In *Proceedings of the International Conference on Machine Learning*. 1638–1646.
- [3] Alekh Agarwal and Tong Zhang. 2022. Model-based RL with optimistic posterior sampling: Structural conditions and sample complexity. *Advances in Neural Information Processing Systems* 35, Article No.: 2557 (2022), 35284–35297.
- [4] Shipra Agrawal and Navin Goyal. 2012. Analysis of Thompson sampling for the multi-armed bandit problem. In *Proceedings of the Conference on Learning Theory*. JMLR Workshop and Conference Proceedings, 39–1.
- [5] Shipra Agrawal and Navin Goyal. 2013. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the International Conference on Machine Learning*. PMLR, 127–135.
- [6] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47, 2 (2002), 235–256.
- [7] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing* 32, 1 (2002), 48–77.
- [8] Pierre Bernhard and Alain Rapaport. 1995. On a theorem of Danskin with an application to a theorem of Von Neumann-Sion. *Nonlinear Analysis: Theory, Methods & Applications* 24, 8 (1995), 1163–1181.
- [9] Dimitri P. Bertsekas. 1999. *Nonlinear Programming*. Athena Scientific, Belmont, MA.
- [10] Omar Besbes, Yonatan Gur, and Assaf Zeevi. 2014. Stochastic multi-armed-bandit problem with non-stationary rewards. *Advances in Neural Information Processing Systems* 1, 27 (2014), 199–207.
- [11] Vladimir Igorevich Bogachev and Maria Aparecida Soares Ruas. 2007. *Measure Theory*. Vol. 1. Springer.
- [12] Sébastien Bubeck and Ronen Eldan. 2016. Multi-scale exploration of convex functions and bandit convex optimization. In *Proceedings of the Conference on Learning Theory*. PMLR, 583–589.
- [13] Sébastien Bubeck, Yin Tat Lee, and Ronen Eldan. 2017. Kernel-based methods for bandit convex optimization. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*. 72–85.
- [14] Sébastien Bubeck and Aleksandrs Slivkins. 2012. The best of both worlds: Stochastic and adversarial bandits. In *Proceedings of the Conference on Learning Theory*. JMLR Workshop and Conference Proceedings, 42–1.
- [15] Fan Chen, Song Mei, and Yu Bai. 2025. Unified algorithms for RL with decision-estimation coefficients: PAC, reward-free, preference-based learning and beyond. *The Annals of Statistics* 53, 1 (2025), 426–456.
- [16] Varsha Dani, Sham M. Kakade, and Thomas Hayes. 2007. The price of bandit information for online optimization. *Advances in Neural Information Processing Systems* 20 (2007), 345–352.
- [17] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. 2020. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics* 20, 4 (2020), 633–679.
- [18] Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. 2021. Bilinear classes: A structural framework for provable generalization in RL. In *Proceedings of the International Conference on Machine Learning*. PMLR, 2826–2836.

- [19] Dylan Foster and Alexander Rakhlin. 2020. Beyond UCB: Optimal and efficient contextual bandits with regression oracles. In *Proceedings of the International Conference on Machine Learning*. PMLR, 3199–3210.
- [20] Dylan J. Foster, Claudio Gentile, Mehryar Mohri, and Julian Zimmert. 2020. Adapting to misspecification in contextual bandits. *Advances in Neural Information Processing Systems* 33, Article No.: 963 (2020), 11478–11489.
- [21] Dylan J. Foster, Noah Golowich, and Yanjun Han. 2023. Tight guarantees for interactive decision making with the decision-estimation coefficient. In *The Thirty Sixth Annual Conference on Learning Theory*. PMLR, 3969–4043.
- [22] Dylan J. Foster, Noah Golowich, Jian Qian, Alexander Rakhlin, and Ayush Sekhari. 2023. Model-free reinforcement learning with the decision-estimation coefficient. *Advances in Neural Information Processing Systems* 36, 881 (2023), 20080–20117.
- [23] Dylan J. Foster, Sham M. Kakade, Jian Qian, and Alexander Rakhlin. 2021. The statistical complexity of interactive decision making. arXiv:2112.13487. Retrieved from <https://arxiv.org/abs/2112.13487>
- [24] Dylan J. Foster, Alexander Rakhlin, Ayush Sekhari, and Karthik Sridharan. 2022. On the complexity of adversarial decision making. *Advances in Neural Information Processing Systems* 35, Article No.: 2566 (2022), 35404–35417.
- [25] Sara A. Geer. 2000. *Empirical Processes in M-estimation*. Vol. 6. Cambridge University Press.
- [26] Botao Hao and Tor Lattimore. 2022. Regret bounds for information-directed reinforcement learning. *Advances in Neural Information Processing Systems* 35, Article No.: 2071 (2022), 28575–28587.
- [27] Elad Hazan and Zohar Karnin. 2016. Volumetric spanners: An efficient exploration basis for learning. *The Journal of Machine Learning Research* 17, 1 (2016), 4062–4095.
- [28] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E. Schapire. 2017. Contextual decision processes with low Bellman rank are PAC-learnable. In *Proceedings of the International Conference on Machine Learning*. 1704–1713.
- [29] Chi Jin, Qinghua Liu, and Sobhan Miryoosefi. 2021. Bellman Eluder dimension: New rich classes of RL problems, and sample-efficient algorithms. *Advances in Neural Information Processing Systems* 34, Article No.: 1027 (2021), 13406–13418.
- [30] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I. Jordan. 2020. Provably efficient reinforcement learning with linear function approximation. In *Proceedings of the Conference on Learning Theory*. PMLR, 2137–2143.
- [31] Tze Leung Lai and Herbert Robbins. 1985. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics* 6, 1 (1985), 4–22.
- [32] Tor Lattimore. 2020. Improved regret for zeroth-order adversarial bandit convex optimisation. *Mathematical Statistics and Learning* 2, 3 (2020), 311–334.
- [33] Tor Lattimore and Andras Gyorgy. 2021. Mirror descent and the information ratio. In *Proceedings of the Conference on Learning Theory*. PMLR, 2965–2992.
- [34] Tor Lattimore and Csaba Szepesvári. 2019. An information-theoretic approach to minimax regret in partial monitoring. In *Proceedings of the Conference on Learning Theory*. PMLR, 2111–2139.
- [35] Tor Lattimore and Csaba Szepesvári. 2020. *Bandit Algorithms*. Cambridge University Press.
- [36] Tor Lattimore and Csaba Szepesvári. 2020. Exploration by optimisation in partial monitoring. In *Proceedings of the Conference on Learning Theory*. PMLR, 2488–2515.
- [37] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. 661–670.
- [38] Haolin Liu, Chen-Yu Wei, and Julian Zimmert. 2025. Decision making in hybrid environments: A model aggregation approach. arXiv:2502.05974. Retrieved from <https://arxiv.org/abs/2502.05974>
- [39] Teodor Vanislavov Marinov and Julian Zimmert. 2021. The pareto frontier of model selection for general contextual bandits. *Advances in Neural Information Processing Systems* 34, Article No.: 1374 (2021), 17956–17967.
- [40] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.
- [41] Yury Polyanskiy and Yihong Wu. 2025. Information theory: From coding to learning. Cambridge university press.
- [42] Ralph Tyrell Rockafellar. 2015. *Convex Analysis*. Princeton University Press.
- [43] Daniel Russo and Benjamin Van Roy. 2013. Eluder dimension and the sample complexity of optimistic exploration. In *Proceedings of the International Conference on Neural Information Processing Systems*. 2256–2264.
- [44] Daniel Russo and Benjamin Van Roy. 2014. Learning to optimize via information-directed sampling. *Advances in Neural Information Processing Systems* 1, 27 (2014), 1583–1591.
- [45] Daniel Russo and Benjamin Van Roy. 2016. An information-theoretic analysis of Thompson sampling. *The Journal of Machine Learning Research* 17, 1 (2016), 2442–2471.
- [46] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.

- [47] David Simchi-Levi and Yunzong Xu. 2022. Bypassing the monster: A faster and simpler optimal algorithm for contextual bandits under realizability. *Mathematics of Operations Research* 47, 3 (2022), 1904–1931.
- [48] Maurice Sion. 1958. On general minimax theorems. *Pacific Journal of Mathematics* 8, 1 (1958), 171–176.
- [49] Myunghyun Song. 2019. Proving that the conditional entropy of a probability measure is concave. *Mathematics Stack Exchange*. Retrieved September 1, 2025 from <https://math.stackexchange.com/q/3080334>
- [50] Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. 2019. Model-based RL in contextual decision processes: PAC bounds and exponential improvements over model-free approaches. In *Proceedings of the Conference on Learning Theory*. PMLR, 2898–2933.
- [51] William R. Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3-4 (1933), 285–294.
- [52] Andrew J. Wagenmaker and Dylan J. Foster. 2023. Instance-optimality in interactive decision making: Toward a non-asymptotic theory. In *Proceedings of the 36th Annual Conference on Learning Theory*. PMLR, 1322–1472.
- [53] Ruosong Wang, Russ R. Salakhutdinov, and Lin Yang. 2020. Reinforcement learning with general value function approximation: Provably efficient approach via bounded eluder dimension. *Advances in Neural Information Processing Systems* 33, Article No.: 514 (2020), 6123–6135.
- [54] Chen-Yu Wei and Haipeng Luo. 2018. More adaptive algorithms for adversarial bandits. In *Proceedings of the Conference on Learning Theory*. PMLR, 1263–1291.
- [55] Chen-Yu Wei and Haipeng Luo. 2021. Non-stationary reinforcement learning without prior knowledge: An optimal black-box approach. In *Proceedings of the Conference on Learning Theory*. PMLR, 4300–4354.
- [56] Yunbei Xu and Assaf Zeevi. 2025. Towards optimal problem dependent generalization error bounds in statistical learning theory. *Mathematics of Operations Research* 50, 1 (2025), 40–67.
- [57] Yunbei Xu and Assaf Zeevi. 2020. Upper counterfactual confidence bounds: A new optimism principle for contextual bandits. arXiv:2007.07876. Retrieved from <https://arxiv.org/abs/2007.07876>
- [58] Lin Yang and Mengdi Wang. 2019. Sample-optimal parametric Q-learning using linearly additive features. In *Proceedings of the International Conference on Machine Learning*. PMLR, 6995–7004.
- [59] Tong Zhang. 2006. From  $\epsilon$ -entropy to KL-entropy: Analysis of minimum information complexity density estimation. *The Annals of Statistics* 34, 5 (2006), 2180–2210.
- [60] Tong Zhang. 2022. Feel-good Thompson sampling for contextual bandits and reinforcement learning. *SIAM Journal on Mathematics of Data Science* 4, 2 (2022), 834–857.
- [61] Yinglun Zhu, Dylan J. Foster, John Langford, and Paul Mineiro. 2022. Contextual bandits with large action spaces: Made practical. In *Proceedings of the International Conference on Machine Learning*. PMLR, 27428–27453.

## Appendix

### A Extensions and Proofs for AIR

#### A.1 Extensions of AIR

**A.1.1 Extension to General Bregman Divergence.** We can generalize AIR from using KL divergence to using general Bregman divergence. And all the results in Section 3 can be extended as well. This generalization is inspired by [33], which defines IR and studies algorithm design using general Bregman divergence.

Let  $\Psi : \Delta(\Pi) \rightarrow \mathbb{R} \cup \infty$  be a convex Legendre function. Denote  $D_\Psi$  to be the Bregman divergence of  $\Psi$ , and  $\text{diam}(\Psi)$  to be the diameter of  $\Psi$ . We refer to Appendix 4.4 for the background of these concepts. Given a reference probability  $q \in \text{int}(\Delta(\Pi))$  in the interior of the simplex and learning rate  $\eta > 0$ , we define the generalized AIR with potential function  $\Psi$  for decision  $p$  and distribution  $v$  by

$$\text{AIR}_{q,\eta}^\Psi(p, v) = \mathbb{E} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} D_\Psi(v_{\pi^*|\pi,o}, v_{\pi^*}) - \frac{1}{\eta} D_\Psi(v_{\pi^*}, q) \right]. \quad (23)$$

We generalize Theorems 3.1 and 3.4 to general Bregman divergence, with the  $\frac{\log |\Pi|}{\eta}$  term in the regret bounds being replaced by a  $\frac{\text{diam}(\Psi)}{\eta}$  term. Using the extension, we can generalize Theorem 3.2 (regret of APS) and Theorem 3.3 (regret of AMS) to generalized Bregman divergence as well, where the definition of IR will also use the corresponding Bregman divergence as in [33]. We state the extension of Theorem 3.4 here.

**THEOREM A.1 (USING GENERAL BREGMAN DIVERGENCE).** *Assume  $\Psi : \Delta(\Pi) \rightarrow \mathbb{R} \cup \infty$  is Legendre and has bounded diameter. Let  $\Pi$  be a finite decision space and  $\mathcal{M}$  a compact model class. Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and maintain a sequence of beliefs  $v_1, \dots, v_t, \dots$  where  $q_{t+1} := (v_t)_{\pi^*|_{\pi_t, o_t}} \in \text{int}(\Delta(\Pi))$  for all rounds. Then for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\text{diam}(\Psi)}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{AIR}_{q_t, \eta}^\Psi(p_t, v_t) + \sup_{v^*} \left( \frac{\partial \text{AIR}_{q_t, \eta}^\Psi(p_t, v)}{\partial v} \Big|_{v=v_t}, v^* - v_t \right) \right) \right].$$

Note that an analogous result to Theorem 7.2 can also be derived using the general Bregman divergence version of MAIR in the stochastic setting.

**A.1.2 Extension to High Probability Bound.** We remark that our regret guarantees focus on expected regret (pseudo-regret), and an important next step is to extend these results to high-probability (actual) regret bounds. We conjecture that the results presented in Section 3 may be extendable to high-probability regret bounds by modifying our algorithmic design—for instance, replacing the expectation of the KL divergence in AIR with either (1) the log of the **moment-generating function (MGF)** of the KL divergence, or (2) the squared Hellinger distance. Indeed, Appendix A of [36] demonstrates that the EBO algorithm can be adapted to yield high-probability bounds by incorporating the log-MGF, and [24] similarly obtains high-probability results for adversarial DMSO problems using a connection between the MGF of log-loss and the squared Hellinger distance (see page 7 therein). Since the log-MGF is known to always preserve convexity, the resulting optimization problem is likely to remain concave in terms of the algorithmic belief, allowing our general approach to extend naturally to this setting. However, formally proving this conjecture—and determining whether the resulting MAB algorithm would remain the same as our Algorithm 4—is a nontrivial task that deserves a separate treatment. We therefore highlight this extension as an open question and refer to [24, 36] as potential starting points to approach this important direction.

## A.2 Proof of Theorem 3.1

**THEOREM 3.1 (GENERIC REGRET BOUND FOR ARBITRARY LEARNING ALGORITHM).** *Given a finite decision space  $\Pi$ , a compact model class  $\mathcal{M}$ , the regret of an arbitrary learning algorithm ALG is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right].$$

By the discussion in Sections 5.1 and 5.2, we only need to prove Lemma 5.2 in order to prove Theorem 3.1.

**LEMMA 5.2 (IDENTITY BY NASH EQUILIBRIUM).** *Given  $q \in \text{int}(\Delta(\Pi))$ ,  $\eta > 0$  and  $p \in \Delta(\Pi)$ , denote  $\bar{v} \in \Delta(\mathcal{M} \times \Pi)$ . Then we have*

$$\sup_{M, \bar{\pi}} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ f_M(\bar{\pi}) - f_M(\pi) - \frac{1}{\eta} \log \frac{\bar{v}_{\pi^*|_{\pi, o}}(\bar{\pi})}{q(\bar{\pi})} \right] = \text{AIR}_{q, \eta}(p, \bar{v}).$$

**PROOF OF LEMMA 5.2:** Let  $\mathcal{Q}$  be the space of all mappings from  $\Pi \times \mathcal{O}$  to  $\Delta(\Pi)$ . For every mapping  $Q \in \mathcal{Q}$ , denote  $Q[\pi, o](\cdot) \in \Delta(\Pi)$  as the image of  $(\pi, o)$ , which corresponds to a distribution of  $\pi^*$ .

Given the fixed decision probability  $\pi \sim p$ , define  $B : \Delta(\mathcal{M} \times \Pi) \times \mathcal{Q} \rightarrow \mathbb{R}$  by

$$B(v, Q) = \mathbb{E}_{(M, \pi^*) \sim v, \pi \sim p} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \log \frac{Q[\pi, o](\pi^*)}{q(\pi^*)} \right]. \quad (24)$$

This is a strongly convex function with respect to  $Q$ . For every belief  $v \in \Delta(M \times \Pi)$ , denote  $Q_v = \arg \min_{Q \in \mathcal{Q}} B(v, Q)$  as the unique minimizer of  $B(v, Q)$  given  $v$ . By the first-order optimality condition, we have that  $Q_v$  corresponds to the posterior functional that always maps the observation pair  $(\pi, o)$  to the marginal Bayesian posterior  $v_{\pi^*|\pi, o}$  given prior  $v$ . Plugging in such minimizer  $Q_v$  into  $B(v, Q)$ , we have

$$\begin{aligned} & \inf_{Q \in \mathcal{Q}} B(v, Q) \\ &= B(v, Q_v) \\ &= \mathbb{E}_{(M, \pi^*) \sim v, \pi \sim p} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{\pi^*|\pi, o}, q) \right] \\ &= \text{AIR}_{q, \eta}(p, v). \end{aligned}$$

Applying our proposed constructive minimax theorem for minimax solutions (Lemma 5.1), by choosing the worst-case prior belief

$$\bar{v} \in \arg \max \text{AIR}_{q, \eta}(p, v),$$

we can establish that  $(\bar{v}, Q_{\bar{v}})$  will be a Nash equilibrium and  $Q_{\bar{v}}$  is a construction of the minimax solution of the minimax optimization problem  $\min_Q \max_v B(v, Q)$ . As a result, we have

$$\begin{aligned} & \text{AIR}_{q, \eta}(p, \bar{v}) \\ &= B(\bar{v}, Q_{\bar{v}}) \\ &= \sup_v B(v, Q_{\bar{v}}) \\ &= \sup_{M, \bar{\pi}} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ f_M(\bar{\pi}) - f_M(\pi) - \frac{1}{\eta} \log \frac{\bar{v}_{\pi^*|\pi, o}(\bar{\pi})}{q(\bar{\pi})} \right]. \end{aligned}$$

Finally, note that in order to apply our proposed minimax theorem for minimax solutions (Lemma 5.1) to the concave-convex objective function  $B$ , we need to verify that the sets  $\mathcal{Q}$  and  $\Delta(\mathcal{M} \times \Pi)$  are convex and compact sets, and  $B$  is continuous with respect to both  $Q \in \mathcal{Q}$  and  $v \in \Delta(\mathcal{M} \times \Pi)$ . This verification step assumes a basic understanding of general topology, as it involves infinite sets (compactness and continuity for finite sets are trivial). We demonstrate the verification step below, which is optional for readers with a general background.

*Verification of the conditions of our constructive minimax theorem (Lemma 5.1):* It is straightforward to see convexity of the sets  $\mathcal{Q}$  and  $\Delta(\mathcal{M} \times \Pi)$ . As a collection of mappings,  $\mathcal{Q}$  is compact with respect to the product topology by Tychonoff's theorem, and  $B$  is continuous with respect to  $Q$  by the definition of product topology. Because the probability measure on the compact set is compact with respect to the weak\*-topology,  $\Delta(\mathcal{M} \times \Pi)$  is a compact set. Finally,  $B$  is continuous with respect to  $v$  because  $B$  is linear in  $v$ . For foundational knowledge in general topology, we direct readers to [11]. A similar verification process concerning compactness and continuity can be found in [33], where we borrow the technique.  $\square$

### A.3 Proof of Lemma 5.1

LEMMA 5.1 (A CONSTRUCTIVE MINIMAX THEOREM FOR MINIMAX SOLUTIONS). *Let  $\mathcal{X}$  and  $\mathcal{Y}$  be convex and compact sets, and  $\psi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  a function which for all  $y$  is strongly convex and continuous*

in  $x$  and for all  $x$  is concave and continuous in  $y$ . For each  $y \in \mathcal{Y}$ , let  $x_y = \min_{x \in \mathcal{X}} \psi(x, y)$  be the corresponding unique minimizer. Then, by maximizing the concave objective

$$\bar{y} \in \arg \max_{y \in \mathcal{Y}} \psi(x_{\bar{y}}, y),$$

we can establish that  $x_{\bar{y}}$  is a minimax solution of  $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y)$  and  $(x_{\bar{y}}, \bar{y})$  is a Nash equilibrium.

**PROOF OF LEMMA 5.1:** Since  $\mathcal{X}$  and  $\mathcal{Y}$  are convex sets, and  $\psi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  is a function such that, for all  $y \in \mathcal{Y}$ , it is convex and continuous in  $x$ , and for all  $x \in \mathcal{X}$ , it is concave and continuous in  $y$ , all the conditions of Sion's minimax theorem (Lemma 4.2) are satisfied. By applying Sion's minimax theorem, we have the following equality:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \psi(x, y). \quad (25)$$

By the definition of  $\bar{y}$ , we know that  $\bar{y} \in \arg \max_{y \in \mathcal{Y}} \psi(x, y)$  is a maximin solution of the convex-concave game  $\psi$ . Similarly, denote  $\bar{x} \in \arg \min_{x \in \mathcal{X}} \psi(x, y)$  to be a minimax solution of the convex-concave game  $\psi$ . We claim that  $(\bar{x}, \bar{y})$  is a Nash equilibrium of the game, meaning  $\bar{x} \in \arg \min_{x \in \mathcal{X}} \psi(x, \bar{y})$  and  $\bar{y} \in \arg \max_{y \in \mathcal{Y}} \psi(\bar{x}, y)$ . This claim can be proved as follows:

Since  $\bar{x}$  is a minimax solution, we have

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y) = \max_{y \in \mathcal{Y}} \psi(\bar{x}, y) \geq \psi(\bar{x}, \bar{y}), \quad (26)$$

where the last inequality will be an equality if and only if  $\bar{y} \in \max_{y \in \mathcal{Y}} \psi(\bar{x}, y)$ . Similarly, because  $\bar{y}$  is a maximin solution, we have

$$\max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \psi(x, y) = \min_{x \in \mathcal{X}} \psi(x, \bar{y}) \leq \psi(\bar{x}, \bar{y}), \quad (27)$$

where the last inequality will be an equality if and only if  $\bar{x} \in \min_{x \in \mathcal{X}} \psi(x, \bar{y})$ .

Combining the equality Equation (25) from Sion's minimax theorem with Equations (26) and (27), we find that all five values in Equations (26) and (27) are equal. This implies that  $(\bar{x}, \bar{y})$  is a Nash equilibrium of the game, satisfying  $\bar{x} \in \arg \min_{x \in \mathcal{X}} \psi(x, \bar{y})$  and  $\bar{y} \in \arg \max_{y \in \mathcal{Y}} \psi(\bar{x}, y)$ .

The traditional minimax theorem only provides the identity between values. To give a concrete construction of the minimax solution, we can impose strong convexity of  $\psi$  with respect to  $x$ . Note that we have defined  $\bar{y}$  as a worst-case choice  $\arg \max_{y \in \mathcal{Y}} \psi(x_{\bar{y}}, y)$ , and  $x_{\bar{y}}$  as the unique minimizer of  $\psi(x, \bar{y})$ . From the fact that  $(\bar{x}, \bar{y})$  is a Nash equilibrium of the game, and  $\bar{x} \in \arg \min_{x \in \mathcal{X}} \psi(x, \bar{y})$  (proved in the last paragraphs), along with the uniqueness of  $x_{\bar{y}}$  due to strong convexity, we can conclude that  $\bar{x} = x_{\bar{y}}$ . Therefore,  $x_{\bar{y}}$  is a minimax solution of  $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y)$  and  $(x_{\bar{y}}, \bar{y})$  is a Nash equilibrium.  $\square$

#### A.4 Proof of Theorem 3.2

**THEOREM 3.2 (REGRET OF APS).** Assume that  $f_M(\pi) \in [0, 1]$  for all  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . The regret of Algorithm 2 with  $\eta = \sqrt{2 \log |\Pi| / ((\text{IR}_H(\text{TS}) + 4) \cdot T)}$  and  $T \geq 5 \log |\Pi|$  is bounded by

$$\mathfrak{R}_T \leq \sqrt{2 \log |\Pi| (\text{IR}_H(\text{TS}) + 4) T},$$

where  $\text{IR}_H(\text{TS}) := \sup_v \text{IR}_H(v, v_{\pi^*})$  is the maximal value of IR for TS. Moreover, the regret of Algorithm 2 with any  $\eta \in (0, 1/3]$  is bounded as follows, for all  $T \in \mathbb{N}_+$ ,

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + T \cdot \left( \text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) + 2\eta \right),$$

where  $\text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) := \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \sup_{\mu \in \Delta(\text{conv}(\mathcal{M}))} \mathbb{E}_{\mu, p^{\text{TS}}} [f_M(\pi_M) - f_M(\pi) - \frac{1}{2\eta} D_H^2(M(\pi), \bar{M}(\pi))]$  is DEC of  $\text{conv}(\mathcal{M})$  for the TS strategy  $p^{\text{TS}}(\pi) = \mu(\{M : \pi_M = \pi\})$ .

As stated in the footnote of Theorem 3.2, we define the squared-Hellinger-distance version of IR by

$$\text{IR}_H(v, p) = \frac{\mathbb{E}_{v,p} [(f_M(\pi^*) - f_M(\pi))^2]}{\mathbb{E}_{v,p} [D_H^2(v_{\pi^*|\pi,o}, v_{\pi^*})]}.$$

We prove the following lemma that upper bounds  $\text{AIR}_{q,\eta}(q_t, v_t)$  by  $\text{IR}_H$  and DEC of TS. The main goal of Lemma A.2 is to replace the decision probability  $\pi \sim q$  in  $\text{AIR}_{q,\eta}(q, v)$  (which is the strategy of APS) with the decision probability  $\pi \sim v_{\pi^*}$  (which is the actual strategy of Bayesian TS).

LEMMA A.2 (BOUNDING AIR BY DEC AND IR FOR TS). *Assume that  $f_M(\pi)$  is bounded in  $[0, 1]$  for all  $M, \pi$ . Then for  $\eta \in (0, 1/3]$  and all  $q \in \text{int}(\Delta(\Pi))$ , we have*

$$\text{AIR}_{q,\eta}(q, v) \leq \text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) + 2\eta \leq \frac{\eta}{2} \cdot \text{IR}_H(\text{TS}) + 2\eta.$$

Theorem 3.2 will be a straightforward consequence of Theorem 3.1 and Lemma A.2. By the regret bound (Equation (5)) in Theorem 3.1, for the APS algorithm where  $p_t = q_t$  for all rounds, we have

$$\begin{aligned} \mathfrak{R}_T &\leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t,\eta}(q_t, v_t) \right] \\ &\leq \frac{\log |\Pi|}{\eta} + \frac{\eta T}{2} (\text{IR}_H(\text{TS}) + 4). \end{aligned}$$

In particular, for  $\eta = \sqrt{2 \log |\Pi| / ((\text{IR}_H(\text{TS}) + 4) \cdot T)}$  and  $T \geq 5 \log |\Pi|$  (the condition on  $T$  is to ensure the condition  $\eta \leq 1/3$  in Lemma A.2), we have

$$\mathfrak{R}_T \leq \sqrt{2 \log |\Pi| (\text{IR}_H(\text{TS}) + 4) T}.$$

□

PROOF OF LEMMA A.2: Given a probability measure  $v$ , Denote  $v_{o|\pi^*, \pi} = \mathbb{E}_{M \sim v_{M|\pi^*}} [M(\pi)]$  to be the posterior belief of observation  $o$  conditioned on  $\pi^*$  and  $\pi$ , and  $v_{o|\pi} = \mathbb{E}_{(M, \pi^*) \sim v} [M(\pi)]$  to be posterior belief of  $o$  conditioned solely on  $\pi$ .

Denote the  $|\Pi|$ -dimensional vector  $X, Y$  by

$$\begin{aligned} X(\pi) &= \mathbb{E}_{(M, \pi^*) \sim v} [f_M(\pi^*) - f_M(\pi)], \\ Y(\pi) &= \mathbb{E}_{(M, \pi^*) \sim v} [D_H^2(v_{o|\pi^*, \pi}, v_{o|\pi})]. \end{aligned}$$

Note that the AIR can always be written as

$$\begin{aligned} \text{AIR}_{q,\eta}(q, v) &= \mathbb{E}_{v,q} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{\pi^*|\pi,o}, q) \right] \\ &= \mathbb{E}_{v,q} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{\pi^*|\pi,o}, v_{\pi^*}) - \frac{1}{\eta} \text{KL}(v_{\pi^*}, q) \right] \\ &= \mathbb{E}_{v,q} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \text{KL}(v_{o|\pi^*, \pi}, v_{o|\pi}) - \frac{1}{\eta} \text{KL}(v_{\pi^*}, q) \right], \end{aligned} \quad (28)$$

where the first equality is the definition of AIR; the second equality is because the expectation of posterior is equal to prior; and the third equality is due to the symmetry property of mutual information.

By Equation (28) we have that

$$\begin{aligned}
& \text{AIR}_{q,\eta}(q, v) \\
& \leq \mathbb{E}_{v,q} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} D_H^2(v_{o|\pi^*,\pi}, v_{o|\pi}) \right] - \frac{1}{\eta} \text{KL}(v_\pi^*, \rho_t) \\
& = \langle q, X \rangle - \frac{1}{2\eta} \text{KL}(v_{\pi^*}, q) - \frac{1}{\eta} \langle q, Y \rangle - \frac{1}{2\eta} \text{KL}(v_{\pi^*}, q) \\
& \leq \langle v_{\pi^*}, X \rangle + 2\eta - \frac{1}{\eta} \langle q, Y \rangle - \frac{1}{2\eta} \text{KL}(v_{\pi^*}, q) \\
& \leq \langle v_{\pi^*}, X \rangle + 2\eta - \frac{1}{\eta} \langle q, Y \rangle - \frac{1}{2\eta} D_H^2(v_{\pi^*}, q) \\
& \leq \langle v_{\pi^*}, X \rangle - \frac{(1-\eta)}{(1+\eta)\eta} \langle v_{\pi^*}, Y \rangle + 2\eta \\
& \leq \mathbb{E}_{v,v_{\pi^*}} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{2\eta} D_H^2(v_{o|\pi^*,\pi}, v_{o|\pi}) \right] + 2\eta,
\end{aligned} \tag{29}$$

where the first inequality is by Lemma 4.8; the second inequality is by Lemma 4.11 and the fact  $f_M(\pi) \in [0, 1]$  for all  $M \in \mathcal{M}$  and  $\pi \in \Pi$ ; the third inequality is by Lemma 4.8; the fourth inequality is a consequence of Lemma 4.10, the AM-GM inequality, and  $\eta \leq 1$ ; and the last inequality uses the condition  $\eta \leq \frac{1}{3}$ .

Therefore, we have

$$\begin{aligned}
& \text{AIR}_{q,\eta}(q, v) \\
& \leq \mathbb{E}_{v,v_{\pi^*}} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{2\eta} D_H^2(v_{o|\pi^*,\pi}, v_{o|\pi}) \right] + 2\eta \\
& \leq \text{DEC}_{2\eta}^{\text{TS}}(\text{conv}(\mathcal{M})) + 2\eta \\
& \leq \frac{\eta}{2} \cdot \text{IR}_H(\text{TS}) + 2\eta,
\end{aligned}$$

where the first inequality is by Equation (29); the second inequality follows the same proof as in Lemma 2.7 (see below Lemma 2.7); and the last inequality is by the definition of  $\text{IR}_H(\text{TS})$  defined in Theorem 3.2 and the AM-GM inequality.  $\square$

## A.5 Proof of Theorem 3.3

**THEOREM 3.3 (REGRET OF AMS).** *For a finite decision space  $\Pi$  and a compact model class  $\mathcal{M}$ , the regret of Algorithm 3 with any  $\eta > 0$  is always bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \text{DEC}_\eta^{\text{KL}}(\text{conv}(\mathcal{M})) \cdot T.$$

In particular, the regret of Algorithm 3 with  $\eta = 2\sqrt{\log |\Pi| / (\text{IR}(\text{IDS}) \cdot T)}$  and all  $T \in \mathbb{N}_+$  is bounded by

$$\mathfrak{R}_T \leq \sqrt{\log |\Pi| \cdot \text{IR}(\text{IDS}) \cdot T},$$

where  $\text{IR}(\text{IDS}) := \sup_v \inf_p \text{IR}(v, p)$  is the maximal IR of IDS.

**PROOF OF THEOREM 3.3:** The proof of Theorem 3.3 is straightforward. For the DEC upper bound, we have

$$\begin{aligned}\mathfrak{R}_T &\leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right] \\ &\leq \frac{\log |\Pi|}{\eta} + \text{DEC}_{\eta}^{\text{KL}}(\text{conv}(\mathcal{M})) \cdot T,\end{aligned}$$

where the first inequality is by Theorem 3.1, and the second inequality is by Lemma 2.7 and an application of the minimax theorem (Lemma 4.2) to swap sup and inf. Moreover, for the IR upper bound, we have

$$\begin{aligned}\mathfrak{R}_T &\leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right] \\ &\leq \frac{\log |\Pi|}{\eta} + \frac{\eta}{4} \text{IR}(\text{IDS}) \cdot T,\end{aligned}$$

where the first inequality is by Theorem 3.1, and the second inequality is by Lemma 2.5 and the definition of  $\text{IR}(\text{IDS})$  in Theorem 3.3. In particular, taking  $\eta = 2\sqrt{\log |\Pi| / (\text{IR}(\text{IDS}) \cdot T)}$  we have

$$\mathfrak{R}_T \leq \sqrt{\log |\Pi| \cdot \text{IR}(\text{IDS}) \cdot T}.$$

□

## A.6 Proof of Theorem 3.4

**THEOREM 3.4 (GENERIC REGRET BOUND USING APPROXIMATE MAXIMIZERS).** Let  $\Pi$  be a finite decision space and  $\mathcal{M}$  a compact model class. Fix any algorithm  $\text{ALG}$  that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and maintain a sequence of beliefs  $v_1, \dots, v_t, \dots$  where  $q_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}} \in \text{int}(\Delta(\Pi))$  for all rounds. Then for all  $T \in \mathbb{N}_+$ ,

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{AIR}_{q_t, \eta}(p_t, v_t) + \sup_{v^*} \left( \frac{\partial \text{AIR}_{q_t, \eta}(p_t, v)}{\partial v} \Big|_{v=v_t}, v^* - v_t \right) \right) \right].$$

The proof of Theorem 3.4 has two steps. The first step is regret decomposition, which is almost the same with the proof of Theorem 3.1, and we have illustrate this step in Section 5.1. The second step is to use the identity for arbitrary algorithmic beliefs (Lemma 5.3).

**PROOF OF THEOREM 3.4:** By Equation (14), and the first equality and the inequality ( $\diamond$ ) in Equation (15) (without the last equality), we have that for every  $\bar{\pi} \in \Pi$ ,

$$\mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\bar{\pi}) - \sum_{t=1}^T f_{M_t}(\pi_t) \right] - \frac{\log |\Pi|}{\eta} \leq \mathbb{E} \left[ \sum_{t=1}^T \sup_{M, \bar{\pi}} \mathbb{E}_{t-1} \left[ f_M(\bar{\pi}) - f_M(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t, o_t}}(\bar{\pi})}{q_t(\pi^*)} \right] \right],$$

which implies

$$\mathfrak{R}_T \leq \frac{\log |\Pi|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \sup_{M, \bar{\pi}} \mathbb{E}_{t-1} \left[ f_M(\bar{\pi}) - f_M(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t, o_t}}(\bar{\pi})}{q_t(\pi^*)} \right] \right].$$

By Lemma 5.3, we have

$$\sup_{M, \bar{\pi}} \mathbb{E}_{t-1} \left[ f_M(\bar{\pi}) - f_M(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t, o_t}}(\bar{\pi})}{q_t(\pi^*)} \right] = \text{AIR}_{q_t, \eta}(p_t, v_t) + \left\langle \frac{\partial \text{AIR}_{q_t, \eta}(p_t, v)}{\partial v} \Big|_{v=v_t}, \mathbb{1}(M, \bar{\pi}) - v_t \right\rangle.$$

Combining the above two inequalities proves Theorem 3.4. □

**LEMMA 5.3 (IDENTITY FOR ARBITRARY ALGORITHMIC BELIEF AND ANY ENVIRONMENT).** *Given  $q \in \text{int}(\Delta(\Pi))$ ,  $\eta > 0$ ,  $p \in \Delta(\Pi)$ , and an arbitrary algorithmic belief  $\hat{v} \in \Delta(\mathcal{M} \times \Pi)$  selected by the agent and an arbitrary environment  $(M, \bar{\pi})$  specified by the nature. Then we have*

$$\mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ f_M(\bar{\pi}) - f_M(\pi) - \frac{1}{\eta} \log \frac{\hat{v}_{\pi^*|_{\pi,o}}(\bar{\pi})}{q(\bar{\pi})} \right] = \text{AIR}_{q,\eta}(p, \hat{v}) + \left\langle \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial v} \Bigg|_{v=\hat{v}}, \mathbb{1}(M, \bar{\pi}) - \hat{v} \right\rangle,$$

where  $\mathbb{1}(M, \bar{\pi})$  is a vector in simplex  $\Delta(\mathcal{M} \times \Pi)$  where all the weights are given to  $(M, \bar{\pi})$ .

**PROOF OF LEMMA 5.3:** Given a fixed  $p$ , we define

$$B(v, Q) = \mathbb{E}_{(M, \pi^*) \sim v, \pi \sim p} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} \log \frac{Q[\pi, o](\pi^*)}{q(\pi^*)} \right]$$

as in Appendix A.2. We have that  $B(v, Q)$  is a strongly convex function in  $Q$  and is a linear function in  $v$ . For every  $v$ , the optimal choice of the mapping  $Q : \Pi \times \mathcal{O} \rightarrow \Delta(\Pi)$  will be the marginal posterior mapping  $Q = v_{\pi^*|.,.}$ , which is specified by

$$Q[\pi, o] = v_{\pi^*|_{\pi,o}}, \quad \forall \pi \in \Pi, o \in \mathcal{O}.$$

And the value of the  $B$  function at this  $(v, Q) = (v, v_{\pi^*|.,.})$  will be

$$B(v, v_{\pi^*|.,.}) = \text{AIR}_{q,\eta}(p, v).$$

Since  $B$  is a linear function with respect to  $v$ , we have that for any other  $v^*$ ,

$$\begin{aligned} B(v^*, v_{\pi^*|.,.}) &= B(v, v_{\pi^*|.,.}) + \left\langle \frac{\partial B(v, Q)}{\partial v} \Bigg|_{Q=v_{\pi^*|.,.}}, v^* - v \right\rangle \\ &= \text{AIR}_{q,\eta}(p, v) + \left\langle \frac{\partial B(v, Q)}{\partial v} \Bigg|_{Q=v_{\pi^*|.,.}}, v^* - v \right\rangle. \end{aligned} \tag{30}$$

Now we use Danskin's theorem (Lemma 4.3), which establishes equivalence between the gradient of the optimized objective and the partial derivative at the optimizer. We refer to Appendix 4.3 for the background of Danskin's theorem. By Danskin's theorem (Lemma 4.3), we have that

$$\frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial v} = \frac{\partial B(v, Q)}{\partial v} \Bigg|_{Q=v_{\pi^*|.,.}}. \tag{31}$$

Taking Equations (31) to (30) proves Lemma 5.3. □

## 1.7 Proof of Theorem A.1

In this section we prove Theorem A.1, which is a more general extension to Theorem 3.4: Theorem A.1 applies to general Bregman divergence while Theorem 3.4 is stated with the KL divergence.

**THEOREM A.1 (USING GENERAL BREGMAN DIVERGENCE).** *Assume  $\Psi : \Delta(\Pi) \rightarrow \mathbb{R} \cup \infty$  is Legendre and has bounded diameter. Let  $\Pi$  be a finite decision space and  $\mathcal{M}$  a compact model class. Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and maintain a sequence of beliefs  $v_1, \dots, v_t, \dots$  where  $q_{t+1} = (v_t)_{\pi^*|_{\pi_t, o_t}} \in \text{int}(\Delta(\Pi))$  for all rounds. Then for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\text{diam}(\Psi)}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{AIR}_{q_t, \eta}^\Psi(p_t, v_t) + \sup_{v^*} \left\langle \frac{\partial \text{AIR}_{q_t, \eta}^\Psi(p_t, v)}{\partial v} \Bigg|_{v=v_t}, v^* - v_t \right\rangle \right) \right].$$

PROOF OF THEOREM A.1: Recall the definition of generalized AIR in Appendix A.1.1, where  $\Psi$  is a convex Legendre function:

$$\text{AIR}_{q,\eta}^{\Psi}(p, v) = \mathbb{E}_{v,p} \left[ f_M(\pi^*) - f_M(\pi) - \frac{1}{\eta} D_{\Psi}(v_{\pi^*|\pi,o}, v_{\pi^*}) - \frac{1}{\eta} D_{\Psi}(v_{\pi^*}, q) \right].$$

Similar to the  $B$  function (Equation (24)) we use in Appendix A.2 and Appendix A.6, given the decision probability  $\pi \sim p$ , for all  $v \in \Delta(\mathcal{M} \times \Pi)$  such that  $v_{\pi^*} \in \text{int}(\Delta(\Pi))$  and  $R : \Pi \times \mathcal{O} \rightarrow \mathbb{R}^{|\Pi|}$ , we define

$$B(v, R) = \mathbb{E}_{v,p} \left[ f_M(\pi^*) - f_M(\pi) + \frac{1}{\eta} \langle \nabla \Psi(q) - R[\pi, o], \mathbf{1}(\pi^*) - q \rangle + \frac{1}{\eta} D_{\Psi^*}(R[\pi, o], \nabla \Psi(q)) \right],$$

where  $\mathbf{1}(\pi^*)$  is the vector whose  $\pi^*$ -coordinate is 1 but all other coordinates are 0. Note that  $B(v, R)$  is a strongly convex function with respect to  $R$ . We have the following formula:

$$\text{AIR}_{q,\eta}^{\Psi}(p, v) = B(v, \nabla \Psi(v_{\pi^*|\cdot})) = \inf_R B(v, R),$$

where the first equality is by Lemma 4.7 and the property (b) in Lemma 4.6, and the second equality is because that the first-order optimal condition implies that the minimizer of  $\inf_R B(v, R)$  will be  $R = \nabla \Psi(v_{\pi^*|\cdot})$ , which is specified by

$$R[\pi, o] = \nabla \Psi(v_{\pi^*|\pi,o}), \quad \forall \pi \in \Pi, o \in \mathcal{O}.$$

By Danskin's theorem (Lemma 4.3), we further have

$$\frac{\partial \text{AIR}_{q,\eta}^{\Psi}(p, v)}{\partial v} = \frac{\partial B(v, R)}{\partial v} \Big|_{R=\nabla \Psi(v_{\pi^*|\cdot})}.$$

By the above identity and the linearity of  $B(v, Q)$  with respect to  $v$ , we have

$$B(v^*, \nabla \Psi(v_{\pi^*|\cdot})) = \text{AIR}_{q,\eta}^{\Psi}(p, v) + \left\langle \frac{\partial \text{AIR}_{q,\eta}^{\Psi}(p, v)}{\partial v}, v^* - v \right\rangle, \quad \forall v^*. \quad (32)$$

Following the very similar steps in proving Theorem 3.4, we are able to prove Theorem A.1. By Lemma 4.7 and the property (b) in Lemma 4.6, for every  $\bar{\pi} \in \Pi$  we have

$$\begin{aligned} & \frac{1}{\eta} \langle \nabla \Psi(q_{t+1}) - \nabla \Psi(q_t), \mathbf{1}(\bar{\pi}) - q_t \rangle - \frac{1}{\eta} D_{\Psi^*}(\nabla \Psi(q_{t+1}), \nabla \Psi(q_t)) \\ &= \frac{1}{\eta} \langle \nabla \Psi(q_{t+1}) - \nabla \Psi(q_t), \mathbf{1}(\bar{\pi}) - q_t \rangle - \frac{1}{\eta} D_{\Psi}(q_t, q_{t+1}) \\ &= D_{\Psi}(\mathbf{1}(\bar{\pi}), q_t) - D_{\Psi}(\mathbf{1}(\bar{\pi}), q_{t+1}). \end{aligned}$$

Given an arbitrary algorithm and an arbitrary algorithmic belief sequence, we set the reference probability  $q_{t+1} = (v_t)_{\pi^*|\pi_t, o_t}$  to aid our analysis. By the above equation we have

$$\mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ \frac{1}{\eta} \left\langle \nabla \Psi((v_t)_{\pi^*|\pi,o}) - \Psi(q_t), \mathbf{1}(\pi^*) - q_t \right\rangle - \frac{1}{\eta} D_{\Psi^*}(\nabla \Psi((v_t)_{\pi^*|\pi,o}), \nabla \Psi(q_t)) \right] \right] \leq \frac{\text{diam}(\Psi)}{\eta}. \quad (33)$$

Subtracting the telescope sum 33 from regret, we have

$$\begin{aligned}
& \mathfrak{R}_T - \frac{\text{diam}(\Psi)}{\eta} \\
& \leq \mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\pi^*) - \sum_{t=1}^T f_{M_t}(\pi_t) \right] - \frac{\text{diam}(\Psi)}{\eta} \\
& \leq \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ f_{M_t}(\pi^*) - f_{M_t}(\pi_t) + \frac{1}{\eta} \langle \nabla \Psi(q_t) - \nabla \Psi((v_t)_{\pi^*|\pi, o}), \mathbb{1}(\pi^*) - q_t \rangle + \frac{1}{\eta} D_{\psi^*}(\nabla \Psi((v_t)_{\pi^*|\pi, o}), \nabla \Psi(q_t)) \right] \right] \\
& \leq \mathbb{E} \left[ \sum_{t=1}^T \sup_{v^*} B_t(v^*, \nabla \Psi((v_t)_{\pi^*|\pi, o})) \right], \tag{34}
\end{aligned}$$

where the first inequality is by the definition of regret; the second inequality is by Equation (14); and the third inequality is by taking the worst-case environment at every round and the definition of the  $B$  function. Finally, taking the equivalent characterization (Equation (32)) for the  $B$  function into Equation (34) proves Theorem A.1.  $\square$

## 1.8 Proof of Theorem 3.5

**THEOREM 3.5 (GENERIC DYNAMIC REGRET BOUND)** *Given a finite decision space  $\Pi$  and a compact model class  $\mathcal{M}$ . Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and  $\gamma \in (0, 1)$ , generate a sequence of beliefs  $v_1, \dots, v_t, \dots$  where  $v_t$  solves  $\sup_{v \in \Delta(\mathcal{M} \times \Pi)} \text{AIR}_{q_t, \eta}(p_t, v)$ , and set  $q_{t+1} := (1 - \gamma)(v_t)_{\pi^*|\pi_t, o_t} + \gamma \text{Unif}(\Pi)$ . Then for all  $T \in \mathbb{N}_+$ , the dynamic regret of ALG is bounded by*

$$\mathfrak{R}_T^D(\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi| + \frac{T\gamma}{1-\gamma} + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right] \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right],$$

where  $\sum_{t=2}^T \log \frac{q_t(\pi_{t+1}^*)}{q_t(\pi_t^*)}$  measures the total shift in the comparator sequence  $\{\pi_t^*\}_{t=1}^T$  up to round  $T$ . Specifically, denote

$$D_T = \#\{t : \pi_t^* \neq \pi_{t+1}^*, 1 \leq t < T\}$$

as the number of changes in the optimal decision before round  $T$ , and set  $\gamma = 1/T$ , then the dynamic regret is bounded by

$$\mathfrak{R}_T^D(\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi|(D_T \log T + 1) + \frac{T}{T-1} \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right].$$

**PROOF OF THEOREM 3.5:** For every comparator sequence  $\{\pi_t^*\}_{t=1}^T$ , we have

$$\sum_{t=1}^T \left[ \log \frac{q_{t+1}(\pi_t^*)}{q_t(\pi_t^*)} \right] = \log \frac{q_{T+1}(\pi_T^*)}{q_1(\pi_1^*)} + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right] \leq \log |\Pi| + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right]. \tag{35}$$

By subtracting the additive elements on the left-hand side of Equation (35) (divided by  $\eta$ ) from the per-round regrets against  $\{\pi_t^*\}_{t=1}^T$ , we obtain

$$\begin{aligned}
& \mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\pi_t^*) - \sum_{t=1}^T f_{M_t}(\pi_t) - \frac{1}{\eta} \cdot \sum_{t=1}^T \log \frac{q_{t+1}(\pi_t^*)}{q_t(\pi_t^*)} \right] \\
& = \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ f_{M_t}(\pi_t^*) - f_{M_t}(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|\pi_t, o_t}(\pi_t^*)}{q_t(\pi_t^*)} \right] + \frac{1}{\eta} \sum_{t=1}^T \mathbb{E}_{t-1} \left[ \log \frac{(v_t)_{\pi^*|\pi_t, o_t}(\pi_t^*)}{(1-\gamma)(v_t)_{\pi^*|\pi_t, o_t}(\pi_t^*) + \frac{\gamma}{|\Pi|}} \right] \right]
\end{aligned}$$

$$\begin{aligned}
&\leq \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ f_{M_t}(\pi_t^*) - f_{M_t}(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t, o_t}}(\pi_t^*)}{q_t(\pi_t^*)} \right] \right] + \frac{T\gamma}{\eta(1-\gamma)} \\
&\stackrel{(\diamond)}{\leq} \mathbb{E} \left[ \sum_{t=1}^T \sup_{M, \pi_t^*} \mathbb{E}_{t-1} \left[ f_M(\pi_t^*) - f_M(\pi_t) - \frac{1}{\eta} \log \frac{(v_t)_{\pi^*|_{\pi_t, o_t}}(\pi_t^*)}{q_t(\pi_t^*)} \right] \right] + \frac{T\gamma}{\eta(1-\gamma)} \\
&\stackrel{(*)}{=} \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right] + \frac{T\gamma}{\eta(1-\gamma)}, \tag{36}
\end{aligned}$$

where the first equality is by taking  $q_{t+1} := (1-\gamma)(v_t)_{\pi^*|_{\pi_t, o_t}} + \gamma \text{Unif}(\Pi)$ , and the conditional expectation notation is introduced in Section 2.1; the first inequality is by the monotonicity of  $\log \frac{p}{(1-\gamma)p+\gamma/|\Pi|}$  with respect to  $p \in [0, 1]$ , and the standard derivation

$$-\log(1 - \gamma(1 - 1/|\Pi|)) \leq \frac{\gamma(1 - 1/|\Pi|)}{1 - \gamma(1 - 1/|\Pi|)} \leq \frac{\gamma}{1 - \gamma};$$

the inequality  $(\diamond)$  is by taking supremum at each rounds; and the last equality  $(*)$  in Equation (36) is by Lemma 5.2, an important identity to be explained in Section 5.2, which is derived from the fact that the pair of maximizer  $v_t$  and posterior functional is a Nash equilibrium of a convex-concave function. We note that the second-to-last formula  $(\diamond)$  is true without imposing any restrictions on  $v_t$ ; only the last formula  $(*)$  relies on the fact that  $v_t$  maximizes AIR.

Combining Equations (36) and (35), we obtain the following inequality for every  $\bar{\pi} \in \Pi$ :

$$\mathbb{E} \left[ \sum_{t=1}^T f_{M_t}(\bar{\pi}) - \sum_{t=1}^T f_{M_t}(\pi_t) \right] - \frac{1}{\eta} \left( \log |\Pi| + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right] \right) \leq \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right] + \frac{T\gamma}{\eta(1-\gamma)}.$$

By taking the supremum over  $\bar{\pi} \in \Pi$  on the left-hand side of the inequality, we are able to prove Theorem 3.5:

$$\mathfrak{R}_T(\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi| + \frac{T\gamma}{1-\gamma} + \sum_{t=2}^T \left[ \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \right] \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right].$$

Finally, from the above dynamic regret bound, it is easy to derive

$$\mathfrak{R}_T^D(\{\pi_t^*\}_{t=1}^T) \leq \frac{1}{\eta} \left( \log |\Pi|(D_T \log T + 1) + \frac{T}{T-1} \right) + \mathbb{E} \left[ \sum_{t=1}^T \text{AIR}_{q_t, \eta}(p_t, v_t) \right],$$

where  $D_T = \#\{t : \pi_t^* \neq \pi_{t+1}^*, 1 \leq t < T\}$  is the number of changes in the optimal decision before round  $T$ . This is because the shift term  $\sum_{t=2}^T \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)}$  in Equation (10) can be trivially upper-bounded by

$$\sum_{t=2}^T \log \frac{q_t(\pi_{t-1}^*)}{q_t(\pi_t^*)} \leq D_T \cdot \log \frac{|\Pi|}{\gamma}.$$

□

Moreover, combining this proof with the proof of Theorem 3.4 lets us extend the dynamic-regret bound to any sequence of beliefs  $v_t$ —for example, to approximate rather than exact AIR maximizers. Specifically, Fix any algorithm ALG that produces decision probability  $p_1, \dots, p_T$ . Initialize  $q_1 = \text{Unif}(\Pi)$  and  $\gamma = 1/T$ , generate an arbitrary sequence of beliefs  $v_1, \dots, v_t, \dots$ , and set  $q_{t+1} := (1-\gamma)(v_t)_{\pi^*|_{\pi_t, o_t}} + \gamma \text{Unif}(\Pi)$ . Then for all  $T \in \mathbb{N}_+$ , the dynamic regret of ALG is

bounded by

$$\begin{aligned} \mathfrak{R}_T^D(\{\pi_t^*\}_{t=1}^T) &\leq \frac{1}{\eta} \left( \log |\Pi| (D_T \log T + 1) + \frac{T}{T-1} \right) \\ &+ \mathbb{E} \left[ \sum_{t=1}^T \left( \text{AIR}_{q_t, \eta}(p_t, v_t) + \sup_{v^*} \left( \frac{\partial \text{AIR}_{q_t, \eta}(p_t, v)}{\partial v} \Big|_{v=v_t}, v^* - v_t \right) \right) \right]. \end{aligned} \quad (37)$$

We will leverage this more flexible corollary in Appendix 2.7.

## 1.9 Information Ratio Bounds Using the Squared Hellinger Distance

We use the squared Hellinger distance to define  $\text{IR}_H$  (instead of KL as in the definition (Equation (1)) of IR) in Theorem 3.2 (regret of APS), because technically a bounded divergence is more convenient to apply change of measure arguments. Note that there is no essential difference between the definitions of IR and  $\text{IR}_H$  in practical applications, since all currently known bounds on the IR hold for the stronger definition  $\text{IR}_H$  with added absolute constants. This fact is illustrated in the following lemma.

**LEMMA 1.3 (INFORMATION RATIO BOUNDS USING THE SQUARED HELLINGER DISTANCE).** *Define IR for the squared Hellinger distance by*

$$\text{IR}_H(v, p) = \frac{(\mathbb{E}_{v,p}[f_M(\pi^*) - f_M(\pi)])^2}{\mathbb{E}_{v,p}[D_H^2(v_{\pi^*|\pi,o}, v_{\pi^*})]},$$

and define IR for TS by  $\text{IR}_H(\text{TS}) = \sup_v \text{IR}_H(v, v_{\pi^*})$ . Then for  $K$ -armed bandits, we have  $\text{IR}_H(\text{TS}) \leq 4K$ ; for problems with full information, we have  $\text{IR}_H(\text{TS}) \leq 4$ ; and for  $d$ -dimensional linear bandits, we have  $\text{IR}_H(\text{TS}) \leq 4d$ .

**PROOF OF LEMMA 1.3:** We refer to [45] for the standard IR bounds using the KL divergence for  $K$ -armed bandits (Proposition 3 in [45]), problems with full information (Proposition 4 in [45]), and  $d$ -dimensional linear bandits (Proposition 5 in [45]). A common step in these proof is applying Pinsker's inequality for the KL divergence to convert the KL divergence in the IR into the square distance. For any  $h : \mathcal{X} \rightarrow [0, 1]$ , Pinsker's inequality is expressed as follows:

$$(\mathbb{E}_{\mathbb{P}}[h(X)] - \mathbb{E}_{\mathbb{Q}}[h(X)])^2 \leq \frac{1}{2} \text{KL}(\mathbb{P}, \mathbb{Q}). \quad (38)$$

Now by Lemma 4.10, we establish a Pinsker-type inequality using the squared Hellinger distance: for any  $h : \mathcal{X} \rightarrow [0, 1]$ ,

$$(\mathbb{E}_{\mathbb{P}}[h(X)] - \mathbb{E}_{\mathbb{Q}}[h(X)])^2 \leq 4D_H^2(\mathbb{P}, \mathbb{Q}). \quad (39)$$

We can establish Lemma 1.3 by closely following Propositions 3–5 in [45] line by line. The only modification in the new proof is replacing the KL divergence with the squared Hellinger distance and substituting Pinsker's inequality (Equation (38)) with the new Pinsker-type inequality (Equation (39)). The final results are nearly identical, differing only in absolute constants.  $\square$

## 2 Details and Proofs for Bandit Problems

The section is organized as follows. In Appendix 2.1, we demonstrate the parameterization of AIR for structured bandit problems with Bernoulli rewards. In Appendix 2.2, we derive and analyze Simplified APS for Bernoulli MAB (Algorithm 4). In Appendix 2.4, we showcase the parameterization of AIR for structured bandit problems with Gaussian rewards. In Appendix 2.5, we present Simplified APS for Gaussian linear bandits (Algorithm 5). It is important to note that all the calculations in Appendices 2.1 to 2.5 can be placed within the broader framework of the exponential family, as

explained in Appendix 2.6. Within this unified framework, the calculations are simplified, and the intuition becomes more immediate. In particular, minimizing the derivatives has the immediate interpretation of making the natural parameters and the log-partition functions nearly “mean-zero” across different conditions. While Appendices 2.1 to 2.5 provide concrete examples, readers who are comfortable with generic derivations can also start by reading Appendix 2.6 first. Lastly, we prove a supporting lemma to analyze Algorithm 4 in Appendix 2.3.

## 2.1 Concave Parameterization with Bernoulli Reward

We consider Bernoulli structured bandit with an action set  $\Pi \subset \mathbb{R}^d$  and a mean reward function class  $\mathcal{F} \subset (\Pi : \mapsto [0, 1])$  that is convex. (As discussed in the beginning of Section 4, every bounded-reward bandit problem can equivalently be reduced to a Bernoulli bandit problem.) For simplicity we make the standard assumption that  $\Pi$  is finite, which can be removed using standard discretization and covering argument. The goal here is to make the computation complexity to be independent of the size of model class  $\mathcal{M}$ , but only depends on  $|\Pi|$ . The general principle to achieve this goal is as follows. For each possible value of  $\pi^*$ , we assign an “effective model”  $M_{\pi^*}$  to  $\pi^*$  so that the optimization problem (Equation (16)) reduces to selecting those  $|\Pi|$  “effective models,” as well as the probability distribution over them.

We introduce the following parameterization:  $\forall a, a^* \in \Pi$ ,

$$\begin{aligned}\theta_{a^*}(a) &= \mathbb{E}[r(a)|\pi^* = a^*], \\ \alpha(a^*) &= v_{\pi^*}(a^*), \\ \beta_{a^*}(a) &= \alpha(a^*) \cdot \theta_{a^*}(a).\end{aligned}\tag{40}$$

Then we have represent AIR by  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$ :

$$\begin{aligned}\text{AIR}_{q,\eta}(p, v) &= \sum_{\pi^* \in \Pi} \beta_{\pi^*}(\pi^*) - \sum_{\pi^*, \pi \in [K]} p(\pi) \beta_{\pi^*}(\pi) \\ &\quad - \frac{1}{\eta} \sum_{\pi^*, \pi \in [K]} p(\pi) \alpha(\pi^*) \text{kl}\left(\frac{\beta_{\pi^*}(\pi)}{\alpha(\pi^*)}, \sum_{\pi^* \in \Pi} \beta_{\pi^*}(\pi)\right) - \frac{1}{\eta} \text{KL}(\alpha, q),\end{aligned}\tag{41}$$

and the constraint of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$  is that the functions parameterized by  $\theta_{a^*}$  belong to the mean reward function class  $\mathcal{F}$ . We know the constraint set of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$  to be convex because the convexity of perspective function.

Now we want to prove that the AIR objective in the maximization problem (Equation (16)) is concave. Using the definition (Equation (24)) of  $B$  function in Appendix A.2 (this  $B$  function will also be the key to simplify our calculation of derivatives in the next section), we have

$$\begin{aligned}B(v, Q) &= \mathbb{E}_{(M, \pi^*) \sim v, \pi \sim p, o \sim M(\pi)} \left[ f_M(\pi^*) - f_M(\pi) + \frac{1}{\eta} \log \frac{q(\pi^*)}{Q[\pi, o](\pi^*)} \right] \\ &= \sum_{\pi^*} \beta_{\pi^*}(\pi^*) - \sum_{\pi^*, \pi} p(\pi) \beta_{\pi^*}(\pi) + \frac{1}{\eta} \sum_{\pi, \pi^*} p(\pi) \beta_{\pi^*}(\pi) \log \frac{q(\pi^*)}{Q[\pi, 1](\pi^*)} \\ &\quad + \frac{1}{\eta} \sum_{\pi, \pi^*} p(\pi) (\alpha(\pi^*) - \beta_{\pi^*}(\pi^*)) \log \frac{q(\pi^*)}{Q[\pi, 0](\pi^*)}.\end{aligned}\tag{42}$$

This means that after parameterizing  $v$  with  $\alpha$  and  $\{\beta_{\pi^*}\}_{\pi^* \in \Pi}$ ,  $B(v, Q)$  will be a linear function of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$ . By the relationship between the  $B$  function and AIR showed in Appendix 2.1, we have

$$\text{AIR}_{q,\eta}(p, v) = \inf_Q B(v, Q),$$

which implies that AIR will be a concave function of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$ . So the optimization problem to maximize AIR is a convex optimization problem, whose computational complexity will be poly-logarithmic to the cardinality of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$ . As a result, the computational complexity to maximize AIR is polynomial in  $|\Pi|$  and does not depends on cardinality of the model class. This discussion shows that we give finite-running-time algorithm with computational complexity  $\text{poly}(e^d)$  even when the cardinality of model class is double-exponential. Still, the computation is only efficient for simple problems such as  $K$ -armed bandits, but we also give efficient algorithm for linear bandits in Appendix 6.2.

## 2.2 Simplified APS for Bernoulli MAB

For Bernoulli  $K$ -armed bandits discussed in Section 4.1, we give the details about how to use first-order optimality conditions to derive Algorithm 4. For Bernoulli  $K$ -armed bandits, the decision space  $\Pi = [K] = \{1, \dots, K\}$  is a finite set of  $K$  actions. We take the parameterization (Equation (40)) introduced in Appendix 2.1, and utilize the expression Equation (41) of AIR and the expression Equation (42) of the  $B$  function. By Equation (41), we have the following concave parameterization of AIR by the  $K(K + 1)$ -dimensional vector  $(\alpha, \beta) = (\alpha, \beta_1, \dots, \beta_K)$ :

$$\begin{aligned} \text{AIR}_{q,\eta}(p, v) &= \sum_{i \in [K]} \beta_i(i) - \sum_{i,j \in [K]} p(j)\beta_i(j) \\ &\quad - \frac{1}{\eta} \sum_{i,j \in [K]} p(j)\alpha(i)\text{kl}\left(\frac{\beta_i(j)}{\alpha(i)}, \sum_{i \in [K]} \beta_i(j)\right) - \frac{1}{\eta} \text{KL}(\alpha, q), \end{aligned} \quad (43)$$

By Equation (42), we have the following expression of the  $B$  function:

$$\begin{aligned} B(v, Q) &= \sum_i \beta_i(i) - \sum_{j,i} p(j)\beta_i(j) + \frac{1}{\eta} \sum_{j,i} p(j)\beta_i(j) \log \frac{q(i)}{Q[j, 1](i)} \\ &\quad + \frac{1}{\eta} \sum_{j,i} p(j)(\alpha(i) - \beta_i(i)) \log \frac{q(i)}{Q[j, 0](i)}, \end{aligned} \quad (44)$$

which satisfies

$$\text{AIR}_{q,\eta}(p, v) = \inf_Q B(v, Q),$$

and will serve the role to simplify our following calculation of the derivatives. We denote  $v_{\pi^*|j,1}(i)$  as the marginal posterior  $\mathbb{P}(\pi^* = i | \pi = j, o = 1)$  when the underlying probability measure is  $v$ .

**2.2.1 Derivation of Algorithm 4.** By Equation (44) and Lemma 4.3 (i.e., using Danskin's theorem and the bivariate function (Equation (44)) to calculate the derivatives is easier than directly calculating the derivatives of the AIR parameterization (Equation (43))), we have for each  $i \in [K]$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \beta_i(i)} &= \frac{\partial B(v, Q)}{\partial \beta_i(i)} \Big|_{Q=v_{\pi^*|:,i}} \\ &= (1 - p(i)) - \frac{1}{\eta} p(i) \left( \log v_{\pi^*|i,1}(i) - \log v_{\pi^*|i,0}(i) \right). \end{aligned} \quad (45)$$

And for every  $i \neq j \in [K]$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \beta_i(j)} &= \frac{\partial B(v, Q)}{\partial \beta_i(j)} \Big|_{Q=v_{\pi^*|j,\cdot}} \\ &= -p(j) - \frac{1}{\eta} p(j) (\log v_{\pi^*|j,1}(i) - \log v_{\pi^*|j,0}(i)). \end{aligned} \quad (46)$$

Lastly, for each  $i \in [K]$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(i)} &= \frac{\partial B(v, Q)}{\partial \alpha(i)} \Big|_{Q=v_{\pi^*|j,\cdot}} \\ &= \frac{1}{\eta} \sum_{j \in [K]} p(j) (\log q(i) - \log v_{\pi^*|j,0}(i)). \end{aligned} \quad (47)$$

We let the derivatives in Equations (45) and (46) be zero, which means that the derivatives with respect to all coordinates of  $\beta$  are zero. We have for all  $j \in [K]$

$$\begin{aligned} \log \frac{v_{\pi^*|j,1}(j)}{v_{\pi^*|j,0}(j)} &= \frac{\eta}{p(j)} - \eta, \\ \log \frac{v_{\pi^*|i,1}(j)}{v_{\pi^*|i,0}(j)} &= \log \frac{1 - v_{\pi^*|j,1}(j)}{1 - v_{\pi^*|j,0}(j)} = -\eta, \quad \forall i \neq j. \end{aligned} \quad (48)$$

Solving the above two equations in Equation (48) we obtain closed form solutions for  $v_{\pi^*}(j|j, 1)$  and  $v_{\pi^*}(j|j, 0)$ . Therefore we have

$$\begin{aligned} v_{\pi^*|j,1}(j) &= \frac{1 - \exp(-\eta)}{1 - \exp(-\eta/p(j))}, \quad \forall j \in [K], \\ v_{\pi^*|j,1}(i) &= \frac{\exp(-\eta) - \exp(-\eta/p(j))}{1 - \exp(-\eta/p(j))} \cdot \frac{\beta_i(j)}{\sum_i \beta_i(j) - \beta_j(j)}, \quad \forall i \neq j \in [K], \\ v_{\pi^*|j,0}(j) &= \frac{\exp(\eta) - 1}{\exp(\eta/p(j)) - 1}, \quad \forall j \in [K], \\ v_{\pi^*|j,0}(i) &= \frac{\exp(\eta/p(j)) - \exp(\eta)}{\exp(\eta/p(j)) - 1} \cdot \frac{\beta_i(j)}{\sum_i \beta_i(j) - \beta_j(j)}, \quad \forall i \neq j \in [K]. \end{aligned} \quad (49)$$

And from Equations (47) and (48) we have for every  $i \in [K]$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(i)} &= \frac{1}{\eta} \sum_{j \in [K]} p(j) \log \frac{q(i)}{v_{\pi^*|j,0}(i)} \\ &= \frac{1}{\eta} \sum_{j \in [K]} p(j) \log \frac{q(i)}{v_{\pi^*|j,1}(i)}, \end{aligned} \quad (50)$$

where the first equality is because of Equation (47) and the last equality is because of Equation (48).

Now we set  $\alpha = p = q$  so that the posterior updates (Equation (49)) all have closed forms. This selection results in Algorithm 4. In the following part we present the regret analysis of Algorithm 4 and the proof to Theorem 4.1.

**2.2.2 Regret Analysis of Algorithm 4.** We have shown that the derivatives of AIR with respect to all  $\{\beta_i(j)\}_{i,j \in [K]}$  are zeros, so the optimization error in Theorem 3.4 will happen in the coordinates in  $\alpha$  rather than  $\beta$ . We then need to control the optimization error incurred because of the derivatives of  $\alpha$ . For this purpose we prove the following lemma (proof deferred to Appendix 2.3).

**LEMMA 2.1 (BOUNDEDNESS OF DERIVATIVES).** *In Bernoulli MAB, under Equation (48) (which is the result of setting the derivatives of AIR with respect to all  $\{\beta_i(j)\}_{i,j \in [K]}$  to be zeros), if we set  $p = q = \alpha \in \text{int}(\Delta([K]))$ , then we have*

$$-\eta \leq \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(i)} \leq \eta, \quad \forall i \in [K].$$

Now we have shown that the derivatives of AIR with respect to all  $\{\beta_i(j)\}_{i,j \in [K]}$  are zeros, and the derivatives of AIR with respect to all  $\{\alpha(i)\}_{i \in [K]}$  are suitably bounded. Combining Theorem 3.4, Lemmas 2.1, and A.2, we prove Theorem 4.1 in a straightforward manner.

**THEOREM 4.1 (REGRET OF SIMPLIFIED APS FOR BERNOUlli MAB).** *The regret of Algorithm 4 with  $\eta = \sqrt{\log K / (2KT + 4T)}$  and  $T \geq 3$  is bounded by*

$$\mathfrak{R}_T \leq 2\sqrt{2(K+2)T \log K}.$$

**PROOF OF THEOREM 4.1:** By Theorem 3.4 (regret bound using approximate belief maximizers) and Lemma 2.1 (boundedness of derivatives), we have

$$\mathfrak{R}_T \leq \frac{\log K}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T (\text{AIR}_{p_t, \eta}(p_t, v_t) + 2\eta) \right]. \quad (51)$$

By Lemma A.2 (bounding AIR by IR for TS) and the fact that IR of TS is bounded by  $4K$  for MAB (Lemma 1.3), for any  $\eta \in (0, 1/3]$  we have

$$\text{AIR}_{p_t, \eta}(p_t, v_t) \leq 2\eta(K+1). \quad (52)$$

Combining Equations (51) and (52) and optimizing for  $\eta$ , we prove Theorem 4.1.  $\square$

We would like to comment that, by using the language of the exponential family and the log-partition function introduced in Appendix 2.6, Lemma 2.1 is equivalent to demonstrating that a “mean-zero” property of the natural parameter implies an approximate “mean-zero” property of the log-partition function. This conclusion appears quite natural, given that the log-partition function for the Bernoulli distribution has a flat growth and exhibits the so-called “local norm” property. Such analysis is generic and can be applied broadly, rather than solely serving for the single lemma here. The formal proof of Lemma 2.1 can be found in Appendix 2.3.

We believe that our approach is no more complicated than the standard proofs in the bandit and RL literature. Once one understands the generic machinery for maximizing AIR and why the derivatives are bounded (e.g., for simple bandit problems, this can often be explained through the natural parameter and the log-partition function as in Appendix 2.6), it becomes convenient to principally derive optimal algorithms.

### 2.3 Proof of Lemma 2.1

**LEMMA B.1 (BOUNDEDNESS OF DERIVATIVES).** *In Bernoulli MAB, under Equation (48) (which is the result of setting the derivatives of AIR with respect to all  $\{\beta_i(j)\}_{i,j \in [K]}$  to be zeros), if we set  $p = q = \alpha \in \text{int}(\Delta([K]))$ , then we have*

$$-\eta \leq \frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(i)} \leq \eta, \quad \forall i \in [K].$$

*Equivalent expression of Lemma 2.1 in the language of log-partition function:* We firstly rewrite Lemma 2.1 in the language of exponential family. Intuition introduced in Appendix 2.6 is quite helpful for understanding the naturalness of our approach, while such knowledge is not required and the proof here will be self-contained.

It is easy to check that Algorithm 4 satisfies the optimality condition stated in Equation (48):

$$\log \frac{\nu_{\pi^*|j,1}(i)}{\nu_{\pi^*|j,0}(i)} = \eta \frac{\mathbb{1}(i=j)}{p(j)} - \eta, \quad \forall i, j \in [K] \quad (53)$$

and  $\alpha = p = q$  at every round (we omit the subscript  $t$  for  $\alpha, p, q, v$ ). Furthermore, it is easy to check that Equation (53) is equivalent with the following “mean-zero” property (with respect to  $j \sim p$ ) of the functions we refer to as “natural parameters”:

$$g_i(j) - g_{\text{avg}}(j) = \eta \frac{\mathbb{1}(i=j)}{p(j)} - \eta, \quad \forall i, j \in [K], \quad (54)$$

where we define

$$\begin{aligned} g_i(j) &= g(\theta_i(j)) = \log \frac{\theta_i(j)}{1 - \theta_i(j)}, \\ g_{\text{avg}}(j) &= g(\theta_{\text{avg}}(j)) = \log \frac{\theta_{\text{avg}}(j)}{1 - \theta_{\text{avg}}(j)} \end{aligned}$$

for the conditional mean  $\theta_i(j) = \mathbb{E}[r(j)|\pi^* = i]$  and mean  $\theta_{\text{avg}}(j) = \mathbb{E}[r(j)]$ . One can straightforwardly check the equivalence between Equations (53) and (54) from the posterior expression

$$\begin{aligned} \nu_{\pi^*|j,1}(i) &= \frac{\alpha(i)\theta_i(j)}{\sum_{i \in [K]} \alpha(i)\theta_i(j)} = \frac{\alpha(i)\theta_i(j)}{\theta_{\text{avg}}(j)}, \\ \nu_{\pi^*|j,0}(i) &= \frac{\alpha(i)(1 - \theta_i(j))}{\sum_{i \in [K]} \alpha(i)(1 - \theta_i(j))} = \frac{\alpha(i)(1 - \theta_i(j))}{1 - \theta_{\text{avg}}(j)}. \end{aligned} \quad (55)$$

Alternatively, if the reader has been familiar with our derivation in Appendix 2.2 and the exponential family interpretation in Appendix 2.6, then it is immediate to understand that both Equation (54) is the consequence of setting all the derivatives w.r.t.  $\beta$  to be 0.

In addition, we can verify that when  $p = q$ ,

$$\frac{\partial \text{AIR}_{q,\eta}(p,v)}{\partial \alpha(i)} = \frac{1}{\eta} \mathbb{E}_{j \sim p} [\bar{A}(g_i(j)) - \bar{A}(g_{\text{avg}}(j))], \quad (56)$$

where  $\bar{A}$  is the log-partition function with respect to the natural parameter  $g$ :

$$\bar{A}(g) = \log(1 + e^g).$$

Equation (56) can be verified by using the derivative formula (50) and the posterior expression (55) straightforwardly. Again, if the reader has been familiar with our general derivation about the exponential family in Appendix 2.6, the derivative expression (56) using the log-partition function will be an immediate result.

Given Equations (54) and (56), we are now ready to equivalently rewrite Lemma 2.1 as the following: the “mean-zero” property (Equation (54)) of the natural parameter with respect to  $j \sim p$  implies an approximate “mean-zero” property of the log-partition function with respect to  $j \sim p$ .

**LEMMA 2.2 (EQUIVALENT EXPRESSION OF LEMMA 2.1).** *Given Equation (54) and  $\alpha = p = q$ , for every fixed  $i \in [K]$ , we have that*

$$-\eta \leq \mathbb{E}_{j \sim p} [\bar{A}(g_i(j)) - \bar{A}(g_{\text{avg}}(j))] \leq \eta,$$

where  $\bar{A}$  is the log-partition function defined by

$$\bar{A}(g) = \log(1 + e^g).$$

PROOF OF LEMMA 2.2 AND LEMMA 2.1: Now we prove Lemma 2.2 in order to prove Lemma 2.1.

Because  $0 \leq \log((1+e^x)/(1+e^y)) \leq x-y$  for all  $x \geq y \in \mathbb{R}$ , and  $x-y \leq \log((1+e^x)/(1+e^y)) \leq 0$  for all  $x \leq y \in \mathbb{R}$ , we have that

$$\min\{0, x-y\} \leq \bar{A}(x) - \bar{A}(y) \leq \max\{x-y, 0\} \quad (57)$$

Therefore, by Equations (57) and (54) we have the upper bound

$$\begin{aligned} & \mathbb{E}_{j \sim p} [\bar{A}(g_i(j)) - \bar{A}(g_{\text{avg}}(j))] \\ & \leq \mathbb{E}_{j \sim p} [\max\{g_i(j) - g_{\text{avg}}(j), 0\}] \\ & \leq \mathbb{E}_{j \sim p} \left[ \eta \frac{\mathbb{1}(i=j)}{p(j)} \right] \\ & = \eta, \end{aligned}$$

and the lower bound

$$\begin{aligned} & \mathbb{E}_{j \sim p} [\bar{A}(g_i(j)) - \bar{A}(g_{\text{avg}}(j))] \\ & \geq \mathbb{E}_{j \sim p} [\min\{0, g_i(j) - g_{\text{avg}}(j)\}] \\ & \geq -\eta. \end{aligned}$$

As a result, we finish the proof of Lemma 2.2 and consequently prove the original Lemma 2.1.  $\square$

## 2.4 Concave Parameterization with Gaussian Reward

For structured bandit with Gaussian reward structure, we can formulate the optimization problem as follows. To facilitate the handling of mixture distributions, we introduce the “homogeneous noise” assumption as follows: for all actions  $\pi \in \Pi$ , the reward of  $\pi$  is denoted as  $r(\pi) = \mathbb{E}[r(\pi)] + \epsilon$ , where  $\epsilon \sim N(0, 1)$  is Gaussian noise that is identical across all actions (meaning all actions share the same randomness within the same rounds). This “homogeneous noise” simplifies our expression of AIR, and the resulting algorithms remain applicable to independent noise models and the broader sub-Gaussian setting.

We introduce the following parameterization of belief  $v$ :  $\forall a, a^* \in \Pi$ ,

$$\begin{aligned} \theta_{a^*}(a) &= \mathbb{E}_v [r(a)|\pi^* = a^*], \\ \alpha(a^*) &= v_{\pi^*}(a^*), \\ \beta_{a^*}(a) &= \alpha(a^*) \cdot \theta_{a^*}(a). \end{aligned}$$

Note that AIR in this “homogeneous noise” setting can be expressed as

$$\text{AIR}_{q,\eta}(p, v) = \mathbb{E} \left[ \theta_{\pi^*}(\pi^*) - \theta_{\pi^*}(\pi) - \frac{1}{\eta} \text{KL}(N(\theta_{\pi^*}(\pi), \sigma^2), N(\theta_{\text{avg}}(\pi), \sigma^2)) - \frac{1}{\eta} \text{KL}(\alpha, q) \right], \quad (58)$$

where we denote

$$\theta_{\text{avg}} = \sum_{\pi^* \in \Pi} \alpha(\pi^*) \theta_{\pi^*}.$$

Similar to the definition of  $B$  function in Appendices A.2 and 2.1, we define the following  $\bar{B}$  function, which can simplify our calculation of derivatives in the next section. Note that in the following definition of the  $\bar{B}$  function, its second argument is a belief  $\omega \in \Delta(\mathcal{M} \times \Pi)$ , and we replace

the mapping  $Q : \Pi \times \mathcal{O} \rightarrow \Delta(\Pi)$  in the original definition (Equation (24)) of the  $B$  function in Appendix A.2 to the marginal posterior mapping  $\omega_{\pi^*| \cdot, \cdot} : \Pi \times \mathcal{O} \rightarrow \Delta(\Pi)$ . To be specific, we define

$$\bar{B}(v, \omega) = B(v, \omega_{\pi^*| \cdot, \cdot}) = \mathbb{E}_{(M, a^*) \sim v, a \sim p, o \sim M(a)} \left[ \theta_{a^*}(a^*) - \theta_{a^*}(a) + \frac{1}{\eta} \log \frac{q(a^*)}{\omega_{\pi^*| a, o}(a^*)} \right],$$

which can be expressed as

$$\begin{aligned} & \bar{B}(v, \omega) \\ &= \mathbb{E}_{(M, a^*) \sim v, a \sim p} \left[ \theta_{a^*}(a^*) - \theta_{a^*}(a) + \frac{1}{2\eta} \left( (\theta_{a^*}^\omega(\pi) - r(a))^2 - (\theta_{\text{avg}}^\omega(a) - r(a))^2 \right) - \frac{1}{\eta} \log \frac{\alpha^\omega(a^*)}{q(a^*)} \right] \\ &= \sum_{a^*} \beta_{a^*}(a^*) - \sum_{a, a^*} p(a) \beta_{a^*}(a) + \frac{1}{2\eta} \sum_{a, a^*} p(a) \alpha(a^*) \left( \theta_{a^*}^\omega(a)^2 - \theta_{\text{avg}}^\omega(a)^2 \right) \\ &\quad + \frac{1}{\eta} \sum_{a, a^*} p(a) (\theta_{\text{avg}}^\omega(a) - \theta_{a^*}^\omega(a)) \beta_{a^*}(a) - \frac{1}{\eta} \sum_{a^*} \alpha(a^*) \log \frac{\alpha^\omega(a^*)}{q(a^*)}, \end{aligned} \quad (59)$$

where  $\alpha^\omega(a^*) = \omega_{\pi^*}(a^*)$  and  $\theta_{a^*}^\omega(a) = \mathbb{E}_\omega[r(a)|\pi^* = a^*]$  following the parameterization of  $\omega$  (which is independent with the parameterization  $(\alpha, \beta)$  of  $v$  without the superscript). By the relationship between the  $B$  function and AIR showed in Appendix 2.1, we have

$$\begin{aligned} & \text{AIR}_{q, \eta}(p, v) \\ &= \inf_{\omega} \bar{B}(v, \omega) \\ &= \int_{\mathcal{A}} \beta_{a^*}^\top a^* da^* - \int_{\mathcal{A}} \int_{\mathcal{A}} p(a) \beta_{a^*}^\top a da da^* \\ &\quad - \frac{1}{2\eta} \int_{\mathcal{A}} \int_{\mathcal{A}} p(a) \alpha(a^*) \left( \frac{\beta_{a^*}^\top a}{\alpha(a^*)} - \int_{\mathcal{A}} \beta_{a^*}^\top a da^* \right)^2 da - \frac{1}{\eta} \text{KL}(\alpha, q). \end{aligned}$$

This means that  $\text{AIR}_{q, \eta}(p, v)$  will be a concave function of  $(\alpha, \{\beta_{\pi^*}\}_{\pi^* \in \Pi})$ .

## 2.5 Simplified APS for Gaussian Linear Bandits

In this subsection we derive Algorithm 5 for adversarial linear bandits with Gaussian reward. We work in the setting discussed in Section 6.2, where the mean reward is bounded in  $[-1, 1]$  and all the rewards follow Gaussian distribution with variance 1. As the decision space is a  $d$ -dimensional action set  $\Pi = \mathcal{A} \subseteq \mathbb{R}^d$ , we will use the notations  $\mathcal{A}$  (as action set),  $a$  (as action), and  $a^*$  (as optimal action) to follow the tradition of literature about linear bandits.

By Equation (59) and Lemma 4.3, we have for each  $a^* \in \mathcal{A}$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q, \eta}(p, v)}{\partial \beta_{a^*}} &= \frac{\partial \bar{B}(v, \omega)}{\partial \beta_{a^*}} \Big|_{\omega=v} \\ &= a^* - \mathbb{E}_{a \sim p}[a] - \frac{1}{\eta} \mathbb{E}_{a \sim p}[aa^\top] (\theta_{a^*} - \theta_{\text{avg}}). \end{aligned} \quad (60)$$

And for each  $a^* \in \mathcal{A}$ ,

$$\begin{aligned} \frac{\partial \text{AIR}_{q, \eta}(p, v)}{\partial \alpha(a^*)} &= \frac{\partial \bar{B}(v, \omega)}{\partial \alpha(a^*)} \Big|_{\omega=v} \\ &= \frac{1}{2\eta} \int_{\mathcal{A}} p(a) (\theta_{a^*}^\top a)^2 da - \frac{1}{\eta} \log \frac{\alpha(a^*)}{q(a^*)}. \end{aligned} \quad (61)$$

Let the derivatives in Equation (60) be close to zero. If the matrix  $\mathbb{E}_{a \sim p}[aa^\top]$  have full rank, then we have

$$\begin{aligned}\theta_{a^*} - \theta_{\text{avg}} &\approx \eta(\mathbb{E}_{a \sim p}[aa^\top])^{-1}(a^* - \mathbb{E}_{a \sim p}[a]), \quad \forall a^* \in \mathcal{A}, \\ \alpha &\approx p.\end{aligned}$$

We ask  $\mathbb{E}_{a \sim p}[aa^\top]$  to be positive definite, and let

$$\theta_{a^*} = \frac{\beta_{a^*}}{\alpha(a^*)} = \eta(\mathbb{E}_{a \sim p}[aa^\top])^{-1}a^*, \quad \forall a^* \in \mathcal{A}. \quad (62)$$

Taking the relationship Equations (62) into (61), we have

$$\frac{\partial \text{AIR}_{q,\eta}(p,v)}{\partial \alpha(a^*)} = \frac{\eta}{2}(a^*)^\top(\mathbb{E}_{a \sim p}[aa^\top])^{-1}a^* - \frac{1}{\eta} \log \frac{\alpha(a^*)}{q(a^*)}. \quad (63)$$

Aiming to make the derivatives Equation (63) to be zero, we propose the following approximate solutions that makes the derivatives in Equation (60) to be nearly zero:

$$\begin{aligned}p &= q, \\ \theta_{a^*} &= \frac{\beta_{a^*}}{\alpha(a^*)} = \eta(\mathbb{E}[aa^\top])^{-1}a^*, \quad \forall a^* \in \mathcal{A}, \\ \alpha(a) &\propto q(a) \exp\left(\frac{\eta}{2}a^\top(\mathbb{E}_{a \sim p}[aa^\top])^{-1}a\right), \quad \forall a^* \in \mathcal{A}\end{aligned} \quad (64)$$

By Bayes' rule and Equation (62), the posterior update  $v_{a^*|a,r(a)}$  can be expressed as follows. Given  $\bar{a} \in \mathcal{A}$ , we have

$$\begin{aligned}v_{a^*|a,r(a)}(\bar{a}) &= \frac{\alpha(\bar{a}) \exp(-\frac{1}{2}(r(a) - \theta_a^\top a)^2)}{\int_{\mathcal{A}} \alpha(a^*) \exp(-\frac{1}{2}(r(a) - \theta_{a^*}(a))^2) da^*} \\ &= \frac{\alpha(\bar{a}) \exp(r(a)\theta_{\bar{a}}^\top a - \frac{1}{2}(\theta_{\bar{a}}^\top a)^2)}{\int_{\mathcal{A}} \alpha(a^*) \exp(r(a)\theta_{a^*}^\top a - \frac{1}{2}(\theta_{a^*}^\top a)^2) da^*}.\end{aligned}$$

The resulting algorithm is an exponential weight algorithm

$$p_t(a) \propto \exp\left(\eta \sum_{i=1}^{t-1} \hat{r}_i(a)\right),$$

with the modified IPW estimator

$$\hat{r}_t(a) = \underbrace{a^\top(\mathbb{E}_{a \sim p}[aa^\top])^{-1}a_t r_t(a_t)}_{\text{IPW estimator}} + \underbrace{\frac{\eta}{2} \left( a^\top(\mathbb{E}_{a \sim p_t}[aa^\top])^{-1}a - (a^\top(\mathbb{E}_{a \sim p_t}[aa^\top])^{-1}a_t)^2 \right)}_{\text{mean zero regularizer}}.$$

The requirement that  $\mathbb{E}_{a \sim p}[aa^\top]$  is positive definite can be ensured with the help of the volumetric spanners constructed in [27]. In this way we have derived Algorithm 5.

## 2.6 Solving AIR with General Exponential Family Distributions

For structured bandits with reward distribution belonging to the exponential family, we formulate the optimization problem as follows. To facilitate the handling of mixture distributions, we consider the “homogeneous noise” assumption as follows: for all actions  $a \in \Pi$ , the reward  $r(a)$  of  $a$  follows the exponential family distribution  $r(a) \sim P_{\theta(a)}$ , and all actions share the same randomness within the same rounds. This “homogeneous noise” simplifies our expression of AIR, and the

resulting algorithms remain applicable to independent noise models. We assume the exponential family distribution has the probability density function

$$P_{\theta(a)}(r(a) = o) \propto h(o) \exp(g(\theta(a))T(o) - A(\theta(a))), \quad \forall a \in \Pi.$$

We introduce the following parameterization of belief  $v$ :  $\forall a, a^* \in \Pi$ ,

$$\begin{aligned} \theta_{a^*}(a) &= \mathbb{E}_v [r(a)|\pi^* = a^*], \\ \alpha(a^*) &= v_{\pi^*}(a^*), \\ \beta_{a^*}(a) &= \alpha(a^*) \cdot \theta_{a^*}(a). \end{aligned}$$

Note that AIR in this “homogeneous noise” setting can be expressed as

$$\text{AIR}_{q,\eta}(p, v) = \mathbb{E} \left[ \theta_{\pi^*}(\pi^*) - \theta_{\pi^*}(\pi) - \frac{1}{\eta} \text{KL}(P_{\theta_{\pi^*}(\pi)}, P_{\theta_{\text{avg}}(\pi)}) - \frac{1}{\eta} \text{KL}(\alpha, q) \right],$$

where we denote

$$\theta_{\text{avg}} = \sum_{\pi^* \in \Pi} \alpha(\pi^*) \theta_{\pi^*}.$$

Similar to Appendix 2.4, we define the following  $\bar{B}$  function, which can simplify our calculation of derivatives. To be specific, we define

$$\bar{B}(v, \omega) = B(v, \omega_{\pi^*| \cdot, \cdot}) = \mathbb{E}_{(M, a^*) \sim v, a \sim p, o \sim M(a)} \left[ \theta_{a^*}(a^*) - \theta_{a^*}(a) + \frac{1}{\eta} \log \frac{q(a^*)}{\omega_{\pi^*| a, o}(a^*)} \right],$$

which can be expressed as

$$\begin{aligned} &\bar{B}(v, \omega) \\ &= \sum_{a^*} \beta_{a^*}(a^*) - \sum_{a, a^*} p(a) \beta_{a^*}(a) + \frac{1}{\eta} \log \frac{q(a^*)}{\alpha^\omega(a^*)} \\ &\quad + \frac{1}{\eta} \sum_{a^*, a} p(a) \alpha(a^*) \int_{\mathcal{O}} (g(\theta_{\text{avg}}^\omega(a)) - g(\theta_{a^*}^\omega(a))) T(o) do + \frac{1}{\eta} \sum_{a^*, a} p(a) \alpha(a^*) [A(\theta_{a^*}^\omega(a)) - A(\theta_{\text{avg}}^\omega(a))]. \end{aligned}$$

where  $\alpha^\omega(a^*) = \omega_{\pi^*}(a^*)$  and  $\theta_{a^*}^\omega(a) = \mathbb{E}_\omega [r(a)|\pi^* = a^*]$  following the parameterization of  $\omega$  (which is independent with the parameterization  $(\alpha, \beta)$  of  $v$  without the superscript). By the relationship between the  $B$  function and AIR showed in Appendix 2.1, we have

$$\text{AIR}_{q,\eta}(p, v) = \inf_{\omega} \bar{B}(v, \omega).$$

Now we consider exponential family where the sufficient statistics  $T(o) = o$ . Common exponential families with sufficient statistics  $T(o) = o$  include Bernoulli distribution, Poisson distribution, exponential distribution, Gaussian distribution with known variance 1, and so on. Then we have

$$\begin{aligned} &\bar{B}(v, \omega) \\ &= \sum_{a^*} \beta_{a^*}(a^*) - \sum_{a, a^*} p(a) \beta_{a^*}(a) + \frac{1}{\eta} \sum_{a^*} \alpha(a^*) \log \frac{q(a^*)}{\alpha^\omega(a^*)} \\ &\quad + \frac{1}{\eta} \sum_{a^*, a} p(a) (g(\theta_{\text{avg}}^\omega(a)) - g(\theta_{a^*}^\omega(a))) \beta_{a^*}(a) + \frac{1}{\eta} \sum_{a^*, a} p(a) \alpha(a^*) [A(\theta_{a^*}^\omega(a)) - A(\theta_{\text{avg}}^\omega(a))]. \end{aligned}$$

Clearly,  $\bar{B}$  is a linear function with respect to the parameterization  $(\alpha, \beta)$  of  $v$ .

We assume that the action set  $\Pi = [K]$  is a finite set of  $K$  actions. By Danskin's theorem (Lemma 4.3), we can calculate the gradient of AIR with respect to  $\beta$  as

$$\begin{aligned}\frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \beta_{a^*}(a)} &= \left. \frac{\partial \bar{B}(v, \omega)}{\partial \beta_{a^*}(a)} \right|_{\omega=v} \\ &= \mathbb{1}(a = a^*) - p(a) + \frac{1}{\eta} p(a) \left[ g(\theta_{\text{avg}}(a)) - g(\theta_{a^*}(a)) \right],\end{aligned}$$

and we can calculate the gradient of AIR with respect to  $\alpha$  as

$$\begin{aligned}\frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(a^*)} &= \left. \frac{\partial \bar{B}(v, \omega)}{\partial \alpha(a^*)} \right|_{\omega=v} \\ &= \frac{1}{\eta} \sum_a p(a) \left[ A(\theta_{a^*}(a)) - A(\theta_{\text{avg}}(a)) \right] + \frac{1}{\eta} \log \frac{q(a^*)}{\alpha(a^*)}.\end{aligned}$$

In statistics theory, the function  $g$  is called the “natural parameter.” Denote

$$g_{a^*}(a) = g(\theta_{a^*}(a)), \quad g_{\text{avg}}(a) = g(\theta_{\text{avg}}(a)),$$

and write  $A(\theta)$  as a function of  $g$  defined by

$$\bar{A}(g) = A(\theta).$$

In statistics theory the function  $\bar{A}(g)$ , which is a function of the natural parameter  $g$ , is called the log-partition function, because it is the logarithm of a normalization factor, without which  $P_\theta(o)$  will not be a probability distribution. Mathematically, it can be defined as  $\bar{A}(g) = \log \left( \int_{\mathcal{O}} h(o) \exp(g(\theta) \cdot T(o)) do \right)$ . However, for the sake of simplicity, we will directly use the expression  $\bar{A}(g) = A(\theta)$ .

Then we can rewrite the derivatives with respect to  $\beta$  in the form of

$$\frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \beta_{a^*}(a)} = \mathbb{1}(a = a^*) - p(a) + \frac{1}{\eta} p(a) \left[ g_{\text{avg}}(a) - g_{a^*}(a) \right], \quad (65)$$

and rewrite the derivatives with respect to  $\alpha$  in the form of

$$\frac{\partial \text{AIR}_{q,\eta}(p, v)}{\partial \alpha(a^*)} = \frac{1}{\eta} \sum_a p(a) \left[ \bar{A}(g_{a^*}(a)) - \bar{A}(g_{\text{avg}}(a)) \right] + \frac{1}{\eta} \log \frac{q(a^*)}{\alpha(a^*)}. \quad (66)$$

Now we discuss whether we should set Equation (66) to be zeros (i.e., making natural parameters “mean-zero”), or we should set Equation (65) to be zeros (i.e., making the log-partition functions “mean-zeros”). Note that we only have  $\mathbb{E}_{a^* \sim \alpha}[\theta_{a^*}] = \theta_{\text{avg}}$  by definition, and both Equation (66) and Equation (65) are our targets to make nearly zero by setting the correct  $(\alpha, \beta)$ . As we will see, there will be tradeoff between natural parameters and log-partition functions, and one need to look at the geometry of the exponential family to determine the formula to choose. We discuss the following two attempts:

(1) **Setting derivatives w.r.t.  $\beta$  to be 0:** we set the following equations:

$$g_{a^*}(a) - g_{\text{avg}}(a) = \eta \frac{\mathbb{1}(a = a^*)}{p(a)} - \eta, \quad \forall a^*, a \in \mathcal{A} \quad (67)$$

together with  $\alpha = p$ . In this way so that all the derivatives with respect to  $\beta$  are all zeros, and we have the following “mean-zero” property for natural parameters: for all  $a^* \in [K]$ .

$$\mathbb{E}_{a \sim p}[g_{a^*}(a) - g_{\text{avg}}(a) = 0].$$

In order to make sure that the derivatives with respect to  $\alpha$  are also nearly zero, we consider the posterior sampling strategy  $p = q$  and want to show that

$$\left| \mathbb{E}_{a \sim p} [\bar{A}(g_{a^*}(a)) - \bar{A}(g_{a^*}(a))] \right| = O(\eta^2).$$

To achieve this, we need the log-partition function  $\bar{A}$  to be very “flat,” in the sense that its Hessian at  $z$  grows no faster than  $1/z$  so that the error  $\bar{A}(g_{a^*}(a)) - \bar{A}(g_{a^*}(a))$  can be bounded by the “local norm”

$$|\bar{A}(g_{a^*}(a)) - \bar{A}(g_{a^*}(a))| \leq O\left(\sum_a p(a) [g_{a^*}(a) - g_{a^*}(a)]^2\right).$$

This is true for Bernoulli distribution, where the log-partition function is in the form  $\bar{A}(g) = \log(1 + e^g)$ , as well as the exponential distribution, where the log-partition function is in the form  $\bar{A}(g) = -\log(-g)$ .

(2) **Setting derivatives w.r.t.  $\alpha$  to be 0:** If the log-partition function is not flat but increases “sharply”, then we instead set the derivatives with  $\alpha$  to be zeros. Then we need to show that the derivatives with respect to  $\beta$  are nearly zeros. This is the case for Gaussian distribution, where the log-partition function  $\bar{A}(g) = g^2/2$ . Taking Gaussian distribution for example, we need to make  $\alpha$  slightly different than  $p$  to make their derivatives to be zero as in Equation (64). But then it is convenient to show that the derivatives respect to  $\beta$  are nearly zero and the Equation (67) is approximately satisfied up to the difference between  $\alpha$  and  $p$ .

So we conclude that in order to get closed-form solutions and avoid tedious boundary conditions, there may be tradeoff between making Equation (66) zeros (i.e., making natural parameters “mean-zero”), and making Equation (65) zeros (i.e., making the log-partition functions “mean-zero”). But nice calculations can often been done with care. We believe it is valuable to further explore this tradeoff between the natural parameter and log-partition function and its role in automatically seeking the optimal bias-variance tradeoff.

Finally, we would like to comment on the IPW-type formulation for the natural parameter  $\eta$  in Equation (67). Following the discussion concerning the connection between APS and mirror descent (see the end of Section 4.1), we conclude that the vanilla IPW estimator and traditional EXP3-type algorithms correspond to using exponential or Gaussian distributions (or any exponential family distribution with a linear natural parameter  $g(\theta) = a\theta + b$  where  $a$  and  $b$  are constants). In contrast, Algorithm 4 employs a more advanced estimator, which can be interpreted from both Bayesian and frequentist perspectives (see Equation (12)).

## 2.7 Dynamic Regret of Simplified APS

In this subsection we establish the dynamic-regret bound claimed in Equation (13) (Section 4.2.3). We show that a slightly modified version of Algorithm 4—presented below as Algorithm 9—satisfies that bound, which follows directly from Theorem 3.5.

The only change from Algorithms 4 to 9 is the decision distribution: in Step 4 we replace  $p_t$  by a forced-exploration mixture. Crucially, this modification is recursive—it affects the posterior-type update every round and thus the entire evolution of the algorithm, rather than merely tweaking the sampling probabilities as in the classical EXP3 mixing rule (stated in Section 4.1).

We now prove the dynamic regret claim Equation (13) (Section 4.2.3) for Algorithm 9 as follows.

**THEOREM 2.3 (REGRET OF SIMPLIFIED APS FOR BERNOULLI MAB).** *The dynamic regret of Algorithm 9 with  $\gamma = 1/T$ ,  $\eta = \sqrt{(\log K(D_T \log T + 1) + 2)/(2KT + 4T)}$ , and  $T \geq 3D_T \log(3D_T) + 6$  is bounded by*

$$\mathfrak{R}_T \leq 2\sqrt{2(K + 2)T(\log K(D_T \log T + 1) + 2)}.$$

**ALGORITHM 9:** Simplified APS with forced exploration for Bernoulli MAB

**Require:** Input learning rate  $\eta > 0$ . Initialize  $p_1 = \text{Unif}(\Pi)$  and exploration rate  $\gamma \in (0, 1)$ .

- 1: **for** round  $t = 1, 2, \dots, T$  **do**
- 2:   Sample action  $\pi_t \sim p_t$  and receives  $r_t(\pi_t)$ .
- 3:   Update  $p_{t+1}$  by

$$p_{t+1}(\pi_t) = \begin{cases} (1-\gamma) \frac{1-\exp(-\eta)}{1-\exp(-\eta/p_t(\pi_t))} + \frac{\gamma}{K}, & \text{if } r_t(\pi_t) = 1 \\ (1-\gamma) \frac{1-\exp(\eta)}{1-\exp(\eta/p_t(\pi_t))} + \frac{\gamma}{K}, & \text{if } r_t(\pi_t) = 0 \end{cases}, \text{ and}$$

$$p_{t+1}(\pi) = (1-\gamma)p_t(\pi) \cdot \frac{1-p_{t+1}(\pi_t)}{1-p_t(\pi_t)} + \frac{\gamma}{K}, \quad \forall \pi \neq \pi_t.$$

**PROOF OF THEOREM 2.3:** By Theorem 3.5 (specifically the more flexible corollary stated in Equation (37)) and Lemma 2.1 (boundedness of derivatives), we have

$$\mathfrak{R}_T \leq \frac{\log K(D_T \log T + 1) + \frac{T}{T-1}}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T (\text{AIR}_{p_t, \eta}(p_t, v_t) + 2\eta) \right]. \quad (68)$$

By Lemma A.2 (bounding AIR by IR for TS) and the fact that IR of TS is bounded by  $4K$  for MAB (Lemma 1.3), for any  $\eta \in (0, 1/3]$  we have

$$\text{AIR}_{p_t, \eta}(p_t, v_t) \leq 2\eta(K + 1). \quad (69)$$

Combining Equations (68) and (69) and optimizing for  $\eta$ , we prove Theorem 2.3.  $\square$

### 3 Proofs for MAIR and RL

#### 3.1 Details for Applications to Reinforcement Learning

We illustrate how the general problem formulation in Section 2.1 covers RL problems as follows. The definitions and theoretical results are primarily based on adapting the results and proofs in [23].

*Example 3.1 (Reinforcement Learning).* An episodic finite-horizon RL problems is defined as follows. Let  $H$  be the horizon and  $\mathcal{A}$  be a finite action set. Each model  $M \in \mathcal{M}$  specifies a non-stationary **Markov decision process (MDP)**  $\{(S^{(h)})_{h=1}^H, \mathcal{A}, \{P_M^{(h)}\}_{h=1}^H, \{R_M^{(h)}\}_{h=1}^H, \mu\}$ , where  $\mu$  is the initial distribution over states; and for each layer  $h$ ,  $S^{(h)}$  is a finite state space,  $P_M^{(h)} : S^{(h)} \times \mathcal{A} \rightarrow (S^{(h+1)})$  is the probability transition kernel, and  $R_M^{(h)} : S^{(h)} \times \mathcal{A} \rightarrow \Delta([0, 1])$  is the reward distribution. We allow the transition kernel and loss distribution to be different for different  $M \in \mathcal{M}$  but assume  $\mu$  to be fixed for simplicity. Let  $\Pi_{\text{NS}}$  be the space of all deterministic non-stationary policies  $\pi = (u^{(1)}, \dots, u^{(H)})$ , where  $u^{(h)} : S^{(h)} \rightarrow \mathcal{A}$ . Given an MDP  $M$  and policy  $\pi$ , the MDP evolves as follows: beginning from  $s^{(1)} \sim \mu$ , at each layer  $h = 1, \dots, H$ , the action  $a^{(h)}$  is sampled from  $u^{(h)}(s^{(h)})$ , the loss  $r^{(h)}(a^{(h)})$  is sampled from  $R_M(s^{(h)}, a^{(h)})$  and the state  $s^{(h+1)}$  is sampled from  $P_M(\cdot | s^{(h)}, a^{(h)})$ . Define  $f_M(\pi) = \mathbb{E}[\sum_{h=1}^H r^{(h)}(a^{(h)})]$  to be the expected reward under MDP  $M$  and policy  $\pi$ . The general framework covers episodic RL problems by taking the observation  $o_t$  to be the trajectory  $(s_t^{(1)}, a_t^{(1)}, r_t^{(1)}), \dots, (s_t^{(H)}, a_t^{(H)}, r_t^{(H)})$  and  $\Pi$  be a subspace of  $\Pi_{\text{NS}}$ . While our framework and complexity measures allow for agnostic policy classes, recovering existing results often requires us to make realizability-type assumptions.

We now focus on a broad class of structured RL problems called “bilinear class” [18]. The following definition of the bilinear class is from [23].

*Definition 3.2 (Bilinear Class).* A model class  $\mathcal{M}$  is said to be bilinear relative to reference model  $\bar{M}$  if:

- (1) There exist functions  $W_h(\cdot; \bar{M}) : \mathcal{M} \rightarrow \mathbb{R}^d$ ,  $X_h(\cdot; \bar{M}) : \mathcal{M} \rightarrow \mathbb{R}^d$  such that for all  $M \in \mathcal{M}$  and  $h \in [H]$ ,

$$|\mathbb{E}^{\bar{M}, \pi_M}[Q_h^{M,*}(s_h, a_h) - r_h - V_h^{M,*}(s_{h+1})]| \leq |\langle W_h(M; \bar{M}), X_h(M; \bar{M}) \rangle|.$$

We assume that  $W_h(M; \bar{M}) = 0$ .

- (2) Let  $z^{(h)} = (s^{(h)}, a^{(h)}, r^{(h)}, s^{(h+1)})$ . There exists a collection of estimation policies  $\{\pi_M^{\text{est}}\}_{M \in \mathcal{M}}$  and estimation functions  $\{\ell_M^{\text{est}}(\cdot; \cdot)\}_{M \in \mathcal{M}}$  such that for all  $M, M' \in \mathcal{M}$  and  $h \in [H]$ ,

$$\langle X_h(M; \bar{M}), W_h(M'; \bar{M}) \rangle = \mathbb{E}^{\bar{M}, \pi_M \circ h \pi_M^{\text{est}}}[\ell_M^{\text{est}}(M'; z_h)].$$

If  $\pi_M^{\text{est}} = \pi_M$ , we say that estimation is on-policy.

If  $M$  is bilinear relative to all  $\bar{M} \in \mathcal{M}$ , we say that  $\mathcal{M}$  is a bilinear class. We let  $d_{\text{bi}}(\mathcal{M}, \bar{M})$  denote the minimal dimension  $d$  for which the bilinear class property holds relative to  $\bar{M}$ , and define  $d_{\text{bi}}(\mathcal{M}) = \sup_{\bar{M} \in \mathcal{M}} d_{\text{bi}}(\mathcal{M}, \bar{M})$ . We let  $L_{\text{bi}}(\mathcal{M}; \bar{M}) \geq 1$  denote any almost sure upper bound on  $|\ell_M^{\text{est}}(M'; z_h)|$  under  $\bar{M}$ , and let  $L_{\text{bi}}(\mathcal{M}) = \sup_{\bar{M} \in \mathcal{M}} L_{\text{bi}}(\mathcal{M}; \bar{M})$ .

For  $\gamma \in [0, 1]$ , let  $\pi_M^\gamma$  be the randomized policy that—for each  $h$ —plays  $\pi_{M,h}$  with probability  $1 - \gamma/H$  and  $\pi_{M,h}^{\text{est}}$  with probability  $\gamma/H$ . Combining the upper bounds of  $\text{DEC}_\eta^{\text{TS}}$  proved in Theorem 7.1 in [23] with our Theorem 7.2 (regret of MAMS) and Theorem 7.4 (regret of MAPS), we can immediately obtain regret guarantees for RL problems in the bilinear class.

**THEOREM 3.3 (REGRET OF MAMS AND MAPS FOR RL PROBLEMS IN THE BILINEAR CLASS).** *In the on-policy case, Model-index AMS (Algorithm 3) and Model-index APS (Algorithm 2) with optimally tuned  $\eta$  achieve regret*

$$\mathfrak{R}_T \leq O(H^2 L_{\text{bi}}^2 d_{\text{bi}}(\mathcal{M}) \cdot T \cdot \log |\mathcal{M}|).$$

*In the general case, Model-index AMS (Algorithm 3) and Model-index APS (Algorithm 2) with forced exploration rate  $\gamma = (8\eta H^3 L_{\text{bi}}^2(\mathcal{M}) d_{\text{bi}}(\mathcal{M}, \bar{M}))^{1/2}$  and optimally tuned  $\eta$  achieves regret*

$$\mathfrak{R}_T \leq O((H^3 L_{\text{bi}}^2 d_{\text{bi}}(\mathcal{M}) \log |\mathcal{M}|)^{1/3} \cdot T^{2/3}).$$

In particular, as a closed-form algorithm that may be computed through sampling techniques, MAPS matches the sharp results for E2D, MAMS, and the non-constructive Bayesian Posterior Sampling algorithm used in the proof of Theorem 7.1 in [23]; and MAPS achieves better regret bounds than the closed-form “inverse gap weighting” algorithm provided in the same paper. Its regret bound for RL problems in the bilinear class also match the E2D algorithms in [22, 23] that are not in closed-form and require more challenging minimax optimization.

### 3.2 Proof of Lemma 2.10

**LEMMA 2.8 (BOUNDING MAIR BY DEC).** *Given model class  $\mathcal{M}$  and  $\eta > 0$ , we have*

$$\sup_{\rho \in \text{int}(\Delta(\mathcal{M}))} \sup_{\mu} \inf_p \text{MAIR}_{\rho, \eta}(p, \nu) = \text{DEC}_\eta^{\text{KL}}(\mathcal{M}) \leq \text{DEC}_\eta(\mathcal{M}).$$

PROOF OF LEMMA 2.10: By the definitions of MAIR and DEC we have the following:

$$\begin{aligned}
& \sup_{\rho \in \text{int}(\Delta(\mathcal{M}))} \sup_{\mu \in \Delta(\mathcal{M})} \inf_{p \in \Delta(\Pi)} \text{MAIR}_{\rho, \eta}(p, \mu) \\
&= \sup_{\rho \in \text{int}(\Delta(\mathcal{M}))} \sup_{\mu \in \Delta(\mathcal{M})} \inf_{p \in \Delta(\Pi)} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \mu_{0|\pi}) - \frac{1}{\eta} \text{KL}(\mu, \rho) \right] \\
&= \inf_{p \in \Delta(\Pi)} \sup_{\mu \in \Delta(\mathcal{M})} \sup_{\rho \in \text{int}(\Delta(\mathcal{M}))} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \mu_{0|\pi}) - \frac{1}{\eta} \text{KL}(\mu, \rho) \right] \\
&= \inf_{p \in \Delta(\Pi)} \sup_{\mu \in \Delta(\mathcal{M})} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \mu_{0|\pi}) \right] \\
&= \inf_{p \in \Delta(\Pi)} \sup_{\mu \in \Delta(\mathcal{M})} \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \bar{M}(\pi)) \right] \\
&= \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \sup_{\mu \in \Delta(\mathcal{M})} \inf_{p \in \Delta(\Pi)} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \bar{M}(\pi)) \right] \\
&= \sup_{\bar{M} \in \text{conv}(\mathcal{M})} \inf_{p \in \Delta(\Pi)} \sup_{M \in \mathcal{M}} \mathbb{E}_{\pi \sim p, M \sim \mu} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(M(\pi), \bar{M}(\pi)) \right] \\
&= \text{DEC}_{\eta}^{\text{KL}}(\mathcal{M}), \\
&\leq \text{DEC}_{\eta}(\mathcal{M}),
\end{aligned}$$

where the first equality is by Equation (19); the second inequality is an application of the minimax theorem (Lemma 4.2); the third equality is by taking the supremum over  $\rho$  and performing limit analysis; the fourth inequality is because the supremum over  $\bar{M}$  will be achieved at  $\bar{M} = \mathbb{E}_{\mu}[M]$ , as per the property of the KL divergence; the fifth and the sixth inequalities are applications of the minimax theorem (Lemma 4.2); the seventh equality is by the definition of the KL version of DEC; and the last inequality is because the squared Hellinger distance is bounded by the KL divergence (Lemma 4.8).  $\square$

### 3.3 Proof of Theorem 7.1

**THEOREM 7.1 (GENERIC REGRET BOUND FOR ARBITRARY LEARNING ALGORITHM LEVERAGING MAIR).** *In the stochastic setting, let  $\mathcal{M}$  be a finite model class, and  $M^* \in \mathcal{M}$  be the ground truth model. Fix any algorithm  $\text{ALG}$  that produces decision probability  $p_1, \dots, p_T$ . Initialize  $\rho_1 = \text{Unif}(\mathcal{M})$  and maintain a sequence of beliefs  $\mu_1, \dots, \mu_t, \dots$  where  $\mu_{t+1} = (\mu_t)_{\pi^*|_{\pi_t, o_t}} \in \text{int}(\Delta(\mathcal{M}))$  for all rounds. Then for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \left( \text{MAIR}_{\rho_t, \eta}(p_t, \mu_t) + \left\langle \frac{\partial \text{MAIR}_{\rho_t, \eta}(p_t, \mu)}{\partial \mu} \Big|_{\mu=\mu_t}, \mathbb{1}(M^*) - \mu_t \right\rangle \right) \right],$$

where  $\mathbb{1}(M^*)$  is the vector whose  $M^*$ -coordinate is 1 but all other coordinates are 0.

PROOF OF THEOREM 7.1: In this subsection we prove Theorem 7.1, a generic regret bound using MAIR. Recall the definition of MAIR:

$$\text{MAIR}_{\rho, \eta}(p, \mu) = \mathbb{E}_{\mu, p} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{\eta} \text{KL}(\mu_{\pi^*|_{\pi, o}}, \mu_{\pi^*}) - \frac{1}{\eta} \text{KL}(\mu_{\pi^*}, \rho) \right].$$

Similar to the  $B$  function (Equation (24)) we use in Appendices A.2 and A.6, given the decision probability  $\pi \sim p$ , we define

$$B(\mu, Q) = \mathbb{E}_{\mu, p} \left[ f_M(\pi_M) - f_M(\pi) + \frac{1}{\eta} \frac{\log \rho(M)}{\log Q[\pi, o](M)} \right].$$

Note that  $B(v, Q)$  is a strongly convex function with respect to  $Q$ . We have the following formula:

$$\text{MAIR}_{\rho, \eta}(p, \mu) = B(\mu, \mu(\cdot| \cdot, \cdot)) = \inf_Q B(\mu, Q),$$

where  $\mu(\cdot| \cdot, \cdot)$  denote the “posterior functional” that maps the observation  $(\pi, o)$  to posterior  $\mu(M|\pi, o) \in \Delta(\mathcal{M})$ . Here, the first equality is by the definition of MAIR, and the second equality is by the first-order optimal condition. By Danskin’s theorem (Lemma 4.3), we have

$$\frac{\partial \text{MAIR}_{\rho, \eta}(p, \mu)}{\partial \mu} = \frac{\partial B(\mu, Q)}{\partial \mu} \Big|_{Q=\mu(\cdot| \cdot, \cdot)}.$$

By the above identity and the linearity of  $B(\mu, Q)$  with respect to  $\mu$ , we have

$$B(1(M^*), \mu(\cdot| \cdot, \cdot)) = \text{MAIR}_{\rho, \eta}(p, \mu) + \left\langle \frac{\partial \text{MAIR}_{\rho, \eta}(p, \mu)}{\partial \mu}, 1(M^*) - \mu \right\rangle, \quad (70)$$

where  $M^*$  is the underlying true model in the stochastic setting.

Following the very similar steps in proving Theorems 3.1 and 3.4, we are able to prove Theorem 7.1. For every  $M^* \in \mathcal{M}$  we have

$$\sum_{t=1}^T \left[ \log \frac{\rho_{t+1}(M^*)}{\rho_t(M^*)} \right] = \log \frac{\rho_{T+1}(M^*)}{\rho_1(M^*)} \leq \log |\mathcal{M}|.$$

Given an arbitrary algorithm and an arbitrary algorithmic belief sequence, we set the reference probability  $\rho_{t+1} = \mu_t(M|\pi_t, o_t)$  to aid our analysis. By the above equation we have

$$\mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ \frac{1}{\eta} \log \frac{\mu_t(M^*|\pi_t, o_t)}{\rho_t(M^*)} \right] \right] \leq \frac{\log |\mathcal{M}|}{\eta}. \quad (71)$$

Subtracting the telescope sum Equation 33 from regret, we have

$$\begin{aligned} & \mathfrak{R}_T - \frac{\log |\mathcal{M}|}{\eta} \\ & \leq \mathbb{E} \left[ \sum_{t=1}^T f_{M^*}(\pi_{M^*}) - \sum_{t=1}^T f_{M^*}(\pi_t) \right] - \frac{\log |\mathcal{M}|}{\eta} \\ & \leq \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{t-1} \left[ f_{M^*}(\pi_{M^*}) - f_{M^*}(\pi_t) - \frac{1}{\eta} \log \frac{\mu_t(M^*|\pi_t, o_t)}{\rho_t(M^*)} \right] \right] \\ & = \mathbb{E} \left[ \sum_{t=1}^T B_t(1(M^*), \mu_t(\cdot| \cdot, \cdot)) \right], \end{aligned} \quad (72)$$

where the first inequality is by the definition of regret; the second inequality is by Equation (14); and the third inequality by the definition of the  $B$  function and the stochastic environment. Finally, taking the equivalent characterization Equation (70) for the  $B$  function into Equation (72) proves Theorem 7.1.  $\square$

### 3.4 Proof of Theorem 7.2

**THEOREM 7.2 (REGRET OF MAMS).** *For a finite and compact model class  $\mathcal{M}$ , the regret of Algorithm 6 with any  $\eta > 0$  is always bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{MAIR}_{\rho_t, \eta}(p_t, \mu_t) \right] \leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_\eta^{\text{KL}}(\mathcal{M}) \cdot T.$$

**PROOF OF THEOREM 7.2:** Combining Theorem 7.1 and Lemma 2.10, we prove Theorem 7.2 in a straightforward manner:

$$\begin{aligned} \mathfrak{R}_T &\leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \text{MAIR}_{\rho_t, \eta}(p_t, \nu_t) \right] \\ &\leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_\eta^{\text{KL}}(\mathcal{M}) \cdot T, \end{aligned}$$

where the first inequality is by Theorem 3.4 and the fact that MAMS always pick

$$\mu_t = \arg \max_{\mu \in \Delta(\mathcal{M})} \text{MAIR}_{\rho_t, \eta}(p_t, \mu),$$

and the second inequality is by Lemma 2.10.  $\square$

### 3.5 Proof for Theorem 7.3

**THEOREM 7.3 (REGRET FOR ARBITRARY ALGORITHM WITH CLOSED-FORM BELIEFS).** *Given a finite model class  $\mathcal{M}$  where the underlying true model is  $M^* \in \mathcal{M}$ , and  $f_M(\pi) \in [0, 1]$  for every  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . For an arbitrary algorithm ALG and any  $\eta > 0$ , the regret of algorithm ALG is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{\mu_t, p_t} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), (\rho_t)_{o|\pi}) - \frac{1}{3\eta} \text{KL}(\mu_t, \rho_t) \right] \right].$$

where  $\mu_t$  and  $\rho_t$  are closed-form beliefs generated according the Algorithm 7.

**PROOF OF THEOREM 7.3:** From Theorem 7.1 we have that

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \sum_{t=1}^T \left( \text{MAIR}_{\rho_t, \eta}(p_t, \mu_t) + \left\langle \frac{\partial \text{MAIR}_{\rho_t, \eta}(p_t, \mu)}{\partial \mu} \Big|_{\mu=\mu_t}, \mathbb{1}(M^*) - \mu_t \right\rangle \right), \quad (73)$$

By Lemma 4.3 we have

$$\begin{aligned} \frac{\partial \text{MAIR}_{\rho, \eta}(p, \mu)}{\partial \mu(M)} &= f_M(\pi_M) - \mathbb{E}_{\pi \sim p} [f_M(\pi)] + \frac{1}{\eta} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ \log \frac{\rho(M)}{\mu(M|\pi, o)} \right] \\ &= f_M(\pi_M) - \mathbb{E}_{\pi \sim p} [f_M(\pi)] - \frac{1}{\eta} \log \frac{\mu(M)}{\rho(M)} - \frac{1}{\eta} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} \left[ \log \frac{[\pi](o)}{\mu_o(o|\pi)} \right] \\ &= f_M(\pi_M) - \mathbb{E}_{\pi \sim p} [f_M(\pi)] - \frac{1}{\eta} \log \frac{\mu(M)}{\rho(M)} - \frac{1}{\eta} \mathbb{E}_{\pi \sim p, o \sim M(\pi)} [\text{KL}(M(\pi), \mu_{o|\pi})]. \end{aligned}$$

By using the updating rule in Equation (20), we have

$$\frac{\partial \text{MAIR}_{\rho, \eta}(p, \mu)}{\partial \mu_t(M)} = -\frac{1}{\eta} \mathbb{E}_{\pi \sim p_t} [\text{KL}(M(\pi), (\mu_t)_{o|\pi})] + \frac{1}{3\eta} \mathbb{E}_{\pi \sim p} [D_H^2(M(\pi), (\rho_t)_{o|\pi})] + C_1, \forall M \in \mathcal{M},$$

where  $C_1$  is some value that is the same for all  $M \in \mathcal{M}$ . This implies that

$$\begin{aligned} & \left\langle \frac{\partial \text{MAIR}_{\rho_t, \eta}(p_t, \mu)}{\partial \mu} \Big|_{\mu=\mu_t}, \mathbb{1}(M^*) - \mu_t \right\rangle \\ &= \frac{1}{\eta} \mathbb{E}_{\mu_t, p_t} \left[ \text{KL}(M(\pi), (\mu_t)_{o|\pi}) \right] - \frac{1}{\eta} \mathbb{E}_{\pi \sim p_t} \left[ \text{KL}(M^*(\pi), (\mu_t)_{o|\pi}) \right] \\ &\quad - \frac{1}{3\eta} \mathbb{E}_{\mu_t, p_t} \left[ D_H^2(M(\pi), (\rho_t)_{o|\pi}) \right] + \frac{1}{3\eta} \mathbb{E}_{\mu_t, p_t} \left[ D_H^2(M^*(\pi), (\rho_t)_{o|\pi}) \right]. \end{aligned} \quad (74)$$

Taking Equations (74) into (73), we have

$$\begin{aligned} \mathfrak{R}_T &\leq \frac{\log |\mathcal{M}|}{\eta} + \sum_{t=1}^T \mathbb{E}_{\mu_t, p_t} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), (\rho_t)_{o|\pi}) - \frac{1}{\eta} \text{KL}(\mu_t, \rho_t) \right. \\ &\quad \left. + \frac{1}{3\eta} D_H^2(M^*(\pi), (\rho_t)_{o|\pi}) - \frac{1}{\eta} \mathbb{E} \left[ \text{KL}(M^*(\pi), (\mu_t)_{o|\pi}) \right] \right] \\ &\leq \frac{\log |\mathcal{M}|}{\eta} + \sum_{t=1}^T \mathbb{E}_{\mu_t, p_t} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), (\rho_t)_{o|\pi}) - \frac{1}{3\eta} \text{KL}(\mu_t, \rho_t) \right], \end{aligned} \quad (75)$$

where the last inequality is because by information-theoretical inequalities, we have

$$\begin{aligned} \frac{1}{3} D_H^2(M^*(\pi), (\rho_t)_{o|\pi}) &\leq \frac{2}{3} D_H^2(M(\pi), (\mu_t)_{o|\pi}) + \frac{2}{3} D_H^2((\mu_t)_{o|\pi}, (\rho_t)_{o|\pi}) \\ &\leq \frac{2}{3} D_H^2(M(\pi), (\mu_t)_{o|\pi}) + \frac{2}{3} D_H^2(\mu_t, \rho_t) \\ &\leq \frac{2}{3} \text{KL}(M(\pi), (\mu_t)_{o|\pi}) + \frac{2}{3} \text{KL}(\mu_t, \rho_t), \end{aligned}$$

where the first inequality is by Lemma 4.12; the second inequality is by Lemma 4.9; and the last inequality is by Lemma 4.8. Finally, by Equation (75) we have that

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \mathbb{E} \left[ \sum_{t=1}^T \mathbb{E}_{\mu_t, p_t} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), (\rho_t)_{o|\pi}) - \frac{1}{3\eta} \text{KL}(\mu_t, \rho_t) \right] \right].$$

Therefore, we prove Theorem 7.3.  $\square$

### 3.6 Proof of Theorem 7.4

**THEOREM 7.4 (REGRET OF MODEL-INDEX ADAPTIVE POSTERIOR SAMPLING).** *Given a finite model class  $\mathcal{M}$  where  $f_M(\pi) \in [0, 1]$  for every  $M \in \mathcal{M}$  and  $\pi \in \Pi$ . The regret of Algorithm 8 with  $\eta \in (0, 1/3]$  is bounded as follows, for all  $T \in \mathbb{N}_+$ ,*

$$\mathfrak{R}_T \leq \frac{\log |\mathcal{M}|}{\eta} + \text{DEC}_{6\eta}^{\text{TS}}(\mathcal{M}, \bar{M}) \cdot T + 6\eta T.$$

**PROOF OF THEOREM 7.4:** Consider the Bayesian posterior sampling strategy induced by  $\mu \in \Delta(M)$ , which samples  $M \sim \mu$  and plays  $\pi_M$ . Denote the induced decision probability as

$$\mu_{\pi_M}(\pi) = \sum_{M \in \mathcal{M}, \pi_M = \pi} \mu(M).$$

For arbitrary  $\mu \in \Delta(M)$ , denote the  $|\Pi|$ -dimensional vectors  $X, Y$  by

$$\begin{aligned} X(\pi) &= \mathbb{E}_\mu [f_M(\pi_M) - f_M(\pi)], \\ Y(\pi) &= \mathbb{E}_\mu \left[ D_H^2(M(\pi), (\rho)_{o|\pi}) \right]. \end{aligned}$$

Then

$$\begin{aligned}
& \mathbb{E}_{M \sim \mu, \pi \sim \rho_{\pi_M}} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{3\eta} D_H^2(M(\pi), \rho_{o|\pi}) \right] - \frac{1}{3\eta} \text{KL}(\mu, \rho) \\
& \leq \langle \rho_{\pi_M}, X \rangle - \frac{1}{3\eta} \langle \rho_{\pi_M}, Y \rangle - \frac{1}{3\eta} \text{KL}(\mu_{\pi_M}, \rho_{\pi_M}) \\
& \leq \langle \mu_{\pi_M}, X \rangle + 6\eta - \frac{1}{3\eta} \langle \rho_{\pi_M}, Y \rangle - \frac{1}{6\eta} \text{KL}(\mu_{\pi_M}, \rho_{\pi_M}) \\
& \leq \langle \mu_{\pi_M}, X \rangle + 6\eta - \frac{1}{3\eta} \langle \rho_{\pi_M}, Y \rangle - \frac{1}{6\eta} D_H^2(\mu_{\pi_M}, \rho_{\pi_M}) \\
& \leq \langle \mu_{\pi_M}, X \rangle - \frac{(1-\eta)}{3(1+\eta)\eta} \langle \mu_{\pi_M}, Y \rangle + 6\eta \\
& \leq \mathbb{E}_{\mu, \pi \sim \mu_{\pi_M}} \left[ f_M(\pi_M) - f_M(\pi) - \frac{1}{6\eta} D_H^2(M(\pi), (\rho)_{o|\pi}) \right] + 6\eta,
\end{aligned}$$

where the first inequality is by the data processing inequality (Lemma 4.9); the second inequality is by Lemma 4.11 and the fact  $f_M(\pi) \in [0, 1]$  for all  $M$  and  $\pi$ ; the third inequality is by Lemma 4.8; the fourth inequality is a consequence of Lemma 4.10 and the AM-GM inequality; and the last inequality uses the condition  $\eta \leq \frac{1}{3}$ .

Combining the above inequality with Theorem 7.3, we prove Theorem 7.4.  $\square$

### 3.7 Proof of Theorem 3.3

**THEOREM 7.6** (Regret of MAMS and MAPS for RL Problems in the Bilinear Class). *In the on-policy case, Model-index AMS (Algorithm 3) and Model-index APS (Algorithm 2) with optimally tuned  $\eta$  achieve regret*

$$\mathfrak{R}_T \leq O(H^2 L_{bi}^2 d_{bi}(\mathcal{M}) \cdot T \cdot \log |\mathcal{M}|).$$

*In the general case, Model-index AMS (Algorithm 3) and Model-index APS (Algorithm 2) with forced exploration rate  $\gamma = (8\eta H^3 L_{bi}^2(\mathcal{M}) d_{bi}(\mathcal{M}, \bar{M}))^{1/2}$  and optimally tuned  $\eta$  achieves regret*

$$\mathfrak{R}_T \leq O((H^3 L_{bi}^2 d_{bi}(\mathcal{M}) \log |\mathcal{M}|)^{1/3} \cdot T^{2/3}).$$

**PROOF OF THEOREM 3.3:** By Theorem 7.1 in [23], we have upper bounds for  $\text{DEC}_\eta^{\text{TS}}$  as follows.

**LEMMA 3.4 (UPPER BOUNDS FOR BILINEAR CLASS REINFORCEMENT LEARNING).** *Let  $\mathcal{M}$  be a bilinear class and let  $\bar{M} \in \text{conv}(\mathcal{M})$ . Let  $\mu \in \Delta(\mathcal{M})$  be given, and consider the modified Bayesian posterior sampling strategy that samples  $M \sim \mu$  and plays  $\pi_M^\alpha$ , where  $\alpha \in [0, 1]$  is a parameter.*

(1) *If  $\pi_M^{\text{est}} = \pi_M$  (i.e., estimation is on-policy), this strategy with  $\alpha = 0$  certifies that*

$$\text{DEC}_\eta^{\text{TS}}(\mathcal{M}, \bar{M}) \leq 4\eta H^2 L_{bi}^2(\mathcal{M}) d_{bi}(\mathcal{M}; \bar{M})$$

*for all  $\eta > 0$ .*

(2) *For general estimation policies, this strategy with  $\gamma = (8\eta H^3 L_{bi}^2(\mathcal{M}) d_{bi}(\mathcal{M}, \bar{M}))^{1/2}$  certifies that*

$$\text{DEC}_\eta^{\text{TS}}(\mathcal{M}, \bar{M}) \leq (32\eta H^3 L_{bi}^2(\mathcal{M}) d_{bi}(\mathcal{M}; \bar{M}))^{1/2}.$$

*whenever  $\gamma \geq 32H^3 L_{bi}^2(\mathcal{M}) d_{bi}(\mathcal{M}, \bar{M})$ .*

By applying the upper bounds on  $\text{DEC}_\eta^{\text{TS}}$  from Lemma 3.4 to Theorem 3.3 and Theorem 7.4, we immediately prove the regret bounds in Theorem 3.3 for Model-index AMS and Model-index APS.  $\square$

## 4 Technical Background

### 4.1 Conditional Entropy

In the discussion after Definition 2.3, we utilize the following result stating that conditional entropy is concave. The reference provides a succinct proof to this result.

LEMMA 4.1 (CONDITIONAL ENTROPY OF A PROBABILITY MEASURE IS CONCAVE, [49]). *Let  $\mathbb{P}$  be a probability measure on locally compact space  $\mathcal{X}$ , and let  $\mathfrak{E}, \mathfrak{F}$  be countable partitions of the space. Define the entropy with respect to the partition  $\mathfrak{E}$  as*

$$H(\mathbb{P}, \mathfrak{E}) = - \sum_{E \in \mathfrak{E}} \mathbb{P}(E) \log \mathbb{P}(E),$$

*and the conditional entropy as*

$$H(\mathbb{P}, \mathfrak{E}|\mathfrak{F}) = \sum_{F \in \mathfrak{F}} \mathbb{P}(F) H(\mathbb{P}(\cdot|F), \mathfrak{E}).$$

*Then the conditional entropy  $H(\mathbb{P}, \mathfrak{E}|\mathfrak{F})$  is a concave function with respect to  $\mathbb{P}$ .*

### 4.2 Minimax Theorem

We introduce the classical minimax theorem for convex-concave game.

LEMMA 4.2 (SION'S MINIMAX THEOREM FOR VALUES, [48]). *Let  $\mathcal{X}$  and  $\mathcal{Y}$  be convex and compact sets, and  $\psi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  a function which for all  $y \in \mathcal{Y}$  is convex and continuous in  $x$  and for all  $x \in \mathcal{X}$  is concave and continuous in  $y$ . Then*

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \psi(x, y) = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \psi(x, y).$$

### 4.3 Danskin's Theorem

We reference Danskin's theorem, and the version we apply in this context is a consequence of the general result in [8].

LEMMA 4.3 (DANSKIN'S THEOREM). *Let  $\mathcal{X}$  be a convex and compact subset of a topological space,  $\mathcal{Y}$  be a convex and compact subset of a Hilbert space, and  $\phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  a function such that for all  $y \in \mathcal{Y}$ ,  $\phi(\cdot, y)$  is a strongly convex in  $x$ ; and for all  $x \in \mathcal{X}$ ,  $\phi(x, \cdot)$  is a continuously differentiable function in  $y$ . For each  $y \in \mathcal{Y}$ , let  $x_y$  be the unique minimizer of the optimization problem*

$$\min_{x \in \mathcal{X}} \phi(x, y).$$

*Then for all  $y \in \mathcal{Y}$ , we have*

$$\nabla_y \phi(x_y, y) = \left. \frac{\partial \phi(x, y)}{\partial y} \right|_{x=x_y}.$$

The directional derivative version of this lemma follows from Theorem D1 in [8], which imposes the following requirements:  $\mathcal{X}$  is a compact subset of a topological space,  $\mathcal{Y}$  is a subset a Banach space,  $\phi$  has minimizers in  $x$ , and  $\phi$  satisfies various upper semi-continuity conditions in  $y$ . If  $\phi$  is differentiable in  $y$ , then directional derivatives exist along every direction and are equal to the inner product between the direction and the derivative of  $\phi$ . This leads us to Lemma 4.3 by the property of inner product. In our application of the lemma,  $\mathcal{X}$  is a functional space,  $\mathcal{Y}$  is a probability simplex, and  $\phi$  exhibits strong convexity in  $x$  while being linear in  $y$ . In fact, slightly generalizing Danskin's theorem for the differentiable case in [9] to a general compact  $\mathcal{X}$  suffices for our purpose. Beyond its applicability to general theorems, we frequently utilize this fact in our calculations as a tool for simplifying computations (without the need to check any conditions in practical calculations).

Intuitively (although not fully rigorous due to differentiability considerations), Lemma 4.3 can be understood as follows:

$$\nabla_y \phi(x_y, y) = \frac{\partial \phi(x, y)}{\partial x} \Big|_{x=x_y} \cdot \nabla_y x_y + \frac{\partial \phi(x, y)}{\partial y} \Big|_{x=x_y} = \frac{\partial \phi(x, y)}{\partial y} \Big|_{x=x_y},$$

where the first equality follows from the elementary rules of derivative and the second equality arises from the optimality condition

$$\frac{\partial \phi(x, y)}{\partial x} \Big|_{x=x_y} = 0.$$

#### 4.4 Convex Analysis

We review some background in convex analysis, and refer to Part V in [42] for their proofs.

*Definition 4.4 (Legendre Function, [35]).* A proper convex function  $f : \mathbb{R}^d \rightarrow \mathbb{R} \cup \infty$  is Legendre if

- (1)  $\mathcal{D} = \text{int}(\text{dom}(\Psi))$  is non-empty;
- (2)  $\Psi$  is differentiable and strictly convex on  $\mathcal{D}$ ; and
- (3)  $\lim_{n \rightarrow \infty} \|\nabla f(u_n)\|_2 = \infty$  for any sequence  $\{u_n\}$  with  $u_n \in \mathcal{D}$  for all  $n$  and  $\lim_{n \rightarrow \infty} u_n = u$  and some  $u \in \partial \mathcal{D}$ .

It is known that the minimizer of a Legendre function is always in its interior:

**LEMMA 4.5 (MINIMIZER OF LEGENDRE FUNCTION IS IN ITS INTERIOR, [35]).** *If  $\Psi$  is Legendre and  $u \in \arg \min_{u \in \text{dom}(\Psi)} \Psi(u)$ , then  $u^* \in \text{int}(\text{dom}(\Psi))$ .*

We assume the  $d$ -dimensional set  $\mathcal{W} \subset \text{dom}(\Psi) := \{u \in \mathbb{R}^d : \Psi(u) < \infty\}$  has bounded diameter, i.e.,

$$\text{diam}(\mathcal{W}) := \sup_{u, v \in \mathcal{W}} |\Psi(u) - \Psi(v)| < \infty.$$

We denote the Fenchel–Legendre dual of  $\Psi$  as

$$\Psi^*(a) = \sup_{u \in \mathcal{W}} \langle a, u \rangle - \Psi(u), \quad \forall a \in \mathbb{R}^d,$$

and denote the Bregman divergences with respect to  $\Psi$  and  $\Psi^*$  as

$$\begin{aligned} D_\Psi(u, v) &= \Psi(u) - \Psi(v) - \langle \nabla \Psi(v), u - v \rangle, \\ D_{\Psi^*}(a, b) &= \Psi^*(a) - \Psi^*(b) - \langle \nabla \Psi^*(b), a - b \rangle. \end{aligned}$$

We have the following properties of Legendre functions.

**LEMMA 4.6 (PROPERTIES OF LEGENDRE FUNCTION, [35]).** *If  $\Psi$  is a Legendre function, then*

- (a)  $\nabla \Psi$  is a bijection between  $\text{int}(\text{dom}(\Psi))$  and  $\text{int}(\text{dom}(\Psi^*))$  with the inverse  $(\nabla \Psi)^{-1} = \nabla \Psi^*$ . That is, for  $u \in \text{int}(\text{dom}(\Psi))$ , if  $a = \nabla \Psi(u)$ , then  $a \in \text{int}(\text{dom}(\Psi^*))$  and  $\nabla \Psi^*(a) = u$ ;
- (b)  $D_\Psi(u, v) = D_{\Psi^*}(\nabla \Psi(v), \nabla \Psi(u))$  for all  $u, v \in \text{int}(\text{dom}(\Psi))$ ;
- (c) the Fenchel conjugate  $\Psi^*$  is Legendre; and

We also introduce the generalized Pythagorean theorem of Bregman divergence.

**LEMMA 4.7 (GENERALIZED PYTHAGOREAN THEOREM).** *For a convex function  $\Psi : \mathcal{W} \rightarrow \mathbb{R} \cup \infty$  and  $u, v, w \in \mathcal{W}$ , we have*

$$D_\Psi(u, w) = D_\Psi(u, v) + D_\Psi(v, w) + \langle u - v, \nabla \Psi(v) - \nabla \Psi(w) \rangle.$$

#### 4.5 Information Theory

We have the following result stating that the squared Hellinger distance between two probability measures are smaller than the KL divergence between those two probability measures.

LEMMA 4.8 (SQUARED HELLINGER DISTANCE SMALLER THAN KL DIVERGENCE, [41]). *For probability measures  $\mathbb{P}$  and  $\mathbb{Q}$ , the following inequalities hold:*

$$D_H^2(\mathbb{P}, \mathbb{Q}) \leq \text{KL}(\mathbb{P}, \mathbb{Q}).$$

For general  $f$ -divergences (which include KL divergence and squared Hellinger distance), we have the following data processing inequality.

LEMMA 4.9 (DATA PROCESSING INEQUALITY, [41]). *Consider a channel that produces  $Y$  given  $X$  based on the conditional law  $\mathbb{P}_{Y|X}$ . Let  $\mathbb{P}_Y$  (resp.  $\mathbb{Q}_Y$ ) denote the distribution of  $Y$  when  $X$  is distributed as  $\mathbb{P}_X$  (resp.  $\mathbb{Q}_X$ ). For any  $f$ -divergence  $D_f(\cdot||\cdot)$ ,*

$$D_f(\mathbb{P}_Y \parallel \mathbb{Q}_Y) \leq D_f(\mathbb{P}_X \parallel \mathbb{Q}_X).$$

We introduce a localized version of Pinsker-type inequality using squared Hellinger distance (which will be stronger than using the KL divergence).

LEMMA 4.10 (MULTIPLICATIVE PINSKER-TYPE INEQUALITY FOR HELLINGER DISTANCE, [23]). *Let  $\mathbb{P}$  and  $\mathbb{Q}$  be probability measures on compact space  $\mathcal{X}$ . For all  $h : \mathcal{X} \rightarrow \mathbb{R}$  with  $0 \leq h(X) \leq R$  almost surely under  $\mathbb{P}$  and  $\mathbb{Q}$ , we have*

$$|\mathbb{E}_{\mathbb{P}}[h(X)] - \mathbb{E}_{\mathbb{Q}}[h(X)]| \leq \sqrt{2R(\mathbb{E}_{\mathbb{P}}[h(X)] + \mathbb{E}_{\mathbb{Q}}[h(X)]) \cdot D_H^2(\mathbb{P}, \mathbb{Q})}.$$

We introduce a standard one-sided bound using KL divergence. Compared with Lemma 4.10, the upper bound in Lemma 4.11 only depends on the probability measure  $q$ , while the bound is one-sided and it does not take the square-root form as in Lemma 4.10.

LEMMA 4.11 (DRIFTED ERROR BOUND USING KL DIVERGENCE). *For any  $p, q \in \Delta(\Pi)$ ,  $\eta > 0$ , and any vector  $y \in \mathbb{R}^\Pi$  where  $y(\pi) \leq 1/\eta$  for all  $\pi \in \Pi$ , we have*

$$\langle y, p - q \rangle - \frac{1}{\eta} \text{KL}(p, q) \leq \eta \sum_{\pi \in \Pi} q(\pi) y(\pi)^2.$$

PROOF OF LEMMA 4.11: Consider the KL divergence  $\psi_{q,\eta}(p) = \frac{1}{\eta} \text{KL}(p||q)$ , it is known that the convex conjugate duality of  $\psi_q$  is the log partition function

$$\begin{aligned} \psi_{q,\eta}^*(y) &:= \sup_{p \in \Delta(\Pi)} \left\{ \langle y, p \rangle - \frac{1}{\eta} \text{KL}(p||q) \right\} \\ &= \frac{1}{\eta} \log \left( \sum_{\pi \in \Pi} q(\pi) \exp(\eta y(\pi)) \right). \end{aligned} \tag{76}$$

We have

$$\begin{aligned}
& \langle y, p \rangle - \frac{1}{\eta} \text{KL}(p||q) \\
& \leq \frac{1}{\eta} \log \left( \sum_{\pi \in \Pi} q \exp(\eta y(\pi)) \right) \\
& \leq \frac{1}{\eta} \log \left( \sum_{\pi} q(\pi) (1 + \eta y(\pi) + \eta^2 y(\pi)^2) \right) \\
& = \frac{1}{\eta} \log \left( 1 + \eta \langle y, q \rangle + \eta^2 \sum_{\pi \in \Pi} q(\pi) y(\pi)^2 \right) \\
& \leq \langle y, q \rangle + \eta \sum_{\pi \in \Pi} q(\pi) y(\pi)^2,
\end{aligned} \tag{77}$$

where the first equality is because of Equation (76); the second inequality is because  $e^z \leq 1 + z + z^2$  for all  $z \leq 1$  and the last inequality is due to  $\log(1 + z) \leq z$  for all  $z \in \mathbb{R}$ . Therefore we have

$$\langle y, p - q \rangle - \frac{1}{\eta} \text{KL}(p||q) \leq \eta \sum_{\pi \in \Pi} q(\pi) y(\pi)^2$$

for all  $y \in \mathbb{R}^{|\Pi|}$  where  $y(\pi) \leq 1/\eta$  for all  $\pi$ .

□

By the fact that the Hellinger distance satisfies the canonical triangle inequality, we have the following triangle-type inequality for the squared Hellinger distance.

**LEMMA 4.12 (TRIANGLE-TYPE INEQUALITY FOR SQUARED HELLINGER DISTANCE).** *For every distributions  $p, \bar{p}, q \in \Delta(\mathcal{X})$ , we have*

$$D_H^2(p, \bar{p}) \leq 2(D_H^2(p, q) + D_H^2(q, \bar{p})).$$

As a result, given any mixture  $\mu \in \Delta(\Delta(\mathcal{X}))$ , for every  $q \in \Delta(\mathcal{X})$ , we have

$$\mathbb{E}_{p \sim \mu, \bar{p} \sim \mu} [D_H^2(p, \bar{p})] \leq 4\mathbb{E}_{p \sim \mu} [D_H^2(p, q)].$$

Received 21 February 2024; revised 29 July 2025; accepted 3 August 2025



**Journal of the ACM**  
<https://jacm.acm.org/>

**Guide to Manuscript Submission**

Submission to the *Journal of the ACM* is done electronically through <https://mc.manuscriptcentral.com/jacm>. Once you are at that site, you can create an account and password with which you can enter the ACM Manuscript Central manuscript review tracking system. Proceed to the Author Center to submit your manuscript and your accompanying files.

You will be asked to create an abstract that will be used throughout the system as a synopsis of your paper. You will also be asked to classify your submission using the ACM Computing Classification System through a link provided at the Author Center. For completeness, please select at least one primary-level classification followed by two secondary-level classifications. To make the process easier, you may cut and paste from the list. Remember, you, the author, know best which area and sub-areas are covered by your paper; in addition to clarifying the area where your paper belongs, classification often helps in quickly identifying suitable reviewers for your paper. So it is important that you provide as thorough a classification of your paper as possible.

The ACM Production Department prefers that your manuscript be prepared in either LaTeX or MS Word format. Style files for manuscript preparation can be obtained at the following location: <https://www.acm.org/publications/authors/submissions>. For editorial review, the manuscript should be submitted as a PDF or Postscript file. Accompanying material can be in any number of text or image formats, as well as software/documentation bundles in zip or tar-gzipped formats.

Questions regarding editorial review process should be directed to the Editor-in-Chief. Questions regarding the post-acceptance production process should be addressed to the Editor, Yubing Zhai at [zhai@hq.acm.org](mailto:zhai@hq.acm.org).

**Subscription and Membership Information.**

Send orders to:

ACM Member Services Dept.  
General Post Office  
PO Box 30777  
New York, NY 10087-0777

For information, contact:

**Mail:** ACM Member Services Dept.  
1601 Broadway, 10th Floor  
New York, NY 10019-7434  
**Phone:** +1-212-626-0500  
**Fax:** +1-212-944-1318  
**Email:** [acmhelp@acm.org](mailto:acmhelp@acm.org)  
**Catalog:** <https://www.acm.org/publications/alacarte>

**About ACM.** ACM is the world's largest educational and scientific computing society, uniting educators, researchers and professionals to inspire dialogue, share resources and address the field's challenges. ACM strengthens the computing profession's collective voice through strong leadership, promotion of the highest standards, and recognition of technical excellence. ACM supports the professional growth of its members by providing opportunities for life-long learning, career development, and professional networking.

**Visit ACM's Web site:** <https://www.acm.org>.

## Economics and Computation

- Article 34**   **A. Hollender**   Envy-Free Cake-Cutting for Four Agents  
(54 pages)   **A. Rubinstein**

## Computational Complexity and Algorithms

- Article 35**   **A. Wein**   The Kikuchi Hierarchy and Tensor PCA  
(40 pages)   **A. El Alaoui**  
                 **C. Moore**

## Learning Theory

- Article 36**   **Z. Zhang**   Optimal Multi-Distribution Learning  
(71 pages)   **W. Zhan**  
                 **Y. Chen**  
                 **S. S. Du**  
                 **J. Lee**

- Article 37**   **Y. Xu**   Bayesian Design Principles for Frequentist Sequential  
(65 pages)   **A. Zeevi**   Learning