The Aldous–Lyons Conjecture II: Undecidability

Lewis Bowen

lpbowen@math.utexas.edu

Michael Chapman

mc9578@nyu.edu

Thomas Vidick

thomas.vidick@weizmann.ac.il

January 3, 2025

Abstract

This paper, and its companion [BCLV24], are devoted to a negative resolution of the Aldous–Lyons Conjecture [AL07, Ald07].

In this part we study *tailored non-local games*. This is a subclass of *non-local games* — combinatorial objects which model certain experiments in quantum mechanics, as well as interactive proofs in complexity theory. Our main result is that, given a tailored non-local game $\mathfrak{G}$, it is undecidable to distinguish between the case where $\mathfrak{G}$ has a special kind of perfect strategy, and the case where every strategy for $\mathfrak{G}$ is far from being perfect. Using a reduction introduced in the companion paper [BCLV24], this undecidability result implies a negative answer to the Aldous–Lyons conjecture. Namely, it implies the existence of unimodular networks that are non-sofic.

To prove our result, we use a variant of the *compression* technique developed in $\mathrm{MIP}^* = \mathrm{RE}$ [JNV$^+$21]. Our main technical contribution is to adapt this technique to the class of tailored non-local games. The main difficulty is in establishing *answer reduction*, which requires a very careful adaptation of existing techniques in the construction of probabilistically checkable proofs. As a byproduct, we are reproving the negation of Connes' embedding problem [Con76] — i.e., the existence of a II_1-factor which cannot be embedded in an ultrapower of the hyperfinite II_1-factor — first proved in [JNV$^+$21], using an arguably more streamlined proof. In particular, we incorporate recent simplifications from the literature [dlS22b, Vid22] due to de la Salle and the third author.

Contents

1 Introduction

In Part I [BCLV24] we proved that if the following theorem is true then there are non-sofic unimodular networks, resolving the Aldous–Lyons conjecture [AL07] in the negative:

Theorem 1.1 (Main Theorem. See Theorem 2.31 for a formal version. Compare to Theorem 7.4 in [BCLV24])**.** *There exists a polynomial time algorithm that takes as input a Turing machine $\mathcal{M}$ and outputs a **tailored** non-local game $\mathfrak{G}_{\mathcal{M}}$ such that:*

1. Completeness*: If $\mathcal{M}$ halts then there exists a perfect $\mathbb{Z}$-**aligned permutation strategy that commutes along edges** for $\mathfrak{G}_{\mathcal{M}}$.*

2. Soundness*: If $\mathcal{M}$ never halts then the **synchronous** quantum value of $\mathfrak{G}_{\mathcal{M}}$ is bounded from above by $1/2$.*

The reader who is unfamiliar with the study of non-local games should not be discouraged, as all definitions regarding the above theorem are explained within this introduction. The reader familiar with the work $\mathrm{MIP}^* = \mathrm{RE}$ by Ji–Natarajan–Vidick–Wright–Yuen [JNV$^+$21] notices that the above theorem is very similar to their main theorem. Actually, it is a strengthening of their result, namely, this paper reproves that the class of multi-prover interactive proofs with entangled provers contains the Halting problem, which implies a negative solution to Connes' embedding problem [Con76] (see also [Bro06, Proposition 6.3.5]) as well as to Tsirelson's problem [Tsi06]. In the statement above we emphasized the main differences in bold. Elaborating on these differences:

- The game $\mathfrak{G}_{\mathcal{M}}$ must belong to the class of *tailored non-local games*, which is a strict subclass of the synchronous games used in [JNV$^+$21]. Tailored games are a generalization of an important class of games considered in the literature, called *linear constraint system games* (LCS, see [CM14, KPS18]).

- The provers are only allowed to use *synchronous* [PSS$^+$16] quantum strategies.

- The allowed perfect strategies for $\mathfrak{G}_{\mathcal{M}}$ in the complete case, $\mathbb{Z}$-*aligned permutation strategies that commute along edges* (ZPC strategies), is a stricter subfamily of the PCC (projective, consistent and commuting) strategies used in the complete case in [JNV$^+$21].

Because the class of games considered is more restricted, and because the class of strategies available to show the completeness property is more limited, Theorem 2.31 is more difficult to show than the corresponding reduction from [JNV$^+$21]. (The restriction to synchronous strategies in the soundness case does play in our favor; however, as we shall see later, this restriction has relatively mild and well-understood consequences.)

The following few subsections of the introduction recall various notions from the theory of non-local games and then introduce the class of tailored games. In the process, tailored games are suggested as a "middle ground" between synchronous games (which were used in [JNV$^+$21]) and linear constraint system games; we specifically address a folklore effort to "linearize" $\mathrm{MIP}^* = \mathrm{RE}$ (cf. [PS23]) — which would have resulted in the existence of non-hyperlinear groups, and thus refute the Aldous–Lyons conjecture — and offer our approach as "semi-linearization". Finally, our proof method is discussed, and in particular the similarities and differences between this work and [JNV$^+$21].

We do not motivate or survey the Aldous–Lyons conjecture nor Connes' embedding problem; even the complexity theoretic aspects of our strengthened version of $\mathrm{MIP}^* = \mathrm{RE}$ are discussed only briefly. Such motivational introductions are already provided both in our companion paper [BCLV24], for the Aldous–Lyons conjecture, and in [JNV$^+$21], for $\mathrm{MIP}^* = \mathrm{RE}$ and Connes' embedding problem.

Non-local games. A non-local game consists of two finite sets X, A, a probability distribution μ over $X \times X$, and a decision predicate $D\colon X \times X \times A \times A \to \{0, 1\}$. The set X is commonly called the *question set* and A the *answer set*.[1] The game is called *synchronous* if $D(\mathbf{x}, \mathbf{x}, a, b) = 0$ for every $\mathbf{x} \in X$ and $a \neq b \in A$; this condition will always be satisfied for us.

---

[1] The answer set may depend on the specific question $\mathbf{x} \in X$, namely, when $\mathbf{x}$ is asked, the allowed answers are from $A_{\mathbf{x}} \subseteq A$. For simplicity, in the introduction, we ignore such dependence.

The data $\mathfrak{G} = (X, A, \mu, D)$ is called a "game" because of the following interpretation. We may imagine a referee challenging two players, colloquially referred to as "Alice" and "Bob", by sending them a pair (x, y) that was sampled according to the distribution μ, such that Alice receives x and Bob receives y. Alice then has to respond with some $a \in A$, while Bob has to respond with some $b \in A$. The players are said to win if and only if $D(x, y, a, b) = 1$.

Given a game $\mathfrak{G}$, its *value* is defined as the maximum probability, over the referee's choice of a pair of questions and the players' choice of an answer, that the players win the game. To make this formal, one needs to specify how the players may determine their answers, i.e., to define the class of allowed strategies for them. This is where things get interesting, as there are several natural choices. The most restricted choice is to require the players to choose a function $f \colon X \to A$ and return $a = f(x)$ and $b = f(y)$. For any game $\mathfrak{G}$, maximizing the players' success probability over all such functions leads to what is known as the (synchronous) *classical value* $\mathrm{val}(\mathfrak{G})$ of the game. Concretely,

$$\mathrm{val}(\mathfrak{G}) = \max_{f \colon X \to A} \left(\sum_{x, y \in X} \mu(x, y) D(x, y, f(x), f(y)) \right).$$

It is not hard to see that allowing "randomized" functions, namely letting the players choose f according to some distribution, does not change the value.

In full generality, a strategy for the players is specified by a *correlation*, which is a family of distributions $p(\cdot, \cdot | x, y)$ on $A \times A$, for every pair $(x, y) \in X \times X$. The restriction considered in the previous paragraph leads to the family of (synchronous) classical strategies. Let us give two other examples of families of strategies. The first example is known as *synchronous quantum strategies*. To define these, first recall the notion of a *projective valued measure* (PVM). A PVM is a collection of operators $\{\mathcal{P}_a\}_{a \in A}$ acting on a Hilbert space $\mathcal{H}$, where A is any finite set, the operators $\mathcal{P}_a$ are orthogonal projections ($\mathcal{P}_a^* = \mathcal{P}_a = \mathcal{P}_a^2$), and $\sum_a \mathcal{P}_a = \mathrm{Id}_{\mathcal{H}}$. A synchronous quantum strategy is then specified by a *finite-dimensional* Hilbert space $\mathcal{H}$ and PVMs $\{\mathcal{P}_a^x\}_{a \in A}$ for each $x \in X$. Such a strategy is said to be *commuting along edges*[2] (or just commuting in [JNV+21]), if for every pair of questions (x, y) that can be sampled in the game (namely, in the support of μ), the projections $\mathcal{P}_a^x$ and $\mathcal{P}_b^y$ commute for every $a, b \in A$. The correlation that the strategy $\mathcal{P}$ induces is

$$p(a, b | x, y) = \tau(\mathcal{P}_a^x \mathcal{P}_b^y), \tag{1}$$

where τ is the dimension-normalized trace on $\mathcal{H}$. The resulting maximum success probability is called the (synchronous) *quantum value* of the game and is denoted by $\mathrm{val}^*(\mathfrak{G})$. Concretely,

$$\mathrm{val}^*(\mathfrak{G}) = \sup_{\mathcal{H}, \{\mathcal{P}_a^x\}} \left(\sum_{x, y} \mu(x, y) \sum_{a, b} \tau(\mathcal{P}_a^x \mathcal{P}_b^y) D(x, y, a, b) \right). \tag{2}$$

In general, $\mathrm{val}(\mathfrak{G}) \leq \mathrm{val}^*(\mathfrak{G})$ always holds, and furthermore the inequality can be strict.[3] This is demonstrated, for example, by the *magic square game* described in Example 2.30. The fact that $\mathrm{val}(\mathfrak{G}) < \mathrm{val}^*(\mathfrak{G})$ is interpreted as a witness of the *non-locality* of quantum mechanics. It has led to experiments (e.g. [HBD+15]) which verify that the quantum mechanical prediction for $\mathrm{val}^*(\mathfrak{G})$ is indeed achievable using a physical system (such as a pair of photons). Such experiments demonstrate that non-classical aspects of quantum mechanics are necessary to explain the physical world.

In this paper, we force the perfect strategies in the complete case to be *Z-aligned permutation strategies that commute along edges*, or ZPC strategies for short. Let us define this subfamily of synchronous quantum strategies. Assume that the answer set of the game is $A = \mathbb{F}_2^\Lambda$ for some fixed integer Λ.[4] In a *permutation strategy*, a finite set Ω is chosen, and we let $\Omega_\pm = \{\pm\} \times \Omega$ be the signed version of Ω, and $\sigma_J \in \mathrm{Sym}(\Omega_\pm)$ be the sign flip; namely $\sigma_J(\pm, \star) = (\mp, \star)$ for every $\star \in \Omega$.[5] Then, to each $x \in X$, a family of Λ pairwise commuting, involutive permutations that commute with the sign flip $\{\sigma_{x,i}\}_{i=1}^\Lambda \subseteq \mathrm{Sym}(\Omega_\pm)$ are associated — this is the same as choosing for every vertex x, a *signed permutations* representation

---

[2]The reason for the name commuting along edges, is that the support of μ induces a graph structure on X, and the condition is indeed that PVMs that are associated with neighboring vertices must commute.

[3]The first to provide an example with a strict inequality was John Bell in [Bel64].

[4]In general we allow Λ to depend on the question x.

[5]We later denote this sign flip by $-\mathrm{Id}$ instead of σ_J, but for the sake of clarity decided on this different notation in the introduction.

of $\mathbb{F}_2^\Lambda$ (acting on $\Omega_\pm$). Using the natural embedding of permutations acting on $\Omega_\pm$ in the unitary matrices acting on $\mathbb{C}^{\Omega_\pm}$, and as all the $\sigma_{x,i}$'s commute with the sign flip permutation, it can be checked that

$$\forall x \in X, \ a \in \mathbb{F}_2^\Lambda : \quad \mathcal{P}_a^x = \frac{\mathrm{Id} - \sigma_J}{2} \cdot \prod_{i=1}^\Lambda \left(\frac{\mathrm{Id} + (-1)^{a_i}\sigma_{x,i}}{2} \right) \tag{3}$$

induces PVMs on the $|\Omega|$-dimensional Hilbert space $\mathcal{H} = \frac{\mathrm{Id}-\sigma_J}{2}\mathbb{C}^{\Omega_\pm}$, which is the space of anti-symmetric functions from $\Omega_\pm$ to $\mathbb{C}$, namely functions satisfying $f(-,\star) = -f(+,\star)$ for every $\star \in \Omega$. These PVMs form a quantum strategy $\mathcal{P}$ called the *quantum strategy associated with the permutation strategy* σ. The permutation strategy σ is said be *commuting along edges* if the associated $\mathcal{P}$ is commuting along edges. The correlation $p(\cdot, \cdot|\cdot, \cdot)$ induced by the PVMs $\{\mathcal{P}_a^x\}$, as in (1), is said to be *induced* by the permutation strategy σ. In words, $p(a, b|x, y)$ is the relative dimension in $\mathcal{H}$ of the joint eigenspace of each $\sigma_{x,i}$ associated with eigenvalue $(-1)^{a_i}$, and of each $\sigma_{y,j}$ associated with eigenvalue $(-1)^{b_j}$. The notion of σ being Z-aligned can be described only after we introduce the class of tailored non-local games.

Tailored games. A tailored game is a non-local game that has the following structure. First, the answer set is $\mathbb{F}_2^{\Lambda_\mathfrak{R}} \times \mathbb{F}_2^{\Lambda_\mathfrak{L}}$, where $\Lambda_\mathfrak{R}$ and $\Lambda_\mathfrak{L}$ are integers, and let $\Lambda = \Lambda_\mathfrak{R} + \Lambda_\mathfrak{L}$.[6] So, the answer $a = (a^\mathfrak{R}, a^\mathfrak{L})$ to a question x consists of two parts: a *readable* part $a^\mathfrak{R}$ and an *unreadable* (or *linear*) part $a^\mathfrak{L}$. Furthermore, the decision procedure of a tailored game is required to be *controlled-linear*: Given a pair of questions (x, y) and answers $(a, b) = ((a^\mathfrak{R}, a^\mathfrak{L}), (b^\mathfrak{R}, b^\mathfrak{L}))$, it first reads only the pair $(a^\mathfrak{R}, b^\mathfrak{R})$, and depending on it returns a system of linear equations with $\mathbb{F}_2$-coefficients $L = L_{xy}(a^\mathfrak{R}, b^\mathfrak{R})$ over 2Λ variables. Then, the pair $(a, b) \in \mathbb{F}_2^{2\Lambda}$ is accepted by the decision procedure, namely $D(x, y, a, b) = 1$, if and only if L is satisfied by the assignment (a, b).

Of course, a tailored game such that the entire answer is marked as readable, i.e. $\Lambda_\mathfrak{R} = \Lambda$ for all questions x, is nothing but a general non-local game. For more restricted choices of $\Lambda_\mathfrak{R} < \Lambda$ to be useful, we need to describe the kinds of strategies which we consider for tailored games. A permutation strategy

$$\left\{ \sigma_{x,\mathfrak{R},i}, \sigma_{x,\mathfrak{L},j} \mid x \in X, \ i \in [\Lambda_\mathfrak{R}], \ j \in [\Lambda_\mathfrak{L}] \right\}$$

for a tailored non-local game, acting on $\Omega_\pm$, is said to be Z-*aligned* if the readable permutations act as controlled sign flips. I.e., for every $\star \in \Omega$, $i \in [\Lambda_\mathfrak{R}]$ and $x \in X$, the permutation $\sigma_{x,\mathfrak{R},i}$ maps the set $\{(+,\star), (-,\star)\}$ to itself. This means, in particular, that the readable permutations are mutually diagonalizable in the standard basis of $\mathcal{H} = \frac{\mathrm{Id}-\sigma_J}{2}\mathbb{C}^{\Omega_\pm}$, which consists of the functions $\mathbf{1}_{(+,\star)} - \mathbf{1}_{(-,\star)}$ (with $\mathbf{1}_.$ being the indicator function). A ZPC strategy for a tailored non-local game is a permutation strategy that commutes along edges and is Z-aligned, and the ZPC value of a game will be the maximum success probability of a ZPC strategy in the game. **The reader can now parse our main theorem**.

One can now see why the tailoring of $\mathfrak{G}$ may affect the value of a game if we restrict it to use only ZPC strategies: the more answer bits are marked as readable, the more restricted the class of strategies that is allowed; thus an "aggressive" tailoring (e.g. marking all answer bits as readable) may lead to a smaller ZPC-value, while a more "relaxed" tailoring of the same game would have higher ZPC-value. In fact, one can easily verify that, for any game $\mathfrak{G}$ such that $\Lambda_\mathfrak{R} = \Lambda$, the ZPC-value agrees with the classical one. Naturally, "fully relaxed" tailoring of a given game (e.g. marking all answer bits as unreadable) is not always possible, because the decision function D may simply not be linear. But, when such a relaxation is possible, the resulting game is said to be a *linear constraint system game* (LCS, [CM14, KPS18]). So, tailored games are a natural generalization of LCS games. LCS games are widely studied, and their values are related to approximation properties — such as hyperlinearity and soficity — of a certain finitely presented group associated with the LCS game. Let us say more about this subclass.

Linear constraint system games. LCS games are a restricted class of non-local games such that the function $D(x, y, a, b)$ is a conjunction of *linear* functions of its input (a, b), seen as an element of $\mathbb{F}_2^{2\Lambda}$. Namely, for every x, y that can be sampled

---

[6]In the formal definition, $\Lambda_\mathfrak{R}$ and $\Lambda_\mathfrak{L}$ may vary depending on the question x.

by μ, there is a system of linear equations $\mathscr{A}_{\mathrm{xy}}\vec{x} = \vec{c}_{\mathrm{xy}}$ with $\mathbb{F}_2$-coefficients and with $\vec{x} \in \mathbb{F}_2^{2\Lambda}$, and $D(\mathrm{x},\mathrm{y},a,b) = 1$ if and only if $\vec{x} = (a,b)$ is a solution to this system of equations.[7]

A natural C^*-algebra $\mathcal{A}(\mathfrak{G})$, known as the *game algebra*, can be associated to every synchronous game. In case $\mathfrak{G}$ is an LCS, $\mathcal{A}(\mathfrak{G})$ happens to be a group von Neumann algebra. Namely, there is a finitely presented group $\Gamma(\mathfrak{G})$, often referred to as the *solution group* (cf. [Slo19]), such that $\mathcal{A}(\mathfrak{G})$ is the von Neumann closure of (a quotient of) the group ring $\mathbb{C}[\Gamma(\mathfrak{G})]$.

There is an additional game value val_{qc}, known as the (synchronous) *quantum commuting value*, defined by taking the supremum as in (2) over all tracial von Neumann algebras $(\mathcal{M}, \tau)$ (instead of only finite dimensional ones). Using known connections between the existence of perfect strategies and $*$-homomorphisms of $\mathcal{A}(\mathfrak{G})$ [KPS18], it is a folklore result that the existence of an LCS game such that $\mathrm{val}_{qc}(\mathfrak{G}) = 1 > \mathrm{val}^*(\mathfrak{G})$ implies the existence of a non-hyperlinear group, which is thus non-sofic, and in turn refutes the Aldous–Lyons conjecture.

In MIP* = RE [JNV$^+$21], synchronous games that satisfy $\mathrm{val}_{qc}(\mathfrak{G}) = 1 > \mathrm{val}^*(\mathfrak{G})$ are constructed. Unfortunately, these games are not LCS. Moreover, it seems essential for some of the key steps of the construction from [JNV$^+$21] that the game decision function D is allowed to depend non-linearly on the answers a, b — this is due to the use of techniques from the field of efficient proof verification in computer science; we describe this obstacle in more detail when discussing *answer reduction* in Section 1.1. In turn, the results of [PS23] demonstrate that implementing the non-linear OR function cannot be done in a "naive" way using LCS only.

Now, tailored non-local games *are* allowed to have decision functions that depend non-linearly on the answers — at least, on the readable part of the answers. Crucially however, the form of strategies which we consider is required to be more limited, as a function of the tailoring. Thus, tailored non-local games are a broader class of games than LCS, but ones with a restricted class of strategies. The combination of these two ingredients allow us to carry through the proof approach from [JNV$^+$21] (because our class of games is sufficiently general) while, to some extent, maintaining the connection with group theory (through the reduction to subgroup tests proved in the companion paper [BCLV24]). However, we are not able to determine whether there exists a non-sofic group; that remains an open problem.

The complexity theoretic angle. Theorem 1.1 is formulated as a reduction from the problem of deciding if a Turing machine $\mathcal{M}$ halts to the problem of deciding if a game $\mathfrak{G}_{\mathcal{M}}$, that is polynomial-time computable from the description of $\mathcal{M}$, has ZPC value 1 or synchronous quantum value at most $\frac{1}{2}$. The existence of such a reduction can be reformulated succinctly as the equality of two complexity classes.

Let RE be the class of problems that are polynomial-time reducible to the Halting Problem. Here, RE stands for "recursively enumerable." An equivalent definition of RE is that it consists of all problems such that there is an algorithm which, given an instance of the problem, always terminates with the answer "yes" when indeed the answer should be yes; when the answer should be no, the algorithm can either say "no", or it is also allowed to never terminate.[8] Turing showed that the Halting Problem is a complete problem for this class.

Let TailoredMIP* be the class of languages that are polynomial-time reducible to the problem of deciding if the ZPC value of a tailored game provided as input is 1, or if its synchronous value is at most $\frac{1}{2}$ (given that one of these is promised to be the case).[9] Then Theorem 1.1 can be formulated succinctly as

$$\mathrm{TailoredMIP}^* = \mathrm{RE} \, .$$

Reformulated in this way, our result bears a clear analogy with the result MIP* = RE. It is also clear that it is a strengthening of the latter, as TailoredMIP$^* \subseteq$ MIP* (and the inclusion MIP$^* \subseteq$ RE is not hard; it is the reverse inclusion that requires work). Such characterization inscribes itself in a long tradition of complexity theory, where equalities such as IP = PSPACE [LFKN90, Sha90] or MIP = NEXP [BFL91] are taken as fundamental statements about the nature of computation, which tend to have important consequences in areas ranging from cryptography to hardness of approximation. In

---

[7]The formal definition of an LCS is slightly more restricted, see Example 2.29, but this generalized setup is essentially equivalent to the standard definition.

[8]We provide an overview of complexity classes in Section 5.1, which includes a formal definition of RE as well.

[9]To make this definition precise, one needs to clarify how a tailored game is represented; this is discussed in Section 2.5.

a different direction, extending the class of strategies allowed for the provers has led to analogues of $\text{MIP}^* = \text{RE}$ for higher classes of the arithmetical hierarchy [MNY22].

1.1 Proof ideas

While our proof follows the same template as [JNV$^+$21], and indeed re-uses the most important ideas therein, it is arguably more streamlined. In particular we are able to take advantage of some simplifications that were discovered after the publication of [JNV$^+$21]. Most notably, we take advantage of the fact that synchronous games can without loss of generality be analyzed by considering their synchronous value only [KPS18, Vid22],[10] and the simplification of [dlS22b] for the step of question reduction (further discussed below).

At the heart of our work is a result about *compression* of non-local games. Informally, compression reduces the size of a game (measured by the number of questions and answers) while preserving its quantum value. The fact that a form of compression implies undecidability as in Theorem 1.1 is very general, as shown in [MSNY24]. For this to be possible, of course, one must introduce certain computational considerations; in particular the procedure which achieves compression must be computable. For clarity of this introduction, we, for the most part, set computational aspects aside, and focus on compression as a *combinatorial* transformation. In this respect, the following is what needs to be done.

Compression. Let $N = 2^n$ for some integer n. Our starting point is a tailored game $\mathfrak{G}$, that has questions and answers of length N, i.e. the sets X, A each have cardinality 2^N. The goal of compression is to construct a new tailored game $\mathfrak{Compr}(\mathfrak{G})$ with the following properties:

1. Questions and answers in $\mathfrak{Compr}(\mathfrak{G})$ have length $\text{poly}(n)$.[11] Namely, an exponential reduction in the length of questions and answers.

2. $\mathfrak{Compr}(\mathfrak{G})$ *simulates* $\mathfrak{G}$, as follows:

   (a) *Completeness*: If there exists a perfect ZPC strategy for $\mathfrak{G}$, then there is also such a strategy for $\mathfrak{Compr}(\mathfrak{G})$.
   (b) *Soundness*: If $\text{val}^*(\mathfrak{G}) \leq \frac{1}{2}$, then $\text{val}^*(\mathfrak{Compr}(\mathfrak{G})) \leq \frac{1}{2}$.

Compression is composed of three main steps:

1. In the first step the length of questions is reduced through a technique referred to as "introspection": informally, each player is instructed to generate its own question by itself; shorter questions are used to enforce that the player samples according to the right question distribution μ. This step produces a game $\mathfrak{G}' = \mathfrak{QueRed}(\mathfrak{G})$, whose questions have length $\text{poly}\log(N) = \text{poly}(n)$ and answers have length $O(N)$.

2. In the second step the length of the answers is reduced. This is achieved using techniques from probabilistic proof checking. Loosely speaking, the players encode their answers in $\mathfrak{G}'$ using an error-correcting code that allows probabilistic checking of computational statements (such as "this answer is a valid answer to that question") by reading only a small number of bits of the encoding — which constitute the player's new answer. This step results in a game $\mathfrak{G}'' = \mathfrak{AnsRed}(\mathfrak{G}')$ whose questions and answers have length $\text{poly}\log(N) = \text{poly}(n)$.[12]

---

[10]This simplification is already taken into account in the expression (2), which technically represents the synchronous value.

[11]We use the O and poly notations although we have not yet specified the asymptotics. For now, it can be assumed that there is a universal constant C such that $\text{poly}(n)$ is bounded from above by Cn^C and $O(N)$ is bounded by CN (cf. Remark 1.2) This is a bit misleading, as the length of the encoding of $\mathfrak{G}$ plays a role as well, but we are trying to postpone complexity theoretic considerations for now. Note that this guarantees a genuine compression only for large enough values of n, which is enough for the undecidability result to hold.

[12]As opposed to the previous step, this step depends heavily on the way $\mathfrak{G}$ is encoded. More specifically, $\mathfrak{G}$ needs to be encoded **succinctly**. In our case, we encode an infinite family of games in a uniform manner, and compress them all at once, which means in particular that the games are succinctly encoded as needed.

3. The combination of the two preceding transformations does not quite satisfy item 2(b) above. Instead, whenever $\text{val}^*(\mathfrak{G}) \leq \frac{1}{2}$ we only have $\text{val}^*(\mathfrak{G}'') \leq 1 - 1/\text{poly}(n)$. To remedy this, the game is repeated in parallel $\text{poly}(n)$ times to yield $\mathfrak{G}''' = \mathfrak{ParRep}(\mathfrak{G}'')$, which still has $\text{poly}(n)$-question and answer length, and moreover satisfies item 2(b).

All in all,

$$\mathfrak{Compr}(\mathfrak{G}) = \mathfrak{G}''' = \mathfrak{ParRep}(\mathfrak{AnsRed}(\mathfrak{QueRed}(\mathfrak{G}))) \,.$$

Each of the three transformations satisfies item 2(a), and so at the end both 2(a) and 2(b) are satisfied. We now discuss each step in more detail.

Question reduction. The introspection technique goes back to the work of Natarajan and Wright [NW19]. Intuitively, the idea is to "force the players to sample their own questions". Let μ be the question distribution in $\mathfrak{G}$. In the game $\mathfrak{QueRed}(\mathfrak{G})$, there is a special pair of questions $(\texttt{Intro}_A, \texttt{Intro}_B)$ such that answers to this pair of questions are expected to take the form $((\mathtt{x}, a), (\mathtt{y}, b))$ (each answer thus has length $2N$). We would like that three conditions hold. Firstly, it should be that, whenever this pair of questions is asked, the marginal distribution of the players' answers on $(\mathtt{x}, \mathtt{y})$ is exactly μ. Secondly, it should be that the a part of the answer is determined using only the question $\texttt{Intro}_A$ and $\mathtt{x}$ part of the answer, namely without "peeking" into the other players' question $\mathtt{y}$ (and similarly for b with all roles reversed). Finally, it should be that (a, b) are valid answers to $(\mathtt{x}, \mathtt{y})$ in $\mathfrak{G}$.

The last condition is easy to verify, as the referee in $\mathfrak{QueRed}(\mathfrak{G})$ can check it by themselves. The first two conditions require work. In particular, one may not expect to enforce a condition on the *distribution* of an answer from a test that depends on that answer only; as this could only restrain the support of the answer, but not its distribution. To achieve the first requirement one must thus consider a more complicated test that involves additional questions in the game. The method for forcing the distribution also allows to limit peeking, by leveraging the Heisenberg uncertainty principle — which states that information stored in a quantum state can be destroyed by performing a measurement in the complementary basis. Let us describe now the underlying ideas.

The main tool for forcing $(\mathtt{x}, \mathtt{y})$ to be distributed according to a specific distribution μ, is to verify that the PVMs associated with the bits of $\mathtt{x}$ and $\mathtt{y}$ come from (the Fourier transform of) a non-commutative representation of the Pauli group. To explain this a little more, let us focus on the case where μ is uniform on pairs $(\mathtt{x}, \mathtt{y}) \in \mathbb{F}_2^N \times \mathbb{F}_2^N$. The technique we describe generalizes to more complex, although far from arbitrary, distributions — it is known to apply to the class of *conditionally linear* distributions introduced in [JNV$^+$21] (see Section 4.3).

Let $\mathbb{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ and $\mathbb{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. These are commonly referred to as the $\mathbb{X}$ and $\mathbb{Z}$ Pauli matrices. The Pauli group acting on k qubits, sometimes called the Weyl–Heisenberg group or the k-dimensional Heisenberg group over $\mathbb{F}_2$, is the subgroup of unitaries acting on $(\mathbb{C}^2)^{\otimes k}$ generated by length k Kronecker tensor products of $\mathbb{X}$ and $\mathbb{Z}$ Pauli matrices. Namely, if for every $\alpha, \beta \in \mathbb{F}_2^k$ we define $\mathbb{X}^{\otimes \alpha} = \bigotimes_{i=1}^k \mathbb{X}^{\alpha_i}$ and $\mathbb{Z}^{\otimes \beta} = \bigotimes_{j=1}^k \mathbb{Z}^{\otimes \beta}$, then $P_k = \{\pm \mathbb{X}^{\otimes \alpha} \mathbb{Z}^{\otimes \beta} \mid \alpha, \beta \in \mathbb{F}_2^k\}$. What is relevant for us is that this group contains two copies of $\mathbb{F}_2^k$ as subgroups, the $\mathbb{X}$-subgroup $\{\mathbb{X}^{\otimes \alpha} \mid \alpha \in \mathbb{F}_2^k\}$ and the $\mathbb{Z}$-subgroup $\{\mathbb{Z}^{\otimes \beta} \mid \beta \in \mathbb{F}_2^k\}$, and that the above representation is its unique non-abelian irreducible representation. By taking the Fourier transform of the $\mathbb{Z}$-subgroup, we get a PVM $\{\mathscr{F}_a^{\mathbb{Z}}\}_{a \in \mathbb{F}_2^k}$, and measuring according to it provides a string a of length k which, following (1), is uniformly distributed. So, by choosing $k = 2N$, if we are able to force the provers to use a non-commutative representation of P_{2N}, we are able to force them to sample a uniform $(\mathtt{x}, \mathtt{y}) \in \mathbb{F}_2^N \times \mathbb{F}_2^N$ as required.

A key property that is used to force the provers to measure according to the representation of P_k described above is that there exists a specific presentation of it (with generators and relations) that is very *stable* (cf. [GH17, HS18, CL23, GR09]). The chosen presentation is due to de la Salle [dlS22b], who showed that in our setup the stability result is quite easily deduced by combining a spectral gap argument with a technique that translates anti-commutation to commutation due to Natarajan–Vidick [NV18a]. Another important property of P_k is that $\mathbb{X}^{\otimes \alpha}$ and $\mathbb{Z}^{\otimes \beta}$ anti-commute whenever $\langle \alpha, \beta \rangle = 1$; this is commonly referred to as mutual non-measurability, or the Heisenberg uncertainty principle. This property is used to guarantee the "no peeking" requirement mentioned above, by comparing the answers of the players in a clever way to certain $\mathbb{X}$-measurement outcomes.

All in all, the method we use for question reduction is a combination of [dlS22b] and [JNV$^+$21]. The main new observation is that the irreducible representation introduced above can be described as a permutation strategy, which is essential to showing that the introspected game has a perfect ZPC strategy (when the original game does).

Answer reduction. To reduce the length of answers in the game we use techniques from the area of probabilistically checkable proofs (PCPs) in computer science. At a high level, the idea is that instead of directly providing an answer that the referee checks, the player will first encode its answer in a suitable error-correcting code. The referee then requests a small number of bits from the encoded answer, and this will suffice for him to verify that these symbols are taken from a well-encoded answer (or one that is sufficiently close to such) that would have satisfied the original checks.

This step crucially relies on complexity theoretic assumptions on the way the original game $\mathfrak{G}$ was encoded. Specifically, it should be possible to represent the decision procedure from $\mathfrak{G}$ using a circuit of size $\mathrm{poly}(n)$ — this is possible, in our case, due to the scaled up Cook–Levin theorem together with the fact that the decision procedure runs in exponential time. The reason why translating the decision procedure to a circuit is crucial, is that the provers are not asked to encode their original answer, but instead to encode an assignment to all wires in the verifier's verification circuit,[13] with the question inputs $(\mathtt{x}, \mathtt{y})$ being hard coded. The assignment to all the wires must include the answers (a, b) themselves, but also a lot of additional information that is relevant for verifying that the answer would have satisfied the original checks, without actually reading the entire answer and performing the entire original computation.

Because of this new answer, the encoded assignment to all wires in the verification circuit depends on *both* questions $\mathtt{x}, \mathtt{y}$ and both answers a, b, and only a player that has access to this entire information may compute it. To make this possible, before answer reduction is performed the game $\mathfrak{G}'$ is *oracularized*. The referee in the oracularized game sends $(\mathtt{x}, \mathtt{y})$ to one player and $\mathtt{x}$ or $\mathtt{y}$ to the other. It checks the first player's answers according to the game's decision function, and the second player's answer for consistency with the first.[14]

Because the verification circuit is in general non-linear, some of the bits of the assignment described in the previous paragraph are obtained as e.g. the "AND" of some of the bits of the original answers. This creates a difficulty: if two answer bits a_i and a_j are computed by permutations σ_i, σ_j in a perfect strategy for the original game (in the sense of the associated quantum strategy (3) and the measurement rule (1)), the bit $a_i \wedge a_j$ may not have a permutation that measures to it. Indeed the natural way to define a permutation that "computes" the AND bit is to take the minus of the projection on the joint $(-1, -1)$ eigenspace of σ_i and σ_j, plus the projection on all other eigenspaces;[15] while this operation is an involution it is easy to see that it may not be a permutation — cf. (27).

A possible attempt to overcome this obstacle would be to require the decision function to be linear, as the parity of two answer bits computed by commuting permutations σ_i and σ_j is naturally computed by the permutation $\sigma_i \sigma_j$. Unfortunately, there are known obstacles to implementing answer reduction using error-correcting codes in a way that requires only a linear decision function. In particular, it is well-known in the classical literature on probabilistic proof checking that linear constraints can only lead to probabilistic checkers that do not have perfect completeness, i.e. even in the ideal case one has to abandon the requirement that the value of the game equals 1 — intuitively this is because linear systems of equations can be solved in polynomial time (using Gaussian elimination); so, deciding if there is a perfect strategy (in the classical model) for a linear verifier can be done efficiently; hence, there is no advantage to considering a nonlocal game in the first place. One may hope that the situation changes when one considers quantum (or permutation) strategies. However, even in that case there are strong obstacles to performing answer reduction in a liniar manner. In particular, it was shown in [PS23] that, in general, games that involve an AND verification predicate cannot be embedded in LCS games; this is shown using the fact that the algebra of an LCS game has a more structured collection of representations than a general game algebra—we refer to [PS23] for further discussion.

It is necessary to overcome this issue, and the generalized setup of tailored games (compared to LCS games) enables us

---

[13]Circuits are a standard model of computation, alongside Turing machines. In this paper we need very little about them; whatever we need is recalled in detail in Section 5.1.4.

[14]It is to guarantee that the oracularized game has a perfect strategy whenever the original game does that we need to restrict to strategies that are commuting along edges.

[15]Here we use the association True $\mapsto -1 = (-1)^1$ and False $\mapsto 1 = (-1)^0$.

to resolve it. Loosely speaking, in a tailored game, the checks performed on the unreadable part of the answer are linear, and hence amenable to linear checking. The checks performed on the readable part require the full power of non-linear proof checking techniques; but because perfect ZPC strategies are required to be Z-aligned, it is possible to define a permutation that computes the AND of two (or more) Z-aligned permutations — this fact is captured in Corollary 3.41.

To perform answer reduction while preserving the category of tailored games we need to design a bespoke probabilistic proof checker (PCP). This is one of the contributions of our paper. Indeed at first it is not obvious that the computation performed by a tailored verifier can be encoded and verified in a way that preserves its structure. In particular, any bit of the encoded answer that depends on an unreadable bit of the original answer must do so in a linear manner only; that is, the bit cannot be multiplied by any other unreadable bit. We carefully design the required PCP using standard techniques in probabilistic proof checking, including the use of multivariate polynomials and the Reed–Muller code; combined with observations specific to the linear case from [BSGH$^+$04].

Parallel repetition   The two preceding transformations have successfully reduced the size of the game; however, the soundness parameter has degraded. To restore this we perform parallel repetition. This consists in executing k instances of the game in parallel, and accepting only if all tuples of answers are valid with respect to the corresponding tuple of questions. Similar to [JNV$^+$21], we apply "anchored" repetition, which is known (by [BVY17]) to reduce the game value at an exponential rate whenever it was initially strictly smaller than 1. (Here there is a small subtlety, as we are in the synchronous soundness setup, and parallel repetition assumes a stronger soundness assumption, but this is resolved using the results from [Vid22].) It is not hard to verify that this transformation preserves perfect completeness with ZPC strategies and we do not describe it any further here.

1.2   Organization of the paper

In Section 2, we provide minimal preliminaries so to be able to formulate our main theorem TailoredMIP* = RE (Theorem 2.31) and the Compression theorem (Theorem 2.53), and to deduce the former assuming the latter. In Section 3 we proivde a wide range of preliminaries which are needed for the proof of Compression, some are quite standard and some are very particular to this paper. The next three sections are devoted to the three transformations, described in this introduction, which are the components of the Compression transformation: Section 4 describes the Question Reduction transformation, Section 5 describes the Answer Reduction transformation, and Section 6 describes the Parallel Repetition transformation. Lastly, in Section 7 we prove Compression by composing these three transformations.

1.3   Notations, naming conventions, and some general remarks

Remark 1.2 (Asymptotic notation). We often use the asymptotic notation O, Θ, Ω; namely, for two functions $f, g \colon \mathbb{N} \to \mathbb{N}$, $f(n) = O(g(n))$ if there is a universal constant $C > 0$ such that $f(n) \leq C \cdot g(n)$ for every $n \in \mathbb{N}$ (similarly, $f(n) = \Omega(g(n))$ if $f(n) \geq C \cdot g(n)$, and Θ is the combination of the two). In addition, we use the following somewhat less conventional notation. For positive integers a, b, c and so on, that we treat as growing to infinity, we write $\mathrm{poly}(a, b, c, ...)$ to denote a function bounded by $C(a^C + b^C + c^C + ...)$ for some universal constant $C \geq 1$. For non-negative real numbers $0 \leq \alpha, \beta, \gamma, ... < 1$, that we treat as going to 0, we write $\mathrm{poly}(\alpha, \beta, \gamma, ...)$ to denote a function bounded by $C(\alpha^{1/C} + \beta^{1/C} + \gamma^{1/C} + ...)$ for some universal constant $C \geq 1$. In either case, the universal constant C can vary each time the $\mathrm{poly}(\cdot)$ notation is used. We write $\mathrm{polylog}(a, b, c, ...)$ for $\mathrm{poly}(\log a, \log b, \log c, ...)$ and $\exp(a, b, c, ...)$ for $2^{\mathrm{poly}(a,b,c,...)}$. Finally, we may use $O_h, \Theta_h, \Omega_h, \mathrm{poly}_h, \mathrm{polylog}_h, \exp_h$ and so on, which means that the constant C involved in the bound is some function of h.

Remark 1.3 (Additional conventions throughout the paper).
- Questions in our games (namely, vertices in the underlying graphs of the games) are denoted using the typewriter style (mathtt): `x, y, Intro, Hide, var, row` and so on.
- Games, and combinatorial transformations on games, are denoted using the gothic style (mathfrak): $\mathfrak{G}$, $\mathfrak{PauliBasis}$, $\mathfrak{M}$, $\mathfrak{Lcs}$, $\mathfrak{QueRed}$ and so on.

- Formal variables that control the bits of the player's answers in our games are denoted using the sans serif style (mathsf): X, Y, Ans, Que, Var, ReadQue, and so on. This style is also used for certain transformations applied by Turing machines (algorithms) such as Compress, Decouple and so on. It is also used for certain acronyms such as ZPC and TailoredMIP*.
- PVMs and observables are denoted using the caligraphic style (mathcal): $\mathcal{P}, \mathcal{Q}, \mathcal{U}$. Some Turing machines also use this style, usually with the letter $\mathcal{M}$, as well as the components of a tailored normal form verifier $\mathcal{V}$ which are $\mathcal{S}, \mathcal{A}, \mathcal{L}$ and $\mathcal{D}$.
- $\mathbb{X}, \mathbb{Z}$ are the Pauli matrices.
- For a positive integer n, $[n]$ is the set $\{1, ..., n\}$.
- We often use σ for permutations, and for permutation strategies. The elements in the sets Ω on which our permutations act are usually denoted by $\star$ and $\diamond$. The elements of the signed set $\Omega_\pm$ are often denoted by $\spadesuit, \diamondsuit$, by which we mean $\spadesuit$ is $+\star$ or $-\star$ for some $\star \in \Omega$.
- We use $\cdot$ (and less frequently $\star$ and $\circ$) as an input which is not specified. It should be understood from context what are the possible inputs for $\cdot$ (respectively $\star, \circ$). Sometimes this notation actually means "for all possible inputs" in this position, and again this should be understood from context.
- We use $|\cdot|$ to denote the length of a word in some finite alphabet Σ. Usually, this word is over bits $\mathbb{F}_2 = \{0, 1\}$, but occasionally, it is over larger alphabets. The set Σ^* is the free monoid over the alphabet Σ, namely all strings with letters from Σ. Namely, $\{0,1\}^*$ it is the set of all bit strings with the concatenation $*$ of words as the product. Similar to other products, we often write ww' instead of $w * w'$ for the concatenation of two words w, w'.
- The sign ε is used in various contexts in the text: as a small positive real number; as an $\mathbb{F}_2$-exponent of order-2 elements in a group; or as the empty word in the mononid $\{0,1\}^*$. These use cases should be understood from context.
- We usually write Id for the identity element Id_Γ of a group Γ. The specific group should be understood from context. Most commonly, Id is used instead of Id_k for the $k \times k$ matrix, where k should be understood from context.
- Bits with 2-modular arithmetic $\{0,1\}$ and the field with two elements $\mathbb{F}_2$ are used interchangeably throughout the paper. Given a vector space $\{0,1\}^S$ where S is a finite set, we use $\mathbf{1}_\mathsf{X} : S \to \{0,1\}$ for the indicator function of $\mathsf{X} \in S$, and $\langle u, v \rangle = \sum_{\mathsf{X} \in S} u(\mathsf{X})v(\mathsf{X})$ for the *standard bilinear form* on it. If $S = [k]$, we often denote by e_i the indicator of $i \in [k]$, instead of $\mathbf{1}_i$, and $\{e_1, ..., e_k\}$ is commonly called the *standard basis*. Given an (ordered) set S, we commonly think of $u : S \to \mathbb{F}_2$ both as a **function** and as a **string of bits** parameterized by the set S.

Acknowledgements

2 Tailored games and deducing TailoredMIP* = RE from Compression

The goal of this section is to provide the minimal preliminaries so that our main theorem (Theorem 2.31) can be rigorously formulated, and then show how Compression of tailored normal form verifiers (Theorem 2.53) implies it. The rest of the paper is devoted to the proof of Compression.

Throughout this section we use $\{0,1\}$ and $\mathbb{F}_2$ interchangeably to describe the field with two elements, namely bits with 2-modular arithmetic. Given a vector space $\{0,1\}^S$ where S is a finite set, let $\mathbf{1}_X \colon S \to \{0,1\}$ be the indicator function of $X \in S$, and let $\langle u, v \rangle = \sum_{X \in S} u(X)v(X)$ be the *standard bilinear form* on it (referred to also as the *dot product* of u and v). If $S = [k]$, we often denote by e_i the indicator of $i \in [k]$, instead of $\mathbf{1}_i$. Given an (ordered) set S, we commonly think of $u \colon S \to \mathbb{F}_2$ both as a **function** and as a **string of bits** parameterized by the set S.

2.1 Measurements

The concept of *measurement* plays a key role in quantum mechanics. Measurements are modeled using *positive operator valued measures*, which can be viewed as non-commutative counterparts of standard probability measures. A finite probability measure is, on the one hand, just a tuple of non-negative real numbers that adds up to 1, and on the other hand a *sampling scheme* with finitely many results. In a similar way:

Definition 2.1 (POVMs and PVMs). A *positive operator valued measure* (POVM) of dimension n with outcomes in a (finite) set A is a mapping $\mathcal{P} \colon A \to M_{n \times n}(\mathbb{C})$ such that for every $a \in A$, $\mathcal{P}_a$ is a positive matrix — i.e., $\mathcal{P}_a = C^*C$ for some matrix C, where $*$ is the conjugate transpose operation — and $\sum_{a \in A} \mathcal{P}_a = \mathrm{Id}_n$, where Id_n is the $n \times n$ identity matrix. It is called a projective valued measure (PVM) if in addition $\mathcal{P}_a$ is an orthogonal projection for every $a \in A$, namely $(\mathcal{P}_a)^2 = \mathcal{P}_a = (\mathcal{P}_a)^*$.

As its name suggests, every POVM $\mathcal{P}$ defines a probability distribution over its outcome set A as follows:[16]

$$\mathbb{P}[a \text{ is sampled}] := \tau(\mathcal{P}_a) \,, \tag{4}$$

where $\tau = \frac{1}{n}\mathrm{Tr}$ is the dimension normalized trace on $n \times n$ matrices. Such an answer is said to be *sampled according* to $\mathcal{P}$ and we denote it by $a \sim \mathcal{P}$.

This definition alone seems like a complicated way of generating probability distributions on finitely many outcomes. The following definition is what makes this model interesting:

Definition 2.2 (Joint measurements). Given two POVMs $\mathcal{P}^{\mathtt{x}}$ and $\mathcal{P}^{\mathtt{y}}$ of the same dimension n, where $\mathcal{P}^{\mathtt{x}}$ is with outcomes in A and $\mathcal{P}^{\mathtt{y}}$ is with outcomes in B, we define their *joint measurement* to be the following n-dimensional POVM with outcomes in $A \times B$:

$$\mathcal{P}^{\mathtt{xy}}_{a,b} = (\mathcal{P}^{\mathtt{x}}_a)^{1/2} \mathcal{P}^{\mathtt{y}}_b (\mathcal{P}^{\mathtt{x}}_a)^{1/2} \,,$$

where $A^{1/2}$ is a well defined (positive) matrix given A is a positive matrix. The joint measurement of $\mathcal{P}^{\mathtt{x}}$ and $\mathcal{P}^{\mathtt{y}}$ defines a probability distribution over $\gamma = (a, b) \in A \times B$, which we refer to as their *joint sampling*:

$$\mathbb{P}[a, b \text{ are sampled}] := \tau(\mathcal{P}^{\mathtt{xy}}_{a,b}) = \tau(\mathcal{P}^{\mathtt{x}}_a \mathcal{P}^{\mathtt{y}}_b) \,. \tag{5}$$

Remark 2.3. In case $\mathcal{P}^{\mathtt{x}}$ and $\mathcal{P}^{\mathtt{y}}$ are projective measurements, namely PVMs, there is a "procedural" viewpoint of jointly sampling according to them. Let $\mathscr{B}$ be an orthonormal basis of eigenvectors for all of the matrices $\{\mathcal{P}^{\mathtt{x}}_a\}_{a \in A}$ and $\mathscr{C}$ an orthonormal basis of eigenvectors for $\{\mathcal{P}^{\mathtt{y}}_b\}_{b \in B}$. Sample $\vec{v} \in \mathscr{B}$ uniformly at random. Sample $\vec{w} \in \mathscr{C}$ with probability $|\langle \vec{v} | \vec{w} \rangle|^2$, where $\langle \cdot | \cdot \rangle$ is the standard inner product on $\mathbb{C}^n$. As $\mathcal{P}^{\mathtt{x}}$ and $\mathcal{P}^{\mathtt{y}}$ are PVMs, there is only one $a \in A$ and $b \in B$ such that $\vec{v} \in \mathrm{Im}(\mathcal{P}^{\mathtt{x}}_a)$ and $\vec{w} \in \mathrm{Im}(\mathcal{P}^{\mathtt{y}}_b)$. Output (a, b).

---

[16]The reader who is familiar with quantum measurements may notice that this is not the most general setup of finite dimensional measurements, as the normalized trace occurs when measuring a system in a specific mixed state. In the vast majority of this paper, this special case is all we need. But, for the soundness analysis of the parallel repetition theorem, we need the more general theory, which is discussed in Section 3.6.

In case $A = \mathbb{F}_2^S$ for some finite set S, there is a close connection between unitary representations of (the group) $\mathbb{F}_2^S$ and PVMs with outcomes in $\mathbb{F}_2^S$. As the images of a unitary representation of $\mathbb{F}_2^S$ are commuting, they have mutual eigenspaces, and there is an algebraic way of extracting the orthogonal projections onto them.

Definition 2.4 (The Fourier transform of a representation). Let $\mathcal{U} \colon \mathbb{F}_2^S \to U(n)$ be a unitary representation. The *Fourier transform* of $\mathcal{U}$ is a PVM $\mathcal{P} \colon \mathbb{F}_2^S \to M_{n \times n}(\mathbb{C})$ defined as follows

$$\forall a \in \mathbb{F}_2^S : \quad \mathcal{P}_a = \mathop{\mathbb{E}}_{\alpha \in \mathbb{F}_2^S} \left[(-1)^{\langle a, \alpha \rangle} \mathcal{U}(\alpha) \right] \ .$$

Indeed for every $\vec{v} \in \mathrm{Im}(\mathcal{P}_a)$ and $\alpha \in \mathbb{F}_2^S$ we have $\mathcal{U}(\alpha)\vec{v} = (-1)^{\langle a, \alpha \rangle}\vec{v}$. The inverse Fourier transform in this case is

$$\forall \alpha \in \mathbb{F}_2^S : \quad U(\alpha) = \sum_{a \in \mathbb{F}_2^S} (-1)^{\langle a, \alpha \rangle} \mathcal{P}_a \ .$$

Definition 2.5 (Projective, Representation and Observable form of a PVM). Let S be a finite set. The following three objects contain the same data:

- *Projective form*: A map $\mathcal{P} \colon \mathbb{F}_2^S \to M_{n \times n}(\mathbb{C})$ whose images are orthogonal projections that sum up to the identity.

- *Representation form*: A unitary representation $\mathcal{U} \colon \mathbb{F}_2^S \to U(n)$.

- *Observable form*: A map $\mathcal{U} \colon S \to U(n)$ whose images are commuting involutions (i.e., square to the identity).

So, a PVM can be given in any of these forms, and we refer to them as the projective, representation, and observable form of the PVM respectively. Furthermore, if we have a PVM $\mathcal{U}$ in representation (or observable) form, we still denote by $a \sim \mathcal{U}$ an outcome sampled according to the PVM (and similarly $(a, b) \sim (\mathcal{U}^{\mathsf{x}}, \mathcal{U}^{\mathsf{y}})$ for the joint measurement).

Remark 2.6. Note that we use the same notation $\mathcal{U}$ for the representation and observable form of a PVM. This may be a bit confusing, as for $\mathsf{X} \in S$, $\mathcal{U}(\mathbf{1}_{\mathsf{X}})$ in representation form is the same as $\mathcal{U}(\mathsf{X})$ in observable form. But it is in fact a natural choice, as we use the "universal property" of $\mathbb{F}_2^S$, which says that any map $\mathcal{U} \colon S \to U(n)$ whose images are commuting involutions can be extended to a unitary representation of $\mathbb{F}_2^S$ through the embedding of S in $\mathbb{F}_2^S$ through the map $\mathsf{X} \mapsto \mathbf{1}_{\mathsf{X}}$.

Definition 2.7 (Diagonal PVM). A PVM $\mathcal{P} \colon A \to M_{n \times n}(\mathbb{C})$ is *diagonal* if all its images $\mathcal{P}_a$ are diagonal matrices. Namely, the projections are on spaces spanned by subsets of the standard basis. In case $A = \mathbb{F}_2^S$, this property is preserved under the Fourier transform. Namely, it is equivalent to the representation (and thus observable) form $\mathcal{U}$ of the PVM to consist of only diagonal unitaries.

Definition 2.8 (Readably Z-aligned PVM). Let $S^{\mathfrak{R}}$ and $S^{\mathfrak{L}}$ be disjoint finite sets.[17] A PVM in observable form $\mathcal{U} \colon S^{\mathfrak{R}} \sqcup S^{\mathfrak{L}} \to U(n)$ is said to be *readably Z-aligned* if its restriction to $S^{\mathfrak{R}}$ is diagonal (Definition 2.7).

Remark 2.9. The standard basis in quantum information theory is commonly called the Z-basis, as it is the mutual eigenbasis of the $\mathbb{Z}$-matrices in the Pauli group (more on that in Section 3.7). Hence the term "readably Z-aligned" for one whose readable observables are diagonal with respect to the standard basis.

2.2 Permutations and Signed permutations

As described in the introduction, the perfect strategies in our category should be induced by permutation representations — actually, by signed permutation representations. To that end we give the following definition.

---

[17]This notation is $\mathfrak{R}$ for **readable** variables and $\mathfrak{L}$ for **linear** or **unreadable** variables.

Definition 2.10 (Permutation matrices and representations). Let Ω be a finite set. As $\mathrm{Sym}(\Omega)$ acts naturally on Ω, its action extends to $\mathbb{C}^\Omega$ as follows: Given $f \colon \Omega \to \mathbb{C}$ and $\sigma \in \mathrm{Sym}(\Omega)$, let $\sigma.f(\star) := f(\sigma^{-1}.\star)$. The standard basis of $\mathbb{C}^\Omega$ consists of the indicators $\mathbf{1}_\star$ for every $\star \in \Omega$, and we have

$$\forall \diamond \in \Omega : \quad \sigma.\mathbf{1}_\star(\diamond) = \mathbf{1}_\star(\sigma^{-1}.\diamond) = \begin{cases} 1 & \sigma^{-1}.\diamond = \star, \\ 0 & \sigma^{-1}.\diamond \neq \star, \end{cases}$$

namely $\sigma.\mathbf{1}_\star = \mathbf{1}_{\sigma.\star}$. Representing $\mathrm{Sym}(\Omega)$ via this action as $\Omega \times \Omega$ matrices gives rise to the subset of $U(\mathbb{C}^\Omega)$ consisting of all $0/1$ matrices with exactly one 1 in every row and column. Unsurprisingly, these matrices are called *permutation matrices*. An action is a homomorphism from a group to $\mathrm{Sym}(\Omega)$, and by using the above embedding of permutations into $U(\mathbb{C}^\Omega)$, we get a unitary representation of the group. Such representations are called *permutation representations*.

Definition 2.11 (Signed sets). Given a finite set Ω, we define its signed version $\Omega_\pm$ to be $\{\pm\} \times \Omega$; we commonly denote $+\star$ and $-\star$ instead of $(+,\star)$ and $(-,\star)$. We commonly use $\star, \diamond$ for elements of Ω and $\spadesuit, \diamondsuit$ for elements of $\Omega_\pm$.

Definition 2.12 (The sign flip). The *sign flip* $-\mathrm{Id}$ is a permutation on $\Omega_\pm$ that, as its name suggests, flips the sign of every vertex. Namely,

$$\forall \star \in \Omega : \quad -\,\mathrm{Id}. \pm \star = \mp \star\,.$$

A function $f \colon \Omega_\pm \to \mathbb{C}$ is said to be *symmetric* if $f(+\star) = f(-\star)$ for every $\star \in \Omega$ and *anti-symmetric* if $f(+\star) = -f(-\star)$. The symmetric functions are the $(+1)$-eigenspace of $-\mathrm{Id}$ and we denote them by $W^+ \subseteq \mathbb{C}^{\Omega_\pm}$, while the anti-symmetric functions are its (-1)-eigenspace and are denoted by $W^- \subseteq \mathbb{C}^{\Omega_\pm}$. Let $\Xi \colon \mathbb{C}^{\Omega_\pm} \to W^-$ be the orthogonal projection on W^-. We fix

$$B^+ = \left\{ \frac{\mathbf{1}_{+\star} + \mathbf{1}_{-\star}}{\sqrt{2}} \right\}_{\star \in \Omega} \quad , \quad B^- = \left\{ \frac{\mathbf{1}_{+\star} - \mathbf{1}_{-\star}}{\sqrt{2}} \right\}_{\star \in \Omega} \tag{6}$$

to be the standard orthonormal bases for W^+ and W^- respectively. Note that these bases are indeed the images (up to a sign in case of B^-) of the standard basis of $\mathbb{C}^{\Omega_\pm}$ via its orthogonal projection onto the symmetric and anti-symmetric functions (i.e., Ξ) respectively.

Definition 2.13 (Signed permutations and representations). A *signed permutation* is a permutation $\sigma \in \mathrm{Sym}(\Omega_\pm)$ that commutes with the sign flip, and we denote by $\mathrm{Sym}_\pm(\Omega)$ the subgroup of all signed permutations. The action of the signed permutations on $\mathbb{C}^{\Omega_\pm}$ preserves the spaces of anti-symmetric functions W^-, which induces an embedding $\mathrm{Sym}_\pm(\Omega) \hookrightarrow U(W^-)$. The image of this embedding is called the group of *signed permutations*. By representing the matrices in $\mathrm{End}(W^-)$ with respect to the basis B^- from (6), the image of $\mathrm{Sym}_\pm(\Omega)$ consists of all matrices with coefficients in $\{0, +1, -1\}$, such that in each row and column there is a single non-zero entry (which must be either $+1$ or -1). A *signed action* is a homomorphism of a group into $\mathrm{Sym}_\pm(\Omega)$, and by composing it with the above embedding into $U(W^-)$ we get a *signed permutation representation*.

Remark 2.14 (Signed permutations as a semidirect product). Every signed permutation matrix $\mathscr{A} \in U(n)$ can be written (uniquely) as a product $\mathscr{B} \cdot \mathscr{D}$, where $\mathscr{B}$ is a (non-signed) permutation matrix, and $\mathscr{D}$ is a diagonal matrix with ± 1 on the diagonal. As the subgroup of diagonal matrices with ± 1 on the diagonal is normal in the signed permutations, and is isomorphic to $\mathbb{F}_2^n$, we deduce that

$$\mathrm{Sym}_\pm(\Omega) \cong \mathrm{Sym}(\Omega) \ltimes \mathbb{F}_2^\Omega\,.$$

Definition 2.15 (Signed permutation PVM). Let S be a finite set. A *signed permutation PVM* (in representation form and with outcomes in $\mathbb{F}_2^S$) is a signed permutation representation of $\mathbb{F}_2^S$, namely a homomorphism $\mathcal{U} \colon \mathbb{F}_2^S \to \mathrm{Sym}_\pm(\Omega) \subseteq U(W^-)$. We seldomly extend $\mathcal{U}$ (in observable form) to be defined on an additional element $\mathrm{J} \notin S$ such that $\mathcal{U}(\mathrm{J}) = -\mathrm{Id}$ — this yields a representation of $\mathbb{F}_2^{S \cup \{\mathrm{J}\}}$.

2.3 Non-local Games

Definition 2.16 (Games). A (2-player, 1-round, synchronous non-local) game $\mathfrak{G}$ consists of a finite (oriented) graph $G = (V, E)$, a length function $\ell \colon V \to \mathbb{N}$, (distinct[18]) formal sets of generators $S_{\mathtt{x}}$ of size $\ell(\mathtt{x})$ for every vertex $\mathtt{x} \in V$, a distribution μ over the edge set E, and decision functions $D_{\mathtt{xy}} \colon \{0,1\}^{S_{\mathtt{xy}}} \to \{0,1\}$ for every edge $\mathtt{xy} \in E$, where $S_{\mathtt{xy}} = S_{\mathtt{x}} \cup S_{\mathtt{y}}$.[19] We denote by S the set $\bigcup_{\mathtt{x} \in V} S_{\mathtt{x}}$ consisting of all formal variables used in the game.

Remark 2.17 (Standard definition of a game). It is common to define a game with less data, as follows: It consists of two finite sets X, A, a probability distribution μ over $X \times X$, and a decision predicate $D \colon X \times X \times A \times A \to \{0,1\}$. The set X is commonly called the *question set* and A the *answer set*. Such a game is called *synchronous* if $D(\mathtt{x}, \mathtt{x}, a, b) = 0$ for every $a \neq b \in A$ and $\mathtt{x} \in X$.

One can extract the data of Definition 2.16 from the above as follows: Let ℓ be the constant function $\Lambda = \lceil \log |A| \rceil$, and fix an embedding of A into $\{0,1\}^{\Lambda}$. The vertices V of the underlying graph G will be X, and the support of μ will be the edge set $E \subseteq X \times X$. There is a unique formal generator in $S_{\mathtt{x}}$ that corresponds to each bit of the answer a when $\mathtt{x} \in X$ is asked as a question — this is the case as all the $S_{\mathtt{x}}$ are disjoint. Then, given that $\mathtt{x} \neq \mathtt{y}$ were asked, a pair of answers a, b can be encoded as a map $\gamma \colon S_{\mathtt{x}} \cup S_{\mathtt{y}} \to \{0,1\}$, where $\gamma|_{S_{\mathtt{x}}} = a$ and $\gamma|_{S_{\mathtt{y}}} = b$. Lastly, $D_{\mathtt{xy}}(\gamma) = D(\mathtt{x}, \mathtt{y}, a, b)$, where γ is the aforementioned encoding. Note that under this formulation, if $\mathtt{x} = \mathtt{y}$, then $S_{\mathtt{x}} = S_{\mathtt{y}}$, which implies that $D_{\mathtt{xx}}$ is a function only of $a = \gamma|_{S_{\mathtt{x}}} = \gamma|_{S_{\mathtt{y}}} = b$. As our strategies are (almost) always synchronous (Definition 2.18), this will mostly not be an issue — see Section 3.6 for the non-synchronous setup, which is used only in the soundness argument of the parallel repetition theorem.

Definition 2.18 (Strategies). A (synchronous, quantum) n-dimensional strategy $\mathscr{S}$ for a game $\mathfrak{G}$ (Definition 2.16) is a map that associates to every vertex $\mathtt{x} \in V$ a n-dimensional PVM (Definition 2.1) with outcomes in $\mathbb{F}_2^{S_{\mathtt{x}}}$. I.e.,

- *Projective form*: A function $\mathcal{P}$ that takes as input a vertex $\mathtt{x} \in V$ and a bit string $a \colon S_{\mathtt{x}} \to \{0,1\}$ and outputs a $n \times n$ matrix with complex coefficients, where for every $\mathtt{x} \in V$ the restriction $\mathcal{P}^{\mathtt{x}} = \mathcal{P}(\mathtt{x}, \cdot) \colon \{0,1\}^{S_{\mathtt{x}}} \to M_{n \times n}(\mathbb{C})$ is a PVM in projective form. In such a case, we denote

$$\mathscr{S} = \{\mathcal{P}\} = \{\mathcal{P}_a^{\mathtt{x}} \mid \mathtt{x} \in V, a \colon S_{\mathtt{x}} \to \mathbb{F}_2\} .$$

- *Representation form*: A function $\mathcal{U}$ that takes as input a vertex $\mathtt{x} \in V$ and a vector $\alpha \in \mathbb{F}_2^{S_{\mathtt{x}}}$ and outputs an $n \times n$ unitary $\mathcal{U}^{\mathtt{x}}(\alpha)$, where for every $\mathtt{x}$ the map $\mathcal{U}^{\mathtt{x}}(\cdot) \colon \mathbb{F}_2^{S} \to U(n)$ is a unitary representation. In such a case we denote

$$\mathscr{S} = \{\mathcal{U}\} = \{\mathcal{U}^{\mathtt{x}}(\alpha) \mid \mathtt{x} \in V, \alpha \in \mathbb{F}_2^{S_{\mathtt{x}}}\} .$$

- *Observable form*: A function $\mathcal{U}$ that takes as input a formal variable $\mathsf{X} \in S$ and outputs an $n \times n$ unitary $\mathcal{U}(\mathsf{X})$, such that its restriction to $S_{\mathtt{x}}$ consists of commuting unitary involutions for every fixed $\mathtt{x} \in V$. In such a case we denote

$$\mathscr{S} = \{\mathcal{U}\} = \{\mathcal{U}(\mathsf{X}) \mid \mathsf{X} \in S\} .$$

We say that the strategy $\mathscr{S}$ *commutes along edges* if for every $\mathtt{xy} \in E$, the images of $\mathcal{P}^{\mathtt{x}}$ and $\mathcal{P}^{\mathtt{y}}$ commute (equivalently, the images of $\mathcal{U}^{\mathtt{x}}$ and $\mathcal{U}^{\mathtt{y}}$ commute, or for every $\mathsf{X} \in S_{\mathtt{x}}$ and $\mathsf{Y} \in S_{\mathtt{y}}$ the matrices $\mathcal{U}(\mathsf{X})$ and $\mathcal{U}(\mathsf{Y})$ commute). We say that $\mathscr{S}$ is a (signed) *permutation strategy* if it associates to each vertex a signed permutation PVM (Definition 2.15).

The game distribution μ specifies a way to sample edges $\mathtt{xy} \in E$. After an edge $\mathtt{xy}$ is sampled, one can jointly measure according to the PVMs at the vertices $\mathtt{x}$ and $\mathtt{y}$, which gives an outcome (a, b) where $a \colon S_{\mathtt{x}} \to \mathbb{F}_2$ and $b \colon S_{\mathtt{y}} \to \mathbb{F}_2$. Namely,

$$\mathbb{P}[(a, b) \in \mathbb{F}_2^{S_{\mathtt{x}}} \times \mathbb{F}_2^{S_{\mathtt{y}}} \text{ is sampled} \mid \mathtt{x}, \mathtt{y} \text{ were sampled}] = \tau(\mathcal{P}_a^{\mathtt{x}} \mathcal{P}_b^{\mathtt{y}}) . \tag{7}$$

We often denote the concatenation of a and b as $\gamma = ab \colon S_{\mathtt{x}} \cup S_{\mathtt{y}} \to \mathbb{F}_2$. A function $\gamma = ab$ sampled as in (7) is said to be *sampled according* to the strategy $\mathscr{S}$, and we denote it by $\gamma \sim \mathscr{S}$ (with the dependence on $\mathtt{x}, \mathtt{y}$ usually left implicit).

---

[18]Namely, there are no formal generators that belong to $S_{\mathtt{x}}$ and to $S_{\mathtt{y}}$ for any $\mathtt{x} \neq \mathtt{y}$.

[19]In case $\mathtt{x} \neq \mathtt{y}$, $S_{\mathtt{xy}}$ is the disjoint union $S_{\mathtt{x}} \sqcup S_{\mathtt{y}}$, but in case $\mathtt{x} = \mathtt{y}$ then $S_{\mathtt{xy}} = S_{\mathtt{x}} = S_{\mathtt{y}}$.

Remark 2.19. In [BCLV24] permutation strategies were defined slightly differently. There, we distinguished between the signed permutation PVMs (in observable form) associated to each vertex, which have images in $\mathrm{Sym}_\pm(\Omega)$ — the collection of them was called the *permutation strategy* (Definition 6.11 therein, where the image of J plays the role of the sign flip $-\mathrm{Id}$) — and the quantum strategy induced by embedding $\mathrm{Sym}_\pm(\Omega)$ in $U(W^-)$ — which is called *the quantum strategy induced by a permutation strategy* (Definition 6.14 therein). As these obejcts provide the same information, here we decided to drop the distinction between them and just think of $\mathrm{Sym}_\pm(\Omega)$ as embedded in the natural way in the unitaries on anti-symmetric functions $U(W^-)$.

Example 2.20 (Classical strategies). The subgroup $\{\pm1\} \subseteq U(1)$ is the collection of 1×1 signed permutation matrices (Definition 2.13). Let $\mathfrak{G}$ be a game and S its formal set of generators. For every fixed $f \colon S \to \{0,1\}$, we can define a strategy $\mathcal{U} \colon S \to \{\pm1\}$ as follows

$$\mathcal{U}(\mathsf{X}) = \begin{cases} +1 & f(\mathsf{X}) = 0\,, \\ -1 & f(\mathsf{X}) = 1\,. \end{cases}$$

Given that we have sampled an edge xy according to μ, it is straightforward that $\gamma \colon S_{\mathsf{xy}} \to \{0,1\}$ which is sampled according to $\mathscr{S} = \{\mathcal{U}\}$ (Definition 2.18) is deterministically $f|_{S_{\mathsf{xy}}}$. Such strategies are usually called *deterministic*. By taking direct sums of such deterministic strategies (for potentially different f's) — which is the same as requiring that the strategy associates to every vertex a diagonal PVM (Definition 2.7) — we can get any (rational) distribution over deterministic strategies. Such strategies are usually called *classical*. Hence, every (rational) classical strategy can be obtained as a permutation strategy.

Definition 2.21 (Value). We can "run" the strategy $\mathscr{S}$ against the game $\mathfrak{G}$: sample $\mathsf{xy} \in E$ according to μ; sample $\gamma \colon S_{\mathsf{xy}} \to \{0,1\}$ according to $\mathscr{S}$; *Accept* if $D_{\mathsf{xy}}(\gamma) = 1$, and otherwise *Reject*. The *value* of $\mathscr{S}$ against $\mathfrak{G}$ is its acceptance probability in the above procedure, namely

$$\begin{aligned} \mathrm{val}(\mathfrak{G}, \mathscr{S}) &= \mathop{\mathbb{E}}_{\mathsf{xy} \sim \mu} \mathop{\mathbb{E}}_{\gamma \sim \mathscr{S}} [D_{\mathsf{xy}}(\gamma)] \\ &= \sum_{\mathsf{xy} \in E} \sum_{\substack{a \colon S_{\mathsf{x}} \to \mathbb{F}_2 \\ b \colon S_{\mathsf{y}} \to \mathbb{F}_2}} \mu(\mathsf{xy}) \tau\left(\mathcal{P}_a^{\mathsf{x}} \mathcal{P}_b^{\mathsf{y}}\right) D_{\mathsf{xy}}(ab)\,. \end{aligned}$$

We say that a strategy $\mathscr{S}$ is *perfect* (for $\mathfrak{G}$) if $\mathrm{val}(\mathfrak{G}, \mathscr{S}) = 1$. The (synchronous quantum) value $\mathrm{val}^*(\mathfrak{G})$ of $\mathfrak{G}$ is the supremum of its value against every quantum strategy $\mathscr{S}$.

Remark 2.22 (Correlations). Usually, the collection of conditional distributions

$$\mathbb{P}[\gamma = (a,b) \text{ is sampled by } \mathscr{S} \mid \mathsf{x}, \mathsf{y} \text{ were sampled}]$$

for every pair $\mathsf{x}, \mathsf{y} \in V$ is called the *correlation* induced by the quantum strategy, and is denoted by $p(a, b|\mathsf{x}, \mathsf{y})$ (or $p_{\mathscr{S}}$ when wanting to emphasize the dependence on $\mathscr{S}$).

Remark 2.23 (Dramatization of a game). The reason for the name "game" for the data described in Definition 2.16, and for the name "strategy" for the collection of PVMs described in Definition 2.18 is the following:

Two players, that can share a maximally entangled state of any dimension n, are separated spatially — e.g., they are seated in far away rooms. A referee samples a pair of questions — i.e., an edge $\mathsf{xy} \in E$ — and sends one question to each player — namely, x to player A and y to player B. The players agreed beforehand, for every possible question in the game, how they will measure their part of the state — namely, they chose a map from V to PVMs acting on $\mathbb{C}^n$. After receiving their questions, each player measures their part of the state as agreed beforehand, comes up with answers — a for player A and b for player B — according to what they have measured, and send them back to the referee. The referee then decides, using the decision predicate $D_{\mathsf{xy}}(ab)$, whether the players *won* or *lost*. The decision predicate D as well as the distribution μ over possible questions are assumed to be known to the players before they choose their *strategy*, namely the dimension of their maximally entangled state and the projective measurements associated to each vertex.

16

2.4 Tailored games

Definition 2.24 (Tailored games). Colloquially, a *tailored* game is one where D_{xy} reads **part** of (the answer pair) $\gamma = ab$, and decides according to this partial view which **parity checks** to apply on **the whole** of γ.[20]

Formally, a tailored (non-local) game $\mathfrak{G}$ is equipped with extra structure, described shortly, and its decision functions D_{xy} behave **canonically** with respect to this extra data. Instead of a single length function ℓ, $\mathfrak{G}$ has two length functions $\ell^{\mathfrak{R}}: V \to \mathbb{N}$ and $\ell^{\mathfrak{L}}: V \to \mathbb{N}$, and $\ell = \ell^{\mathfrak{R}} + \ell^{\mathfrak{L}}$. Before, the length function described the size of the formal set of generators at each vertex. Now, the formal set of generators S_x at $x \in V$ will be a disjoint union of the sets $S_x^{\mathfrak{R}}$ and $S_x^{\mathfrak{L}}$, where $S_x^{\mathfrak{R}}$ is of size $\ell^{\mathfrak{R}}(x)$ and $S_x^{\mathfrak{L}}$ is of size $\ell^{\mathfrak{L}}(x)$. The elements of $S_x^{\mathfrak{R}}$ are called the *readable* variables at $x \in V$ and the elements of $S_x^{\mathfrak{L}}$ the *linear* or *unreadable* variables at x. In addition, $\mathfrak{G}$ is equipped with a collection of *controlled linear constraints* functions L_{xy} that take as input a function $\gamma^{\mathfrak{R}}: S_x^{\mathfrak{R}} \sqcup S_y^{\mathfrak{R}} \to \mathbb{F}_2$, and outputs a sequence of subsets of $S_{xy} \sqcup \{J\}$, where J is a new formal variable not in any other set. Namely,

$$L_{xy}: \mathbb{F}_2^{S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}}} \to \mathbb{F}_2^{\mathbb{F}_2^{S_{xy} \cup \{J\}}} \ .$$

The image of L_{xy} is interpreted as a collection of linear constraints that will be verified by the decision function. The decision function $D_{xy}(\gamma)$ behaves as follows: It restricts γ to the readable variables, namely looks at $\gamma^{\mathfrak{R}} = \gamma|_{S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}}}: S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}} \to \mathbb{F}_2$, and calculates $L_{xy}(\gamma^{\mathfrak{R}})$. Then, it extends γ such that $\gamma(J) = 1$. Finally, for every $c \in L_{xy}(\gamma^{\mathfrak{R}})$, we have $c: S_{xy} \cup \{J\} \to \mathbb{F}_2$, and D_{xy} verifies that

$$\langle c, \gamma \rangle = \sum_{X \in S_{xy} \cup \{J\}} c(X) \cdot \gamma(X) = 0 \ .$$

Namely, $L_{xy}(\gamma^{\mathfrak{R}})$ consists of linear constraints that γ needs to satisfy. If all of the above were satisfied, then $D_{xy}(\gamma) = 1$, and otherwise it is 0. In the spirit of Remark 2.17, we often denote

$$a^{\mathfrak{R}} = \gamma|_{S_x^{\mathfrak{R}}} \ , \quad a^{\mathfrak{L}} = \gamma|_{S_x^{\mathfrak{L}}} \ , \quad b^{\mathfrak{R}} = \gamma|_{S_y^{\mathfrak{R}}} \quad \text{and} \quad b^{\mathfrak{L}} = \gamma|_{S_y^{\mathfrak{L}}} \ , \tag{8}$$

and conversely $\gamma = a^{\mathfrak{R}} a^{\mathfrak{L}} b^{\mathfrak{R}} b^{\mathfrak{L}}$ or $\gamma = (a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$. Hence, it is common, for example, to see $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ instead of $L_{xy}(\gamma^{\mathfrak{R}})$ throughout the paper.

Definition 2.25 (Underlying combinatorial game). Given a tailored (non-local) game, we can refer to its *underlying combinatorial game*. By this, we mean the game with the same graph, a length function that disregards readability $\ell = \ell^{\mathfrak{R}} + \ell^{\mathfrak{L}}$, and the same decision predicates. Note that in any operative way, these are the same game, we just *forget* about the tailored structure that governs D_{xy}.

Remark 2.26 (Naive tailoring of any game). Being tailored may at first seem to be quite a restrictive form for a non-local game. Indeed, while the dependence of the decision function on the readable variables is allowed to be arbitrary, the dependence on the linear variables is restrictive — as not every boolean function can be expressed as a conjunction of affine-linear functions — for example, consider the OR function. However, observe that because the definition allows one to "tailor" according to any partition of the variables in "readable" and "unreadable" variables, every game can be tailored in a trivial manner, as follows. First, all variables are declared readable, namely $\ell^{\mathfrak{R}} = \ell$ and $\ell^{\mathfrak{L}} = 0$. Then, if the decision function D_{xy} decided to accept γ according to the original game, then it lets $L_{xy}(\gamma^{\mathfrak{R}})$ be empty (and thus all linear conditions will be satisfied regardless of what γ is). And, if D_{xy} decided to reject γ according to the original game, then it chooses $L_{xy}(\gamma^{\mathfrak{R}})$ to contain the singleton $\{J\}$ as the single subset appearing in L_{xy}. Note that $\{J\}$ represents the linear equation $1 \cdot \gamma(J) = 0$, which is $1 = 0$, and thus cannot be satisfied by any γ.

This raises the question: What have we gained by defining tailored non-local games, if any game can be tailored in a straightforward manner?

---

[20]We considered calling such games *controlled linear*, since it is more informative. But, since conditionally linear is a term we use in this paper, and terms containing linear are generally overused, we decided to use a less informative notion.

Definition 2.27 (Z-aligned permutation strategies). A strategy $\mathcal{U}$ for a tailored non-local game $\mathfrak{G}$ is said to be a Z-*aligned permutation strategy* if it associates to each vertex $\mathsf{x} \in V$ a readably Z-aligned (Definition 2.8) signed permutation PVM (Definition 2.15). Namely, in observable form, for every $\mathsf{X} \in S$ we have $\mathcal{U}(\mathsf{X}) \in \mathrm{Sym}_{\pm}(\Omega) \subseteq U(W^-)$ (which is the permutation strategy condition) and for every readable variable X the observable $\mathcal{U}(\mathsf{X})$ is diagonal (which is the readably Z-aligned condition). This is equivalent to having a permutation strategy such that each readable variable acts on each point in the signed set $\Omega_{\pm}$ either like the identity or like the sign flip $-\mathrm{Id}$.

We use the acronym ZPC to describe a Z-aligned permutation strategy that commutes along edges.

Remark 2.28. The classical strategies described in Example 2.20 are Z-aligned permutation strategies. But, one can construct permutation strategies that induce a classical strategy in the standard sense (namely, one whose all outputs are commuting) without it being Z-aligned.

It is clearer now why the way one tailors a non-local game matters: The existence of a perfect ZPC strategy for the game depends on it. Let us demonstrate this with binary linear constraint system (LCS) games, and specifically the Mermin–Peres magic square game. For a thorough introduction to the magic square game we refer to [Ara02], and more generally for an introduction to LCS games see [CLS17].

Example 2.29 (Linear constraint system games). Let $\mathscr{A}$ be an $m \times n$ matrix with $\mathbb{F}_2$-coefficients, and let $\vec{b}$ be a column vector in $\mathbb{F}_2^m$. Classically, such a pair defines a system of linear equations $\mathscr{A}\vec{x} = \vec{b}$ over $\mathbb{F}_2$. It also defines a certain non-local game $\mathfrak{Lcs}(\mathscr{A}, \vec{b})$ which is the quantum counterpart of this classical system of equations. In this game, an assignment to a random linear constraint in $\mathscr{A}\vec{x} = \vec{b}$ (i.e., a row) is asked for, and is crossed checked against some "global" assignment to the variables (i.e., columns) for consistency.

The vertices in the underlying graph of $\mathfrak{Lcs}(\mathscr{A}, \vec{b})$ will be indexed by the rows (i.e., linear constraints) and columns (i.e., variables) of the matrix $\mathscr{A}$, namely $\{\mathtt{const}_i \mid i \in [m]\}$ and $\{\mathtt{var}_j \mid j \in [n]\}$. There is an edge between $\mathtt{const}_i$ and $\mathtt{var}_j$ if and only if $\mathscr{A}_{ij} = 1$ — which is saying, the jth variable appears in the ith constraint. The length of every column vertex is 1, and we denote by Var_j the formal variable associated with the jth column $\mathtt{var}_j$. The length of each row vertex is the number of 1's in the row, and we associate formal variables $S_{\mathtt{const}_i} = \{\mathsf{Const}_{ij'} \mid \mathscr{A}_{ij'} = 1\}$ to $\mathtt{const}_i$. The decision function $D_{\mathtt{const}_i \, \mathtt{var}_j}$ gets as input an assignment γ to Var_j and $\{\mathsf{Const}_{ij'} \mid \mathscr{A}_{ij'} = 1\}$, and accepts if and only if

$$\sum_{j' : \mathscr{A}_{ij'} = 1} \gamma(\mathsf{Const}_{ij'}) = b_i \quad \text{and} \quad \gamma(\mathsf{Var}_j) = \gamma(\mathsf{Const}_{ij}), \tag{9}$$

namely, if the assignment induced by γ satisfies the ith constraint, and is consistent with the global assignment to the jth variable. Though for our discussion the distribution μ over edges in this game is not important, one can consider the following standard sampling scheme: 1) Choose a row uniformly at random. 2) Choose a uniform variable out of the support of the chosen row.

Let us describe a **non-trivial** tailoring of the LCS game $\mathfrak{Lcs}(\mathscr{A}, \vec{b})$. First, all variables are chosen to be *unreadable*, namely $\ell^{\mathfrak{R}} = 0$ and $\ell^{\mathfrak{L}} = \ell$. Given that the edge $\mathtt{const}_i \, \mathtt{var}_j$ was sampled, the controlled linear constraints $L_{\mathtt{const}_i \, \mathtt{var}_j}$ will consist of two checks, which are derived from (9):[21]

$$c_{\text{consistency}}(\mathsf{X}) = \begin{cases} 0 & \mathsf{X} \neq \mathsf{Var}_j, \mathsf{Const}_{ij} \\ 1 & \text{otherwise} \end{cases}$$

$$c_{\text{linear}}(\mathsf{X}) = \begin{cases} 0 & \mathsf{X} = \mathsf{Var}_j \\ 1 & \mathsf{X} = \mathsf{Const}_{ij'} \in S_{\mathtt{row}_i} \\ b_i & \mathsf{X} = \mathsf{J} \end{cases}$$

---

[21] Note that, as there are no readable variables, $L_{\mathtt{const}_i \, \mathtt{var}_j}$ is constant.

Then, $c_{\text{consistency}}$ forces the canonical decision procedure $D_{\text{const}_i\,\text{var}_j}$ to check consistency between the constraint assignment to the jth variable and the global one, i.e. $\gamma(\text{Var}_j) = \gamma(\text{Const}_{ij})$, and c_{linear} forces it to check that the ith linear constraint is indeed sarisfied, i.e. $\sum_{j':\,\mathscr{A}_{ij'}=1}\gamma(\text{Const}_{ij'}) = b_i$ — as required by the definition of the LCS game $\mathfrak{Lcs}(\mathscr{A},\vec{b})$.

The difference between the above tailored form of $\mathfrak{Lcs}(\mathscr{A},\vec{b})$ and the one suggested in Remark 2.26 may seem technical. But, here all the variables are unreadable, and in the version of Remark 2.26 all variables are readable. If all variables of a tailored game are readable, a Z-aligned permutation strategy for it in observable form is just a collection of diagonal matrices with ± 1 on the diagonal. These strategies are exactly the *classical* ones described in Example 2.20, and having a perfect strategy of this kind for an LCS game is the same as for the linear system $\mathscr{A}\vec{x} = \vec{b}$ to have a solution. On the other hand, when all the variables are unreadable, there could be a perfect Z-aligned permutation strategy without $\mathscr{A}\vec{x} = \vec{b}$ having a solution. This is demonstrated in the next example, which is used in the proof of Compression (Theorem 2.53).

Example 2.30 (The Peres–Mermin Magic Square game). The system of linear equations associated with the magic square game has 6 constraints and 9 variables, and is defined as follows:

$$\texttt{row}_1 : \quad \text{Var}_{11} + \text{Var}_{12} + \text{Var}_{13} = 0,$$
$$\texttt{row}_2 : \quad \text{Var}_{21} + \text{Var}_{22} + \text{Var}_{23} = 0,$$
$$\texttt{row}_3 : \quad \text{Var}_{31} + \text{Var}_{32} + \text{Var}_{33} = 0,$$
$$\texttt{col}_1 : \quad \text{Var}_{11} + \text{Var}_{21} + \text{Var}_{31} = 1,$$
$$\texttt{col}_2 : \quad \text{Var}_{12} + \text{Var}_{22} + \text{Var}_{32} = 1,$$
$$\texttt{col}_3 : \quad \text{Var}_{13} + \text{Var}_{23} + \text{Var}_{33} = 1.$$

The choice for the names of the variables and constraints comes from visualising the variables positioned in a 3×3 grid, and asking for the values in each row to sum up to 0 while the values in each column should sum up to 1:

$$
\begin{array}{ccccccl}
\text{Var}_{11} & + & \text{Var}_{12} & + & \text{Var}_{13} & = 0 \\
+ & & + & & + & \\
\text{Var}_{21} & + & \text{Var}_{22} & + & \text{Var}_{23} & = 0 \\
+ & & + & & + & \\
\text{Var}_{31} & + & \text{Var}_{32} & + & \text{Var}_{33} & = 0 \\
\| & & \| & & \| & \\
1 & & 1 & & 1 &
\end{array}
$$

It is straightforward to see that this system has no solution (e.g., by adding up all the constraints). Therefore, it has no classical perfect strategy, and thus no perfect Z-aligned permutation strategy according to the naive tailoring of Remark 2.26. But, it **has** a perfect Z-aligned permutation strategy, acting on a signed set of size 8, with respect to the tailoring described in Example 2.29. In Figure 1, 5 permutations are visualized — $-\text{Id}, \mathbb{X}^{\otimes 01}, \mathbb{X}^{\otimes 10}, \mathbb{Z}^{\otimes 01}, \mathbb{Z}^{\otimes 10}$.[22] To get the perfect permutation strategy for the magic square game, take the mapping

$$
\begin{array}{lll}
\text{Var}_{11} \mapsto \mathbb{X}^{\otimes 10} & \text{Var}_{12} \mapsto \mathbb{X}^{\otimes 01} & \text{Var}_{13} \mapsto \mathbb{X}^{\otimes 10}\mathbb{X}^{\otimes 01} \\
\text{Var}_{21} \mapsto \mathbb{Z}^{\otimes 01} & \text{Var}_{22} \mapsto \mathbb{Z}^{\otimes 10} & \text{Var}_{23} \mapsto \mathbb{Z}^{\otimes 10}\mathbb{Z}^{\otimes 01} \\
\text{Var}_{31} \mapsto -\text{Id} \cdot \mathbb{X}^{\otimes 10}\mathbb{Z}^{\otimes 01} & \text{Var}_{32} \mapsto -\text{Id} \cdot \mathbb{X}^{\otimes 01}\mathbb{Z}^{\otimes 10} & \text{Var}_{33} \mapsto -\text{Id} \cdot \mathbb{X}^{\otimes 10}\mathbb{X}^{\otimes 01}\mathbb{Z}^{\otimes 10}\mathbb{Z}^{\otimes 01}
\end{array}
$$

Note that the $\mathbb{Z}$ permutations are Z-aligned. This is no coincidence, and it will be helpful when later used. We leave the discussion on where this strategy comes from to Section 3.7.

As discussed in the introduction, the main result of [JNV$^+$21] is that approximating the quantum value of a game (Definition 2.21) is as hard as the Halting problem. This shown by a reduction: one exhibits a computable mapping from (encodings[23] of) Turing machines $\mathcal{M}$ to (encodings of) games $\mathfrak{G}_{\mathcal{M}}$ such that, if $\mathcal{M}$ halts then $\text{val}^*(\mathfrak{G}_{\mathcal{M}}) = 1$, and if $\mathcal{M}$

---

[22]Throughout this paper, we use $\mathbb{X}$ and $\mathbb{Z}$ for the Pauli matrices. As the notation $\mathbb{Z}$ for integers is rarely used in this paper, this should not be confusing for the reader.

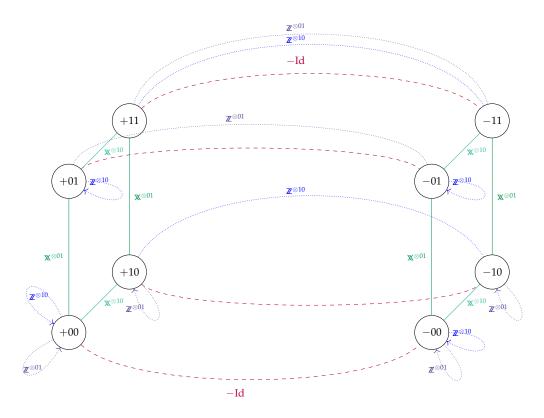[23]See Section 2.5.1 for a discussion of the role played by encodings.

Figure 1: In this figure there are 5 permutations, $-\mathrm{Id}, \mathbb{X}^{\otimes 01}, \mathbb{X}^{\otimes 10}, \mathbb{Z}^{\otimes 01}, \mathbb{Z}^{\otimes 10}$, acting on the set $(\mathbb{F}_2^2)_\pm$. The $\mathbb{X}$ permutations act as bit flips. The $\mathbb{Z}$ permutations are conditional sign changes, namely, they flip the sign depending on whether the associated bit is 0 or 1. Finally, $-\mathrm{Id}$ flips the sign.

does not halt then $\mathrm{val}^*(\mathfrak{G}_\mathcal{M}) \leq \frac{1}{2}$.[24] The goal of this paper is to reprove this result with two extra conditions: The game $\mathfrak{G}_\mathcal{M}$ needs to be tailored, and the perfect strategy $\mathscr{S}$ (in the complete case, i.e. the case where $\mathcal{M}$ halts) needs to be ZPC. Formally:

Theorem 2.31 (TailoredMIP* = RE). *There exists a polynomial time algorithm that takes as input (the encoding of) a Turing machine $\mathcal{M}$ and outputs (the encoding of) a **tailored** game $\mathfrak{G}_\mathcal{M}$ (see Definition 2.24) such that:*

(1) Sampling $\mathrm{xy} \in E$ according to μ and evaluating $D_{\mathrm{xy}}(\cdot)$ from the encoding of the game $\mathfrak{G}_\mathcal{M}$ can be done in time $\mathrm{poly}(|\mathcal{M}|)$, where $|\mathcal{M}|$ is the bit-length of the encoding of $\mathcal{M}$.

*(2) If $\mathcal{M}$ halts, then there exists a perfect Z-**aligned permutation** strategy $\mathscr{S}$ for $\mathfrak{G}_\mathcal{M}$ that commutes along edges (see Definitions 2.18 and 2.27). In particular, $\mathrm{val}^*(\mathfrak{G}_\mathcal{M}) = 1$.*

(3) If $\mathcal{M}$ never halts, then $\mathrm{val}^*(\mathfrak{G}_\mathcal{M}) < 1/2$.

2.5 Encoding tailored games

This section is devoted to an encoding scheme for tailored games, and following [JNV$^+$21] we use the term *tailored normal form verifiers* (TNFV) for it (Section 5.1.1 motivates this term). As seen in Theorem 2.31, some encoding mechanism for games is needed to be able to prove our result, and also to be able to phrase the compression theorem rigorously.

---

[24]Here, $\frac{1}{2}$ is an arbitrary constant chosen for convenience.

2.5.1 Prelude — Encodings, Running time and Description length

Encodings

By *encoding* we mean a correspondence (not necessarily single valued or onto) between a collection of objects — graphs, games, Turing machined, functions, etc. — to the set of (finite) bit strings $\{0,1\}^*$. In computer science, whenever one performs manipulations (e.g. an algorithm) on a collection of objects, one ought to have in mind an encoding thereof. This is because, ultimately, each instance of the collection is meant to be represented, and manipulated, on a computer — which processes strings of bits. Encodings are thus essential as a tool to connect high-level language to concrete implementations. Different choices of encodings can affect, as we will shortly demonstrate, the running time of procedures performed on them. They can also, of course, affect the resilience of stored data to errors, which is the fundamental goal in the theory of error correcting codes. Besides their practical importance, encodings are important theoretical tools — for example encodings enable *self-reference*, which is the backbone of the classical incompleteness and undecidability results of Gödel and Turing [Göd31, Tur37].

Before proceeding let us first fix a computational model. For us, an algorithm is always represented by a *Turing machine*. Informally, a Turing machine (TM) is a finite-state machine that processes data presented on an *input tape* (or maybe several input tapes), using a *memory tape* to store intermediate information and an *output tape* to write its output. The input tape is read-only, the memory tape is read-write and the output tape is write-only. Each tape has infinitely many memory cells, indexed by integers. The Turing machine has one head for each tape, initially positioned at location 0 (of its appropriate tape). At each *time step*, according to its current *internal state* and the bits each of the heads is reading, the Turing machine may move a head by ± 1 (or leave it in place) along its tape, read or write a symbol, and change its internal state (to one of finitely many possible ones). The Turing machine has a designated "halt" state; when it reaches that state its *output* is the contents of the output tape. For a more complete description of this standard model, we refer to [Sip12].

Now let us consider, e.g., a Turing machine that decides if an input graph is connected or not. In high-level language, this can be performed efficiently by, for example, breadth-first search: First, you need to have a queue and a list. Start from an arbitrary vertex, write its name in the beginning of the list, and put it in the rear of the queue (which is also the front of the queue at this point). Then, repeat the following until the queue is empty: Pop the vertex from the front of the queue (namely, remove it from the queue while reading its name); go over the neighbors of the popped vertex — if a neighbor appears in the list, do nothing, and if it does not appear, add it to the list and put it at the rear of the queue. At the end of this process, you will have a list of vertices (and an empty queue). Go over these vertices and check whether all vertices of the input graph have been visited.

While this may be straightforward to understand intuitively, implementing the algorithm as a Turing machine requires one to make a number of choices that affect the running time. An important such choice is the way that the graph passed as input is represented (one also needs to specify the implementation of the queue and list using the memory tape of the TM, but we ignore this intricacy). There are two standard possible encodings. The first represents the graph as an adjacency matrix. If there are n nodes, one will first write the binary representation of the integer n, then a separator symbol $\sqcup$,[25] and then n sequences of n bits representing the n rows of the adjacency matrix. The second encoding is by "adjacency lists:" we first write n in binary, then $\sqcup$, and then n sequences of a multiple of $\lceil \log n \rceil$ bits each, such that the i-th sequence lists the labels of all vertices connected to the i-th vertex. For example, a triangle is represented as

$$11 \sqcup 011 \sqcup 101 \sqcup 110$$

in the first representation, and as

$$11 \sqcup 0110 \sqcup 0010 \sqcup 0001$$

in the second representation. Here, 0110, 0010 and 0001 are unambiguously interpreted as the neighbors 2 and 3 of vertex 1, 1 and 3 of vertex 2, and 1 and 2 of vertex 3; this is because the number of vertices 3 given first specifies how many bits each vertex is represented with and we naturally label vertices starting with 00, 01, etc.

---

[25]This symbol should itself be represented as a string of bits. Namely, an encoding of a larger alphabet which includes $\{0, 1, \sqcup\}$ is needed. A way of doing that is suggested in Definition 2.34.

There are important differences between these representations. Firstly, they generally do not have the same size: for a graph with n vertices and m edges, the first representation has size $O(n^2)$ while the second has size $O((n + m)\log n)$. Secondly, certain algorithms run faster on one or the other representation — here, it should be clear that the breadth-first search algorithm will take advantage of the second representation for the case of sparse graphs, as it immediately gives access to all neighbors without having to parse a long row which may contain mostly 0's.

Now, because Theorem 2.31 states the existence of an algorithm, with certain properties and in particular a certain runtime, that takes as input a Turing machine, for the theorem to be precise we need to fix some encoding of Turing machines. However, in contrast to the encoding of tailored games described in detail in the next section, the specific encoding of Turing machines that we need is not very strict. We henceforth assume that a specific encoding of Turing machines has been fixed, that satisfies the conditions mentioned in Section 3.1 of [JNV$^+$21]; essentially, we need the following:

Fact 2.32 (Cf. [HMU06] and [AB09]). *There is an encoding scheme for Turing machines as bit strings which satisfies:*

(1) *The length of the encoding of a TM is reasonably sized (polynomial) as a function of the number of states it can be in.*

(2) *A Turing machine is able to take the encoding of another Turing machine as input, and execute the latter with some polynomial overhead in the running time;*

(3) *Fixing some of the inputs of a Turing machine enlarges its description length by at most some polynomial in the lengths of the fixed inputs.*

These conditions are easily satisfied with standard encodings (for a detailed reference, including the universal simulation theorem, see e.g. [HMU06]).

Remark 2.33. In the remainder of the paper we describe Turing machines using high-level language and make statements about their runtime and description length; it will always be clear that a low-level formalization, in terms of states and transition functions, of the high-level description can be obtained which satisfies the claimed runtime and description length bounds.

The inputs to Turing machines are assumed to be bit strings, namely elements of the free monoid $\{0, 1\}^*$ which are finite sequences of 0's and 1's. But, as mentioned above, some larger alphabets are sometimes needed to be able to describe certain objects. To that end, we use the following:

Definition 2.34 (The Alphabet). Let $\{0, 1, \sqcup, \mathfrak{error}\}$ be the finite alphabet we wish to encode. To that end, we define an encoding map from the above set as follows:

$$\mathrm{enc}(0) = 00 \quad \mathrm{enc}(1) = 01 \quad \mathrm{enc}(\sqcup) = 10 \quad \mathrm{enc}(\mathfrak{error}) = 11.$$

This map extends naturally to an isomorphism of monoids

$$\mathrm{enc} \colon \{0, 1, \sqcup, \mathfrak{error}\}^* \to \{00, 01, 10, 11\}^* = \{\text{all even length bit strings}\}.$$

Given a bit string $x \in \{0, 1\}^*$, we define a decoding map

$$\mathrm{dec} \colon \{0, 1\}^* \to \{0, 1, \sqcup\}^* \cup \{\mathfrak{error}\}$$

as follows — given an input $x \in \{0, 1\}^*$:

- First, dec checks that the number of bits in x, which we denote by $|x|$, is even. If not, it outputs $\mathfrak{error}$.

- Otherwise, x is of even length, and as we mentioned enc has a unique inverse on bit strings of even length. Let $y = \mathrm{enc}^{-1}(x) \in \{0, 1, \sqcup, \mathfrak{error}\}^*$.

- Finally, if y contains an **error** symbol, then dec will output **error**. Otherwise, dec will output y, which in this case is in $\{0, 1, \sqcup\}^*$.

Definition 2.35 (Encodings of integers). It is common to assume that certain positive integers $\mathbb{N} = \{1, 2, 3, ...\}$ are provided as inputs to Turing machines. There are two natural ways of achieving that, with substantial differences between them:

- **Binary**: There is a bijection $\overline{(\cdot)} \colon \mathbb{N} \to \{0, 1\}^*$ which writes n in binary and chops its leftmost bit. So, e.g., $\overline{1}$ is the empty bit string ε, $\overline{3}$ is 1 and $\overline{4}$ is 00. When we say that a certain input n to a TM $\mathcal{M}$ is an integer in binary, we mean that $\mathcal{M}$ receives $\overline{n}$. We often abuse notation and denote $\mathcal{M}(n)$ instead of $\mathcal{M}(\overline{n})$, though the explicit input must be a bit string.

- **Unary**: The word length function $|\cdot| \colon \{0, 1\}^* \to \mathbb{N}$ translates a bit string into an integer. A TM that ignores the specific input bit string x, and only uses its length $|x|$ in its computation, is said to take the integer $n = |x|$ as an input in unary. This is not the standard notion of unary input, which assumes n is encoded as $1^{*n} = \underbrace{1...1}_{n-times}$, but any length n bit string is an encoding for n.

Running time

Though RE, the class of languages for which the Halting Problem is complete, is defined without any running time constraints — namely, it is a computational class and not a complexity class — the class TailoredMIP* does have running time restrictions in its definition (see Section 5.1.1 for the formal definition of both). Many of the arguments in this paper rely on efficient running time of certain algorithms (Turing machines), and sometimes even not so common variations on efficient running time are needed (e.g., Definition 2.50). To that end, we need to define running time.

Recall that a Turing machine holds a table (function) that tells it given the current reads from its heads (on the input tapes, memory tape and output tape) and the current internal state of the machine, to which state to move, what to write on the current position (in the memory and output tape) and to which direction each of the heads needs to move. The computation of the Turing machine progresses by following the table and transforming the state, content of tapes and position of heads accordingly — each application of the table rules is called a *time step*.[26]

Definition 2.36 (Running Time). Let $\mathcal{M}$ be a k-input Turing machine, and $x_1, ..., x_k \in \{0, 1\}^*$ be k bit strings. The computation of $\mathcal{M}$ given $x_1, ..., x_k$ as inputs may halt or not. If it halted, it took some finite amount of time steps to get there, and we denote this number by $\mathbb{T}(\mathcal{M}; x_1, ..., x_k)$ — if the TM did not halt, this function outputs ∞. In the case it halts, the output of $\mathcal{M}$ is what is written in the output tape in the end of the computation, so $\mathcal{M}$ defines a partial function $\mathcal{M} \colon (\{0, 1\}^*)^k \to \{0, 1\}^*$.

Given a function $f \colon \mathbb{N}^k \to \mathbb{N}$, we say that $\mathcal{M}$ runs in f-time if for every $x_1, ..., x_k \in \{0, 1\}^*$, we have

$$\mathbb{T}(\mathcal{M}; x_1, ..., x_k) \leq f(|x_1|, ..., |x_k|),$$

where $|\cdot|$ is, again, the word length function (in particular, $\mathcal{M}$ needs to halt regardless of the input) — this is denoted by $\mathbb{T}(\mathcal{M}) \leq f$, and if this is true only up to some universal constant $C' > 0$, then we denote it by $\mathbb{T}(\mathcal{M}) = O(f)$ (see Remark 1.2 for our asymptotic notation conventions). As is common, we say that $\mathcal{M}$ runs in *polynomial time* if there is some constant $C > 0$ such that for every $x_1, ..., x_k \in \{0, 1\}^*$, $\mathbb{T}(\mathcal{M}; x_1, ..., x_k) \leq C|x_1|^C \cdot ... \cdot |x_k|^C + C$ — this is often denoted by $\mathbb{T}(\mathcal{M}) = \text{poly}(|x_1|, ..., |x_k|)$. Similarly, we say that it runs in *exponential time* if $\mathbb{T}(\mathcal{M}) = 2^{\text{poly}(|x_1|, ..., |x_k|)}$. For $r < k$, we often use the notation

$$\mathbb{T}(\mathcal{M}; x_1, ..., x_r, \cdot, ..., \cdot) = \sup\{\mathbb{T}(\mathcal{M}; x_1, ..., x_r, x_{r+1}, ..., x_k) \mid x_{r+1}, ..., x_k \in \{0, 1\}^*\},$$

where we emphasize that the supremum on the right-hand side is taken over inputs $x_{r+1}, ..., x_k$ of arbitrary (unbounded) length.

---

[26] Intuitively, for real machines, operating such a step does take physical time, and that is the reason for the name.

Descriptions and description length

Often Turing machines are fed as input to other Turing machines, so they need to be encoded somehow. As we remarked in the encoding part of this Prelude, we do not describe this encoding in detail (only assume it satisfies the condition appearing in Section 3.1 of [JNV⁺21]). But, as we **do care** about running times, following the size of these encodings and the way they change with the various transformations applied on them is necessary. To that end,

Definition 2.37 (Description length)**.** Given a Turing machine $\mathcal{M}$, let $\overline{\mathcal{M}}$ be its *description*, i.e., a bit string which encodes $\mathcal{M}$. Let $|\mathcal{M}| = |\overline{\mathcal{M}}|$ be the *description length* of $\mathcal{M}$, which is the number of bits in the encoding of $\mathcal{M}$.

Remark 2.38. Although we do not describe the fixed encoding of TMs which we use, it is helpful to think of it as follows: Take your favorite programming language, say, Python. Then, every Turing machine can be written as a function in Python (with the appropriate number of inputs). The code for this function is just a string of symbols, and using ASCII, can be translated to a string of bits. The *description* of the TM is then the Python code for it (translated to bit strings using ASCII), and the *description length* is the length of this code.

In various places in this paper, we ask one TM to calculate the description of another TM. By this, we mean retrieve the code for the appropriate algorithm, which is described in a high level fashion along the paper.

2.5.2   Tailored normal form verifiers

Let us motivate the definition of normal form verifiers. The goal of the verifier is to encode an infinite sequence of games $\{\mathfrak{G}_n\}_{n \in \mathbb{N}}$ using a finite amount of data. This is a common theme in theoretical computer science, and is usually referred to as *uniform generation*. So, we want a finite object that "calculates" a function $n \mapsto \mathfrak{G}_n$. A natural choice would be an algorithm (i.e., Turing machine) $\mathcal{V}$ which on input n outputs the full description of $\mathfrak{G}_n$, according to some predefined encoding of underlying graphs, length functions, distributions over edges (which must be rational to be finitely described), and truth tables of decision functions.

We use a different type of encoding which is focused on the procedural manifestation of the nth game. Procedurally, games require both a sampling mechanism of an edge $\mathtt{xy} \in E$ (also known as a pair of questions), and the calculation of the decision predicate $D_{\mathtt{xy}}(\cdot) = D(\mathtt{x}, \mathtt{y}, \cdot, \cdot)$. Hence, our verifiers will consist of algorithms that perform the sampling and the decision process required in the nth game. There is a subtlety here, which is that the resulting game needs to be tailored (Definition 2.24) — a restriction not present in [JNV⁺21]. To model this we introduce two additional Turing machines in the definition of normal form verifiers compared to [JNV⁺21]. One of them calculates the (answer) length functions $\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}}$ of $\mathfrak{G}$, and the other calculates the controlled linear constraints function $L_{\mathtt{xy}}$.

For the sampling procedure, $\mathcal{S}$ will be a randomized Turing machine that on input n outputs a pair of bit strings $\mathtt{x}, \mathtt{y}$ interpreted as two vertices in V. A (bounded running time) randomized Turing machine can be assumed to first read a string of random bits from its randomness (whose length depends on the input n), and then apply a deterministic algorithm on the input that consists of n and the string of random bits, to finally produce $\mathtt{x}, \mathtt{y}$. Though this is a good benchmark for what a sampler is, for compression (Theorem 2.53) to work we need the sampler to be able to provide us with additional details on its inner workings: For example, the number of random bits it uses in the nth game, partial computations of its output, and so on — this appears in Definitions 4.29 and 4.44. At this point, let us stick to the simpler to follow definition.

For the decision algorithm, since $\mathfrak{G}_n$ is tailored, we can assume it is done in two steps. First, there is a Turing machine $\mathcal{A}$ which takes the index of the game n and a vertex $x \in V$ as input, and outputs $\ell^{\mathfrak{R}}(\mathtt{x}), \ell^{\mathfrak{L}}(\mathtt{x})$. Then, another Turing machine $\mathcal{L}$ takes $\mathtt{xy}$ and a bit string $\gamma^{\mathfrak{R}} = a^{\mathfrak{R}} b^{\mathfrak{R}}$ and calculates $L_{\mathtt{xy}}(\gamma^{\mathfrak{R}})$. Finally, a canonical Turing machine $\mathcal{D}$ takes as input a suggested answer $\gamma = a^{\mathfrak{R}} a^{\mathfrak{L}} b^{\mathfrak{R}} b^{\mathfrak{L}}$ together with the lengths $\ell^{\mathfrak{R}}(\mathtt{x}), \ell^{\mathfrak{L}}(\mathtt{x}), \ell^{\mathfrak{R}}(\mathtt{y}), \ell^{\mathfrak{L}}(\mathtt{y})$ and the sequence of linear constraints $L_{\mathtt{xy}}(\gamma^{\mathfrak{R}})$. It first checks that the restrictions $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ of γ are of the appropriate length, and that the linear constraints in $L_{\mathtt{xy}}(\gamma^{\mathfrak{R}})$ are properly formatted. Then, it verifies that γ satisfies the constraints in $L_{\mathtt{x}}(\gamma^{\mathfrak{R}})$. We now describe this encoding more rigorously.

Definition 2.39 (Sampler). A sampler $\mathcal{S}$ is a 1-input randomized Turing machine that gets as input an integer n in binary,[27] and outputs a pair of bit strings $\mathtt{x}, \mathtt{y}$.

Remark 2.40. We later restrict the family of samplers that we consider (see Definitions 4.29 and 4.44), and assume, given $\mathcal{S}$, to have access to certain subroutines of its calculation.

Definition 2.41 (Answer length calculator). An *answer length calculator* $\mathcal{A}$ is a 3-input Turing machine. The input tuple $(n, \mathtt{x}, \kappa)$ consists of an integer n in binary (Definition 2.35), a bit string $\mathtt{x}$, and a symbol $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$.

Remark 2.42.

- We expect $\mathtt{x}$ in the above definition of $\mathcal{A}$ to be the name of one of the vertices sampled by $\mathcal{S}$.

- The input κ should actually be a single bit 0 or 1, where 0 is interpreted as $\mathfrak{R}$ (namely, $\mathfrak{R}$ is encoded as 0) and 1 is interpreted as $\mathfrak{L}$ (namely, $\mathfrak{L}$ is encoded as 1). The reason we use the $\mathfrak{R}$ and $\mathfrak{L}$ symbols is mainly for readability, as is clarified in the next clause.

- The **decoded** (Definition 2.34) output of $\mathcal{A}$ is interpreted as the **unary** representation of an integer (Definition 2.35), which in turn, indicates the length functions in the nth game. This is done as follows (and is repeated in the description of the canonical decider $\mathcal{D}$, Definition 2.45): Say that $\mathtt{y} \in \{0,1\}^*$ is the output of $\mathcal{A}(n, \mathtt{x}, \kappa)$. First, we decode $\mathtt{y}$ using dec from Definition 2.34 — resulting in an element $\mathrm{dec}(\mathtt{y}) = \mathtt{z}$ in $\{0, 1, \sqcup\}^* \cup \{\mathtt{error}\}$. If $\mathtt{z} = \mathtt{error}$, the decider rejects. Otherwise, it uses the **length** of $\mathtt{z}$, $|\mathtt{z}|$, as the readable answer length $\ell^{\mathfrak{R}}(\mathtt{x})$ if $\kappa = \mathfrak{R}$, and as the linear answer length $\ell^{\mathfrak{L}}(\mathtt{x})$ if $\kappa = \mathfrak{L}$, in the nth game.

Definition 2.43 (Linear constraints processor). A *Linear constraints processor* $\mathcal{L}$ is a 5-input Turing machine. The input tuple $(n, \mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ consists of an integer n (in binary, Definition 2.35), signifying the index of the game, and four bit strings $\mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}$.

Remark 2.44. Note that **any** 5-input Turing machine can play the role of a linear constraints processor, in particular, one that does not halt. This is important for the way compression is used to deduce Theorem 2.31. In the above definition of $\mathcal{L}$, the input bit strings $\mathtt{x}, \mathtt{y}$ are expected to be the endpoints of the edge sampled by $\mathcal{S}$. The bit strings $a^{\mathfrak{R}}$ and $b^{\mathfrak{R}}$ are expected to be the restrictions of γ to the readable variables at $\mathtt{x}$ and $\mathtt{y}$ respectively (which we denoted by $\gamma^{\mathfrak{R}}$ beforehand). The output of $\mathcal{L}$ is expected to be (the encoding of) a sequence of bit strings $(c^1, ..., c^k)$, that will be interpreted by $\mathcal{D}$ as linear constraints on γ — namely, c_i is the ith row of a binary matrix, representing a system of linear equations over $\mathbb{F}_2$. This is done by encoding first the alphabet $\{0, 1, \sqcup\}$ as pairs of bits (as is done in Definition 2.34), and then writing $c^1 \sqcup c^2 \sqcup ... \sqcup c^k$ as the encoded version.

Definition 2.45 (Canonical Decider). The *canonical decider* is a 9-input Turing machine $\mathcal{D}$ that either accepts (i.e., outputs 1) or rejects (i.e., outputs 0). The input 9-tuple of $\mathcal{D}$ is

$$\left(\ell_a^{\mathfrak{R}}, \ell_a^{\mathfrak{L}}, \ell_b^{\mathfrak{R}}, \ell_b^{\mathfrak{L}}, a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}, L \right),$$

and all are bit strings. The canonical decider works in several steps. First, it checks that the inputs are properly formatted. This includes checking that

$$\mathrm{dec}(\ell_a^{\mathfrak{R}}), \ \mathrm{dec}(\ell_b^{\mathfrak{R}}), \ \mathrm{dec}(\ell_a^{\mathfrak{L}}), \ \mathrm{dec}(\ell_b^{\mathfrak{L}}), \ \mathrm{dec}(L) \neq \mathtt{error},$$

where dec is the decoding function from Definition 2.34. Then, $\mathcal{D}$ checks that

$$|a^{\mathfrak{R}}| = |\mathrm{dec}(\ell_a^{\mathfrak{R}})|, \ |a^{\mathfrak{L}}| = |\mathrm{dec}(\ell_a^{\mathfrak{L}})|, \ |b^{\mathfrak{R}}| = |\mathrm{dec}(\ell_b^{\mathfrak{R}})|, \ |b^{\mathfrak{L}}| = |\mathrm{dec}(\ell_b^{\mathfrak{L}})|,$$

---

[27]i.e., $\overline{n} \in \{0,1\}^*$ is the input to $\mathcal{S}$, as explained in Definition 2.35.

and lets
$$\Delta = |\text{dec}(\ell_a^{\mathfrak{R}})| + |\text{dec}(\ell_a^{\mathfrak{L}})| + |\text{dec}(\ell_b^{\mathfrak{R}})| + |\text{dec}(\ell_b^{\mathfrak{L}})| + 1.$$

Then, it checks that
$$\text{dec}(L) = c^1 \sqcup ... \sqcup c^k$$

for some sequence of bit strings $c^1, ..., c^k \in \{0,1\}^*$ each of which of length Δ, namely
$$\forall 1 \le i \le k : \quad |c^i| = \Delta.$$

If L is the empty string, then it is decoded to the empty sequence of constraints, which is assumed to be well formatted (and signifies the *no constraints* situation). If the inputs are not properly formatted, then $\mathcal{D}$ rejects. Otherwise, let w be the concatenation of the bit strings $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ together with an extra 1 at the end, namely $w = a^{\mathfrak{R}} a^{\mathfrak{L}} b^{\mathfrak{R}} b^{\mathfrak{L}} 1$. Since the inputs are well formatted, w and c^i are bit strings of the same length Δ. The canonical decider $\mathcal{D}$ evaluates the dot product (over $\mathbb{F}_2$, namely $(\text{mod } 2)$) between c^i and w; i.e.,

$$\forall 1 \le i \le k : \quad \langle c^i, w \rangle = \sum_{j=1}^{\Delta} c_j^i w_j.$$

Then, $\mathcal{D}$ accepts if all of the above dot products are zero, and rejects otherwise.

Remark 2.46. In the above definition of $\mathcal{D}$, the inputs $\ell_a^{\mathfrak{R}}$, $\ell_a^{\mathfrak{L}}$, $\ell_b^{\mathfrak{R}}$ and $\ell_b^{\mathfrak{L}}$ are expected to be the outputs of $\mathcal{A}(n, \text{x}, \mathfrak{R})$, $\mathcal{A}(n, \text{x}, \mathfrak{L})$, $\mathcal{A}(n, \text{y}, \mathfrak{R})$ and $\mathcal{A}(n, \text{y}, \mathfrak{L})$ respectively, where x, y is the pair sampled by $\mathcal{S}$. As mentioned in Remark 2.42, (the decodings of) these outputs are expected to be the (unary representation of the) readable and linear answer lengths

$$\ell^{\mathfrak{R}}(\text{x}), \ell^{\mathfrak{L}}(\text{x}), \ell^{\mathfrak{R}}(\text{y}), \ell^{\mathfrak{L}}(\text{y})$$

in the encoded game. If $\gamma : S_{\text{xy}} \to \mathbb{F}_2$ is the answer produced by running the strategy $\mathscr{S}$, then we use the notation of (8) to obtain

$$a^{\mathfrak{R}} = \gamma|_{S_{\text{x}}^{\mathfrak{R}}}, \quad a^{\mathfrak{L}} = \gamma|_{S_{\text{x}}^{\mathfrak{L}}}, \quad b^{\mathfrak{R}} = \gamma|_{S_{\text{y}}^{\mathfrak{R}}} \quad \text{and} \quad b^{\mathfrak{L}} = \gamma|_{S_{\text{y}}^{\mathfrak{L}}}.$$

Hence, the bit string w is exactly the extension of γ such that $\gamma(\mathsf{J}) = 1$. The last input L is expected to be the output of $\mathcal{L}(n, \text{x}, \text{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$. We intentionally did not require this output to be formatted in a specific way, and thus $\mathcal{D}$ needs to check on its own that this bit string is indeed an encoding of a sequence $(c^1, c^2, ..., c^k)$, where each c^i is a bit string of length Δ.

Definition 2.47 (Tailored normal form verifiers). A *tailored normal form verifier* (TNFV) is a quadruple of Turing machines $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$, where $\mathcal{S}$ is a sampler as in Definition 2.39, $\mathcal{A}$ is an answer length calculator as in Definition 2.41, $\mathcal{L}$ is a linear constraint processor as in Definition 2.43, and $\mathcal{D}$ is the canonical decider as in Definition 2.45.

Although $\mathcal{D}$ is fixed, we keep it in the notation.

Note that while the quadruple $\mathcal{V}$ seems to encode an infinite sequence of tailored games, it may not. This is because we did not restrict them enough — e.g., the sampler, answer length calculator and linear constraints processor may never halt (as opposed to the canonical decider that always halts, and in time which is linear in its input length). This leads us to the following.

Definition 2.48 (The nth game defined by a tailored normal form verifier). Let $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ be a TNFV (Definition 2.47), and let n be a positive integer. Assume:

- The sampler $\mathcal{S}(n)$, which is a randomized TM, always halts in at most $T \in \mathbb{N}$ time steps. In addition, by Definition 2.39, when $\mathcal{S}$ halts, (the encoding) of its output is a pair of bit strings $\text{x} \sqcup \text{y}$.

- The answer length calculator $\mathcal{A}(n, \text{x}, \kappa)$ halts for every bit string x of length at most T, and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$.

- The linear constraints processor $\mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ halts for every pair x, y of bit strings of length at most T and every pair of bit strings $a^{\mathfrak{R}}, b^{\mathfrak{R}}$ of lengths $|\mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{R}))|$ and $|\mathrm{dec}(\mathcal{A}(n, \mathrm{y}, \mathfrak{R}))|$ respectively, where dec is the decoding function (Definition 2.34) and $|\cdot|$ the *word length* function.

Then $\mathcal{V}_n$, the nth *game corresponding to* $\mathcal{V}$, is the following tailored non-local game: As $\mathcal{S}(n)$ runs for at most T steps, the output pair x, y consists of bit strings of length at most T. Then, the vertex set V of the graph underlying $\mathcal{V}_n$ consists of all bit strings of length at most T — indeed, the output of $\mathcal{S}(n)$ will always be some ordered pair from V. The edge set E will consist of all pairs $\mathrm{xy} \in V \times V$ that are possible outputs of $\mathcal{S}(n)$, and $\mu(\mathrm{xy})$ is the probability $\mathrm{x} \sqcup \mathrm{y}$ was the output of $\mathcal{S}(n)$.

As $\mathcal{A}(n, \mathrm{x}, \kappa)$ halts whenever $|\mathrm{x}| \leq T$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the readable length at $\mathrm{x} \in V$ can be defined to be $\ell^{\mathfrak{R}}(\mathrm{x}) = |\mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{R}))|$ and the unreadable length at x can be defined to be $\ell^{\mathfrak{L}}(\mathrm{x}) = |\mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{L}))|$.

For any $\gamma \colon S_{\mathrm{xy}} \to \mathbb{F}_2$, let us use the notation in (8), namely

$$a^{\mathfrak{R}} = \gamma|_{S_{\mathrm{x}}^{\mathfrak{R}}}, \quad a^{\mathfrak{L}} = \gamma|_{S_{\mathrm{x}}^{\mathfrak{L}}}, \quad b^{\mathfrak{R}} = \gamma|_{S_{\mathrm{y}}^{\mathfrak{R}}} \quad \text{and} \quad b^{\mathfrak{L}} = \gamma|_{S_{\mathrm{y}}^{\mathfrak{L}}}.$$

The output of $\mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ is either (an encoding of) a sequence of bit strings $c^1 \sqcup \ldots \sqcup c^k$ of lengths

$$\Delta = \ell^{\mathfrak{R}}(\mathrm{x}) + \ell^{\mathfrak{L}}(\mathrm{x}) + \ell^{\mathfrak{R}}(\mathrm{y}) + \ell^{\mathfrak{L}}(\mathrm{y}) + 1 ,$$

or not. If not, then we let $L_{\mathrm{xy}}(\gamma^{\mathfrak{R}})$ be the singleton $\{\mathtt{J}\}$ (which translates to definite rejection). Similarly, if one of the decodings $\mathrm{dec}(\mathcal{A}(\overline{n}, \mathrm{x}, \mathfrak{R})), \mathrm{dec}(\mathcal{A}(\overline{n}, \mathrm{x}, \mathfrak{L})), \mathrm{dec}(\mathcal{A}(\overline{n}, \mathrm{y}, \mathfrak{R}))$ or $\mathrm{dec}(\mathcal{A}(\overline{n}, \mathrm{y}, \mathfrak{L}))$ is $\mathfrak{error}$, then $L_{\mathrm{xy}}(\gamma^{\mathfrak{R}})$ will also be the singleton $\{\mathtt{J}\}$. If the output is well formatted, then we can interpret each term c^i in the sequence as an indicator function $c^i \colon S_{\mathrm{xy}} \cup \{\mathtt{J}\} \to \mathbb{F}_2$. Then we can add c^i to $L_{\mathrm{xy}}(\gamma^{\mathfrak{R}})$. This way we get some controlled linear constraint function $L_{\mathrm{xy}} \colon \mathbb{F}_2^{S_{\mathrm{x}}^{\mathfrak{R}} \cup S_{\mathrm{y}}^{\mathfrak{R}}} \to \mathbb{F}_2^{\mathbb{F}_2^{S_{\mathrm{xy}} \cup \{\mathtt{J}\}}}$. Note that the same indicator may appear more than once in the output of $\mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$; namely, there may be $i \neq j$ such that $c^i = c^j$. But, this does not affect the function L_{xy} nor the decision process of the canonical decider $\mathcal{D}$.

All in all, it is straightforward to check that the canonical $D_{\mathrm{xy}}(\gamma)$ from Definition 2.24 agrees with the output of

$$\mathcal{D}(\mathcal{A}(n, \mathrm{x}, \mathfrak{R}), \mathcal{A}(n, \mathrm{x}, \mathfrak{L}), \mathcal{A}(n, \mathrm{y}, \mathfrak{R}), \mathcal{A}(n, \mathrm{y}, \mathfrak{L}), a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}, \mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})) . \tag{10}$$

Remark 2.49. A TNFV $\mathcal{V}$ that satisfies the three bullets from Definition 2.47, is said to have a well defined corresponding n^{th} game $\mathcal{V}_n$. We often claim that transformations on TNFVs have a combinatorial effect on the level of $\mathcal{V}_n$ *when it is defined*, by which we mean the above restrictions apply.

Definition 2.50 (λ-bounded tailored normal form verifiers). Let $\lambda \in \mathbb{N}$ be an integer. A tailored normal form verifier $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ is λ-*bounded* if the following two conditions hold

1. Let n be a positive integer (in binary). The running times of $\mathcal{S}(n)$, $\mathcal{A}(n, \cdot, \cdot)$ and $\mathcal{L}(n, \cdot, \cdot, \cdot, \cdot)$ are bounded by n^{λ} for $n \geq 2$.[28] Namely

$$\forall 2 \leq n \in \{0,1\}^*, \ \forall \mathrm{x}, \kappa, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}} \in \{0,1\}^* : \ \mathbb{T}(\mathcal{S}; n), \ \mathbb{T}(\mathcal{A}; n, \mathrm{x}, \kappa), \ \mathbb{T}(\mathcal{L}; n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}) \leq n^{\lambda} .$$

2. The description length $|\mathcal{V}|$ of the verifier $\mathcal{V}$ (Definition 2.37) is bounded by λ.

Remark 2.51. A few things to note about λ-bounded TNFVs:

- First, for every n, the nth game of a λ-bounded TNFV is well defined (Remark 2.49). Namely, such verifiers do define an infinite sequence of tailored games in a uniform way.

---

[28]We do not require the bound to hold for $n = 1$, as 1^{λ} is always 1 and hence it is usually not satisfied.

- At this point, it is not clear what motivates the above running time restriction on $\mathcal{S}, \mathcal{A}$ and $\mathcal{L}$. This will be clarified in Section 5.1.1. Note the restriction is both very strict and somewhat weak. Usually, the running time of a TM is bounded as a function of the total length of all of its inputs, and we expect the TMs to be efficient, namely run in time polynomial in these lengths. Here, we ask the running times to be bounded only in terms of the first input, which means that these Turing machines often need to halt before even reading the entirety of their non-n inputs (as they may be too long). But, the running time is exponential in this first input — the length $|\overline{n}|$ is $\Theta(\log n)$ and thus $n^\lambda = 2^{\lambda \log n}$ is exponential in it.

- An easy observation that is later used in a somewhat subtle manner, is that for $\lambda > \lambda'$, a λ'-bounded TNFV is also a λ-bounded TNFV.

2.6 Proving $\mathsf{TailoredMIP}^* = \mathsf{RE}$: A protocol for the Halting Problem

2.6.1 Entanglement bounds

The Halting Problem (HP) is the following decision problem:[29] Given (an encoding of) a Turing machine $\mathcal{M}$, does it ever halt when run on the empty input? If it does halt, then this can be certified in finite time — just run the Turing machine. This shows that HP is recursively enumerable (in RE). The reason HP is undecidable (namely, it is not recursive — in R) is because there is no bound given $\mathcal{M}$ on the needed number of steps for it to halt, or alternatively a method for showing in finite time that it does not halt.[30]

$\mathsf{TailoredMIP}^*$ is the decision problem which was hinted at in Theorem 2.31: Given (an encoding[31] of) a tailored game $\mathfrak{G}$, does it have a perfect Z-aligned permutation strategy that commutes along edges (ZPC), or does every (quantum) strategy for it have value at most $1/2$? These types of decision problems are called "promise languages", as we are not categorizing all possible tailored games, but promised that the input either has a perfect ZPC strategy or is far from having good strategies. At first glance, it is not clear what is complicated about deciding this. To see that, let us demonstrate why $\mathsf{TailoredMIP}^*$ is in RE. For every dimension d, we can define a finite $1/d$-net of quantum strategies in the set of all d-dimensional strategies. This provides a countable sequence of strategies, and it can be proven that the $\lim \sup$ of the value of $\mathfrak{G}$ against these strategies is indeed $\mathrm{val}^*(\mathfrak{G})$. Thus, if $\mathfrak{G}$ has a perfect ZPC strategy, then in particular this sequence tends to 1, and since this sequence can be calculated it will certify that $\mathrm{val}^*(\mathfrak{G}) > 1/2$, which implies we are in the complete case. The complexity comes exactly from the fact that there is no clear d that depends on (the encoding of) $\mathfrak{G}$ which is the correct dimension we should look up to. This leads to the following definition:

Definition 2.52 (Entanglement requirements of a game)**.** Given a game $\mathfrak{G}$ and a threshold $\nu \in [0,1]$, let $\mathscr{E}(\mathfrak{G}, \nu)$ denote the minimum integer d such that there exists a d-dimensional (synchronous, quantum) strategy $\mathscr{S}$ whose value against $\mathfrak{G}$ is at least ν. If there is no such strategy, then define $\mathscr{E}(\mathfrak{G}, \nu)$ to be ∞.

2.6.2 Compression

We can now formulate the compression theorem, and deduce using it Theorem 2.31. The idea behind compression is to substitute a λ-bounded tailored normal form verifier $\mathcal{V}$ by a λ-bounded tailored normal form verifier $\mathcal{V}'$ that simulates it with exponential speedup. Namely, perfect ZPC strategies for $\mathcal{V}_{2^n}$ translate to perfect ZPC strategies for $\mathcal{V}'_n$, and almost perfect quantum strategies for $\mathcal{V}'_n$ translate to almost perfect quantum strategies for $\mathcal{V}_{2^n}$.[32] In addition, there is a lower bound on the entanglement requirements of $\mathcal{V}'_n$ which is exponential in n and independent of $\mathcal{V}$. In the body of the text we prove a slight variation on the following, see Theorem 4.34. The proved variation assumes an extra condition on $\mathcal{V}$ (i.e., it having a sampler

---

[29]See Section 5.1.1 for more on decision problems.

[30]This is a theorem, first shown by Turing [Tur37].

[31]The exact encoding does not matter at this point. As normal form verifiers were defined by now, we can assume the game $\mathfrak{G}$ is encoded as a **pair** consisting of a λ-bounded normal form verifier $\mathcal{V}$ and an index $n \in \mathbb{N}$, and then $\mathfrak{G} = \mathcal{V}_n$ as defined in Definition 2.48.

[32]Almost perfect strategies are discussed in Section 3. Results that translate almost perfect strategies of one game to another are often deep and technical, and usually use ideas from the theory of Robustness of games (see Definition 3.30). Such results sit at the heart of compression.

which is h-level conditionally linear, as defined in Definition 4.29), but ensures that $\mathcal{V}'$ also satisfies the same extra condition. This change does not effect the deduction of Theorem 2.31, as can be verified by the reader — one uses $\mathsf{Compress}_5$ from Theorem 4.34 instead of $\mathsf{Compress}$ from Theorem 2.53.

Theorem 2.53 (Compression theorem for tailored games)**.** *There exists a universal positive integer constant C and a polynomial time 2-input Turing machine $\mathsf{Compress}$ that takes as input a TNFV $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ and a positive integer λ (in binary), and outputs a TNFV $\mathsf{Compress}(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}^\lambda, \mathcal{A}^\lambda, \mathcal{L}', \mathcal{D})$, such that: $\bullet$ $\mathcal{S}^\lambda$ and $\mathcal{A}^\lambda$ depend only on λ, can be calculated from it in time $\mathrm{polylog}(\lambda)$, and run in time $\mathrm{poly}(n, \lambda)$.[33] In addition, given that $\mathtt{x}$ is a possible output of $\mathcal{S}^\lambda(n)$, and that $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the output of $\mathcal{A}^\lambda(n, \mathtt{x}, \kappa)$ never decodes (Definition 2.34) to an $\mathfrak{error}$ sign. $\bullet$ $\mathcal{L}'$ depends on both λ and $\mathcal{V}$, it can be calculated from them in time $\mathrm{poly}(\log \lambda, |\mathcal{V}|)$, and runs in $\mathrm{poly}(n, \lambda)$-time. $\bullet$ The canonical decider $\mathcal{D}$ (Definition 2.45) is fixed and runs in time which is linear in its input length. $\bullet$ If $\mathcal{V}$ is λ-bounded, then $\mathcal{V}'$, the output of $\mathsf{Compress}$, satisfies for all $n \geq C$,*

1. ***Completeness**: If $\mathcal{V}_{2^n}$ has a perfect Z-aligned permutation strategy that commutes along edges (ZPC strategy), then so does $\mathcal{V}'_n$.*

2. ***Soundness**: $\mathscr{E}(\mathcal{V}'_n, \frac{1}{2}) \geq \max\left\{ \mathscr{E}(\mathcal{V}_{2^n}, \frac{1}{2}), 2^{2^{\lambda n} - 1} \right\}$.*

Remark 2.54. It may seem technical, and even unnatural, that the time complexities of the sampler, answer length calculator and linear constraints processor of the compressed verifier are $\mathrm{poly}(n, \lambda)$. One would expect $\mathcal{V}'$ to be λ-bounded, which requires a bound of the form n^λ. Note that for every $c > 0$ there is a large enough λ such that n^λ upper bounds $c(n^c + \lambda^c)$ for $n \geq 2$ (cf. Lemma 12.4 in [JNV$^+$21]). This *better than λ-bounded* condition is crucial for the Halting problem reduction to work out. Specifically, it is used in Lemma 2.61, which plays a key role in the reduction.

Remark 2.55. The formulation of the compression theorem, Theorem 2.53, is hiding the approach to prove it in some sense. Disregarding the complexity theoretic part (which is critical, but independent of what we are emphasizing now), the point is that we transform a game $\mathcal{V}_{2^n}$ to a game $\mathcal{V}'_n$ in a complete and sound way. The completeness and soundness that we prove are actually stronger than what the formulation reveals.

The completeness that is actually proven is that any perfect ZPC strategy for $\mathcal{V}_{2^n}$ can be transformed into a perfect ZPC strategy for $\mathcal{V}'_n$. The soundness that we actually prove is that every strategy for $\mathcal{V}'_n$ with value $1 - \varepsilon$ can be perturbed so that a strategy with value $1 - f(\varepsilon)$ for $\mathcal{V}_{2^n}$ can be extracted out of it. Following the bounds deduced on the function f throughout the steps of compression, one can show that for $\varepsilon < 1/2$, also $f(\varepsilon) < 1/2$, namely $\mathrm{val}^*(\mathcal{V}_{2^n}) < 1/2$ implies $\mathrm{val}^*(\mathcal{V}'_n) < 1/2$. Furthermore, the entanglement needed to win $\mathcal{V}'_n$ with probability $1 - \varepsilon$ is (morally) the product of the entanglement needed to win $\mathcal{V}_{2^n}$ with probability $1 - f(\varepsilon)$ and $(1 - f(\varepsilon)) \cdot 2^{2^{\lambda n}}$, which is substantially larger than the maximum between them. Namely, we can deduce something of the form $\mathscr{E}(\mathcal{V}'_n, 1 - \varepsilon) \geq \mathscr{E}(\mathcal{V}_{2^n}, 1 - f(\varepsilon)) \cdot (1 - f(\varepsilon)) \cdot 2^{2^{\lambda n}}$ for all $\varepsilon > 0$.

This viewpoint is better for understanding the structure of completeness and soundness proofs of the transformations associated with compression. We elaborate on this in Section 3.

2.6.3 A $\mathsf{TailoredMIP}^*$-protocol for the Halting Problem: Proving Theorem 2.31 assuming Theorem 2.53

This section is devoted to the proof of our main theorem, Theorem 2.31, assuming Compression, Theorem 2.53. The idea in the reduction is to transform a Turing machine $\mathcal{M}$ and an integer λ into a tailored normal form verifier $\mathcal{V}^{\mathcal{M}, \lambda}$ that is a "fixed point" of the algorithm $\mathsf{Compress}$ from Theorem 2.53 — this approach is part of a long tradition of fixed point theorems in computation theory, cf. [Rog87]; see also [MSNY24] for a broader perspective on the connection between compression techniques and undecidability. Then, we show that there is a constant $\lambda = \lambda(\mathcal{M})$ — that is bounded by a polynomial in the description length of $\mathcal{M}$ — such that $\mathcal{V}^{\mathcal{M}, \lambda}$ is λ-bounded. Finally, using the properties described in Theorem 2.53 and the fact that $\mathcal{V}^{\mathcal{M}, \lambda}$ is a fixed point, we can choose $\mathfrak{G}_\mathcal{M} = \mathcal{V}_C^{\mathcal{M}, \lambda}$ — the Cth game defined by the verifier $\mathcal{V}^{\mathcal{M}, \lambda}$, as in Definition

---

[33]By saying that their running time is $\mathrm{poly}(n, \lambda)$, we mean that no matter what the other inputs are, the runtime of $\mathcal{S}^\lambda(n)$ and $\mathcal{A}^\lambda(n, \mathtt{x}, \kappa)$ is bounded only by $c \cdot (n^c + \lambda^c)$ for some universal constant c.

2.48, where C is the constant promised by Theorem 2.53 — and it satisfies the requirements of our main theorem, Theorem 2.31.

Recall that we fixed some encoding of Turing machines in Section 2.5.1. In Definition 2.37, we denoted by $\overline{\mathcal{M}}$ a description of $\mathcal{M}$, namely a bit string encoding of $\mathcal{M}$ according to the aforementioned encoding scheme of TMs. Furthermore, $|\mathcal{M}| = |\overline{\mathcal{M}}|$, the description length of $\mathcal{M}$, was the bit length of the description of $\mathcal{M}$.[34] The following is an adaptation of the Turing machine $\mathcal{F}$ described in Section 12.2 of [JNV$^+$21]. Note that in our case $\mathcal{F}$ plays the role of the linear constraints processor $\mathcal{L}$ and not the decider $\mathcal{D}$, which is fixed in the tailored case to be the canonical one.

Definition 2.56. Let $\mathcal{F}$ be an 8-input Turing machine. Its input is

$$(\overline{\mathcal{R}}, \overline{\mathcal{M}}, \lambda, n, \mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}),$$

where $\mathcal{R}$ is an 8-input Turing machine, $\mathcal{M}$ is a single input Turing machine, λ and n are integers in binary, and $\mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}$ are bit strings. The description of $\mathcal{F}$ is as follows:

(1) Run $\mathcal{M}$ on the blank input for n steps. If it halts, then return an empty tape. Continue otherwise.

(2) Compute the description $\overline{\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}}$ of the 5-input Turing machine $\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}$ defined by

$$\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}(\cdot,\cdot,\cdot,\cdot,\cdot) = \mathcal{R}(\overline{\mathcal{R}}, \overline{\mathcal{M}}, \lambda, \cdot, \cdot, \cdot, \cdot, \cdot),$$

i.e., on input $(\cdot,\cdot,\cdot,\cdot,\cdot)$ the TM $\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}$ calculates the output of $\mathcal{R}$ given input $(\overline{\mathcal{R}}, \overline{\mathcal{M}}, \lambda, \cdot, \cdot, \cdot, \cdot, \cdot)$.[35]

(3) Compute the descriptions $\overline{\mathcal{S}^{\lambda}}$ and $\overline{\mathcal{A}^{\lambda}}$ of the TMs $\mathcal{S}^{\lambda}$ and $\mathcal{A}^{\lambda}$ from Theorem 2.53, which are the sampler and answer length calculator that $\mathsf{Compress}(\cdot, \lambda)$ outputs regardless of which input normal form verifier it got.

(4) Compute the description $\overline{\mathcal{D}}$ of the canonical decider $\mathcal{D}$ from Definition 2.45.

(5) Let $\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda} = (\mathcal{S}^{\lambda}, \mathcal{A}^{\lambda}, \mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}, \mathcal{D})$ be a TNFV.

(6) Compute the description $\overline{(\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda})'}$ of the compressed verifier

$$(\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda})' = \mathsf{Compress}(\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda}, \lambda) = \left(\mathcal{S}^{\lambda}, \mathcal{A}^{\lambda}, (\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda})', \mathcal{D}\right),$$

where $\mathsf{Compress}$ is the algorithm discussed in Theorem 2.53.[36]

(7) Output $(\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda})'(n, \mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$; namely, simulate the operation of the compressed linear constraints processor $(\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda})'$ on the 5-tuple input $(n, \mathtt{x}, \mathtt{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$, and provide the same output as it did.

Definition 2.57 (The Halting tailored normal form verifier)**.** For every Turing machine $\mathcal{M}$ and $\lambda \in \mathbb{N}$, define the linear constraints processor $\mathcal{L}^{\mathcal{M},\lambda}$ to be the 5-input Turing machine

$$\mathcal{L}^{\mathcal{M},\lambda}(\cdot,\cdot,\cdot,\cdot,\cdot) = \mathcal{F}(\overline{\mathcal{F}}, \overline{\mathcal{M}}, \lambda, \cdot, \cdot, \cdot, \cdot, \cdot),$$

where $\mathcal{F}$ is the Turing machine from Definition 2.56 — note again, that this is just hard-coding the first three inputs of $\mathcal{F}$ to being $\overline{\mathcal{F}}, \overline{\mathcal{M}}$ and λ respectively, which makes it into a 5-input TM, and thus it can play the role of a linear constraints processor (Definition 2.43).

---

[34]In Remark 2.38 we provided a helpful heuristic way of thinking about these objects — $\overline{\mathcal{M}}$ is the code of some function in a programming language that behaves exactly as $\mathcal{M}$, and $|\overline{\mathcal{M}}|$ is the bit-length of this code.

[35]In the heuristic viewpoint of Remark 2.38, this is the same as taking the code of the 8-input function $\mathcal{R}$, and hard-coding the first three inputs of it to being $\overline{\mathcal{R}}, \overline{\mathcal{M}}$ and λ. The resulting function has only 5 free inputs, and is thus a 5-input TM which can play the role of a linear constraints processor (Definition 2.43).

[36]Note that the sampler and answer length calculator of both $\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda})'$ are **the same**. This is because of the way $\mathsf{Compress}$ operates, and our choice of sampler $\mathcal{S}^{\lambda}$ and answer length calculator $\mathcal{A}^{\lambda}$ for $\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda}$.

Now, define the halting tailored normal form verifier corresponding to $\mathcal{M}$ and λ to be

$$\mathcal{V}^{\mathcal{M},\lambda} = (\mathcal{S}^\lambda, \mathcal{A}^\lambda, \mathcal{L}^{\mathcal{M},\lambda}, \mathcal{D}) \,,$$

where, again, $\mathcal{S}^\lambda$ and $\mathcal{A}^\lambda$ are the sampler and answer length calculator that $\mathsf{Compress}(\cdot, \lambda)$ always outputs (from Theorem 2.53).

Remark 2.58. Let us note some properties of the linear constraints processor $\mathcal{L}^{\mathcal{M},\lambda}$ from Definition 2.57. Specifically, what is the output of

$$\mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}) = \mathcal{F}(\overline{\mathcal{F}}, \overline{M}, \lambda, n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$$

given that $\mathcal{M}$ does not halt in n steps. By inspecting the (high-level) description of the Turing machine $\mathcal{F}$ from Definition 2.56, one can see that the description of $\mathcal{L}^{\mathcal{R},\mathcal{M},\lambda}$ computed by $\mathcal{L}^{\mathcal{M},\lambda}$ at Step (2) is the description of $\mathcal{L}^{\mathcal{M},\lambda}$ itself. So, the TNFV $\mathcal{V}^{\mathcal{R},\mathcal{M},\lambda}$ computed in Step (5) is the Halting TNFV $\mathcal{V}^{\mathcal{M},\lambda}$ (Definition 2.57). Thus, as $\mathcal{M}$ does not halt in n steps, $\mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ will get to Step (7) and output the same output as $(\mathcal{L}^{\mathcal{M},\lambda})'(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$, where $(\mathcal{L}^{\mathcal{M},\lambda})'$ is the linear constraints processor of $\mathsf{Compress}(\mathcal{V}^{\mathcal{M},\lambda}, \lambda)$; namely,

$$\mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}) = (\mathcal{L}^{\mathcal{M},\lambda})'(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$$

whenever $\mathcal{M}$ does not halt in n steps. This is the way in which $\mathcal{L}^{\mathcal{M},\lambda}$, and thus $\mathcal{V}^{\mathcal{M},\lambda}$, is a fixed point of $\mathsf{Compress}(\cdot, \lambda)$. Furthermore, for all $\mathcal{M}$ and λ the linear constraints processor $\mathcal{L}^{\mathcal{M},\lambda}$ halts on all inputs, and $\mathcal{V}^{\mathcal{M},\lambda}$ is a tailored normal form verifier, though not necessarily λ-bounded.

Lemma 2.59. *Let $\mathcal{M}$ be a Turing machine, and λ and n positive integers. Recall the Halting TNFV $\mathcal{V}^{\mathcal{M},\lambda}$ from Definition 2.57, and let $(\mathcal{V}^{\mathcal{M},\lambda})' = \mathsf{Compress}(\mathcal{V}^{\mathcal{M},\lambda}, \lambda)$, where $\mathsf{Compress}$ is the transformation from Theorem 2.53.*

   1. *The underlying graph $G^\lambda = (V^\lambda, E^\lambda)$, length functions $\ell^{\mathfrak{R},\lambda}, \ell^{\mathfrak{L},\lambda}$ and distribution μ^λ over edges in $\mathcal{V}_n^{\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{M},\lambda})'_n$ — the nth games associated with $\mathcal{V}^{\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{M},\lambda})'$ (Definition 2.48) — are the same.*

   2. *If $\mathcal{M}$ does not halt in n steps, then the games $\mathcal{V}_n^{\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{M},\lambda})'_n$ are the same.*

   3. *If $\mathcal{M}$ halts in less than n steps, then $\mathcal{V}_n^{\mathcal{M},\lambda}$ is the always accepting game — namely, $L_{\mathrm{xy}}(\gamma^{\mathfrak{R}})$ is empty regardless of γ and $\mathrm{xy} \in E$, and thus D_{xy} accepts any $\gamma \colon S_{\mathrm{xy}} \to \{0, 1\}$.*

Proof. Clause 1. is immediate from the fact that the underlying graph, length functions and distribution over edges in Definition 2.48 depend only on the sampler and answer length calculator, and both $\mathcal{V}^{\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{M},\lambda})'$ have the same sampler $\mathcal{S}^\lambda$ and same answer length calculator $\mathcal{A}^\lambda$.

Clause 2. is deduced from Remark 2.58, which states that

$$\mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}) = (\mathcal{L}^{\mathcal{M},\lambda})'(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$$

in case $\mathcal{M}$ does not halt in n steps. Thus, as the length functions are the same for both games, L_{xy} is the same for $\mathcal{V}_n^{\mathcal{M},\lambda}$ and $(\mathcal{V}^{\mathcal{M},\lambda})'_n$. Since the rest of the data is the same as well, they are the exact same tailored game.

For clause 3., note that in this case $\mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ outputs an empty tape regardless of what $\mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}$ or $b^{\mathfrak{R}}$ are. By the properties of the answer length calculator in Theorem 2.53, for every vertex x in the underlying graph of $\mathcal{V}_n^{\mathcal{M},\lambda}$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$ we are guaranteed that $\mathcal{A}^\lambda(n, \mathrm{x}, \kappa)$ does not decode to $\mathfrak{error}$. Hence, every quadruple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ of respective lengths $|\mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{R})|, |\mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{L})|, |\mathrm{dec}(\mathcal{A}(n, \mathrm{y}, \mathfrak{R})|, |\mathrm{dec}(\mathcal{A}(n, \mathrm{y}, \mathfrak{L})|$ will make the canonical decider $\mathcal{D}$ (Definition 2.45) output 1 on input

$$(\mathcal{A}(n, \mathrm{x}, \mathfrak{R}), \mathcal{A}(n, \mathrm{x}, \mathfrak{L}), \mathcal{A}(n, \mathrm{y}, \mathfrak{R}), \mathcal{A}(n, \mathrm{y}, \mathfrak{L}), a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}, \mathcal{L}^{\mathcal{M},\lambda}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})) \,.$$

As described in Definition 2.48, the decision function $D_{\mathrm{xy}}(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ (using the notation of (8)) of the nth game $\mathcal{V}_n^{\mathcal{M},\lambda}$ agrees with the canonical decider in this setup. Namely, D_{xy} accept every possible $\gamma \colon S_{\mathrm{xy}} \to \{0, 1\}$, as claimed. $\qquad \square$

Corollary 2.60. *Let $\mathcal{M}$ be a Turing machine and λ an integer. Then the halting TNFV $\mathcal{V}^{\mathcal{M},\lambda}$ (Definition 2.57) has the following properties. For all $n \in \mathbb{N}$:*

1. *If $\mathcal{M}$ halts in n steps, then $\mathcal{V}_n^{\mathcal{M},\lambda}$ has a perfect Z-aligned permutation strategy that commutes along edges (ZPC).*

2. *If $\mathcal{M}$ does not halt in n steps, then $\mathcal{V}_n^{\mathcal{M},\lambda}$ has a perfect ZPC strategy if and only if $(\mathcal{V}^{\mathcal{M},\lambda})'_n$ does, where $(\mathcal{V}^{\mathcal{M},\lambda})' = \mathsf{Compress}(\mathcal{V}^{\mathcal{M},\lambda}, \lambda)$. Furthermore, under the same assumption (that $\mathcal{M}$ does not halt in n steps), it holds that*

$$\mathscr{E}\left(\mathcal{V}_n^{\mathcal{M},\lambda}, \frac{1}{2}\right) = \mathscr{E}\left((\mathcal{V}^{\mathcal{M},\lambda})'_n, \frac{1}{2}\right).$$

Proof. For item 1., by clause 3. of Lemma 2.59, $\mathcal{V}_n^{\mathcal{M},\lambda}$ always accepts. Thus, any deterministic strategy (as in Example 2.20) is a perfect Z-aligned permutation strategy for it. Since deterministic strategies are globally commuting, they in particular commute along edges.

Item 2. follows directly from clause 2. of Lemma 2.59, since they are the same tailored game. $\square$

Lemma 2.61. *There is a polynomial-time computable $\lambda = \lambda(\mathcal{M})$, scaling as $\mathrm{poly}(|\overline{\mathcal{M}}|)$, such that the verifier $\mathcal{V}^{\mathcal{M},\lambda}$ is λ-bounded. Moreover, the time complexities of $\mathcal{S}^\lambda$, $\mathcal{A}^\lambda$ and $\mathcal{L}^{\mathcal{M},\lambda}$ are $\mathrm{poly}(n, |\overline{\mathcal{M}}|)$.*[37]

Proof sketch. This is a combination of:

- The observation from Remark 2.54, i.e., that $\mathrm{poly}(n, \lambda)$ is dominated by n^λ for any large enough λ. Similarly, $\mathrm{polylog}(\lambda)$ is dominated by λ.

- An accounting argument of the running time and description length of $\mathcal{L}^{\mathcal{M},\lambda}$ through the definition of $\mathcal{F}$ (Definition 2.56).

- The time bounds of the sampler, answer length calculator and linear constraint processor of the compressed verifier in Theorem 2.53.

It is probably better for the readers to try and follow these calculations for themselves. In any case, a complete proof of the analogous claim appears in [JNV$^+$21, Lemma 12.5]. $\square$

Proof of Theorem 2.31. For every Turing machine $\mathcal{M}$, let $\lambda = \lambda(\mathcal{M})$ be the parameter promised by Lemma 2.61. Let $\mathcal{V}^{\mathcal{M},\lambda}$ be the tailored normal form verifier from Definition 2.57. Then, let $\mathfrak{G}_{\mathcal{M}} = \mathcal{V}_C^{\mathcal{M},\lambda}$ be the Cth game defined by $\mathcal{V}^{\mathcal{M},\lambda}$ (as in Definition 2.48), where C is the constant promised in Theorem 2.53.

First, let us show that the calculation of (the description of) $\mathcal{V}_C^{\mathcal{M},\lambda}$ takes at most $\mathrm{poly}(|\overline{\mathcal{M}}|)$-time. By Lemma 2.61, calculating λ takes $\mathrm{poly}(|\overline{\mathcal{M}}|)$-time. Now, calculating the description of $\mathcal{S}^\lambda$ and $\mathcal{A}^\lambda$ takes $\mathrm{polylog}(\lambda)$-time, which in turn is $\mathrm{poly}\log(|\overline{\mathcal{M}}|) \leq \mathrm{poly}(|\overline{\mathcal{M}}|)$. Furthermore, the decider $\mathcal{D}$ is fixed. Calculating the description of $\mathcal{L}^{\mathcal{M},\lambda}$ requires $\mathrm{poly}(|\overline{\mathcal{F}}|, |\overline{\mathcal{M}}|, \log \lambda)$ which is again $\mathrm{poly}(|\overline{\mathcal{M}}|)$ (as $|\overline{\mathcal{F}}|$ is a constant). Finally, fixing $n = C$ in all of these Turing machines adds at most a constant to their description. This proves that $\mathfrak{G}_{\mathcal{M}}$ can be calculated in time polynomial in $|\overline{\mathcal{M}}|$.

By Lemma 2.61, $\mathcal{S}^\lambda(C)$ runs in time $\mathrm{poly}(C, \lambda) = \mathrm{poly}(|\overline{\mathcal{M}}|)$. Recall Definition 2.48. For the edge set E and the distribution μ over it, Definition 2.48 took the pushforward along $\mathcal{S}^\lambda(C)$. This means that sampling according to μ is **exactly** running $\mathcal{S}^\lambda(C)$, and that takes $\mathrm{poly}(|\overline{\mathcal{M}}|)$-time. By Lemma 2.61, given $\gamma = (a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$, calculating $\mathcal{L}^{\mathcal{M},\lambda}(C, \mathsf{x}, \mathsf{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ takes at most $\mathrm{poly}(C, \lambda) = \mathrm{poly}(|\overline{\mathcal{M}}|)$ time, and in particular its output length is bounded by $\mathrm{poly}(|\overline{\mathcal{M}}|)$. Also by Lemma 2.61, $\mathcal{A}(C, \mathsf{x}, t)$ takes at most $\mathrm{poly}(|\overline{\mathcal{M}}|)$-time. Since $\mathcal{D}$ runs in time linear in its input, the value

$$D_{\mathsf{xy}}(\gamma) = \mathcal{D}(\mathcal{A}(C, \mathsf{x}, \mathfrak{R}), \mathcal{A}(C, \mathsf{x}, \mathfrak{L}), \mathcal{A}(C, \mathsf{y}, \mathfrak{R}), \mathcal{A}(C, \mathsf{y}, \mathfrak{L}), a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}, \mathcal{L}^{\mathcal{M},\lambda}(C, \mathsf{x}, \mathsf{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})),$$

can be calculated in time at most $\mathrm{poly}(|\overline{\mathcal{M}}|)$. This proves (1) in Theorem 2.31.

---

[37]Note that this extra condition is indeed a strengthening of being λ-bounded, since the dependence on the description length of $\mathcal{M}$ appears in the base and not the exponent — recall Remark 2.54.

Assume that $\mathcal{M}$ halts. Let N be the number of time steps it takes $\mathcal{M}$ to halt. For every $n \geq N$, by Corollary 2.60, $\mathcal{V}_n^{\mathcal{M},\lambda}$ has a perfect ZPC strategy. So, if $C \geq N$, then we are done. Otherwise, let n be such that

$$\max(C, \log N) \leq n < N.$$

By Lemma 2.59, $\mathcal{V}_n^{\mathcal{M},\lambda} = (\mathcal{V}^{\mathcal{M},\lambda})'_n$ in this case. By the compression theorem 2.53, since $n \geq C$, $(\mathcal{V}^{\mathcal{M},\lambda})'_n$ has a perfect ZPC strategy given that $\mathcal{V}_{2^n}^{\mathcal{M},\lambda}$ has one. But $2^n \geq N$, and we already argued that these tailored games have perfect ZPC strategies. Hence, $\mathcal{V}_n^{\mathcal{M},\lambda}$ has a perfect ZPC strategy when $n \geq \max(C, \log N)$. If $C \geq \log N$, then we are done. Otherwise, we can iterate this argument and deduce the same for any $n \geq \max(C, \log \log N)$. Since there exists some t for which $C \geq \underbrace{\log \ldots \log}_{t-\text{times}} N$, we deduce that $\mathfrak{G}_{\mathcal{M}} = \mathcal{V}_C^{\mathcal{M},\lambda}$ has a perfect ZPC strategy. This proves (2) in Theorem 2.31.

Assume that $\mathcal{M}$ does not halt. Then, by Lemma 2.59,

$$\mathcal{V}_n^{\mathcal{M},\lambda} = (\mathcal{V}^{\mathcal{M},\lambda})'_n \tag{11}$$

for every n. If $n \geq C$, then by the compression theorem 2.53, we have

$$\mathscr{E}((\mathcal{V}^{\mathcal{M},\lambda})'_n, 1/2) \geq \mathscr{E}(\mathcal{V}_{2^n}^{\mathcal{M},\lambda}, 1/2) \quad \text{and} \quad \mathscr{E}((\mathcal{V}^{\mathcal{M},\lambda})'_n, 1/2) \geq \underbrace{2^{2^{\lambda n}-1}}_{\geq 2^n}. \tag{12}$$

So, for every positive integer t we can deduce that

$$\mathscr{E}(\mathcal{V}_C^{\mathcal{M},\lambda}, 1/2) =_{(11)} \mathscr{E}((\mathcal{V}^{\mathcal{M},\lambda})'_C, 1/2)$$
$$\geq_{(12)} \mathscr{E}(\mathcal{V}_{2^C}^{\mathcal{M},\lambda}, 1/2) =_{(11)} \mathscr{E}((\mathcal{V}^{\mathcal{M},\lambda})'_{2^C}, 1/2)$$
$$\vdots$$
$$\geq_{(12)} \mathscr{E}(\mathcal{V}_{\underbrace{2^{\cdot^{\cdot^{2^C}}}}_{t-times}}^{\mathcal{M},\lambda}, 1/2) =_{(11)} \mathscr{E}((\mathcal{V}^{\mathcal{M},\lambda})'_{\underbrace{2^{\cdot^{\cdot^{2^C}}}}_{t-times}}, 1/2)$$
$$\geq_{(12)} \underbrace{2^{\cdot^{\cdot^{2^C}}}}_{(t+1)-times}.$$

Since this was true for every t, we can deduce that $\mathscr{E}(\mathcal{V}_C^{\mathcal{M},\lambda}, 1/2) = \infty$, which in turn proves that

$$\mathrm{val}^*(\mathfrak{G}_{\mathcal{M}}) = \mathrm{val}^*(\mathcal{V}_C^{\mathcal{M},\lambda}) < 1/2,$$

proving (3) in Theorem 2.31. $\square$

The rest of the paper is devoted to the proof of Compression, Theorem 2.53.

3 The compression toolbox

In the previous section we provided the minimal amount of preliminaries so that TailoredMIP* = RE (Theorem 2.31) and Compression (Theorem 2.53) can be phrased, and so that the former can be deduced from the latter. This section provides additional preliminaries needed for the proof of Compression. Specifically, we introduce various technical tools that are used in the completeness and soundness analysis of the transformations on games that take part in Compression. Section 3.1 provides useful functional analytic definitions and facts. In Section 3.2 we introduce a notion of distance between strategies; this notion takes into account the need to compare strategies in different dimensions through the use of isometries. In Section 3.3 we consider a frequent transformation on PVMs, *data processing*, and its effect on the distance measure. Section 3.4 contains useful lemmas for manipulating permutation strategies. Section 3.5 defines transformations that can be applied on games, which will be used repeatedly in the paper — specifically, sums, products and double covers of games. In Section 3.6, we review the more general setup of non-synchronous strategies for (synchronous) games, and phrase an important Theorem (Fact 3.63) due to the third author that allows one to move from value and entanglement bounds in the generalized setup back to ours. Finally, in Section 3.7, we recall the definition of P_k the Pauli group acting on k-qubits, and the generalized Pauli basis test (originally due to Natarajan–Vidick [NV18b], but here the version of de la Salle [dlS22b] is used); this is a robust self test (Definition 3.30) that forces any almost perfect strategy for this game to be close to the unique non-commuting irreducible representation of P_k.

3.1 Functional analytic preliminaries

Let $\langle \cdot | \cdot \rangle$ be the standard euclidean inner product on $\mathbb{C}^N$, namely

$$\forall \vec{v}, \vec{w} \in \mathbb{C}^N : \quad \langle \vec{v} | \vec{w} \rangle = \sum_{i=1}^{N} \overline{v_i} \cdot w_i ;$$

note that when $\vec{v}$ and $\vec{w}$ are thought of as column vectors, their inner product is exactly $(\vec{v})^* \cdot \vec{w}$, with $*$ being the conjugate transposition and $\cdot$ the standard product of matrices. An $N \times N$ complex matrix A is said to be *positive* (semi-definite) if for every $\vec{v} \in \mathbb{C}^n$ we have $\langle \vec{v} | A\vec{v} \rangle \geq 0$. Let $\tau = \frac{1}{N}\text{Tr}$ be the normalized trace on $N \times N$ complex matrices. Every such matrix A has a polar decomposition UP where U is unitary and P is positive; the matrix P is unique and often denoted by $|A|$ or $(A^*A)^{1/2}$. By functional calculus (cf. [Bla06, Section I.4.1]), the pth power of $|A|$, which we denote by $|A|^p$, is defined for every $p \geq 0$.

Definition 3.1 (Normalized p-norms). For every $p \geq 1$ we define the *normalized p-norm* $\|A\|_p$ of a $N \times N$ complex matrix A by $(\tau(|A|^p))^{1/p}$. Specifically for the case of $p = 2$, this norm is called the *normalized Hilbert–Schmidt norm*; we denote it by $\|A\|_{hs}$, and we note that it is induced by the inner product $\langle A, B \rangle = \tau(A^*B)$. In addition, the case $p = \infty$ is the operator norm, namely $\|A\|_\infty = \|A\|_{op} = \max\{\|A\vec{v}\| \mid \vec{v} \in \mathbb{C}^n, \|\vec{v}\| = 1\}$, where $\|\cdot\|$ is the euclidean norm on $\mathbb{C}^n$ induced by the inner product $\langle \cdot | \cdot \rangle$; the operator norm is well defined for non-square matrices as well.

Fact 3.2 (Useful equations and inequalities. Cf. Proposition 2.1 in [JNV+22a] and Lemma 6.1 in [GH17]). *Let $A, B \in M_{N \times N}(\mathbb{C})$, and $p, q \in [1, \infty]$. Then:*

(1) Unitary invariance*: If A is unitary, then $\|AB\|_p = \|B\|_p$.*

(2) Cauchy–Schwarz*: $|\tau(A^*B)| \leq \|A\|_{hs}\|B\|_{hs}$.*

(3) 1-norm upper bound*: $|\tau(A)| \leq \tau(|A|) = \|A\|_1$.*

(4) Hölder's inequality*: If $1/p + 1/q = 1$, then $\tau(|AB|) = \|AB\|_1 \leq \|A\|_p\|B\|_q$.*

(5) Triangle inequality ($\triangle$)*: $\|A + B\|_p \leq \|A\|_p + \|B\|_p$.*

(6) Monotonicity: If $p \leq q$, then $\|A\|_p \leq \|A\|_q$.[38]

(7) Sub-multiplicativity with operator norm: $\|AB\|_p \leq \|A\|_p \|B\|_{op}$.[39]

Definition 3.3 (Projections, isometries and partial isometries)**.** An (orthogonal) *projection* is an operator (square complex matrix in the finite dimensional case) A satisfying $A^2 = A = A^*$. An *isometry* is a linear map A between Hilbert spaces that satisfies $A^*A = \mathrm{Id}$. A *partial isometry* $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ is a linear map such that both $\omega^*\omega$ and $\omega\omega^*$ are projections. Any partial isometry can be written as $\omega = \iota \circ \kappa^*$, where $\iota \colon \mathbb{C}^K \to \mathbb{C}^M$ and $\kappa \colon \mathbb{C}^K \to \mathbb{C}^N$ are isometries. Given an $M \times M$ matrix A, the $N \times N$ matrix $\omega^* A \omega$ is often called *a corner* of A (with repsect to ω) — this naming choice is clearer in the case when $M \geq N$ and ω is an isometry embedding $\mathbb{C}^N$ in $\mathbb{C}^M$.

We will sometimes need to compare observables, PVMs, or strategies, that act in different spaces. For example, we may have families of operators $\{A_i\}$, $\{B_i\}$ on $\mathbb{C}^M$ and $\mathbb{C}^N$ respectively. To compare them, we may measure their distance as the infimum, over all partial isometries $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$, of $\sum_i p_i \|A_i - wB_iw^*\|_{hs}^2$, where p_i are some coefficients (e.g. probabilities). The following definition and claims will be useful technical tools in the manipulation of such distance measures.

Definition 3.4 (ε-near bijection)**.** A partial isometry $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ is said to be an *ε-near bijection* if $1 - \tau(\omega^*\omega), 1 - \tau(\omega\omega^*) \leq \varepsilon$.[40]

Claim 3.5. *Let $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ be an ε-near bijection (Definition 3.4). Then, for every contraction $A \in M_{M \times M}(\mathbb{C})$, i.e. $\|A\|_{op} \leq 1$, we have*

$$\left| \|A\|_{hs}^2 - \|\omega^* A \omega\|_{hs}^2 \right| \leq 4\varepsilon.$$

Proof. If $\varepsilon > 1/2$, then the claim is immediate, using $\|A\|_{hs} \leq \|A\|_{op} \leq 1$ and the triangle inequality. Assume otherwise. Let $\omega = \iota \circ \kappa^*$ be the decomposition of ω as an isometry $\iota \colon \mathbb{C}^K \to \mathbb{C}^M$ and co-isometry $\kappa^* \colon \mathbb{C}^N \to \mathbb{C}^K$. First, it is straightforward to check that $\tau(\omega^*\omega) = K/N$ and $\tau(\omega\omega^*) = K/M$. Now,

$$\begin{aligned}
\|\omega^* A \omega\|_{hs}^2 &= \frac{1}{N} \mathrm{Tr}(\omega^* A^* \omega \omega^* A \omega) \\
&=_{\kappa^*\kappa = \mathrm{Id}_K} \frac{1}{N} \mathrm{Tr}(\iota^* A^* \iota \cdot \iota^* A \iota) \\
&= \frac{1}{N} \mathrm{Tr}(A^* \iota \cdot \iota^* A \iota \cdot \iota^*).
\end{aligned}$$

We have

$$\begin{aligned}
\mathrm{Tr}(A^* \iota \cdot \iota^* A) &= \mathrm{Tr}(A^* \iota \cdot \iota^* A \iota \cdot \iota^*) + \mathrm{Tr}(A^* \iota \cdot \iota^* A (\mathrm{Id}_M - \iota \cdot \iota^*)) \\
&\leq_{\text{Hölder}} \mathrm{Tr}(A^* \iota \cdot \iota^* A \iota \cdot \iota^*) + \underbrace{\|A^* \iota \cdot \iota^* A\|_{op}}_{\leq 1} \cdot \underbrace{\mathrm{Tr}(\mathrm{Id}_M - \iota \cdot \iota^*)}_{M-K}.
\end{aligned} \tag{13}$$

The same argument shows that $\mathrm{Tr}(AA^*) \leq \mathrm{Tr}(AA^* \iota \cdot \iota^*) + (M - K)$. Hence,

$$\begin{aligned}
\|A\|_{hs}^2 &= \frac{1}{M} \mathrm{Tr}(AA^*) \\
&\leq \frac{1}{M} \left(\mathrm{Tr}(A^* \iota \cdot \iota^* A \iota \cdot \iota^*) + 2(M - K) \right) \\
&= \frac{N}{M} \|\omega^* A \omega\|_{hs}^2 + 2\varepsilon.
\end{aligned}$$

---

[38] This direction of monotonicity is due the normalized trace τ. Without normalization, the monotonicity property is reversed.

[39] The case $p = 1$ is covered by Hölder.

[40] Note that one of the τ's is the normalized trace on $N \times N$ matrices and the other on $M \times M$ matrices.

Finally, $\frac{N}{M} \leq \frac{N}{K} \leq \frac{1}{1-\varepsilon} \leq 1 + 2\varepsilon$, and since $\|\omega^* A\omega\|_{hs} \leq \|\omega^* A\omega\|_{op} \leq 1$, we deduce that

$$\|A\|_{hs}^2 - \|\omega^* A\omega\|_{hs}^2 \leq 4\varepsilon.$$

On the other hand, as $\iota \cdot \iota^*$ and $\mathrm{Id}_M - \iota \cdot \iota^*$ are both positive, we can deduce that

$$\begin{aligned}
\mathrm{Tr}(\iota^* A^* \iota \cdot \iota^* A\iota) &\leq \mathrm{Tr}(\iota^* A^* \iota \cdot \iota^* A\iota) + \mathrm{Tr}(\iota^* A^* (\mathrm{Id}_M - \iota \cdot \iota^*) A\iota) \\
&= \mathrm{Tr}(\iota^* A^* A\iota) \\
&\leq \mathrm{Tr}(A\iota \cdot \iota^* A^*) + \mathrm{Tr}(A(\mathrm{Id}_M - \iota \cdot \iota^*) A^*) \\
&= \mathrm{Tr}(AA^*).
\end{aligned}$$

Therefore,

$$\|\omega^* A\omega\|_{hs}^2 = \frac{1}{N} \mathrm{Tr}(\iota^* A^* \iota \cdot \iota^* A\iota) \leq \frac{1}{N} \mathrm{Tr}(AA^*) = \frac{M}{N} \|A\|_{hs}^2,$$

and as $\frac{M}{N} \leq \frac{M}{K} \leq \frac{1}{1-\varepsilon} \leq 1 + 2\varepsilon$ and $\|A\|_{hs}^2 \leq \|A\|_{op}^2 \leq 1$, we deduce that

$$\|\omega^* A\omega\|_{hs}^2 - \|A\|_{hs}^2 \leq 2\varepsilon.$$

Combining the two finishes the proof. □

3.2 Notions of distance between measurements, correlations and strategies

As mentioned in Remark 2.55, the proof method of the soundness conditions in Compression (Theorem 2.53) is as follows. Let $\mathfrak{G}$ be a tailored game, and $\mathfrak{T}(\mathfrak{G})$ be some transformation of $\mathfrak{G}$ into a new game. Assume you are given a strategy $\mathscr{S}$ for $\mathfrak{T}(\mathfrak{G})$ with $\mathrm{val}(\mathfrak{T}(\mathfrak{G}), \mathscr{S}) \geq 1 - \varepsilon$. Then, the goal is to extract from $\mathscr{S}$ a strategy $\mathscr{S}'$ for the original game $\mathfrak{G}$ with value at least $1 - f(\varepsilon)$ (controlling this f is a recurring technical hurdle). In the first two transformations applied by Compress, *question reduction* and *answer reduction*, the way $\mathscr{S}'$ is extracted out of $\mathscr{S}$ is by perturbing it until it passes some of the subroutines of $\mathfrak{T}(\mathfrak{G})$ perfectly. After this perturbation, the value of the resulting strategy is not much worse than the value of the original strategy. Using moreover that the new strategy, by definition, passes some subroutines perfectly, then makes it easier for us to extract $\mathscr{S}'$ for the original $\mathfrak{G}$.

3.2.1 Distance between (partial) measurements

To make this notion of "perturbation" formal, we need appropriate notions of distance between POVMs and between quantum strategies, which is the topic of this section. Let us begin by extending the notion of a measurement.

Definition 3.6 (Partial and Corner POVMs). An N-dimensional partial POVM with outcomes in a finite set A is a tuple of positive $N \times N$ matrices $\{\mathcal{P}_a\}_{a \in A}$ such that $\sum \mathcal{P}_a \leq \mathrm{Id}$. It is a partial PVM if every $\mathcal{P}_a$ is an orthogonal projection. A partial POVM defines a tuple of non-negative real numbers $p_a = \tau(\mathcal{P}_a)$ satisfying $\sum p_a \leq 1$, which we keep calling the *distribution* induced by $\mathcal{P}$. The quantity $1 - \sum p_a = 1 - \sum \tau(\mathcal{P}_a)$ is often called the *deficiency* of $\mathcal{P}$.

   Given an M-dimensional partial POVM $\mathcal{P}$ and a partial isometry (Defintion 3.3) $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$, the tuple of $N \times N$ matrices $\mathcal{P}'_a = \omega^* \mathcal{P}_a \omega$ parametrized by A is called *the corner* POVM of $\mathcal{P}$ with respect to ω. We often denote the corner POVM by $\omega^* \mathcal{P} \omega$.

Remark 3.7. The above definition of a partial POVM (called a submeasurement in [JNV$^+$22a]) clearly extends the notion of a POVM (Definition 2.1), and the ideas of measuring and jointly measuring extend with it (Definition 2.2) — though, we may get partial distributions when measuring instead of full ones. When needed, we call a POVM (or PVM), as in Definition 2.1, a *full* or *complete* POVM.

Claim 3.8. *Given an M-dimensional partial POVM $\mathcal{P}$ and a partial isometry $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$, the corner POVM $\omega^* \mathcal{P} \omega$ is indeed a partial POVM. In addition, if the deficiency of $\mathcal{P}$ is δ, and ω is an ε-near bijection (Defintion 3.4), then the deficiency of the corner $\omega^* \mathcal{P} \omega$ is at most $\delta + 2\varepsilon$.*

Proof. The partial order on matrices $A \leq B$ (i.e., $A - B$ being positive) is preserved by corners, namely: If A, B are $M \times M$ complex matrices and $A \leq B$, then $\omega^* A \omega \leq \omega^* B \omega$. This observation implies immediately that the corner POVM consists of positive matrices, and that $\sum_a \omega^* \mathcal{P}_a \omega \leq \omega^* \omega$; as $\omega^* \omega$ is a projection (Definition 3.3), it satisfies $\omega^* \omega \leq \text{Id}_N$, and the proof is complete.

Now, as $\sum \mathcal{P}_a \leq \text{Id}$, it is a contraction, and the argument in (13) shows that for every $a \in A$,

$$\sum_{a \in A} \text{Tr}(\mathcal{P}_a) \leq \left(\sum_{a \in A} \text{Tr}(\omega^* \mathcal{P}_a \omega) \right) + \text{Tr}(\text{Id} - \omega \omega^*) .$$

So, rearranging the above inequality and using the deficiency and near bijection assumptions leads to

$$\sum_{a \in A} \tau(\omega^* \mathcal{P}_a \omega) = \frac{1}{N} \sum_{a \in A} \text{Tr}(\omega^* \mathcal{P}_a \omega) \geq \frac{M}{N}(1 - \delta - \varepsilon) \geq (1 - \varepsilon)(1 - \delta - \varepsilon) \geq 1 - \delta - 2\varepsilon .$$

$\square$

Fact 3.9 (Naimark's dilation theorem, see e.g. Chapter 4 in [Pau03])**.** *Every (finite dimensional) POVM is a corner of a (finite dimensional) PVM.*

As quantum strategies (Definition 2.18), which are the objects of interest for us, are defined using full PVMs, it seems unnecessary to define POVMs, not to mention partial ones. The reason for these intricacies is that we want to be able to compare strategies acting on Hilbert spaces of different dimensions. This will require us to use partial isometries between these spaces, and the conjugation of a PVM by a partial isometry — namely the corner — is only guaranteed to be a partial POVM by Claim 3.8. Similarly, in representation form, the conjugation by a partial isometry of a unitary is no longer a unitary. But, as long as the partial isometry is not too deforming, namely it is an ε-near bijection (Definition 3.4), these properties are "almost" preserved — see Fact 3.21 and the above claim.

Definition 3.10 (Distance and Inconsistency of POVMs)**.** Let $\mathcal{P}$ and $\mathcal{Q}$ be partial POVMs (Definition 3.6) of the same dimension with outcomes in the same finite set A. We say that $\mathcal{P}$ and $\mathcal{Q}$ are *ε-close*, and denote it by $\mathcal{P}_a \approx_\varepsilon \mathcal{Q}_a$, if

$$\sum_{a \in A} \|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}^2 \leq \varepsilon .$$

We say that $\mathcal{P}$ and $\mathcal{Q}$ are *ε-inconsistent*, and denote it by $\mathcal{P} \simeq_\varepsilon \mathcal{Q}$, if

$$\sum_{a \neq b \in A} \tau(\mathcal{P}_a \mathcal{Q}_b) \leq \varepsilon .$$

Remark 3.11. The name inconsistency is appropriate, as by Definition 2.2, if $\mathcal{P}$ and $\mathcal{Q}$ are full POVMs, and we jointly measure $(a, b) \sim (\mathcal{P}, \mathcal{Q})$, then the probability $a \neq b$ is exactly the incosistency of $\mathcal{P}$ and $\mathcal{Q}$. In particular, note that, as opposed to distance, the inconsistency of a POVM with itself is not necessarily 0 — this is true only when the product of $\mathcal{P}_a$ and $\mathcal{P}_b$ is 0 for every $a \neq b$.

Our (tailored) games contain various comparisons between the answers at the endpoints of the sampled edge, and a strategy passing the game along this edge with high probability implies a small inconsistency between the (data processed, Definition 3.32) PVMs at the endpoints of the edge.

Proposition 3.12 (Properties of distance and inconsistency. Cf. [JNV+22a, NW19] and [CVY23])**.** *Let $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ be partial POVMs of dimension N with outcomes in a finite set A, let $p, q, r \in \mathbb{R}^A$ be the distributions associated with them, and let $\| \cdot \|_1$ be the standard L^1 norm on $\mathbb{R}^A$.*

1. *Inconsistency and distance are the same for projective measurements: If $\mathcal{P}$ and $\mathcal{Q}$ are full PVMs then $\mathcal{P}_a \simeq_\varepsilon \mathcal{Q}_a$ if and only if $\mathcal{P}_a \approx_{2\varepsilon} \mathcal{Q}_a$.*

2. **Semi-triangle inequality**: *If $\mathcal{P}_a \approx_\varepsilon \mathcal{Q}_a$ and $\mathcal{Q}_a \approx_\delta \mathcal{R}_a$, then $\mathcal{P}_a \approx_{2\varepsilon + 2\delta} \mathcal{R}_a$. More generally, given $k+1$ many partial POVMs $\mathcal{P}^1, ..., \mathcal{P}^{k+1}$ such that $\mathcal{P}^i \approx_{\varepsilon_i} \mathcal{P}^{i+1}$ for every $1 \le i \le k$, we have*

$$\mathcal{P}^1 \approx_{k(\varepsilon_1 + ... + \varepsilon_k)} \mathcal{P}^{k+1} .$$

3. **Consistent almost full POVMs induce close distributions**: *Assume the deficiency of $\mathcal{P}$ is δ_1 and of $\mathcal{Q}$ is δ_2 — i.e., $\|p\|_1 = \sum \tau(\mathcal{P}_a) = 1 - \delta_1$, $\|q\|_1 = \sum \tau(\mathcal{Q}_a) = 1 - \delta_2$ — and assume they are ε-inconsistent — i.e., $\mathcal{P} \simeq_\varepsilon \mathcal{Q}$. Then $\|p - q\|_1 \le 2(\delta_1 + \delta_2 + \varepsilon)$.*

4. **Small inconsistency to closeness in case both are full** *: Assume $\mathcal{P}, \mathcal{Q}$ are full POVMs. Then, $\mathcal{P} \simeq_\varepsilon \mathcal{Q}$ implies $\mathcal{P} \approx_{2\varepsilon} \mathcal{Q}$.*

5. **Closeness to small inconsistency in case one of them is projective**: *Assume $\mathcal{P}$ is projective. Then, $\mathcal{P} \approx_\varepsilon \mathcal{Q}$ implies $\mathcal{P} \simeq_{\sqrt{\varepsilon}} \mathcal{Q}$.*

Proof.   1. It follows from

$$\sum_{a \in A} \overbrace{\|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}^2}^{\tau((\mathcal{P}_a - \mathcal{Q}_a)^*(\mathcal{P}_a - \mathcal{Q}_a))} = \sum_{a \in A} \tau(\mathcal{P}_a) + \tau(\mathcal{Q}_a) - 2\tau(\mathcal{P}_a \mathcal{Q}_a)$$

$$=_{\mathcal{P}, \mathcal{Q} \text{ full PVMs}} 2\left(1 - \sum_{a \in A} \tau(\mathcal{P}_a \mathcal{Q}_a)\right) \tag{14}$$

$$= 2 \sum_{a \ne b \in A} \tau(\mathcal{P}_a \mathcal{Q}_b) ,$$

where the last equation is since $\sum_{a,b \in A} \tau(\mathcal{P}_a \mathcal{Q}_b) = 1$.

2. The first case is immediate from

$$\forall a \in A : \quad \|\mathcal{P}_a - \mathcal{R}_a\|_{hs}^2 \le_\triangle (\|\mathcal{P}_a - \mathcal{Q}_a\|_{hs} + \|\mathcal{Q}_a - \mathcal{R}_a\|_{hs})^2 \le 2\|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}^2 + 2\|\mathcal{Q}_a - \mathcal{R}_a\|_{hs}^2 .$$

The general case uses the same idea together with the inequality $(\sum_{i=1}^k x_i)^2 \le k \sum_{i=1}^k x_i^2$.

3. Choose a new element $\perp \notin A$, and extend the partial POVMs $\mathcal{P}, \mathcal{Q}$ to full POVMs $\mathcal{P}', \mathcal{Q}'$ on $A' = A \cup \{\perp\}$ by letting

$$\forall a \in A : \mathcal{P}'_a := \mathcal{P}_a ,$$
$$\mathcal{P}'_\perp := \mathrm{Id} - \sum_{a \in A} \mathcal{P}_a , \tag{15}$$

and similarly for $\mathcal{Q}'$. Furthermore, let p' and q' be the distributions induced by $\mathcal{P}', \mathcal{Q}'$. It is immediate that $\|p - q\|_1 \le \|p' - q'\|_1$. In addition,

$$\sum_{a' \ne b' \in A'} \tau(\mathcal{P}'_{a'} \mathcal{Q}'_{b'}) = \sum_{a \ne b \in A} \tau(\mathcal{P}_a \mathcal{Q}_b) + \overbrace{\sum_{a \in A} \tau(\mathcal{P}_a \mathcal{Q}_\perp)}^{\tau((\sum \mathcal{P}_a)\mathcal{Q}_\perp)} + \overbrace{\sum_{b \in A} \tau(\mathcal{P}_\perp \mathcal{Q}_b)}^{\tau(\mathcal{P}_\perp (\sum \mathcal{Q}_b))}$$

$$\le_{\mathcal{P} \simeq_\varepsilon \mathcal{Q} \text{ and Hölder}} \varepsilon + \underbrace{\|\sum \mathcal{P}_a\|_{op}}_{\le 1} \cdot \underbrace{\tau(\mathcal{Q}_\perp)}_{=\delta_1} + \underbrace{\|\sum \mathcal{Q}_a\|_{op}}_{\le 1} \cdot \underbrace{\tau(\mathcal{P}_\perp)}_{=\delta_2} \tag{16}$$

$$\le \varepsilon + \delta_1 + \delta_2 ,$$

which means $\mathcal{P}' \simeq_{\varepsilon + \delta_1 + \delta_2} \mathcal{Q}'$. Now, $\mathcal{P}'$ and $\mathcal{Q}'$ are full POVMs, and hence for every $a' \in A'$ we have

$$|\tau(\mathcal{P}'_{a'}) - \tau(\mathcal{Q}'_{a'})| = \left| \sum_{b' \in A'} \tau(\mathcal{P}'_{a'} \mathcal{Q}'_{b'}) - \tau(\mathcal{Q}'_{a'} \mathcal{P}'_{b'}) \right| \le \sum_{b' \in A' : \, b' \ne a'} |\tau(\mathcal{P}'_{a'} \mathcal{Q}'_{b'})| + |\tau(\mathcal{Q}'_{a'} \mathcal{P}'_{b'})| . \tag{17}$$

Summing up over all $a' \in A$ gives us

$$\|p' - q'\|_1 = \sum_{a' \in A'} |\tau(\mathcal{P}'_{a'}) - \tau(\mathcal{Q}'_{a'})| \leq 2 \sum_{a' \neq b' \in A'} |\tau(\mathcal{P}'_{a'} \mathcal{Q}'_{b'})| \leq 2(\varepsilon + \delta_1 + \delta_2),$$

as needed.

4. This is the same calculation as in (14), except that we need to use the inequality $\tau(\mathcal{P}_a^2) \leq \tau(\mathcal{P}_a)$ and $\tau(\mathcal{Q}_a^2) \leq \tau(\mathcal{Q}_a)$.

5. We have

$$\sum_{a \in A} \sum_{b \in A: b \neq a} \tau(\mathcal{P}_a \mathcal{Q}_b) \leq \sum_{a \in A} \tau(\mathcal{P}_a(\mathrm{Id} - \mathcal{Q}_a))$$

$$=_{\mathcal{P} \text{ is projective}} \sum_{a \in A} \overbrace{\tau(\mathcal{P}_a(\mathcal{P}_a - \mathcal{Q}_a))}^{=|\langle \mathcal{P}_a, \mathcal{P}_a - \mathcal{Q}_a \rangle|}$$

$$\leq_{\text{Cauchy–Schwarz}} \sum_{a \in A} \|\mathcal{P}_a\|_{hs} \|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}$$

$$\leq_{\text{Cauchy–Schwarz}} \sqrt{\sum_{a \in A} \|\mathcal{P}_a\|_{hs}^2} \sqrt{\sum_{a \in A} \|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}^2}$$

$$\leq \sqrt{\varepsilon},$$

where the last inequality uses the fact $\mathcal{P} \approx_\varepsilon \mathcal{Q}$ and the fact that projections satisfy $\|\mathcal{P}_a\|_{hs}^2 = \tau(\mathcal{P}_a)$. $\quad\square$

Remark 3.13. In the full case, ε-inconsistency implies that the POVMs are ε-close (the above clause 4., see also [JNV$^+$22a, Proposition 2.5]), but the reverse is not true in general (see [NW19, Remark 4.15]). Luckily, we have the above clause 5., which states that in case one of them is a PVM, there is a way to infer $\sqrt{\varepsilon}$-inconsistency out of ε-closeness (see also [JNV$^+$22a, Proposition 2.6]). This will be very helpful in the upcoming analysis, as small inconsistency allows to deduce results that closeness cannot (cf. [JNV$^+$22a, Propositions 2.4 and 2.9]).

The value of a measurement being projective leads to the following definition:

Definition 3.14 (Almost projective measurements). A partial POVM $\mathcal{P}$ with outcomes in A is said to be ε-almost projective if $\sum_{a \in A} \|\mathcal{P}_a - \mathcal{P}_a^2\|_1 \leq \varepsilon$.

Remark 3.15. For full POVMs, being ε-almost projective is the same as having ε-self inconsistency, namely satisfying $\mathcal{P} \simeq_\varepsilon \mathcal{P}$. This is because

$$\|\mathcal{P}_a - \mathcal{P}_a^2\|_1 = \tau(\mathcal{P}_a(\mathrm{Id} - \mathcal{P}_a)) = \sum_{b \in A: b \neq a} \tau(\mathcal{P}_a \mathcal{P}_b).$$

Claim 3.16 (Corners of PVMs are almost projective). *Let $\mathcal{P}$ be an M-dimensional PVM with outcomes in A and $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ an ε-near bijection. Then the corner POVM $\omega^* \mathcal{P} \omega$ is ε-almost projective.*

Proof. Let us calculate

$$\sum_{a \in A} \|\omega^* \mathcal{P}_a \omega - (\omega^* \mathcal{P}_a \omega)^2\|_1 = \sum_{a \in A} \tau(\omega^* \mathcal{P}_a^2 \omega - \omega^* \mathcal{P}_a \omega \omega^* \mathcal{P}_a \omega)$$

$$= \sum_{a \in A} \tau(\omega^* \mathcal{P}_a(\mathrm{Id}_M - \omega \omega^*)\mathcal{P}_a \omega)$$

$$= \tau\Big((\mathrm{Id}_M - \omega \omega^*)\Big(\sum_{a \in A} \mathcal{P}_a \omega \omega^* \mathcal{P}_a\Big)\Big)$$

$$\leq_{\text{Hölder}} \tau(\mathrm{Id}_M - \omega \omega^*) \Big\| \sum_{a \in A} \mathcal{P}_a \omega \omega^* \mathcal{P}_a \Big\|_{op}.$$

39

By the ε-near bijection assumption, $\tau(\mathrm{Id}_M - \omega\omega^*) \leq \varepsilon$, and as $\mathcal{P}_a\omega\omega^*\mathcal{P}_a \leq \mathcal{P}_a\omega\omega^*\mathcal{P}_a + \mathcal{P}_a(\mathrm{Id}_M - \omega\omega^*)\mathcal{P}_a = \mathcal{P}_a^2 = \mathcal{P}_a$ we have $\sum \mathcal{P}_a\omega\omega^*\mathcal{P}_a \leq \sum \mathcal{P}_a \leq \mathrm{Id}_M$; as the operator norm respects the order on positive matrices, we are done. $\qquad\square$

Claim 3.17 (Corners of POVMs produce similar joint distributions). *Let $\mathcal{P}$ and $\mathcal{Q}$ be M-dimensional partial POVMs with outcomes in finite sets A and B respectively, and let $\omega\colon \mathbb{C}^N \to \mathbb{C}^M$ be an ε-near bijection. Then, jointly measuring (Definition 2.2) according to $(\mathcal{P}, \mathcal{Q})$ is $4\sqrt{\varepsilon}$-close in L^1-distance to jointly measuring according to the corners $(\omega^*\mathcal{P}\omega, \omega^*\mathcal{Q}\omega)$.*

Proof. If $\varepsilon \geq 1/2$, then the conclusion is immediate (as every two partial probability distributions are at most 2 apart in the L^1-norm). Hence, we can assume $\varepsilon < 1/2$, and in particular, as ω is an ε-near bijection, we have

$$1 - \varepsilon \leq M/N, N/M \leq 1/1-\varepsilon \leq 1 + 2\varepsilon.$$

By applying Hölder (Item (4) of Fact 3.2) twice, and using the fact that a projection is a contraction, we get

$$\forall a \in A, b \in B: \quad \mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) \leq |\mathrm{Tr}(\mathcal{P}_a\omega\omega^*\mathcal{Q}_b)| \|\omega\omega^*\|_{op} \leq \mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b) \|\omega\omega^*\|_{op}^2 \leq \mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b). \tag{18}$$

On the other hand, we have

$$\begin{aligned}
\mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b) &= \mathrm{Tr}(\mathcal{P}_a\omega\omega^*\mathcal{Q}_b) + \mathrm{Tr}(\mathcal{P}_a(\mathrm{Id}_M - \omega\omega^*)\mathcal{Q}_b) \\
&= \mathrm{Tr}(\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega\omega^*) + \mathrm{Tr}(\mathcal{P}_a\omega\omega^*\mathcal{Q}_b(\mathrm{Id} - \omega\omega^*)) + \mathrm{Tr}(\mathcal{P}_a(\mathrm{Id}_M - \omega\omega^*)\mathcal{Q}_b).
\end{aligned} \tag{19}$$

Using the ε-near bijectiveness of ω and the fact that $\mathrm{Id}_M - \omega\omega^*$ is a projection, one gets

$$\mathrm{Tr}((\mathrm{Id}_M - \omega\omega^*)^*(\mathrm{Id}_M - \omega\omega^*)) = \mathrm{Tr}(\mathrm{Id}_M - \omega\omega^*) \leq M\varepsilon.$$

Therefore,

$$\begin{aligned}
\sum_{a,b} |\mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b) - \mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)| &=_{(18)} \sum_{a,b} \mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b) - \mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) \\
&=_{(19)} \sum_{a,b} \mathrm{Tr}(\mathcal{P}_a\omega\omega^*\mathcal{Q}_b(\mathrm{Id} - \omega\omega^*)) + \mathrm{Tr}(\mathcal{Q}_b\mathcal{P}_a(\mathrm{Id}_M - \omega\omega^*)) \\
&= \mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)(\mathrm{Id} - \omega\omega^*)\Big) + \mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)(\mathrm{Id} - \omega\omega^*)\Big).
\end{aligned} \tag{20}$$

By applying Cauchy–Schwartz on the two summands we get

$$\mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)(\mathrm{Id} - \omega\omega^*)\Big) \leq \sqrt{\mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)^*\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)\Big)} \sqrt{\mathrm{Tr}(\mathrm{Id} - \omega^*\omega)^2}$$

and

$$\mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)(\mathrm{Id} - \omega\omega^*)\Big) \leq \sqrt{\mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)^*\Big(\sum_{a,b} \mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)\Big)} \sqrt{\mathrm{Tr}(\mathrm{Id} - \omega^*\omega)^2}.$$

Now, $\sum \mathcal{P}_a, \sum \mathcal{Q}_b \leq \mathrm{Id}_M$ (as they are partial POVMs) which implies $(\sum \mathcal{P}_a)^2, (\sum \mathcal{Q}_b)^2 \leq \mathrm{Id}_M$; therefore

$$\begin{aligned}
\mathrm{Tr}\Big(\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)^*\Big(\sum_{a,b} \mathcal{Q}_b\mathcal{P}_a\Big)\Big) &= \mathrm{Tr}\Big(\Big(\sum_a \mathcal{P}_a\Big)\Big(\sum_b \mathcal{Q}_b\Big)^2\Big(\sum_a \mathcal{P}_a\Big)\Big) \\
&\leq \mathrm{Tr}\Big(\Big(\sum_a \mathcal{P}_a\Big)^2\Big) \\
&\leq M.
\end{aligned}$$

Similarly,

$$\mathrm{Tr}\Big(\Big(\sum_{a,b}\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)^*\Big(\sum_{a,b}\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\Big)\Big) = \mathrm{Tr}\Big(\Big(\sum_b\mathcal{Q}_b\Big)\omega\omega^*\Big(\sum_a\mathcal{P}_a\Big)^2\omega\omega^*\Big(\sum_{b\in B}\mathcal{Q}_b\Big)\Big)$$
$$\leq \mathrm{Tr}\Big(\Big(\sum_b\mathcal{Q}_b\Big)\omega\omega^*\Big(\sum_{b\in B}\mathcal{Q}_b\Big)\Big)$$
$$\leq \mathrm{Tr}\Big(\Big(\sum_b\mathcal{Q}_b\Big)^2\Big)$$
$$\leq M\,.$$

Plugging all of these upper bounds to (20), we get

$$\sum_{a,b}|\mathrm{Tr}(\mathcal{P}_a\mathcal{Q}_b) - \mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)| \leq 2M\sqrt{\varepsilon}\,.$$

If we divide both sides by M, we are almost done; the problem is that $\tau(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) = \frac{1}{N}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)$ and not $\frac{1}{M}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)$. But, for every $a \in A$ and $b \in B$ we have

$$\left|\frac{1}{N}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) - \frac{1}{M}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)\right| = \left|\frac{1}{N} - \frac{1}{M}\right|\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)\,,$$

and as $\sum_{a,b}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) \leq N$, we deduce

$$\sum_{a,b}\left|\frac{1}{N}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega) - \frac{1}{M}\mathrm{Tr}(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)\right| \leq \left|1 - \frac{N}{M}\right| \leq 2\varepsilon\,.$$

Combining all of the above gives

$$\sum_{a,b}|\tau(\mathcal{P}_a\mathcal{Q}_b) - \tau(\omega^*\mathcal{P}_a\omega\omega^*\mathcal{Q}_b\omega)| \leq 2\varepsilon + 2\sqrt{\varepsilon} \leq 4\sqrt{\varepsilon}\,.$$

$\square$

Claim 3.18 (Close almost projective POVMs produce similar joint distributions). *Let $\mathcal{P}$ and $\mathcal{Q}$ be two N-dimensional partial POVMs with outcomes in A such that $\mathcal{P} \approx_\varepsilon \mathcal{Q}$, and let $\mathcal{R}$ an N-dimensional partial POVM with outcomes in B. Assume in addition that $\mathcal{P}$ is δ_1-almost projective (Definition 3.14) and that $\mathcal{Q}$ is δ_2-almost projective. Then, jointly measuring (Definition 2.2) according to $(\mathcal{P}, \mathcal{R})$ is $(\delta_1 + \delta_2 + 2\sqrt{\varepsilon})$-close in L^1-distance to jointly measuring according to $(\mathcal{Q}, \mathcal{R})$.*

Proof. For every $0 \neq z \in \mathbb{C}$ there is a unique complex number α (with absolute value 1) such that $|z| = \alpha z$. Hence, for every $a \in A, b \in B$, there is an $\alpha_{a,b}$ such that

$$|\tau(\mathcal{P}_a\mathcal{R}_b) - \tau(\mathcal{Q}_a\mathcal{R}_b)| = \alpha_{a,b}\tau((\mathcal{P}_a - \mathcal{Q}_a)\mathcal{R}_b)\,.$$

Summing up the above over $b \in B$ gives

$$\sum_{b\in B}|\tau(\mathcal{P}_a\mathcal{R}_b) - \tau(\mathcal{Q}_a\mathcal{R}_b)| = \tau\Big((\mathcal{P}_a - \mathcal{Q}_a)\sum_{b\in B}\alpha_{a,b}\mathcal{R}_b\Big)$$
$$\leq_{\text{Hölder}} \|\mathcal{P}_a - \mathcal{Q}_a\|_1\Big\|\sum_{b\in B}\alpha_{a,b}\mathcal{R}_b\Big\|_{op}\,.$$

If $\mathcal{R}$ consists of projections (i.e., it is a partial PVM), then under an appropriate choice of basis $\sum \alpha_{a,b} \mathcal{R}_b$ is a diagonal matrix with $\alpha_{a,b}$ on the diagonal, which immediately shows that $\|\sum_{b \in B} \alpha_{a,b} \mathcal{R}_b\|_{op} \leq 1$. The general case follows from Naimark's dilation theorem (Fact 3.9). Hence,

$$\sum_{a \in A, b \in B} |\tau(\mathcal{P}_a \mathcal{R}_b) - \tau(\mathcal{Q}_a \mathcal{R}_b)| \leq \sum_{a \in A} \|\mathcal{P}_a - \mathcal{Q}_a\|_1 .$$

We now repeat the argument of [CVY23, Lemma 5.4] to bound the latter. For every $a \in A$, by the triangle inequality,

$$\|\mathcal{P}_a - \mathcal{Q}_a\|_1 \leq \|\mathcal{P}_a - \mathcal{P}_a^2\|_1 + \|\mathcal{P}_a^2 - \mathcal{P}_a \mathcal{Q}_a\|_1 + \|\mathcal{P}_a \mathcal{Q}_a - \mathcal{Q}_a^2\|_1 + \|\mathcal{Q}_a^2 - \mathcal{Q}_a\|_1 . \tag{21}$$

Summing over $a \in A$, the first and latst summands of (21) are bounded by δ_1 and δ_2 respectively, as $\mathcal{P}$ is δ_1-almost projective and $\mathcal{Q}$ is δ_2-almost projective. For the second summand in (21),

$$\sum_{a \in A} \|\mathcal{P}_a(\mathcal{P}_a - \mathcal{Q}_a)\|_1 \leq_{\text{Hölder}} \sum_{a \in A} \|\mathcal{P}_a\|_{hs} \|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}$$

$$\leq_{\text{Cauchy–Scwhartz}} \sqrt{\sum_{a \in A} \|\mathcal{P}_a\|_{hs}^2} \sqrt{\sum_{a \in A} \|\mathcal{P}_a - \mathcal{Q}_a\|_{hs}^2}$$

As $\mathcal{P}$ and $\mathcal{Q}$ are ε-close, the second factor is bounded by $\sqrt{\varepsilon}$. The first factor is bounded by 1 as $\|\mathcal{P}_a\|_{hs}^2 = \tau(\mathcal{P}_a^2) \leq \tau(\mathcal{P}_a)$ and $\sum \mathcal{P}_a \leq \mathrm{Id}_M$. The third summand in (21) is bounded in the exact same way, which leads to

$$\sum_{a \in A} \|\mathcal{P}_a - \mathcal{Q}_a\|_1 \leq \delta_1 + 2\sqrt{\varepsilon} + \delta_2 ,$$

finishing the proof. $\qquad\square$

3.2.2 Distance between measurements with outcomes in $\mathbb{F}_2^S$

The properties in the previous subsection were for general measurements (or partial measurements). As in our games the set A is always of the form $\mathbb{F}_2^S$ for some finite set S, and as in this case PVMs are closely related to representations (Definition 2.5), there are some facts we need to demonstrate in this situation. In this case, we often view strategies and PVMs as given in observable or representation form; it is thus natural to ask what is the analogous formulation of distance. The following claim is a straightforward application of the Fourier transform (Definition 2.4. See also [dlS22b, Lemma 3.4]):

Claim 3.19. *Let $\mathcal{P}$ be an N-dimensional PVM with outcomes in $\mathbb{F}_2^S$, where S is a finite set, and let $\mathcal{U}$ be its representation form. Similarly, let $\mathcal{Q}$ be an M-dimensional PVM with outcomes in the same set $\mathbb{F}_2^S$, with $\mathcal{V}$ being its observable form. Then, for every partial isometry $\omega\colon \mathbb{C}^N \to \mathbb{C}^M$ we have*

$$\mathop{\mathbb{E}}_{\alpha\colon S \to \mathbb{F}_2} \left[\|\mathcal{U}(\alpha) - \omega^* \mathcal{V}(\alpha) \omega\|_{hs}^2 \right] = \sum_{a\colon S \to \mathbb{F}_2} \|\mathcal{P}_a - \omega^* \mathcal{Q}_a \omega\|_{hs}^2 .$$

Hence, when we denote $\mathcal{U} \approx_\varepsilon \omega^* \mathcal{V} \omega$ where the two sides are in representation form, we mean that the left hand-side of the above equation is smaller or equal to ε.

Proof. Recall that by Definition 2.4,

$$\forall \alpha\colon S \to \mathbb{F}_2 : \ \mathcal{U}(\alpha) = \sum_{a\colon S \to \mathbb{F}_2} (-1)^{\langle \alpha, a \rangle} \mathcal{P}_a .$$

Thus, for every $\alpha\colon S \to \mathbb{F}_2$, we have

$$\|\mathcal{U}(\alpha) - \omega^* \mathcal{V}(\alpha) \omega\|_{hs}^2 = \left\| \sum_{a\colon S \to \mathbb{F}_2} (-1)^{\langle \alpha, a \rangle} (\mathcal{P}_a - \omega^* \mathcal{Q}_a \omega) \right\|_{hs}^2$$

$$= \sum_{a,b\colon S \to \mathbb{F}_2} (-1)^{\langle \alpha, a+b \rangle} \tau \left((\mathcal{P}_a - \omega^* \mathcal{Q}_a \omega)(\mathcal{P}_b - \omega^* \mathcal{Q}_b \omega) \right)$$

42

But, for every fixed $a \neq b \colon S \to \mathbb{F}_2$, we have $a + b \neq \vec{0}$ and thus

$$\mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \left[(-1)^{\langle \alpha, a+b \rangle} \right] = 0 \,.$$

Hence,

$$\mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \left[\| \mathcal{U}(\alpha) - \omega^* \mathcal{V}(\alpha) \omega \|_{hs}^2 \right] = \sum_{a \colon S \to \mathbb{F}_2} \tau((\mathcal{P}_a - \omega^* \mathcal{Q}_a \omega)^2)$$

$$= \sum_{a \colon S \to \mathbb{F}_2} \| \mathcal{P}_a - \omega^* \mathcal{Q}_a \omega \|_{hs}^2 \,.$$

$\square$

Remark 3.20. As remarked in [dlS22b], this is just the standard orthogonality of characters argument for the group $\mathbb{F}_2^S$.

The following is a very useful fact, that states that the corners (Definition 3.3) of a representation of $\mathbb{F}_2^S$ with respect to a nearly bijective partial isometry are close to a genuine representation in the appropriate dimension.

Fact 3.21 (Orthonormalization. See Lemma 2.9 in [CVY23] and [dlS22a]). *Let $\mathcal{P}$ be an N-dimensional PVM with outcomes in $\mathbb{F}_2^S$, and $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ be a partial isometry with $1 - \tau(\omega \omega^*), 1 - \tau(\omega^* \omega) \leq \varepsilon$. Then, there is an M-dimensional PVM $\mathcal{Q}$ such that $\omega \mathcal{P}_a \omega^* \approx_{56\varepsilon} \mathcal{Q}_a$, namely*

$$\sum_{a \colon S \to \mathbb{F}_2} \| \mathcal{Q}_a - \omega \mathcal{P}_a \omega^* \|_{hs}^2 \leq 56\varepsilon.$$

In representation form, if $\mathcal{U}$ is an N-dimensional representation of $\mathbb{F}_2^S$, then there is an M-dimensional representation $\mathcal{V}$ of $\mathbb{F}_2^S$ such that $\omega \mathcal{U} \omega^* \approx_{56\varepsilon} \mathcal{V}$, namely

$$\mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \| \mathcal{V}(\alpha) - \omega \mathcal{U}(\alpha) \omega^* \|_{hs}^2 \leq 56\varepsilon.$$

The above distance between representations is of L^1-type. It is also natural to consider the L^∞-distance between representations, as many arguments are easier in this setup. The following allows us to move back and forth between the two notions.

Claim 3.22 (L^1-closeness of representations implies L^∞-closeness). *Let $\chi \colon \mathbb{F}_2^S \to U(N)$ and $\zeta \colon \mathbb{F}_2^S \to U(M)$ be two representations of $\mathbb{F}_2^S$. Let $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ be a partial isometry such that $1 - \tau(\omega^* \omega), 1 - \tau(\omega \omega^*) \leq \varepsilon$. Then, for every $\beta \colon S \to \mathbb{F}_2$, we have*

$$\| \chi(\beta) - \omega^* \zeta(\beta) \omega \|_{hs}^2 \leq 6 \, \mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \left[\| \chi(\alpha) - \omega^* \zeta(\alpha) \omega \|_{hs}^2 \right] + 15\varepsilon.$$

Proof. As χ (and ζ) is a representation of $\mathbb{F}_2^S$, we have $\chi(\beta) = \chi(\beta + \alpha)\chi(\alpha)$ for every $\alpha \in \mathbb{F}_2^S$ (and similarly for ζ), and thus by the triangle and Jensen's inequalities

$$\| \chi(\beta) - \omega^* \zeta(\beta) \omega \|_{hs}^2 = \| \mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \chi(\beta + \alpha)\chi(\alpha) - \omega^* \zeta(\alpha + \beta)\zeta(\alpha)\omega \|_{hs}^2$$

$$\leq \mathbb{E}_{\alpha \colon S \to \mathbb{F}_2} \| \chi(\beta + \alpha)\chi(\alpha) - \omega^* \zeta(\alpha + \beta)\zeta(\alpha)\omega \|_{hs}^2.$$

By the triangle inequality, for every α,

$$\| \chi(\beta + \alpha)\chi(\alpha) - \omega^* \zeta(\alpha + \beta)\zeta(\alpha)\omega \|_{hs} \leq \| \chi(\beta + \alpha)\chi(\alpha) - \omega^* \zeta(\alpha + \beta)\omega\omega^* \zeta(\alpha)\omega \|_{hs}$$

$$+ \| \omega^* \zeta(\alpha + \beta)\zeta(\alpha)\omega - \omega^* \zeta(\alpha + \beta)\omega\omega^* \zeta(\alpha)\omega \|_{hs}$$

43

and

$$\|\chi(\beta + \alpha)\chi(\alpha) - \omega^*\zeta(\alpha + \beta)\omega\omega^*\zeta(\alpha)\omega\|_{hs} \leq \|\chi(\beta + \alpha)\chi(\alpha) - \chi(\beta + \alpha)\omega^*\zeta(\alpha)\omega\|_{hs}$$
$$+ \|\chi(\beta + \alpha)\omega^*\zeta(\alpha)\omega - \omega^*\zeta(\alpha + \beta)\omega\omega^*\zeta(\alpha)\omega\|_{hs}$$
$$\leq \|\chi(\alpha) - \omega^*\zeta(\alpha)\omega\|_{hs} + \|\chi(\beta + \alpha) - \omega^*\zeta(\alpha + \beta)\omega\|_{hs},$$

where the last inequality uses the unitary invariance of the Hilbert–Schmidt norm (Item (1)), the inequality $\|AB\|_{hs} \leq \|A\|_{op}\|B\|_{hs}$ (Item (7)), and the fact that $\|\omega^*\zeta(\alpha)\omega\|_{op} \leq 1$ (which can also be deduced by the non-square analogue of Item (7)). Therefore,

$$\|\chi(\beta) - \omega^*\zeta(\beta)\omega\|_{hs}^2 \leq 3 \underset{\alpha : \, S \to \mathbb{F}_2}{\mathbb{E}} \|\chi(\alpha) - \omega^*\zeta(\alpha)\omega\|_{hs}^2$$
$$+ 3 \underset{\alpha : \, S \to \mathbb{F}_2}{\mathbb{E}} \|\chi(\alpha + \beta) - \omega^*\zeta(\alpha + \beta)\omega\|_{hs}^2$$
$$+ 3 \underset{\alpha : \, S \to \mathbb{F}_2}{\mathbb{E}} \|\omega^*\zeta(\alpha + \beta)\zeta(\alpha)\omega - \omega^*\zeta(\alpha + \beta)\omega\omega^*\zeta(\alpha)\omega\|_{hs}^2.$$

Note that the first and second summand are equal to one another. For the third summand, using Claim 3.5, we have

$$\|\omega^*\zeta(\alpha + \beta)(\mathrm{Id} - \omega\omega^*)\zeta(\alpha)\omega\|_{hs}^2 \leq \|\zeta(\alpha + \beta)(\mathrm{Id} - \omega\omega^*)\zeta(\alpha)\|_{hs}^2 + 4\varepsilon$$
$$= \underbrace{\|\mathrm{Id} - \omega\omega^*\|_{hs}^2}_{=1 - \tau(\omega\omega^*)} + 4\varepsilon$$
$$\leq 5\varepsilon.$$

Combining all of the inequalities,

$$\|\chi(\beta) - \omega^*\zeta(\beta)\omega\|_{hs}^2 \leq 6 \underset{\alpha : \, S \to \mathbb{F}_2}{\mathbb{E}} \|\chi(\alpha) - \omega^*\zeta(\alpha)\omega\|_{hs}^2 + 15\varepsilon.$$

which proves the claim. $\qquad\square$

3.2.3 Distance between correlations and strategies

Definition 3.23 (Distance between correlations). Recall from Remark 2.22 that every strategy $\mathscr{S}$ to a game $\mathfrak{G}$ induces a correlation $p(a, b|x, y)$. The distance between correlations associated with a game $\mathfrak{G}$ is the following L^1-type

$$d(p, q) = \underset{\substack{xy \sim \mu \\ a : \, S_x \to \mathbb{F}_2 \\ b : \, S_y \to \mathbb{F}_2}}{\mathbb{E}} \sum |p(a, b|x, y) - q(a, b|x, y)|.$$

Remark 3.24. The aforementioned distance between correlations is natural in the following way: If $\mathscr{S} = \{\mathcal{P}_a^x\}$ induces the correlation p, and $\mathscr{S}' = \{\mathcal{Q}_a^x\}$ induces the correlation q, then their values are closer than the distance between the correlations, namely

$$|\mathrm{val}(\mathfrak{G}, \mathscr{S}) - \mathrm{val}(\mathfrak{G}, \mathscr{S}')| = \Big| \underset{\substack{xy \sim \mu \\ a : \, S_x \to \mathbb{F}_2 \\ b : \, S_y \to \mathbb{F}_2}}{\mathbb{E}} \sum \Big(\underbrace{\tau(\mathcal{P}_a^x \mathcal{P}_b^y)}_{p(a,b|x,y)} - \underbrace{\tau(\mathcal{Q}_a^x \mathcal{Q}_b^y)}_{q(a,b|x,y)} \Big) D_{xy}(a, b) \Big|$$
$$\leq \underset{\substack{xy \sim \mu \\ a : \, S_x \to \mathbb{F}_2 \\ b : \, S_y \to \mathbb{F}_2}}{\mathbb{E}} \sum |p(a, b|x, y) - q(a, b|x, y)| \underbrace{D_{xy}(a, b)}_{\leq 1}$$
$$\leq d(p, q) .$$

Similar to the measurements case, we need a generalized notion of strategies for the rest of the arguments to be clear.

44

Definition 3.25 (Partial and Corner strategies). Let $\mathfrak{G}$ be a game with vertex (question) set V, formal generating sets $S_\mathtt{x}$ at each vertex[41] $\mathtt{x} \in V$, and distribution μ over edges (pairs of questions) of the underlying graph. An N-dimensional *partial strategy* for $\mathfrak{G}$ is a map $\mathcal{P}$ that for every vertex $\mathtt{x} \in V$ associates a partial POVM (Definition 3.6) $\mathcal{P}^\mathtt{x} \colon \mathbb{F}_2^{S_\mathtt{x}} \to M_{N \times N}(\mathbb{C})$.

Given an M-dimensional (full) strategy $\mathscr{S} = \{\mathcal{P}\}$ (as in Definition 2.18) and a partial isometry $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$, the N-dimensional partial strategy $\mathscr{S}' = \{\mathcal{P}'\}$, defined by $\mathcal{P}_a'^\mathtt{x} = \omega^* \mathcal{P}_a^\mathtt{x} \omega$, is called the *corner strategy* of $\mathcal{P}$ with respect to ω. We often denote the corner strategy by $\omega^* \mathcal{P} \omega$.

The following is the most straightforward notion of distance between (partial) strategies of the same dimension, which just takes the average distance over distance along the POVMs at each vertex.

Definition 3.26 (Strict distance between strategies). Let $\mathscr{S} = \{\mathcal{P}_a^\mathtt{x}\}$ and $\mathscr{S}' = \{\mathcal{Q}_a^\mathtt{x}\}$ be two N-dimensional partial strategies (Definition 3.25). We say that $\mathscr{S}$ is ε-*(strictly)-close* to $\mathscr{S}'$, and denote it by $\mathcal{P}_a^\mathtt{x} \approx_\varepsilon \mathcal{Q}_a^\mathtt{x}$, if

$$\mathbb{E}_{\mathtt{x} \sim \mu} \Big[\sum_{a \colon S_\mathtt{x} \to \mathbb{F}_2} \| \mathcal{P}_a^\mathtt{x} - \mathcal{Q}_a^\mathtt{x} \|_{hs}^2 \Big] \leq \varepsilon \, , \tag{22}$$

where $\mathtt{x} \sim \mu$ is the marginalization of μ to vertices defined by first sampling an edge and then choosing a uniform endpoint of it — i.e., $\mu(\mathtt{x}) = \frac{\sum_{\mathtt{y} \in V} \mu(\mathtt{xy}) + \mu(\mathtt{yx})}{2}$. Namely, on average over the vertices the associated POVMs are ε-(strictly)-close.

As we need to be able to compare strategies of varying dimensions, we define the following generalized notion of distance.

Definition 3.27 (Flexible distance between strategies). Let $\mathscr{S} = \{\mathcal{P}_a^\mathtt{x}\}$ be an N-dimensional strategy and $\mathscr{S}' = \{\mathcal{Q}_a^\mathtt{x}\}$ an M-dimensional strategy for a game $\mathfrak{G}$ with distribution μ over its edges. We say that $\mathscr{S}$ is ε-*(flexibly)-close* to $\mathscr{S}'$ if there exists an ε-near bijection (Definition 3.4) $\omega \colon \mathbb{C}^N \to \mathbb{C}^M$ such that $\mathcal{P}$ is ε-(strictly)-close to the corner strategy (Definition 3.25) $\omega^* \mathcal{Q} \omega$. Namely,

$$\max \Big\{ \mathbb{E}_{\mathtt{x} \sim \mu} \Big[\sum_{a \colon S_\mathtt{x} \to \mathbb{F}_2} \| \mathcal{P}_a^\mathtt{x} - \omega^* \mathcal{Q}_a^\mathtt{x} \omega \|_{hs}^2 \Big], 1 - \tau(\omega^* \omega), 1 - \tau(\omega \omega^*) \Big\} \leq \varepsilon \, . \tag{23}$$

Remark 3.28. Note that in the flexible notion of distance between strategies we measure the expected distance and inconsistency (Definition 3.10) between the PVM $\{\mathcal{P}_a^\mathtt{x}\}$ and the corner POVM $\{\omega^* \mathcal{Q}_a^\mathtt{x} \omega\}$ (over vertices $\mathtt{x} \in V$). The fact that the second object is non-projective and partial causes technical issues when proving various facts, as was already seen in previous proofs. But, in most cases in this paper, we perturb the strategies (and PVMs) in the same dimension, with respect to the trivial isometry $\omega = \mathrm{Id}_n$ — namely, we have small strict distance between the strategies. In this case, many of the technicalities in the proofs become much simpler (and with better parameters).

The following demonstrates that flexibly close by strategies produce close by correlations, and thus that their values against the game are close as well — which shows why this notion is natural in our context. The statement appears in Claim 3.29, and slightly generalizes [CVY23, Lemma 5.5].

Claim 3.29 (Perturbation of strategies). *Let $\mathscr{S} = \{\mathcal{P}_a^\mathtt{x}\}$ be an N-dimensional (full, projective) strategy and $\mathscr{S}' = \{\mathcal{Q}_a^\mathtt{x}\}$ an M-dimensional (full, projective) strategy that are ε-(flexibly)-close. Let p and q be the correlations that they induce (respectively), as in Remark 2.22. Then*

$$d(p, q) \leq 10\sqrt{\varepsilon} \, .$$

In particular, $|\mathrm{val}(\mathfrak{G}, \mathscr{S}) - \mathrm{val}(\mathfrak{G}, \mathscr{S}')| \leq 10\sqrt{\varepsilon}$.

Proof. As ω is an ε-near bijection, by Claim 3.17, for every edge $\mathtt{xy} \in E$ in the game, jointly measuring according to $(\mathcal{Q}^\mathtt{x}, \mathcal{Q}^\mathtt{y})$ is $4\sqrt{\varepsilon}$-close to jointly measuring according to the corners $(\omega \mathcal{Q}^\mathtt{x} \omega^*, \omega \mathcal{Q}^\mathtt{y} \omega^*)$, namely

$$\sum_{\substack{a \colon S_\mathtt{x} \to \mathbb{F}_2 \\ b \colon S_\mathtt{y} \to \mathbb{F}_2}} |\tau(\mathcal{Q}_a^\mathtt{x} \mathcal{Q}_b^\mathtt{y}) - \tau(\omega^* \mathcal{Q}_a^\mathtt{x} \omega \omega^* \mathcal{Q}_b^\mathtt{y} \omega)| \leq 4\sqrt{\varepsilon} \, . \tag{24}$$

---

[41]These sets are unions of readable and unreadable variables at the vertex, but this is irrelevant to this definition, so is ignored.

In addition, as $\mathcal{Q}$ was projective, by Claim 3.16, the corner strategy $\omega^*\mathcal{Q}\omega$ is ε-almost projective.

For every $x \in V$, let ε_x be the distance (Definition 3.10) between the PVM $\mathcal{P}^x$ and corner POVM $\omega^*\mathcal{Q}^x\omega$; by the ε-flexible-closeness of $\mathscr{S}$ and $\mathscr{S}'$, we have

$$\underset{xy\sim\mu}{\mathbb{E}}\left[\frac{\varepsilon_x + \varepsilon_y}{2}\right] = \underset{x\sim\mu}{\mathbb{E}}[\varepsilon_x] \le \varepsilon. \tag{25}$$

For every edge $xy \in E$, by Claim 3.18, jointly measuring according to $(\mathcal{P}^x, \mathcal{P}^y)$ is $\varepsilon + 2\sqrt{\varepsilon_x}$-close to jointly measuring according to $(\omega^*\mathcal{Q}^x\omega, \mathcal{P}^y)$, which in turn is $\varepsilon + 2\sqrt{\varepsilon_y}$-close to jointly measuring according to $(\omega^*\mathcal{Q}^x\omega, \omega^*\mathcal{Q}^y\omega)$. Hence,

$$\sum_{\substack{a:\, S_x\to\mathbb{F}_2 \\ b:\, S_y\to\mathbb{F}_2}} |\tau(\mathcal{P}_a^x\mathcal{P}_b^y) - \tau(\omega^*\mathcal{Q}_a^x\omega\omega^*\mathcal{Q}_b^y\omega)| \le 2\varepsilon + 2\sqrt{\varepsilon_x} + 2\sqrt{\varepsilon_y}. \tag{26}$$

Using Jensen's inequality and the bounds above, we deduce

$$
\begin{aligned}
d(p,q) &= \underset{xy\sim\mu}{\mathbb{E}}\left[\sum_{\substack{a:\, S_x\to\mathbb{F}_2 \\ b:\, S_y\to\mathbb{F}_2}} |p(a,b|x,y) - q(a,b|x,y)|\right] \\
&= \underset{xy\sim\mu}{\mathbb{E}}\left[\sum_{\substack{a:\, S_x\to\mathbb{F}_2 \\ b:\, S_y\to\mathbb{F}_2}} |\tau(\mathcal{P}_a^x\mathcal{P}_b^y) - \tau(\mathcal{Q}_a^x\mathcal{Q}_b^y)|\right] \\
&\le_{\triangle+(24)+(26)} \underset{xy\sim\mu}{\mathbb{E}}\Big[4\sqrt{\varepsilon} + 2\varepsilon + \underbrace{2\sqrt{\varepsilon_x} + 2\sqrt{\varepsilon_y}}_{\le_{\text{Jensen}} 4\sqrt{\frac{\varepsilon_x+\varepsilon_y}{2}}}\Big] \\
&\le_{\text{Jensen}} 4\sqrt{\varepsilon} + 2\varepsilon + 4\sqrt{\underset{xy\sim\mu}{\mathbb{E}}\left[\frac{\varepsilon_x+\varepsilon_y}{2}\right]} \\
&\le_{(25)} 8\sqrt{\varepsilon} + 2\varepsilon \le 10\sqrt{\varepsilon}.
\end{aligned}
$$

$\square$

We end this subsection by recalling a standard definition of *robustness* for games. This notion is commonlu used in the soundness analysis of games, which generally uses the condition $\mathrm{val}(\mathfrak{G}, \mathscr{S}) \ge 1 - \varepsilon$ to deduce many constraints on the structure of $\mathscr{S}$.

Definition 3.30. A game $\mathfrak{G}$ is said to be δ-*robust* (or *rigid* or *stable*), where $\delta\colon [0,1] \to [0,1]$ is a non-decreasing function with $\delta(\varepsilon) \xrightarrow{\varepsilon\to 0} 0$, if for every strategy $\mathscr{S}$ with $\mathrm{val}(\mathfrak{G}, \mathscr{S}) \ge 1 - \varepsilon$, there is a perfect strategy $\mathscr{S}'$ where $d(\mathscr{S}, \mathscr{S}') \le \delta(\varepsilon)$.

A game $\mathfrak{G}$ is a *self test* if all perfect strategies for it are the same up to isometries and corners (Definition 3.3). Namely, it has essentially one perfect strategy.

Remark 3.31. An example of an $O(\varepsilon)$-robust self test is the magic square game from Example 2.30.

3.3 Data processing

Data processing refers to the process of "coarse-graining" a POVM by applying a (generally non-injective) function to its output to define a new POVM. To formalize this, we introduce the following notation.

Definition 3.32 (Data processing POVM). Let $\{\mathcal{P}_a\}_{a\in A}$ be a POVM, and $f\colon A \to A'$ be a function. The f-evaluated POVM $\{\mathcal{P}_{[f(\cdot)=a']}\}_{a'\in A'}$ is defined to be

$$\mathcal{P}_{[f(\cdot)=a']} = \sum_{a\in A:\, f(a)=a'} \mathcal{P}_a.$$

46

One can think of this POVM procedurally as first measuring $a \in A$ and then outputting $f(a)$ — clarifying the term data processing. If $A = \mathbb{F}_2^S$, $A' = \mathbb{F}_2^{S'}$ for finite sets S and S', and $\mathcal{P}$ is projective, then $\mathcal{P}_a$ and $\mathcal{P}_{[f(\cdot)=a']}$ have an observable and representation form. If $\mathcal{U}$ is the observable (or representation) form of $\mathcal{P}$, then we denote by $\mathcal{U}_{[f]}$ the observable form of $\mathcal{P}_{[f(\cdot)=a']}$.

A common function that we data process along is restriction to a substring. For this case, we use the following notation. Let $S = S' \sqcup S''$, and let $f\colon \mathbb{F}_2^S \to \mathbb{F}_2^{S'}$ be the restriction to the S' substring, namely $f(\gamma) = \gamma|_{S'}$. In this case, we commonly denote $\mathcal{P}_{[f(\cdot)=a]}$ by $\mathcal{P}_a^{S'}$. So, sampling $a\colon S' \to \mathbb{F}_2$ according to $\mathcal{P}^{S'}$ is the same as sampling $\gamma\colon S \to \mathbb{F}_2$ according to $\mathcal{P}$ and returning the restriction of γ to S', namely $a = \gamma|_{S'}$. This restriction operation makes sense also in the observable and representation form $\mathcal{U}$ of the PVM: Let $\mathcal{U}^{S'}\colon \{0,1\}^{S'} \to U(n)$ be the composition of the embedding $\iota\colon \{0,1\}^{S'} \to \{0,1\}^S$ — defined by extending every function to be zero outside of S' — with $\mathcal{U}$, i.e.,

$$\forall \alpha\colon S' \to \mathbb{F}_2 : \quad \mathcal{U}^{S'}(\alpha) = \prod_{\mathsf{X} \in S'} \mathcal{U}(\mathsf{X})^{\alpha(\mathsf{X})}.$$

It is straightforward to check that, indeed, $\mathcal{U}^{S'}$ is the Fourier transform of $\mathcal{P}^{S'}$.

Remark 3.33. Note that the general data processing operation is very natural on PVMs in projective form, and usually unnatural in observable or representation forms (except for special cases, such as the restriction). This is a recurrent theme. Some operations and arguments are easier in the projective viewpoint and others in the observable viewpoint. It is good to remember that the object is the same, whether it is viewed in projective or observable (or representation) form, and thus one can apply operations and arguments in the more convenient form.

Observation 3.34. Let $\mathcal{P}$ be a POVM with outcomes in A, $\mathcal{Q}$ a POVM of the same dimension with outcomes in B, and $f\colon A \to C, g\colon B \to C$ two functions. Then, in the spirit of Remark 3.11, the probability a jointly sampled pair $(a,b) \sim (\mathcal{P},\mathcal{Q})$ does not satisfy $f(a) = g(b)$ is exactly the inconsistency of the data processed POVMs $\mathcal{P}_{[f(\cdot)=\cdot]}$ and $\mathcal{Q}_{[g(\cdot)=\cdot]}$. As small inconsistency implies small distance (clause 4. in Proposition 3.12), this will be a useful tool for deducing that the POVMs a strategy associates to the endpoints of an edge are close (after data processing them).

In the other direction, as PVMs are self-consistent, a jointly sampled pair $(a,a') \sim (\mathcal{P},\mathcal{P}_{[f(\cdot)=\cdot]})$ where $\mathcal{P}$ is a PVM always satisfies $f(a) = a'$.

Claim 3.35 (Inconsistency can only decrease by data processing. Cf. Fact 4.25 in [NW19]). *Let $\mathcal{P}$ and $\mathcal{Q}$ be POVMs of the same dimension with outcomes in the set A, and let $f\colon A \to A'$ be a function. Then $\mathcal{P} \simeq_\varepsilon \mathcal{Q}$ implies $\mathcal{P}_{[f(\cdot)=\cdot]} \simeq_\varepsilon \mathcal{Q}_{[f(\cdot)=\cdot]}$.*

Proof. This is immediate from the fact that applying a function on a pair of answers may only increase the probability of them agreeing. Let us provide the calculation in any case:

$$\sum_{a' \neq b' \in A'} \tau(\mathcal{P}_{[f(\cdot)=a']} \mathcal{Q}_{[f(\cdot)=b']}) = \sum_{a' \neq b' \in A'} \tau\left(\left(\sum_{a \in A\,:\, f(a)=a'} \mathcal{P}_a\right)\left(\sum_{b \in A\,:\, f(b)=b'} \mathcal{Q}_b\right)\right)$$
$$= \sum_{a,b \in A\,:\, f(a) \neq f(b)} \tau(\mathcal{P}_a \mathcal{Q}_b)$$
$$\leq \sum_{a \neq b \in A} \tau(\mathcal{P}_a \mathcal{Q}_b) \,.$$

$\square$

As in tailored games the comparisons along edges are linear, the following special case of Observation 3.34 and Claim 3.35 will be repeatedly used. We add the proof for clarity.

Claim 3.36 (Consistency of linear checks). *Let $S_{\mathsf{x}}, S_{\mathsf{y}}$ be finite sets, $\mathcal{U}^{\mathsf{x}}\colon \mathbb{F}_2^{S_{\mathsf{x}}} \to U(n)$ and $\mathcal{U}^{\mathsf{y}}\colon \mathbb{F}_2^{S_{\mathsf{y}}} \to U(n)$ two representations. Fix $\alpha\colon S_{\mathsf{x}} \to \mathbb{F}_2$ and $\beta\colon S_{\mathsf{y}} \to \mathbb{F}_2$. Then, the probability that $\sum_{\mathsf{X} \in S_{\mathsf{x}}} \alpha(\mathsf{X}) \gamma(\mathsf{X}) \neq \sum_{\mathsf{Y} \in S_{\mathsf{y}}} \beta(\mathsf{Y}) \gamma(\mathsf{Y})$ when $\gamma\colon S_{\mathsf{x}} \sqcup S_{\mathsf{y}} \to \mathbb{F}_2$ is jointly sampled (Definitions 2.2 and 2.5) according to $(\mathcal{U}^{\mathsf{x}}, \mathcal{U}^{\mathsf{y}})$ is exactly $1/4 \cdot \|\mathcal{U}^{\mathsf{x}}(\alpha) - \mathcal{U}^{\mathsf{y}}(\beta)\|_{hs}^2$.*

Proof. Denote, as usual, $\gamma = (a, b)$. First, note that $\sum_{\mathsf{X} \in S_\mathsf{x}} \alpha(\mathsf{X})\gamma(\mathsf{X}) \neq \sum_{\mathsf{Y} \in S_\mathsf{y}} \beta(\mathsf{Y})\gamma(\mathsf{Y})$ if and only if $\langle (\alpha, \beta), \gamma \rangle = 1$. Then,

$$\mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 0] = \sum_{\gamma = (a,b) \,:\, \langle \alpha, a \rangle = \langle \beta, b \rangle} \tau \left(\mathcal{P}_a^\mathsf{x} \mathcal{P}_b^\mathsf{y} \right) \,,$$

$$\mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 1] = \sum_{\gamma = (a,b) \,:\, \langle \alpha, a \rangle \neq \langle \beta, b \rangle} \tau \left(\mathcal{P}_a^\mathsf{x} \mathcal{P}_b^\mathsf{y} \right) \,,$$

$$\mathcal{U}^\mathsf{x}(\alpha) = \sum_{a \,:\, \langle a, \alpha \rangle = 0} \mathcal{P}_a^\mathsf{x} - \sum_{a \,:\, \langle a, \alpha \rangle = 1} \mathcal{P}_a^\mathsf{x} \,,$$

$$\mathcal{U}^\mathsf{y}(\beta) = \sum_{b \,:\, \langle b, \beta \rangle = 0} \mathcal{P}_b^\mathsf{y} - \sum_{b \,:\, \langle b, \beta \rangle = 1} \mathcal{P}_b^\mathsf{y} \,.$$

So,

$$\tau \left(\mathcal{U}^\mathsf{x}(\alpha) \mathcal{U}^\mathsf{y}(\beta) \right) = \tau \left(\sum_{\gamma = (a,b) \,:\, \langle a, \alpha \rangle = \langle b, \beta \rangle} \mathcal{P}_a^\mathsf{x} \mathcal{P}_b^\mathsf{y} - \sum_{\gamma = (a,b) \,:\, \langle a, \alpha \rangle \neq \langle b, \beta \rangle} \mathcal{P}_a^\mathsf{x} \mathcal{P}_b^\mathsf{y} \right)$$

$$= \mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 0] - \mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 1]$$

$$= 1 - 2 \mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 1] \,,$$

and

$$\| \mathcal{U}^\mathsf{x}(\alpha) - \mathcal{U}^\mathsf{y}(\beta) \|_{hs}^2 = \| \mathrm{Id} - \mathcal{U}^\mathsf{x}(\alpha) \mathcal{U}^\mathsf{y}(\beta) \|_{hs}^2$$

$$= 2 - 2\mathrm{Re} \left(\tau \left(\mathcal{U}^\mathsf{x}(\alpha) \mathcal{U}^\mathsf{y}(\beta) \right) \right)$$

$$= 2 - 2\mathrm{Re} \left(1 - 2 \mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 1] \right)$$

$$= 4 \mathbb{P}_{\gamma \sim (\mathcal{U}^\mathsf{x}, \mathcal{U}^\mathsf{y})} [\langle (\alpha, \beta), \gamma \rangle = 1] \,,$$

as claimed. $\square$

3.4 Data processing of permutation strategies

As opposed to f-evaluating (Definition 3.32) POVMs and PVMs, which clearly remain POVMs and PVMs respectively, it is not true for a general function $f \colon \mathbb{F}_2^S \to \mathbb{F}_2^{S'}$ that the f-evaluation of a signed permutation PVM (Definition 2.15) remains a signed permutation PVM. A simple example for this phenomenon is the bit product function (which is the arithmetic version of the boolean AND), namely $f(a, b) = a \cdot b$. One can check that if we have two formal variables X, Y that are sent by $\mathcal{U}$ to the commuting involutive permutation matrices

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \,, \quad \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \,, \tag{27}$$

then the observable form of the f-evaluated PVM $\mathcal{U}_{[f]}$ is

$$\frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 \\ -1 & 1 & 1 & 1 \end{bmatrix} \,,$$

which is not a signed permutation matrix. As the completeness in Theorem 2.31 requires working with Z-aligned permutation strategies that commute along edges (ZPC strategies), we need to characterize the functions f for which f-evaluation preserves the ZPC-property.

To that end, we need the following to claims:

Claim 3.37 (Data processing diagonal PVMs). *If $\mathcal{P}$ is a diagonal PVM (Definition 2.7) with outcomes in A, and $f\colon A \to A'$ is a function, then $\mathcal{P}_{[f(\cdot)=\cdot]}$ is a diagonal PVM.*

Proof. As diagonal matrices are closed under addition, the claim follows. $\square$

Given a finite set S, there is a one to one correspondence between functions $\alpha\colon S \cup \{\mathsf{J}\} \to \mathbb{F}_2$ and affine maps (i.e., linear maps plus a constant) $\beta\colon \mathbb{F}_2^S \to \mathbb{F}_2$ given by

$$\forall a\colon S \to \mathbb{F}_2 : \quad \beta(a) = \alpha(\mathsf{J}) + \sum_{\mathsf{X} \in S} \alpha(\mathsf{X})a(\mathsf{X}) . \tag{28}$$

Claim 3.38 (Affine data processing). *Let $\mathcal{U}\colon S \to U(n)$ be a PVM in observable form, let $\alpha\colon S \cup \{\mathsf{J}\} \to \mathbb{F}_2$ be a function, and let β be the corresponding affine map as in (28). Let $\mathcal{U}_{[\beta]}$ be the β-evaluated PVM (Definition 3.32), which consists of a single observable $\mathcal{O}$. Then,*

$$\mathcal{O} = (-\mathrm{Id})^{\alpha(\mathsf{J})} \cdot \prod_{\mathsf{X} \in S} \mathcal{U}(\mathsf{X})^{\alpha(\mathsf{X})} .$$

In particular, $\mathcal{O}$ is in the group generated by $-\mathrm{Id}$ and $\mathrm{Im}(\mathcal{U})$.

Proof. This follows from Observation 3.34 and Claim 3.36. $\square$

Corollary 3.39. *If $\beta\colon \mathbb{F}_2^S \to \mathbb{F}_2$ is an affine function, and $\mathcal{U}\colon S \to \mathrm{Sym}_\pm(\Omega)$ is a signed permutation PVM in observable form (Definition 2.15), then the β-evaluated $\mathcal{U}_{[\beta]}$ is also a signed permutation PVM. Moreover, if $\gamma\colon \mathbb{F}_2^S \to \mathbb{F}_2^{S'}$ is a linear map, then*

$$\forall \alpha\colon S' \to \mathbb{F}_2 : \quad \mathcal{U}_{[\gamma]}(\alpha) = \mathcal{U}(\gamma^*(\alpha)) , \tag{29}$$

where $\gamma^*\colon \mathbb{F}_2^{S'} \to \mathbb{F}_2^S$ is the dual map with respect to the bilinear form $\langle \cdot, \cdot \rangle$. In other words, if we fix the standard bases of $\mathbb{F}_2^S$, $\mathbb{F}_2^{S'}$ to be the indicators $\mathbf{1}_\mathsf{X}$, then γ is a matrix and $\gamma^*(\alpha)$ is the left multiplication

$$\alpha \cdot \gamma = \Big(\sum_{\mathsf{X}' \in S} \alpha_{\mathsf{X}'} \cdot \gamma_{\mathsf{X}'\mathsf{X}} \Big)_{\mathsf{X} \in S} .$$

Observation 3.40. A readably Z-aligned (Definition 2.8) signed permutation PVM (Definition 2.15) in observable form

$$\mathcal{U}\colon S^\mathfrak{R} \cup S^\mathfrak{L} \to \mathrm{Sym}_\pm(\Omega) \subseteq U(W^-) ,$$

when represented with respect to the standard basis B^- from (6), consists of block diagonal matrices, where in each block the readable observables are constant (and diagonal) — this is immediate from the fact that the image of $\mathcal{U}$ consists of commuting matrices for which the readable variables are diagonal ± 1 matrices.

Corollary 3.41. *Let $\mathcal{U}\colon S^\mathfrak{R} \cup S^\mathfrak{L} \to \mathrm{Sym}_\pm(\Omega)$ be a readably Z-aligned (Definition 2.8) signed permutation PVM (Definition 2.15). For every $a^\mathfrak{R} \in \mathbb{F}_2^{S^\mathfrak{R}}$ let $\mathfrak{s}_{1,a^\mathfrak{R}}, ..., \mathfrak{s}_{k,a^\mathfrak{R}}$ be a sequence of affine maps from $\mathbb{F}_2^{S^\mathfrak{R} \cup S^\mathfrak{L}} \to \mathbb{F}_2$ and let $f_1, ..., f_t$ be a collection of functions from $\mathbb{F}_2^{S^\mathfrak{R}} \to \mathbb{F}_2$. Then, by adding a set $\{\mathsf{X}_{f_i}\}_{i=1}^t$ of readable variables and $\{\mathsf{Y}_{\mathfrak{s}_j}\}_{j=1}^k$ of unreadable variables, we can define a readably Z-aligned signed permutation PVM $\mathcal{V}$ that extends[42] $\mathcal{U}$ to the new variables and satisfies the following:*

---

[42]For the joint sampling to make sense, the formal variables of $\mathcal{U}$ and $\mathcal{V}$ should be disjoint. So, the notion of extension is a bit misleading, but it is easier to follow notationally. What we mean is that $\mathcal{V}$ has a copy of the variables at $\mathcal{U}$ and associates the same observables to them.

Given a sampled pair $(a, b) \sim (\mathcal{U}, \mathcal{V})$, denoting $a^{\mathfrak{R}} = a|_{S^{\mathfrak{R}}}$, we have

$$\forall X \in S^{\mathfrak{R}} \cup S^{\mathfrak{L}} : \quad a(X) = b(X) \, , \tag{30}$$

$$\forall 1 \leq i \leq t : \quad b(X_{f_i}) = f_i(a^{\mathfrak{R}}) \, , \tag{31}$$

$$\forall 1 \leq j \leq k : \quad b(Y_{\mathfrak{s}_j}) = \mathfrak{s}_{j, a^{\mathfrak{R}}}(a) \, . \tag{32}$$

In words, we can replace a readably Z-aligned signed permutation PVM with outcomes in $\mathbb{F}_2^S$ with a new readably Z-aligned signed permutation PVM with outcomes in $\mathbb{F}_2^{S'}$, where $S \subseteq S'$, and the new PVM samples the same strings as the original PVM (as part of its output), and in addition has bits which are either functions on the values of the readable variables, or affine combinations of all values, where the specific combinations depend on the values of the readable variables.

Proof. Condition (30) is guaranteed by choosing $\mathcal{V}$ to be an extension of $\mathcal{U}$. For (31), the restriction of $\mathcal{U}$ to the readable variables is diagonal (by the definition of a readably Z-aligned measurement), and thus it is immediate from Claim 3.37. Finally, due to Observation 3.40, Claim 3.38 can be applied to each block individually with the appropriate affine map, as the value of the readable variables there is constant. $\qquad\square$

Remark 3.42. Corollary 3.41 is very useful, as it allows one to take perfect strategies and encode each output of them in various ways. Specifically, one can encode the outputs using error correcting codes, which is an important step in the construction of PCPs. See Section 5 for more on that. It essentially characterizes the types of functions $f \colon \mathbb{F}_2^S \to \mathbb{F}_2^{S'}$ for which the f-evaluated PVM remains a signed permutation PVM and preserves the readably Z-aligned structure.

3.5 Transformations of games

Compression consists of applying various transformations to the input normal form verifier $\mathcal{V}$. Some of these transformations, when observed as acting on the associated games $\mathcal{V}_n$, are applying some form of *game composition*. As it sounds, composing games is just a process that takes two (or more) games, and generates a new game out of them.

3.5.1 Product and sum of games

We describe two straightforward examples: *product* and *sum* of games. The product is the parallel play in both games — namely, each round of the product consists of a round from both games — while the sum is the barycenter of them — namely, with probability $1/2$ it plays a round of one game, and with probability $1/2$ it plays a round in the other game.[43] On the level of the underlying graphs, the product of games has the tensor product of the graphs underlying it, and the sum of games has a disjoint union of the graphs underlying it.

Definition 3.43 (Product of games). Given two games $\mathfrak{G}_1$ and $\mathfrak{G}_2$, their product $\mathfrak{G}_1 \otimes \mathfrak{G}_2$ is defined as follows. If G_1 and G_2 are the underlying graphs of $\mathfrak{G}_1$ and $\mathfrak{G}_2$ respectively, then the underlying graph of the product is $G_1 \otimes G_2$. An edge in $G_1 \otimes G_2$ is of the form $(x_1, x_2)(y_1, y_2)$, where $x_1 y_1$ is an edge in G_1 and $x_2 y_2$ is an edge in G_2. The probability of sampling $(x_1, x_2)(y_1, y_2)$ is $\mu_1(x_1 y_1) \cdot \mu_2(x_2 y_2)$. For the length function (and in the tailored category, functions), $\ell(x_1, x_2) = \ell_1(x_1) + \ell_2(x_2)$, and we think of $S_{(x_1, x_2)}$ as the disjoint union of (its own copies of) S_{x_1} and S_{x_2} (respectively for readable and unreadable variables in the tailored category). Hence, we can think of the answer to (x_1, x_2) as being a pair of answers (a_1, a_2). Finally, $D_{(x_1, x_2)(y_1, y_2)}((a_1, a_2)(b_1, b_2)) = D^1_{x_1 y_1}(a_1 b_1) \cdot D^2_{x_2 y_2}(a_2 b_2)$, where D^1 (respectively D^2) is the decision function of $\mathfrak{G}_1$ (respectively $\mathfrak{G}_2$). This again works out in the tailored category by letting $L_{(x_1, x_2)(y_1, y_2)}((a_1^{\mathfrak{R}}, a_2^{\mathfrak{R}}), (b_1^{\mathfrak{R}}, b_2^{\mathfrak{R}})) = L^1_{x_1 y_1}(a_1^{\mathfrak{R}}, b_1^{\mathfrak{R}}) \sqcup L^2_{x_2 y_2}(a_2^{\mathfrak{R}}, b_2^{\mathfrak{R}})$, where L^1 and L^2 are the respective controlled linear constraints functions of $\mathfrak{G}_1$ and $\mathfrak{G}_2$.[44] Note that the above union of linear constraints makes sense only because the formal generating sets at x_1 and x_2 are "embedded" in $S_{(x_1, x_2)}$ (and similarly for y).

---

[43]The exact probabilities will not necessarily be $1/2$, but it is a good example to hold in mind — so, the sum is more of a convex combination than exactly the barycenter.

[44]It is straightforward to check that indeed, by taking the disjoint union of linear constraints, the canonical decider will accept the answers only when it would have accepted them in each game separately.

Though it is fitting to start and analyze the completeness and soundness properties of the product game, we leave it to the parallel repetition section, Section 6, in which it is used.

Definition 3.44 (Sum of games). Given two games $\mathfrak{G}_1$ and $\mathfrak{G}_2$, their sum $\mathfrak{G}_1 \oplus \mathfrak{G}_2$ is defined as follows. The underlying graph is the disjoint union of the underlying graphs G_1 and G_2 of $\mathfrak{G}_1$ and $\mathfrak{G}_2$. The distribution on edges is

$$\mu(e) = \begin{cases} \frac{\mu_1(e)}{2} & e \in G_1 \,, \\ \frac{\mu_2(e)}{2} & e \in G_2 \,, \end{cases}$$

where μ_1 and μ_2 are the respective distributions on edges in $\mathfrak{G}_1$ and $\mathfrak{G}_2$.[45] The length function is

$$\ell(\mathbf{x}) = \begin{cases} \ell_1(\mathbf{x}) & \mathbf{x} \in G_1 \,, \\ \ell_2(\mathbf{x}) & \mathbf{x} \in G_2 \,, \end{cases}$$

with ℓ_1 and ℓ_2 being the respective length functions (and similarly in the tailored category). We presume $S_{\mathbf{x}}$ remains the same in this case. Finally, every edge $\mathbf{xy}$ is either in G_1 or in G_2. If it is in G_1, then $D_{\mathbf{xy}}(\gamma) = D^1_{\mathbf{xy}}(\gamma)$, and if it is in G_2, then $D_{\mathbf{xy}}(\gamma) = D^2_{\mathbf{xy}}(\gamma)$. Furthermore, in the tailored category, we assume $L_{\mathbf{xy}}(\gamma) = L^1_{\mathbf{xy}}(\gamma)$ and $L_{\mathbf{xy}}(\gamma) = L^2_{\mathbf{xy}}(\gamma)$ with respect to whether $\mathbf{xy} \in G_1$ or $\mathbf{xy} \in G_2$.[46]

Since the sum has a disconnected underlying graph, it is natural to *augment* it, so that the games are forced to be related in some way. There are many ways to do so. Usually, the augmentation involves the addition of vertices and edges between the graphs that check various consistencies between the answers. Both the Pauli basis game (Section 3.8.3) and the question reduction game (Section 4.4) are augmented sums of smaller games.

Definition 3.45 (Augmentation of a game). Given a game $\mathfrak{G}$, we say that another game $\mathfrak{G}'$ is *an augmentation of* $\mathfrak{G}$, or conversely that $\mathfrak{G}$ is *contained* or a *sub-game* of $\mathfrak{G}'$, if there is a subgraph of the underlying graph of $\mathfrak{G}'$ such that the restriction of $\mathfrak{G}'$ to this subgraph is an instance of $\mathfrak{G}$ (up to the distribution on edges μ). In more words, $\mathfrak{G}'$ is defined by adding vertices and edges to the underlying graph of $\mathfrak{G}$, such that the lengths and decision procedure on the "original" edges stays the same.

3.5.2   Product and Sum of PVMs

Recall the notion of the Kronecker tensor product of matrices, which we denote by $\otimes$. Similar to composition of games, we can also compose PVMs (and thus strategies), which results in new PVMs with some new properties.

Definition 3.46 (Product of PVMs). Given two PVMs (in observable form) $\mathcal{U}^1 \colon S \to U(N)$ and $\mathcal{U}^2 \colon S \to U(M)$ over the same variable set S, we define their product to be the PVM $\mathcal{U}^1 \otimes \mathcal{U}^2 \colon S \to U(NM)$ satisfying

$$(\mathcal{U}^1 \otimes \mathcal{U}^2)(\mathsf{X}) = \mathcal{U}^1(\mathsf{X}) \otimes \mathcal{U}^2(\mathsf{X}) \,.$$

Definition 3.47 (Sum of PVMs). Given two PVMs (in observable form) $\mathcal{U}^1 \colon S^1 \to U(N)$ and $\mathcal{U}^2 \colon S^2 \to U(M)$ over **different** variable sets S, we define their sum to be the PVM $\mathcal{U}^1 \oplus \mathcal{U}^2 \colon S^1 \sqcup S^2 \to U(NM)$ defined by

$$\mathcal{U}^1 \oplus \mathcal{U}^2(\mathsf{X}) = \begin{cases} \mathcal{U}^1(\mathsf{X}) \otimes \mathrm{Id}_M & \mathsf{X} \in S^1 \,, \\ \mathrm{Id}_N \otimes \mathcal{U}^2(\mathsf{X}) & \mathsf{X} \in S^2 \,. \end{cases}$$

---

[45] As we remarked before, the distribution may change from exactly $1/2 - 1/2$ to some other one.

[46] This again can be checked to work the same on the level of canonical deciders.

Remark 3.48. It is straightforward to check that given two signed permutation matrices their Kronecker tensor product is a signed permutation as well, and that the tensor product of diagonal matrices is diagonal. Hence, the above two operations on PVMs (sum and product) preserve readable Z-alignment as well as being a signed permutation PVM.

As signed permutations act on a signed sets, when one performs the tensor product of two signed permutations, one acting on $\Omega_\pm^1$ and one on $\Omega_\pm^2$, the resulting signed permutation acts on $(\Omega^1 \times \Omega^2)_\pm$. For explanatory reasons, we define an equivalence relation on $\Omega_\pm^1 \times \Omega_\pm^2$ which bijects it on $(\Omega^1 \times \Omega^2)_\pm$ by letting

$$\forall \star \in \Omega^1, \diamond \in \Omega^2 : \quad (+\star, +\diamond) = (-\star, -\diamond) = +(\star, \diamond) \quad \text{and} \quad (-\star, +\diamond) = (+\star, -\diamond) = -(\star, \diamond). \tag{33}$$

In this guise, the tensor product acts as expected: Given two signed permutation strategies $\sigma_1 \colon S \to \mathrm{Sym}(\Omega_\pm^1)$ and $\sigma_2 \colon S \to \mathrm{Sym}(\Omega_\pm^2)$, we have

$$\forall X \in S, \spadesuit \in \Omega_\pm^1, \diamondsuit \in \Omega_\pm^2 : \quad \sigma_1 \otimes \sigma_2(X).(\spadesuit, \diamondsuit) = \sigma_1(X) \times \sigma_2(X).(\spadesuit, \diamondsuit) = (\sigma_1(X).\spadesuit, \sigma_2(X).\diamondsuit). \tag{34}$$

Remark 3.49. The product and sum operations should be familiar to graph theorists, as these are PVM analogs of the *tensor product and cartesian product of graphs*. Indeed, if one applies these transformations to signed permutation PVMs, and look at the resulting Schreier graph induced by the new PVMs (in observable form), then it is respectively the tensor product and cartesian product of the original Schreier graphs.

Lemma 3.50. *Let $\mathfrak{G}_1$ and $\mathfrak{G}_2$ be two (tailored) games, and let $\mathfrak{G} = \mathfrak{G}_1 \otimes \mathfrak{G}_2$ be their product taken according to Definition 3.43. For $i \in \{1, 2\}$ let $\mathcal{U}^i \colon S \to U(N_i)$ be a strategy for $\mathfrak{G}_i$, in observable form. Let $\mathcal{U} = \mathcal{U}^1 \oplus \mathcal{U}^2$ be their sum, taken according to Definition 3.47. Then the following hold:*

1. *$\mathcal{U}$ is a valid strategy for $\mathfrak{G}$.*

2. *If both $\mathcal{U}^1$ and $\mathcal{U}^2$ have value 1, then so does $\mathcal{U}$.*

3. *If both $\mathcal{U}^1$ and $\mathcal{U}^2$ are Z-aligned, then so is $\mathcal{U}$.*

4. *If both $\mathcal{U}^1$ and $\mathcal{U}^2$ are commuting along edges, then so is $\mathcal{U}$.*

As a consequence, if $\mathcal{U}^1$ and $\mathcal{U}^2$ are perfect ZPC strategies then so is $\mathcal{U}$.

Proof. The first item follows because, according to the definition, the set of generators S for $\mathfrak{G}_1 \otimes \mathfrak{G}_2$ is the disjoint union $S_1 \sqcup S_2$. To show the second item, fix a question pair $(x_1, x_2), (y_1, y_2)$ in $\mathfrak{G}$. Then the strategy $\mathcal{U}$ samples answers γ according to a product distribution, i.e.

$\Pr[\gamma = ((a_1, a_2), (b_1, b_2))$ is sampled by $\mathcal{U} \mid (x_1, x_2), (y_1, y_2)$ were sampled$]$

$= \Pr[\gamma_1 = (a_1, b_1)$ is sampled by $\mathcal{U}^1 \mid x_1, y_1$ were sampled$] \cdot \Pr[\gamma_2 = (a_2, b_2)$ is sampled by $\mathcal{U}^2 \mid x_2, y_2$ were sampled$]$.

Item 2 follows since the decision function of the product game simply checks the conjunction of the decision functions of each individual game. Regarding item 3, its validity was already observed in Remark 3.48 above. Finally, item 4 follows because given an edge $(x_1, x_2), (y_1, y_2)$, the associated permutations are either associated to the edge $x_1 y_1$ from $\mathfrak{G}_1$ and act on the first tensor factor in $U(N_1) \otimes U(N_2)$, or associated with the edge $x_2 y_2$ from $\mathfrak{G}_2$ and act on the first tensor factor. Since both $\mathcal{U}^1$ and $\mathcal{U}^2$ are assumed to commute along edges, and since unitaries acting on different tensor factors commute, the conclusion follows. $\square$

3.5.3 Double cover of a game

Another natural transformation on games is their *double cover*. A double cover of a graph $G = (V, E)$ is the graph $G_\pm = (V_\pm, E_\pm)$ defined as follows: $V_\pm = \{\pm\} \times V$, and we denote, as usual, $+v$ instead of $(+, v)$ and $-v$ instead of $(-, v)$; for any (oriented) edge $e = (v, w) \in E$, there are two appropriate (oriented) edges $+e = (+v, -w)$ and $-e = (-v, +w)$ in $E_\pm$. As its name suggest, the double cover is indeed a combinatorial covering space (cf. [BL06] under the name of *lifts*) of G, and the covering map $\pi \colon G' \to G$ is the one which removes the signs. The following are easy to verify facts about the double cover of a graph.

Fact 3.51.

1. *The double cover of a graph is always bipartite.*

2. *The double cover of a bipartite graph is a disjoint union of two copies of the original graph.*

Definition 3.52 (Double cover of a game)**.** Let $\mathfrak{G}$ be a tailored game. Its double cover $\mathfrak{DoubleCover}(\mathfrak{G}) = \mathfrak{G}'$ is a game whose underlying graph is the double cover $G_\pm$ of the underlying graph G of the game $\mathfrak{G}$. The distribution over edges in $\mathfrak{G}'$ is defined to be

$$\forall \pm e \in E' : \quad \mu'(\pm e) = \mu(e)/2 \,,$$

where μ is the distribution of $\mathfrak{G}$ over the edges in G. Namely, the sampling scheme of the double cover game is as follows: Sample $e \in E$ according to μ, and choose a sign $\varepsilon \in \{\pm\}$ uniformly; output $\varepsilon \cdot e \in E_\pm$. The lengths of the vertex $\varepsilon \cdot \mathsf{x} \in V_\pm$ are the same as the lengths of x in $\mathfrak{G}$. In addition, the elements of the formal generating set $S_{+\mathsf{x}}$ will be of the form $+\mathsf{X}$ for $\mathsf{X} \in S_\mathsf{x}$, and similarly $-\mathsf{X}$ will be the form of elements in $S_{-\mathsf{x}}$. If $+e = (+\mathsf{x}, -\mathsf{y})$ (respectively $-e = (-\mathsf{x}, +\mathsf{y})$) is sampled, then L_{+e} (respectively L_{-e}) treats $S_{+\mathsf{x}}$ (respectively $S_{-\mathsf{x}}$) as S_x and $S_{-\mathsf{y}}$ (respectively $S_{+\mathsf{y}}$) as S_y and outputs the appropriate linear constraints (given the restriction $\gamma^\mathfrak{R} \colon S^\mathfrak{R}_{\pm\mathsf{x}} \sqcup S^\mathfrak{R}_{\mp\mathsf{y}} \to \mathbb{F}_2$). If $\mathsf{x} = \mathsf{y}$, then in addition to the above constraints, it also outputs the consistency checks

$$\forall \mathsf{X} \in S_\mathsf{x} : \quad \gamma(+\mathsf{X}) = \gamma(-\mathsf{X}) \,.$$

On the combinatorial level, the double cover acts as follows: If $a^\mathfrak{R}, a^\mathfrak{L}$ are the answers associated to $+\mathsf{x}$ and $b^\mathfrak{R}, b^\mathfrak{L}$ are the answers associated to $-\mathsf{y}$, for $\mathsf{x} \neq \mathsf{y}$, then the double cover will accept these answers if and only if the original game would accept these answers for x and y respectively. In the case $\mathsf{x} = \mathsf{y}$, the double cover needs (in addition to the checks induced by the original game $\mathfrak{G}$) to check *consistency*, namely that $a^\mathfrak{R} = b^\mathfrak{R}$ and $a^\mathfrak{L} = b^\mathfrak{L}$.

Remark 3.53. The definition of the double cover is natural when trying to relate non-synchronous strategies to synchronous strategies of the same game (see Section 3.6). In addition, it is used in the *detyping transformation* (Definition 4.40).

Claim 3.54. *Let $\mathfrak{G}$ be a tailored game, such that in its underlying graph $G = (V, E)$, all loops xx for $\mathsf{x} \in V$ appear as edges in E. Assume in addition that there is some constant $c > 0$, such that for every $\mathsf{x} \in V$ we have*

$$\frac{\mu(\mathsf{xx})}{\mu(\mathsf{x})} \geq c \,, \tag{35}$$

where μ is the distribution over edges in $\mathfrak{G}$, and $\mu(\mathsf{x})$ is (as before) the marginal on vertices, namely $\mu(\mathsf{x}) = \sum_{\mathsf{y} \in V} \frac{\mu(\mathsf{xy}) + \mu(\mathsf{yx})}{2}$. Then:

- *(Completeness) if $\mathfrak{G}$ has a perfect ZPC strategy, then so does $\mathfrak{DoubleCover}(\mathfrak{G})$;*

- *(Soundness) if $\mathfrak{DoubleCover}(\mathfrak{G})$ has a strategy $\mathscr{S}$ with value $1 - \varepsilon$, then $\mathfrak{G}$ has a strategy with value of at least $1 - O(\sqrt{\varepsilon}/c)$. In particular,*
$$\mathscr{E}(\mathfrak{DoubleCover}(\mathfrak{G}), 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{G}, 1 - O(\sqrt{\varepsilon}/c)) \,.$$

Proof. For completeness, note that if $\sigma \colon S \to \mathrm{Sym}(\Omega_\pm)$ is a perfect ZPC strategy, then $\sigma' \colon S_\pm \to \mathrm{Sym}(\Omega_\pm)$ defined by $\sigma'(\pm\mathsf{X}) = \sigma(\mathsf{X})$ is a perfect ZPC strategy for $\mathfrak{DoubleCover}(\mathfrak{G})$.

For soundness, let $\mathscr{S} = \{\mathcal{U}\}$ pass $\mathfrak{DoubleCover}(\mathfrak{G})$ with probability $1 - \varepsilon$. By (35),

$$\sum_{\mathsf{x} \in V} \mu(\mathsf{xx}) \leq \sum_{\mathsf{x} \in V} \mu(\mathsf{x}) \leq \frac{1}{c} \cdot \sum_{\mathsf{x} \in V} \mu(\mathsf{xx}) \,,$$

and hence

$$\mathbb{P}[\mathscr{S} \text{ loses} \mid \text{a loop was sampled}] \leq \frac{\mathbb{P}[\mathscr{S} \text{ loses}]}{\mathbb{P}[\text{a loop was sampled}]} \leq \frac{\varepsilon}{\sum_{\mathsf{x} \in V} \mu(\mathsf{xx})} \leq \frac{\varepsilon}{c \cdot \sum_{\mathsf{x} \in V} \mu(\mathsf{x})} = \varepsilon/c \,.$$

Let ε_x be the probability $\mathscr{S}$ loses when $(+x, -x)$ or $(-x, +x)$ is sampled. Then by the above derivations and using (35) again,

$$c \cdot \mathop{\mathbb{E}}_{x \sim \mu} [\varepsilon_x] = c \cdot \sum_{x \in V} \mu(x) \varepsilon_x \leq \sum_{x \in V} \mu(xx) \varepsilon_x \leq \frac{\sum_{x \in V} \mu(xx) \varepsilon_x}{\sum_{y \in V} \mu(yy)} = \mathbb{P}[\mathscr{S} \text{ loses} \mid \text{a loop was sampled}] \leq \varepsilon/c \,. \quad (36)$$

On the other hand, whenever $(+x, -x)$ or $(-x, +x)$ is sampled, the answers must be consistent; by the equivalence between inconsistency and distance for projective measurements (14), and the distance notion for PVMs in representation form (Claim 3.19), one deduces

$$\varepsilon_x \geq \mathop{\mathbb{P}}_{(a,b) \sim (\mathcal{U}^{+x}, \mathcal{U}^{-x})} [a \neq b] = 1/2 \cdot \mathop{\mathbb{E}}_{\alpha \colon S_x \to \mathbb{F}_2} \| \mathcal{U}^{+x}(\alpha) - \mathcal{U}^{-x}(\alpha) \|_{hs}^2 \,. \quad (37)$$

Let $\mathscr{S}' = \{\mathcal{U}'\}$ be the strategy that uses the observables of the positive side for both sides of the double cover, namely satisfy

$$\forall X \in S \colon \ \mathcal{U}'(\pm X) = \mathcal{U}(+X) \,. \quad (38)$$

Combining (36) and (37), we deduce that the distance between the strategy $\mathscr{S}$ and $\mathscr{S}'$ is at most $\frac{\varepsilon}{c^2}$. As close by strategies produce similar values (Claim 3.29), the value of $\mathscr{S}'$ is at least $1 - \varepsilon - 10\sqrt{\varepsilon}/c$. Moreover, it is straightforward to check that the strategy $\mathscr{S}'$ for $\mathfrak{DoubleCover}(\mathfrak{G})$ has the same value as the strategy $\mathscr{S}'' = \{\mathcal{U}''\}$ for $\mathfrak{G}$ that is defined by $\mathcal{U}''(X) = \mathcal{U}'(\pm X) = \mathcal{U}(+X)$, which proves the claim. □

Remark 3.55. As the double cover of a bipartite graph is just a disjoint union of two copies of the underlying graph, the double cover is the same game as the original one (with just two copies of the underlying graph instead of one). So, in this case, the double cover is complete and sound without any extra assumptions on self loops.

3.6 Non-synchronous strategies, values and entanglement lower bounds

When defining quantum strategies (Definition 2.18), we marked that our definition is commonly called in the literature "synchronous"; namely, our definition is some specialization of the more general notion of a quantum strategy, which is the topic of this subsection. This notion of "synchronicity" encapsulates three properties of the given strategy: The strategy is "projective", i.e., associates a projective measurement (PVM, Definition 2.1), and not the more general notion of a measurement (POVM), to every vertex in the game. The strategy is "maximally entangled", i.e., the state of the bipartite system on which the measurements are defined is the maximally entangled one. The strategy is "symmetric", i.e., the measurements the strategy associates with each vertex are the same on both sides of the bipartite system. Let us make this discussion formal.

Definition 3.56 (Measuring with respect to a general state. Compare to Definition 2.1)**.** Let $\mathcal{P}$ be an n-dimensional POVM with outcomes in A, and $\psi \in \mathbb{C}^n$ a unit vector. Recall also, from Remark 2.3, that $\langle u | v \rangle = u^* \cdot v = \sum_{i=1}^{n} \overline{u_i} \cdot v_i$ is the standard inner product on $\mathbb{C}^n$, where $\overline{(\cdot)}$ is the complex conjugate. Then, the probability distribution induced by $(\psi, \mathcal{P})$ is

$$\mathbb{P}[a \text{ is sampled}] := \psi^* \mathcal{P}_a \psi = \langle \psi | \mathcal{P}_a \psi \rangle \,.$$

Sampling $a \in A$ as above is often called "measuring according to $(\psi, \mathcal{P})$", and is denoted by $a \sim (\psi, \mathcal{P})$.

In a similar manner to Definition 2.2, given two n-dimensional POVMs, $\mathcal{P}$ with outcomes in A and $\mathcal{Q}$ with outcomes in B, the tensor product $\mathcal{P} \otimes \mathcal{Q}^T$ is a POVM with outcomes in $A \times B$, where $(\cdot)^T$ is the transposition of matrices. Given a unit vector $\psi \in \mathbb{C}^n \otimes \mathbb{C}^n$, we get the probability distribution

$$\mathbb{P}[a, b \text{ are sampled}] := \psi^* (\mathcal{P}_a \otimes \mathcal{Q}_b^T) \psi = \langle \psi | \mathcal{P}_a \otimes \mathcal{Q}_b^T \psi \rangle \,, \quad (39)$$

and again, we call this jointly sampling mechanism "measuring according to $(\psi, \mathcal{P}, \mathcal{Q})$", and denote it by $(a, b) \sim (\psi, \mathcal{P}, \mathcal{Q})$.

Claim 3.57. *Let $\mathcal{P}, \mathcal{Q}$ be n-dimensional POVMs as in Definition 3.56. Assume ψ is the maximally entangled state, namely, that $\psi = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} e_i \otimes e_i$, where $\{e_i\}$ is the standard basis of $\mathbb{C}^n$.[47] Then, jointly measuring according to $(\psi, \mathcal{P}, \mathcal{Q})$ as defined in (39) is the same as jointly measuring according to $(\mathcal{P}, \mathcal{Q})$ as in (5).*

Proof. This is immediate, because for the maximally entangled state ψ, $\psi^* P \otimes Q^T \psi$ is equal to $\tau(PQ)$ for any two $n \times n$ matrices P, Q. $\qquad\square$

Definition 3.58 (General quantum strategies. Compare to Definition 2.18)**.** Let $\mathfrak{G}$ be a (tailored) non local game with underlying graph $G = (V, E)$ and length function $\ell \colon V \to \mathbb{N}$. A (generalized) n-dimensional quantum strategy $\mathscr{S}$ consists of a unit vector $\psi \in \mathbb{C}^n \otimes \mathbb{C}^n$, together with two mappings $\mathcal{P}, \mathcal{Q}$, that given a vertex $\mathrm{x} \in V$, associate to it POVMs $\mathcal{P}^{\mathrm{x}}, \mathcal{Q}^{\mathrm{x}}$ acting on $\mathbb{C}^n$ and with outcomes in $\mathbb{F}_2^{\ell(\mathrm{x})}$. As in Remark 2.22, such a strategy induces a correlation

$$p_{\mathscr{S}}(a, b | \mathrm{x}, \mathrm{y}) = \psi^* \mathcal{P}_a^{\mathrm{x}} \otimes (\mathcal{Q}_b^{\mathrm{y}})^T \psi \,. \tag{40}$$

A generalized strategy $\mathscr{S} = (\psi, \mathcal{P}, \mathcal{Q})$ is called: *projective* if $\mathcal{P}^{\mathrm{x}}, \mathcal{Q}^{\mathrm{x}}$ are PVMs for every vertex x; *symmetric* if $\mathcal{P}^{\mathrm{x}} = (\mathcal{Q}^{\mathrm{x}})^T$ for every vertex x; *maximally entangled* if ψ is the maximally entangled state $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} e_i \otimes e_i$. A projective, symmetric, maximally entangled strategy is called *synchronous*.

The way we defined a game beforehand (Definition 2.16), there was a single generating set at each vertex, and thus when the edge sampled in the game was a loop xx, the answer γ was a bit string parametrized by S_{x} and not $S_{\mathrm{x}} \sqcup S_{\mathrm{x}}$. Once one allows general strategies, it is not clear how to decide about an answer γ in such a case, as $\psi^* \mathcal{P}_a^{\mathrm{x}} \otimes (\mathcal{Q}_b^{\mathrm{x}})^T \psi$ may be positive for $a \neq b$ — namely, there are two answers $a, b \colon S_{\mathrm{x}} \to \mathbb{F}_2$, which one should be γ? Though our discussion on double covers (Definition 3.52) was motivated by other constructions along this paper, the resolution to the aforementioned issue is in it. Usually in the literature, the (synchronous) game $\mathfrak{G}$ *is* its double cover, namely at each vertex there are two distinct sets of formal variables S_{x}^+ and S_{x}^- of size $\ell(\mathrm{x})$, and given that the sampled edge was xy the assignment γ is from $S_{\mathrm{x}}^+ \sqcup S_{\mathrm{y}}^-$ to $\mathbb{F}_2$ and not from $S_{\mathrm{x}} \sqcup S_{\mathrm{y}}$, and it is sampled to be ab with probability (40). Let us define this more general notion of a game.

Definition 3.59 (General game. Compare to Definition 2.16)**.** A general game $\mathfrak{G}$ consists of an underlying graph $G = (V, E)$, a length function $\ell \colon V \to \mathbb{N}$ (or two length $\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}}$ functions in the case of a tailored game), **two distinct** formal generators sets S_{x}^+ and S_{x}^- at each vertex $\mathrm{x} \in V$, a distribution μ over E, and for every $e = \mathrm{xy} \in E$ a decision predicate $D_{\mathrm{xy}} \colon \mathbb{F}_2^{S_{\mathrm{x}}^+} \times \mathbb{F}_2^{S_{\mathrm{y}}^-} \to \mathbb{F}_2$. Such a game is called *synchronous* if $D_{\mathrm{xx}}(ab) = 0$ whenever $a \neq b$.

Remark 3.60. Indeed, a general synchronous game as in Definition 3.59 is exactly the double cover (Definition 3.52) of a game $\mathfrak{G}$ as in Definition 2.16. So, when discussing general strategies, there is no reason to distinguish between the double cover and the game itself.

Definition 3.61 (Non-synchronous value of a general game, and non-synchronous entanglement. Compare to Definitions 2.21 and 2.52)**.** The value of a general strategy $\mathscr{S} = (\psi, \mathcal{P}, \mathcal{Q})$ in a general game $\mathfrak{G}$ (Definition 3.59) is the same as it was in Definition 2.21; the only difference is due to the way the correlation is induced by the strategy, namely using (40) instead of (1). Namely,

$$\mathrm{val}(\mathfrak{G}, \mathscr{S}) = \sum_{\mathrm{xy} \in E} \sum_{\substack{a \colon S_{\mathrm{x}}^+ \to \mathbb{F}_2 \\ b \colon S_{\mathrm{y}}^- \to \mathbb{F}_2}} \mu(\mathrm{xy}) D_{\mathrm{xy}}(ab) \cdot \psi^* \mathcal{P}_a^{\mathrm{x}} \otimes (\mathcal{Q}_b^{\mathrm{y}})^T \psi \,. \tag{41}$$

So, taking the supremum of the value of a general game over **general** strategies gives a new notion of a value which we call the *non-synchronous* value of $\mathfrak{G}$, and denote it by $\mathrm{val}^{\mathrm{non-sync}}(\mathfrak{G})$. In addition, let $\mathscr{E}^{\mathrm{non-sync}}(\mathfrak{G}, 1 - \varepsilon)$ be the smallest n such that there is a an n-dimensional **general** strategy with value of at least $1 - \varepsilon$. This quantity is the non-synchronous entanglement lower bound of $\mathfrak{G}$ (with parameter $1 - \varepsilon$).

---

[47]Note that the maximally entangled state is equal to $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} u_i^* \otimes u_i$ for **any** orthonormal basis $\{u_i\}$ of $\mathbb{C}^n$, and not only with respect to the standard basis — this is a useful fact which is often used in the analysis of measurements.

Remark 3.62. Given an n-dimensional quantum strategy $\mathscr{S} = \{\mathcal{P}\}$ as in Definition 2.18, one can define a general synchronous n-dimensional strategy $\mathscr{S}' = (\frac{1}{\sqrt{n}} \sum_{i=1}^{n} e_i \otimes e_i, \mathcal{P}, \mathcal{P}^T)$. This mapping 'embeds' our notion of a quantum strategy as a special case of general quantum strategies. By Claim 3.57, this mapping preserves the correlations induced by the appropriate strategies, and thus the value of $\mathscr{S}'$ versus $\mathfrak{G}$ is the same as that of $\mathscr{S}$.

Fact 3.63 (Translating non-synchronous bounds to synchronous bounds)**.** *Let $\mathfrak{G}$ be a general synchronous (tailored) game, such that in its underlying graph $G = (V, E)$, all loops $\mathtt{xx}$ for $\mathtt{x} \in V$ appear as edges in E. Assume in addition that there is some constant $c > 0$, such that for every $\mathtt{x} \in V$ we have*

$$\frac{\mu(\mathtt{xx})}{\mu(\mathtt{x})} \geq c \,, \tag{42}$$

where μ is the distribution over edges in $\mathfrak{G}$, and $\mu(\mathtt{x})$ is (as before) the marginal on vertices, namely $\mu(\mathtt{x}) = \sum_{\mathtt{y} \in V} \frac{\mu(\mathtt{xy}) + \mu(\mathtt{yx})}{2}$. Then, from every n-dimensional general strategy $\mathscr{S}$ for $\mathfrak{G}$ with value $1 - \varepsilon$, one can extract an n-dimensional synchronous strategy (i.e., projective, symmetric and maximally entangled) for $\mathfrak{G}$ with value of at least $1 - \mathrm{poly}(\varepsilon/c^2)$. This in particular says that $\mathscr{E}^{\mathrm{non-sync}}(\mathfrak{G}, 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{G}, 1 - \mathrm{poly}(\varepsilon/c^2))$.

Proof idea. By analysing the given strategy $\mathscr{S} = (\psi, \mathcal{P}, \mathcal{Q})$ with value $1 - \varepsilon$ in a similar fashion to the strategy that had high probability of winning in the double cover (cf. (36) and (37)), we can deduce that $\mathscr{S}$ is (ε/c^2)-self inconsistent — which is the generalized quantity of inconsistency between $\mathcal{P}^{\mathtt{x}}$ and $(\mathcal{Q}^{\mathtt{x}})^T$ (on average over all $\mathtt{x} \in V$, see [Vid22, Equation (4)]). Once the strategy has low self inconsistency, it is close to being projective as well as symmetric. Hence, using orthonormalization (Fact 3.21) and naive symmetrization (similar to the choice of $\mathcal{U}'$ in the soundness of the double cover (38)), we can perturb it to being projective and symmetric without enlarging the dimension. The fact that close by strategies provide close by value (even in the general setup, see [Vid22, Lemma 2.10]), means that the value degrades only by some polynomial in ε/c^2, as required. Once this is done, we are only left to make it maximally entangled. It turns out that this cannot be done naively — maybe it is genuinely far from a maximally entangled strategy. But, there is a convex combination of projective, symmetric maximally entangled strategies of dimension at most the dimension of $\mathscr{S}$ that is close (in terms of correlations produced) to it (see [Vid22, Corollary 3.3]). In particular, as the value of the game is linear, one of these strategies provides a value that is at most polynomial in ε/c^2 lower that that of $\mathscr{S}$, finishing the proof (see the paragraph immediately after [Vid22, Corollary 3.3]). $\qquad\square$

3.7 The Pauli group

The Pauli matrices $\mathbb{X}$, $\mathbb{Z}$ (Definition 3.66) are ubiquitous in quantum information theory; together with $\mathbb{Y} = i\mathbb{X}\mathbb{Z}$ and Id they form a linear basis of all observables that can be performed on a qubit, and $\mathbb{X}$, $\mathbb{Z}$ are generally interpreted as the observables associated with two fundamental incompatible degrees of freedom such as the angular momentum, along two orthogonal directions, of an electron, or the position and momentum of a particle (in the infinite-dimensional case).

It turns out that these matrices are characterized, among all 2-dimensional complex observables and up to a global unitary rotation, by the anti-commutation relation $\mathbb{X}\mathbb{Z} = -\mathbb{Z}\mathbb{X}$. In this section we take a (classic) group-theoretic perspective and introduce the generalized Pauli group acting on k qubits. This perspective will be used in the next section, where we introduce a nonlocal game that essentially forces any good strategy to make use of these matrices as observables — namely, it is a robust self-test (Definition 3.30) with the single optimal strategy being induced by the Pauli matrices.

The resulting non-local game, which we call the *generalized Pauli basis game* and is introduced in the next section (following [NV17, NW19, JNV$^+$21, dlS22b, CVY23]), will later enable us to modify the naive introspection game $\mathfrak{Intro}(\mathfrak{G})$ (Section 4.1) so as to force the pair of questions sampled by the strategy to conform to the question distribution μ of the game $\mathfrak{G}$.

Recall that $\mathbb{F}_2 = \{0, 1\}$ is the field with two elements, $\mathbb{F}_2^k$ is the k-dimensional vector space over $\mathbb{F}_2$, and $\langle \cdot, \cdot \rangle \colon \mathbb{F}_2^k \times$

$\mathbb{F}_2^k \to \mathbb{F}_2$ is the bilinear form

$$\forall v, w \in \mathbb{F}_2^k : \quad \langle v, w \rangle = \sum_{i=1}^{k} v_i w_i. \tag{43}$$

This bilinear form induces an isomorphism between $\mathbb{F}_2^k$ and its dual space, $(\mathbb{F}_2^k)^* = \{ f : \mathbb{F}_2^k \to \mathbb{F}_2 \mid f \text{ is linear} \}$, by defining $v \mapsto v^* = \langle v, \cdot \rangle$. Under this isomorphism, the standard basis $\{e_1, ..., e_k\}$ is dual to itself, namely

$$\forall i, j \in [k] : \quad e_i^*(e_j) = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases}$$

All of these choices allow us to think of $\mathbb{F}_2^k$ as column vectors, $(\mathbb{F}_2^k)^*$ as row vectors, the $*$ operation as transposition of matrices, and the bilinear form $\langle \cdot, \cdot \rangle$ as matrix multiplication between row and column vectors.

Definition 3.64. The *Pauli group acting on k qubits* (also known as the Weyl–Heisenberg group, or the k-dimensional Heisenberg group over $\mathbb{F}_2$) is the collection of triples $P_k = \{(v, w, a) \mid v, w \in \mathbb{F}_2^k, a \in \mathbb{F}_2\}$ with multiplication

$$\forall v, v', w, w' \in \mathbb{F}_2^k, \ a, a' \in \mathbb{F}_2 : \quad (v, w, a) \cdot (v', w', a') = (v + v', w + w', a + a' + \langle w, v' \rangle).$$

Remark 3.65. Note that $\{(v, 0, 0)\}$ and $\{(0, w, 0)\}$ are subgroups of P_k isomorphic to $\mathbb{F}_2^k$. We usually call them the X-subgroup and Z-subgroup for reasons that will soon be clear.

There is a faithful $\mathbb{F}_2$-representation of P_k as $(k+2) \times (k+2)$ matrices by mapping

$$(v, w, a) \mapsto \begin{pmatrix} 1 & w^* & a \\ \vec{0}_k & \mathrm{Id}_k & v \\ 0 & (\vec{0}_k)^* & 1 \end{pmatrix} \in GL_{k+2}(\mathbb{F}_2),$$

where $\vec{0}_k$ is the length k all zero column vector, and Id_k is the $k \times k$ identity matrix. In this guise, the group is commonly called the k-dimensional Heisenberg group over $\mathbb{F}_2$.

3.7.1 Complex representations of the Pauli group

The map $(v, w, a) \mapsto (v, w)$ is an epimorphism of P_k onto $\mathbb{F}_2^{2k}$. Hence, all complex irreducible representations of $\mathbb{F}_2^{2k}$ are also irreducible representations of P_k. There are 2^{2k} such 1-dimensional representations. It turns out P_k has only one extra irreducible representation of dimension 2^k, which we will describe shortly. Let $U(\mathcal{H})$ be the group of unitary operators acting on a Hilbert space $\mathcal{H}$.

Definition 3.66. The X and Z *Pauli matrices* are the following **signed permutation matrices**

$$\mathbb{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \ \mathbb{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \in U(\mathbb{C}^2).$$

By viewing $\mathbb{C}^2$ as $\mathbb{C}^{\mathbb{F}_2}$, and letting $\mathbf{1}_a$ be the indicator of $a \in \mathbb{F}_2$, we can see that $\mathbb{X}\mathbf{1}_a = \mathbf{1}_{a+1}$ and that $\mathbb{Z}\mathbf{1}_a = (-1)^a \mathbf{1}_a$. For every $v = (v_1, ..., v_k)$ and $w = (w_1, ..., w_k)$ in $\mathbb{F}_2^k$, let

$$\mathbb{X}^{\otimes v} = \bigotimes_{i=1}^{k} \mathbb{X}^{v_i}, \quad \mathbb{Z}^{\otimes w} = \bigotimes_{i=1}^{k} \mathbb{Z}^{w_i} \in U(\mathbb{C}^{2^k}),$$

where $\mathbb{X}^0 = \mathbb{Z}^0 = \mathrm{Id} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $\otimes$ is (again) the Kronecker tensor product of matrices. These matrices act naturally on $(\mathbb{C}^{\mathbb{F}_2})^{\otimes k} \cong \mathbb{C}^{\mathbb{F}_2^k}$ as follows. Let $\mathbf{1}_v \in \mathbb{C}^{\mathbb{F}_2^k}$ be the indicator function of $v \in \mathbb{F}_2^k$. Then

$$\forall v, w \in \mathbb{F}_2^k : \quad \mathbb{X}^{\otimes v} \mathbf{1}_w = \mathbf{1}_{w+v}, \quad \mathbb{Z}^{\otimes v} \mathbf{1}_w = (-1)^{\langle v, w \rangle} \mathbf{1}_w. \tag{44}$$

57

As the tensor product of signed permutation matrices is a signed permutation matrix, the matrices $\mathbb{X}^{\otimes v}$ and $\mathbb{Z}^{\otimes v}$ (and their products) are signed permutation matrices; the signed set on which they naturally act is the signed standard basis

$$Y_\pm = \{\pm \mathbf{1}_v \mid v \in \mathbb{F}_2^k\} .$$

See Figure 2 for a visualization of the action of these matrices in case $k = 3$.

Definition 3.67. Let $\mathscr{F}_v^{\mathbb{Z}}$ be the (orthogonal) projection on the 1-dimensional subspace in $\mathbb{C}^{\mathbb{F}_2^k}$ spanned by $\mathbf{1}_v$, and $\mathscr{F}_v^{\mathbb{X}}$ the (orthogonal) projection on the 1-dimensional subspace in $\mathbb{C}^{\mathbb{F}_2^k}$ spanned by $\sum_{w \in \mathbb{F}_2^k} (-1)^{\langle v,w \rangle} \mathbf{1}_w$.

Then, $\{\mathscr{F}_v^{\mathbb{X}}\}$ (respectively $\{\mathscr{F}_v^{\mathbb{Z}}\}$) is a PVM with outcomes in $\mathbb{F}_2^k$, and its observable form is $i \mapsto \mathbb{X}^{\otimes e_i}$ (respectively $i \mapsto \mathbb{Z}^{\otimes e_i}$) for $i \in [k]$. Moreover, if z is sampled according to $\{\mathscr{F}_v^{\mathbb{Z}}\}$ (or $\{\mathscr{F}_v^{\mathbb{X}}\}$), then it is a uniform bit string of length k.

Definition 3.68 (The unique non-commuting unitary irreducible representation of the Pauli group). The map $\rho \colon \mathrm{P}_k \to \mathrm{Sym}_\pm(Y) \subseteq U(2^k)$ defined by

$$\rho(v, w, a) = (-1)^a \mathbb{X}^{\otimes v} \mathbb{Z}^{\otimes w} \tag{45}$$

is a faithful irreducible signed permutation representation of P_k. In particular, given $v, v', w, w' \in \mathbb{F}_2^k$, we have

$$(\mathbb{X}^{\otimes v})^2 = \mathrm{Id}, \quad (\mathbb{Z}^{\otimes w})^2 = \mathrm{Id}, \quad \mathbb{X}^{\otimes v} \mathbb{X}^{\otimes v'} = \mathbb{X}^{\otimes v+v'}, \quad \mathbb{Z}^{\otimes w} \mathbb{Z}^{\otimes w'} = \mathbb{Z}^{\otimes w+w'}$$

and

$$\mathbb{X}^{\otimes v} \mathbb{Z}^{\otimes w} = (-1)^{\langle v,w \rangle} \mathbb{Z}^{\otimes w} \mathbb{X}^{\otimes v} .$$

Moreover, $\{\mathbb{X}^{\otimes v} \mid v \in \mathbb{F}_2^k\}$ and $\{\mathbb{Z}^{\otimes w} \mid w \in \mathbb{F}_2^k\}$ are isomorphic to the X and Z subgroups in P_k. For later use, we let $\rho^{\mathbb{Z}}$ and $\rho^{\mathbb{X}}$ be the restrictions of ρ to the Z and X subgroups, namely

$$\forall \alpha \in \mathbb{F}_2^k : \quad \rho^{\mathbb{Z}}(\alpha) = \mathbb{Z}^{\otimes \alpha} \quad \text{and} \quad \rho^{\mathbb{X}}(\alpha) = \mathbb{X}^{\otimes \alpha} . \tag{46}$$

Remark 3.69 (The $\mathscr{F}$-projections as inverse Fourier transform). As usual, the PVM $\{\mathscr{F}_v^{\mathbb{X}}\}$ (respectively $\{\mathscr{F}_v^{\mathbb{Z}}\}$) is the Fourier transform (Definition 2.4) of the representation $\rho^{\mathbb{X}}$ of $\mathbb{F}_2^k$ defined in (46) by $v \mapsto \mathbb{X}^{\otimes v}$ (respectively $\rho^{\mathbb{Z}}$ defined by $v \mapsto \mathbb{Z}^{\otimes v}$), namely

$$\forall v \in \mathbb{F}_2^k : \quad \mathscr{F}_v^{\mathbb{Z}} = \mathop{\mathbb{E}}_{w \in \mathbb{F}_2^k} \left[(-1)^{\langle w,v \rangle} \mathbb{Z}^{\otimes w} \right] \quad \text{and} \quad \mathbb{Z}^{\otimes v} = \sum_{w \in \mathbb{F}_2^k} (-1)^{\langle w,v \rangle} \mathscr{F}_w^{\mathbb{Z}} ,$$

and similarly for $\mathbb{X}$ and $\mathscr{F}_v^{\mathbb{X}}$.

Remark 3.70. The 2^k-dimensional representation ρ is the unique non-commuting irreducible representation of P_k (up to isomorphism). This is because there are 2^{2k} one-dimensional representations, and the squares of the dimensions of the irreducible representations of P_k should sum up to its order, which is 2^{2k+1}.

Corollary 3.71. *The representation $\rho^{\mathbb{X}}$ from (46) is a signed permutation PVM (Definition 2.15). The representation $\rho^{\mathbb{Z}}$ is a diagonal PVM (Definition 2.7). Finally, a vector $z \in \mathbb{F}_2^k$ sampled according to any of these PVMs (as was defined in (4)) is uniformly distributed.*

Figure 2: This is an example of the actions of the Pauli matrices on $Y_\pm$, which consists of all signed bit strings of length 3 in this case. The purple dashed lines are the actions of $-\mathrm{Id}$. The shades of green solid lines are the actions of the $\mathbb{X}$-generators for the standard basis elements. The shades of blue dotted lines are the actions of the $\mathbb{Z}$-generators. We intentionally did not include all actions of $\mathbb{Z}$-generators, for visual clarity.

3.7.2 Error correcting codes and stability of the Pauli group

Let $k \leq n$ be positive integers. Let C be a k-dimensional linear error correcting code of length n, i.e. a linear subspace $C \subseteq \mathbb{F}_2^n$ of dimension k. Let $E \in M_{n \times k}(\mathbb{F}_2)$ be a matrix whose columns span C. We call such matrices *encoding matrices*, since they induce an encoding of $\mathbb{F}_2^k$ as vectors in C via the mapping

$$\forall v \in \mathbb{F}_2^k : \quad v \mapsto Ev .$$

The *(Hamming) weight* of a vector $u = (u_1, ..., u_n) \in \mathbb{F}_2^n$ is the number of non-zero entries in it, namely

$$\omega_H(u) := |\{1 \leq i \leq n \mid u_i \neq 0\}| .$$

We say that C has *distance d* if

$$\forall 0 \neq c \in C : \quad \omega_H(c) \geq d .$$

All in all, C is called a (binary) *linear $[n, k, d]$-code*. Let $A \in M_{m \times n}(\mathbb{F}_2)$ be a matrix whose (right) kernel is C, namely

$$C = \{u \in \mathbb{F}_2^n \mid Au = 0\} .$$

Such matrices are called *parity check matrices* of C. Every ordered set $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$ defines an encoding matrix E by letting w^i be the ith row of E, namely $E_{ij} = w_j^i$. We refer to the image of E in this case as the *code induced by $\mathscr{B}$*.

For the purpose of this section we can use any binary linear code that has linear dimension and distance, and whose encoding matrix E can be efficiently constructed. The existence of such codes is guaranteed by the following well-known fact.

Fact 3.72. *For any $R \in (0,1)$ there is a $\delta > 0$ and a family of binary linear codes $(C_n)_{n \geq 1}$ of dimension $k = \lfloor Rn \rfloor$, length n, and distance $d \geq \delta n$ such that furthermore an encoding matrix E_n for C_n can be computed in time polynomial in n.*

Proof. An example construction is given by the Justesen codes [Jus72], which can be obtained from the concatenation of a Reed–Solomon code over $\mathbb{F}_q$ and a suitably chosen inner code. Better constructions are possible if one is interested in a specific range of (R, δ); for us it suffices that $\delta > 0$ can be guaranteed for any $R < 1$. $\square$

Fact 3.73 (Semi-stability of P_k, cf. Corollary 2.6 in [dlS22b]). *Let χ and ζ be two N-dimensional unitary representations of $\mathbb{F}_2^k$. Let $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$ be an ordered set which induces an $[n, k, d]$-code. Assume that χ and ζ satisfy the following "almost (anti-)commutation relations"*

$$\mathbb{E}_{i,j \in [n]} \left[\|\chi(w^i)\zeta(w^j) - (-1)^{\langle w^i, w^j \rangle} \zeta(w^j)\chi(w^i)\|_{hs}^2 \right] \leq \varepsilon , \tag{47}$$

where $\| \cdot \|_{hs}$ is the normalized Hilbert–Schmidt norm (Definition 3.1). Then, there exists an integer m and a $C(k/d)^2\varepsilon$-near bijection (Definition 3.4) $\omega \colon \mathbb{C}^N \to \mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$ for which χ is $C(k/d)^2\varepsilon$-close (Definition 3.10 and Claim 3.19) to $\omega^*(\rho^X \otimes \mathrm{Id}_m)\omega$ and similarly ζ is $C(k/d)^2\varepsilon$-close to $\omega^*(\rho^Z \otimes \mathrm{Id}_m)\omega$, where C is a universal constant (independent of any other parameter). Namely,

$$\mathbb{E}_{v \in \mathbb{F}_2^k}[\|\chi(v) - \omega^* \cdot X^{\otimes v} \otimes \mathrm{Id}_m \cdot \omega\|_{hs}^2] \leq C (k/d)^2 \varepsilon ,$$

$$\mathbb{E}_{v \in \mathbb{F}_2^k}[\|\zeta(v) - \omega^* \cdot Z^{\otimes v} \otimes \mathrm{Id}_m \cdot \omega\|_{hs}^2] \leq C (k/d)^2 \varepsilon ,$$

$$\tau(\mathrm{Id}_N - \omega^*\omega) , \ \tau(\mathrm{Id}_{2^k} \otimes \mathrm{Id}_m - \omega\omega^*) \leq C (k/d)^2 \varepsilon .$$

In words, any χ and ζ which almost satisfy the appropriate (anti-)commutation relations of P_k are close to (a direct sum of m copies of) the respective restrictions ρ^X, ρ^Z to the X and Z subgroups of the unique non-commuting representation ρ of P_k from Definition 3.68.

Remark 3.74. The proof of the Fact 3.73 is due to de la Salle [dlS22b]. It uses a combination of ideas. The first is a method of Natarajan–Vidick [NV18b] which translates anti-commutation to commutation. The second is a spectral gap argument, standard in the analysis of groups with property (T), that allows to translate almost invariance against a generating sets to almost invariance against the whole group — this is sometimes called the L^2-Poincare inequality of spectral expanders (cf. Theorem 13.9 in [HLW06]). Lastly, an "on average" version of a stability result of finite groups due to Gowers–Hatami [GH17] is used. Though this description may seem intimidating, all the ingredients are quite straightforward (see [CVY23] for more on this).

3.8 The generalized Pauli basis game

We can now describe the generalized Pauli basis game $\mathfrak{Pauli} \, \mathfrak{Basis}_k$. The version provided here is due to de la Salle [dlS22b]. A group theoretic perspective on the Pauli basis game (and the following generalization of it) appears in [CVY23].

In $\mathfrak{Pauli} \, \mathfrak{Basis}_k$, there are two special questions, $\mathtt{Pauli}_X$ and $\mathtt{Pauli}_Z$. Their length will be k, and we expect that a perfect strategy restricted to $S_{\mathtt{Pauli}_X}$ and $S_{\mathtt{Pauli}_Z}$ induces (up to isometry and direct sums) the unique non-commuting irreducible representation ρ of P_k defined in (45) — namely, it is a self test (Definition 3.30). Note that every strategy $\mathscr{S}$, when restricted to $S_{\mathtt{Pauli}_X}$ (or $S_{\mathtt{Pauli}_Z}$), is a representation of $\mathbb{F}_2^k$. These restrictions will play the role of χ and ζ in the semi-stability result in Fact 3.73. So, we need to find a way to force (47) to be satisfied with a small enough ε, namely for the commutator of $\chi(w^i)$ and $\zeta(w^j)$ to be ε-close, on average, to $(-1)^{\langle w^i, w^j \rangle}\mathrm{Id}$ — that will ensure that the Pauli basis game is robust (Definition 3.30).

Throughout this description, n and k are positive integers, $i, j \in [n]$, $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$ and $a, b \in [3]$. The game $\mathfrak{Pauli}\,\mathfrak{Basis}_k(\mathscr{B})$ is an augmented sum (Definitions 3.44 and 3.45) of $\mathfrak{C}^{i,j}$ — each of which is either a commutation game (Section 3.8.1) or a null-commutation game — and $\mathfrak{M}^{i,j}$ — each of which is either an anti-commutation game (Section 3.8.2) or a null-anti-commutation game.

Sub-Structure	Question	Variables (all unreadable)
Augmentation:	$\mathtt{Pauli}_{\mathsf{X}}$	$\{\mathsf{PX}^{\alpha}\}_{\alpha=1}^{k}$
	$\mathtt{Pauli}_{\mathsf{Z}}$	$\{\mathsf{PZ}^{\beta}\}_{\beta=1}^{k}$
	X^i	X^i
	Z^j	Z^j
(null-)Commutation game:	$\mathtt{First}^{i,j}$	$\mathsf{First}^{i,j}$
	$\mathtt{Second}^{i,j}$	$\mathsf{Second}^{i,j}$
	$\mathtt{Both}^{i,j}$	$\{\mathsf{Both}_1^{i,j}, \mathsf{Both}_2^{i,j}\}$
(null-)Anti-commutation game:	$\mathtt{var}_{ab}^{i,j}$	$\mathsf{Var}_{ab}^{i,j}$,
	$\mathtt{row}_a^{i,j}$	$\{\mathsf{Row}_{a1}^{i,j}, \mathsf{Row}_{a2}^{i,j}, \mathsf{Row}_{a3}^{i,j}\}$
	$\mathtt{col}_b^{i,j}$	$\{\mathsf{Col}_{1b}^{i,j}, \mathsf{Col}_{2b}^{i,j}, \mathsf{Col}_{3b}^{i,j}\}$

1. **Commutation**: The (null-)commutation game $\mathfrak{C}^{i,j}$ involves the questions $\mathtt{First}^{i,j}$, $\mathtt{Second}^{i,j}$ and $\mathtt{Both}^{i,j}$. If $\langle w^i, w^j \rangle = 0$, then it forces the observable of $\mathsf{First}^{i,j}$ to commute with the observable of $\mathsf{Second}^{i,j}$.

2. **Anti-commutation** The (null)-anti-commutation game $\mathfrak{M}^{i,j}$ involves the questions $\mathtt{var}_{ab}^{i,j}$, $\mathtt{row}_a^{i,j}$ and $\mathtt{col}_b^{i,j}$. If $\langle w^i, w^j \rangle = 1$, then it forces the observable of $\mathsf{Var}_{11}^{i,j}$ to anti-commute with the observable of $\mathsf{Var}_{22}^{i,j}$.

3. **Consistency of** X: The observable of X^i is forced to be consistent with the observables $\mathsf{First}^{i,j}$ and $\mathsf{Var}_{11}^{i,j}$ for all j.

4. **Consistency of** Z: The observable of Z^j is forced to be consistent with the observables $\mathsf{Second}^{i,j}$ and $\mathsf{Var}_{22}^{i,j}$ for all i.

5. **Linear conditions on** X: The observable of X^i is forced to be consistent with the observable of the product $\prod_{\alpha=1}^{k}(\mathsf{PX}^{\alpha})^{w_{\alpha}^i}$.

6. **Linear conditions on** Z: The observable of Z^j is forced to be consistent with the observable of the product $\prod_{\beta=1}^{k}(\mathsf{PZ}^{\beta})^{w_{\beta}^j}$.

Figure 3: Questions and answers in the generalized Pauli basis game $\mathfrak{Pauli}\,\mathfrak{Basis}_k(\mathscr{B})$. Since the game is tailored as an LCS, all answers are unreadable. We also list the conditions on a strategy's observables that the game enforces.

To that end, for every $i, j \in [n]$, there will be questions $\mathtt{X}^i$ and $\mathtt{Z}^j$ of length 1, whose observables are (expected to be) corresponding to $\chi(w^i)$ and $\zeta(w^j)$ respectively. This is achieved by a consistency check of $\mathtt{X}^i$'s vs. $\mathtt{Pauli_X}$ and $\mathtt{Z}^j$ vs. $\mathtt{Pauli_Z}$. Then, we check that the observables at the vertices $\mathtt{X}^i$ and $\mathtt{Z}^j$ (anti-)commute, according to whether $\langle w^i, w^j \rangle = 0$ or 1. This is done using "small" games that force either commutation or anti-commutation between two observables. See Figure 6 for a partial representation of the underlying graph of $\mathfrak{Pauli\ Basis}_k$.[48]

To implement this last step, we need a game that forces commutation, and a game that forces anti-commutation.

3.8.1 Commutation game

The commutation game $\mathfrak{C}$ has three questions (vertices) in its underlying graph: $\mathtt{First}, \mathtt{Second}$ and $\mathtt{Both}$. The vertex $\mathtt{First}$ is of length 1 and has the associated formal generator $\mathtt{First}$, the vertex $\mathtt{Second}$ is of length 1 and has the associated formal generator $\mathtt{Second}$, and the vertex $\mathtt{Both}$ is of length 2 and has associated formal generators $\mathtt{Both_1}, \mathtt{Both_2}$. The edges in the underlying graph are $\mathtt{First} - \mathtt{Both}$ and $\mathtt{Second} - \mathtt{Both}$. Then, $D_{\mathtt{First\ Both}}$ checks that $\gamma(\mathtt{First}) = \gamma(\mathtt{Both_1})$, and $D_{\mathtt{Second\ Both}}$ checks that $\gamma(\mathtt{Second}) = \gamma(\mathtt{Both_2})$. Note that this is a linear constraint system game, and thus can be tailored a la Example 2.29, in particular without readable variables. The distribution over edges is uniform. As described formally in the next fact, perfect strategies for this game imply commutation of observables, and almost perfect strategies imply almost commutation of observables.
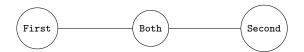

Figure 4: The underlying graph of the commutation game $\mathfrak{C}$.

Fact 3.75 (Completeness and soundness of the comutation game, cf. Lemma 3.5 in [dlS22b]). *If $\mathscr{S}$ is a perfect strategy for the commutation game $\mathfrak{C}$, and $\mathcal{U}$ is $\mathscr{S}$ in observable form, then $\mathcal{U}(\mathsf{First}), \mathcal{U}(\mathsf{Second})$ are commuting involutions, i.e.,*

$$\mathcal{U}(\mathsf{First})\mathcal{U}(\mathsf{Second}) = \mathcal{U}(\mathsf{Second})\mathcal{U}(\mathsf{First}) .$$

Moreover, if $\mathscr{S}$ has value $1 - \varepsilon$, then $\|\mathcal{U}(\mathsf{First})\mathcal{U}(\mathsf{Second}) - \mathcal{U}(\mathsf{Second})\mathcal{U}(\mathsf{First})\|_{hs}^2 \leq 64\varepsilon$.

Remark 3.76. Fact 3.75 can be deduced almost immediately from Claim 3.36.

Claim 3.77 (Extending commuting observables to perfect strategies). *Given two commuting involutions $O_1, O_2 \in U(n)$, there is a perfect strategy $\mathscr{S} = \{\mathcal{U}\}$ for $\mathfrak{C}$ that commutes along edges such that $\mathcal{U}(\mathsf{First}) = O_1$ and $\mathcal{U}(\mathsf{Second}) = O_2$.*

Proof. We are left to define the observables associated to the $\mathtt{Both}$ vertex. For the strategy to be perfect, they need to be consistent with the observables at the other vertices, so we are forced to let $\mathcal{U}(\mathsf{Both_1}) = O_1$ and $\mathcal{U}(\mathsf{Both_2}) = O_2$. This is a well defined strategy, as indeed the observables at $\mathtt{Both}$ are commuting (by assumption) which induces a PVM in observable form at this vertex. $\square$

Let the *null-commutation game* $\mathfrak{C}_{null}$ be the game whose underlying graph, length functions and sets of formal variables are the same as in the commutation game, but it always accepts. This is also a linear constraint system game.

---

[48]Note that for every $\mathtt{X}^i, \mathtt{Z}^j$ there is both a commutation game and an anti-commutation game attached to them. As we see later, the irrelevant one will be ignored. This is a quirk of the way compression works: The running time of the question reduced verifier needs to be exponentially faster, but calculating $\langle w^i, w^j \rangle$ may take a long time. Thus, we delegate this check to the decision process — i.e., linear constraints processor combined with the canonical decider — (which may still run in the original running time), and let it a posteriori ignore irrelevant (anti-)commutation checks that are not part of the presentation of P_k.

3.8.2 Anti-commutation game

We have already seen the anti-commutation game: The magic square game from Example 2.30. Again, we note that this game is an LCS, and thus can be tailored such that all variables are unreadable. As the next fact shows, it has the property that in a perfect strategy the observables of Var_{11} and Var_{22} anti-commute, and in an almost perfect strategy they almost anti-commute.

Figure 5: The underlying graph of the anti-commutation game $\mathfrak{M}$.

Fact 3.78 (Completeness and soundness of the magic square game, cf. Lemma 3.6 in [dlS22b]). *Let $\mathfrak{M}$ be the magic square game from Example 2.30. Let $\mathscr{S}$ be a perfect strategy for $\mathfrak{M}$, and let $\mathcal{U}$ be $\mathscr{S}$ in observable form. Then $\mathcal{U}(\mathsf{Var}_{11})$ and $\mathcal{U}(\mathsf{Var}_{22})$ are anti-commuting involutions, namely*

$$\mathcal{U}(\mathsf{Var}_{11})\mathcal{U}(\mathsf{Var}_{22}) = -\mathcal{U}(\mathsf{Var}_{22})\mathcal{U}(\mathsf{Var}_{11}) \, .$$

Moreover, if $\mathscr{S}$ has value $1 - \varepsilon$, then

$$\|\mathcal{U}(\mathsf{Var}_{11})\mathcal{U}(\mathsf{Var}_{22}) + \mathcal{U}(\mathsf{Var}_{22})\mathcal{U}(\mathsf{Var}_{11})\|_{hs}^2 \leq 432\varepsilon \, .$$

Remark 3.79. Fact 3.78 can also be deduced almost immediately from Claim 3.36.

Claim 3.80 (Extending anti-commuting observables to perfect strategies). *Let $O_{11}, O_{12}, O_{21}, O_{22} \in U(n)$ be involutions satisfying the following four commutation conditions*

$$O_{11}O_{12} = O_{12}O_{11} , \ O_{11}O_{21} = O_{21}O_{11} , \ O_{22}O_{12} = O_{12}O_{22} , \ O_{22}O_{21} = O_{21}O_{22} ,$$

as well as the following two anti-commutation conditions

$$O_{11}O_{22} = -O_{22}O_{11} , \ O_{12}O_{21} = -O_{21}O_{12} .$$

Then, there exists a perfect strategy $\mathscr{S} = \{\mathcal{U}\}$ for the magic square game $\mathfrak{M}$ satisfying for every $a, b \in \{1, 2\}$ that $\mathcal{U}(\mathsf{Var}_{ab}) = O_{ab}$.

Proof. For every $a, b \in [3]$, let $\mathcal{U}(\mathsf{Var}_{ab}) = \mathcal{U}(\mathsf{Row}_{ab}) = \mathcal{U}(\mathsf{Col}_{ab})$ be the ab^{th} entry in Table 1. We leave it to the reader to

O_{11}	O_{12}	$O_{11}O_{12}$
O_{21}	O_{22}	$O_{21}O_{22}$
$-O_{11}O_{21}$	$-O_{21}O_{22}$	$-O_{11}O_{12}O_{21}O_{22}$

Table 1: Perfect strategy for $\mathfrak{M}$ induced by the quadruple $O_{11}, O_{12}, O_{21}, O_{22}$.

verify that this is a well defined, perfect strategy that commutes along edges. We encourage the reader to compare the above general strategy to the one we described in Example 2.30. $\quad\square$

Let the *null-anti-commutation game* $\mathfrak{M}_{null}$ be the game whose underlying graph, length functions and sets of formal variables are the same as in the magic square game, but it always accepts. This is also a linear constraint system game.

3.8.3 Pauli basis game — See Figure 3 for a summary

For any set $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$ we define an appropriate Pauli basis game $\mathfrak{Pauli \ Basis}_k = \mathfrak{Pauli \ Basis}_k(\mathscr{B})$. For every i and j in $[n]$, we let $\mathfrak{C}^{i,j}$ be either a copy of the commutation game $\mathfrak{C}$ (Section 3.8.1) or the null-commutation game $\mathfrak{C}_{null}$: It will be a copy of $\mathfrak{C}$ in case $\mathbb{X}^{w^i}$ should commute with $\mathbb{Z}^{w^j}$, namely when $\langle w^i, w^j \rangle = 0$, and $\mathfrak{C}_{null}$ otherwise. Similarly, we let $\mathfrak{M}^{i,j}$ be either a copy of the anti-commutation game $\mathfrak{M}$ (Section 3.8.2) or the null-anti-commutation game $\mathfrak{M}_{null}$: It will be a copy of $\mathfrak{M}$ if $\mathbb{X}^{w^i}$ should anti-commute with $\mathbb{Z}^{w^j}$, namely when $\langle w^i, w^j \rangle = 1$, and $\mathfrak{M}_{null}$ otherwise. For clarity of notation, the vertices in $\mathfrak{C}^{i,j}$ will be $\mathtt{First}^{i,j}, \mathtt{Second}^{i,j}$ and $\mathtt{Both}^{i,j}$, while the vertices in $\mathfrak{M}^{i,j}$ will be $\{\mathtt{var}_{ab}^{i,j}\}_{a,b=1}^3$, $\{\mathtt{row}_a^{i,j}\}_{a=1}^3$ and $\{\mathtt{col}_b^{i,j}\}_{b=1}^3$ and they are connected as in Figures 4 and 5 respectively.

The Pauli basis game $\mathfrak{Pauli \ Basis}_k$ is an augmented sum (Definitions 3.44 and 3.45) of the $2n^2$ games $\mathfrak{C}^{i,j}$ and $\mathfrak{M}^{i,j}$. It is augmented with $2n + 2$ extra vertices — $\{\mathtt{X}^i\}_{i=1}^n, \{\mathtt{Z}^j\}_{j=1}^n, \mathtt{Pauli}_{\mathbb{X}}$ and $\mathtt{Pauli}_{\mathbb{Z}}$. The lengths of $\mathtt{X}^i$ and $\mathtt{Z}^j$ are 1 with associated generators X^i and Z^j, while $\mathtt{Pauli}_{\mathbb{X}}$ and $\mathtt{Pauli}_{\mathbb{Z}}$ have length k with associated generators $\{\mathsf{PX}^i\}_{i=1}^k$ and $\{\mathsf{PZ}^j\}_{j=1}^k$. The vertex $\mathtt{X}^i$ is connected to $\mathtt{Pauli}_{\mathbb{X}}, \mathtt{First}^{i,j}$ for every $j \in [n]$, and $\mathtt{var}_{11}^{i,j}$ for every $j \in [n]$. The vertex $\mathtt{Z}^j$ is connected to $\mathtt{Pauli}_{\mathbb{Z}}, \mathtt{Second}^{i,j}$ for every $i \in [n]$ and $\mathtt{var}_{22}^{i,j}$ for every $i \in [n]$ (see Figure 6 for a partial view).

Now, if an edge within $\mathfrak{C}^{i,j}$ or $\mathfrak{M}^{i,j}$ is sampled, the decision procedure is already defined. When $\mathtt{X}^i$ (respectively $\mathtt{Z}^j$) is sampled against $\mathtt{First}^{i,j}$ (respectively $\mathtt{Second}^{i,j}$), we check consistency between their values, namely $\gamma(\mathsf{X}^i) = \gamma(\mathsf{First}^{i,j})$ (respectively $\gamma(\mathsf{Z}^j) = \gamma(\mathsf{Second}^{i,j})$). Similarly, when $\mathtt{X}^i$ (respectively $\mathtt{Z}^j$) is sampled against $\mathtt{var}_{11}^{i,j}$ (respectively $\mathtt{var}_{22}^{i,j}$), we check consistency between their values, namely $\gamma(\mathsf{X}^i) = \gamma(\mathsf{Var}_{11}^{i,j})$ (respectively $\gamma(\mathsf{Z}^j) = \gamma(\mathsf{Var}_{22}^{i,j})$). Lastly, if $\mathtt{X}^i$ (respectively $\mathtt{Z}^j$) is sampled against $\mathtt{Pauli}_{\mathbb{X}}$ (respectively $\mathtt{Pauli}_{\mathbb{Z}}$), then we check that $\gamma(\mathsf{X}^i) = \sum_{j=1}^k w_j^i \gamma(\mathsf{PX}^j)$ (respectively $\gamma(\mathsf{Z}^j) = \sum_{i=1}^k w_i^j \gamma(\mathsf{PZ}^i)$).

Note that this is an LCS game, as was defined in Example 2.29. In particular, by tailoring it as described in the aforementioned example, all variables are linear, and the linear constraint processor is implicitly defined by the decision procedure above.

Figure 6: This is a partial picture of the underlying graph of $\mathfrak{Pauli\ Basis}_k$. Note that the nodes $\mathfrak{C}^{i,j}$ and $\mathfrak{M}^{i,j}$ are not single vertices in the graph, but some (constant sized) subgraphs associated with the commutation 3.8.1 and anti-commutation 3.8.2 games (respectively). The $\mathtt{X}^i$ and $\mathtt{Z}^j$ vertices are attached to $\mathfrak{C}^{i,j}$ and $\mathfrak{M}^{i,j}$ in a specific way: $\mathtt{X}^i$ is connected to $\mathtt{First}^{i,j}$ in $\mathfrak{C}^{i,j}$ and to $\mathtt{var}_{11}^{i,j}$ in $\mathfrak{M}^{i,j}$, while $\mathtt{Z}^j$ is connected to $\mathtt{Second}^{i,j}$ in $\mathfrak{C}^{i,j}$ and to $\mathtt{var}_{22}^{i,j}$ in $\mathfrak{M}^{i,j}$. There is a commutation (or null commutation) and anti-commutation (or null anti-commutation) game between every $\mathtt{X}^i$ and $\mathtt{Z}^j$, but we have only drawn the local picture for the pairs $(\mathtt{X}^1, \mathtt{Z}^1)$, $(\mathtt{X}^2, \mathtt{Z}^{n-1})$ and $(\mathtt{X}^n, \mathtt{Z}^n)$.

Finally, we need to describe the distribution used in $\mathfrak{Pauli\ Basis}_k$. For now,[49] let us assume it is the following — with probability $1/8$ do one of the following: sample a uniform edge from a uniformly random $\mathfrak{M}^{i,j}$; sample a uniform edge from a uniformly random $\mathfrak{C}^{i,j}$; sample a uniform edge of the form $\mathtt{X}^i - \mathtt{First}^{i,j}$; sample a uniform edge of the form $\mathtt{Z}^j - \mathtt{Second}^{i,j}$; sample a uniform edge of the form $\mathtt{X}^i - \mathtt{Var}_{11}^{i,j}$; sample a uniform edge of the form $\mathtt{Z}^j - \mathtt{Var}_{22}^{i,j}$; sample a uniform edge of the

---

[49]The distribution we actually use needs to be induced by a conditionally linear sampling scheme (Definition 4.16). See Example 4.39 for the actual distribution we use. We note that in the resulting distribution, the probability each edge is sampled is at least some constant times the distribution we provided here. So, all of our arguments, which anyway use the asymptotic $O(\cdot)$-notation, stay the same.

form $\mathtt{X}^i - \mathtt{Pauli_X}$; sample a uniform edge of the form $\mathtt{Z}^j - \mathtt{Pauli_Z}$.

Remark 3.81. Let us briefly motivate the structure and checks of $\mathfrak{Pauli\ Basis}_k$. As discussed in Definition 2.1, every strategy $\mathscr{S} = \{\mathcal{U}\}$ induces two representations of $\mathbb{F}_2^k$ — $\chi = \mathcal{U}^{\mathtt{Pauli_X}}$ associated with the image of $S_{\mathtt{Pauli_X}}$ and $\zeta = \mathcal{U}^{\mathtt{Pauli_Z}}$ associated with the image of $S_{\mathtt{Pauli_Z}}$. The check $\mathtt{X}^i - \mathtt{Pauli_X}$ forces $\mathscr{S}$ to satisfy $\mathcal{U}(\mathtt{X}^i) = \chi(w^i)$, and the check $\mathtt{Z}^j - \mathtt{Pauli_Z}$ forces $\mathscr{S}$ to satisfy $\mathcal{U}(\mathtt{Z}^j) = \zeta(w^j)$. Then, for i,j such that $\langle w^i, w^j \rangle = 0$, the consistency checks $\mathtt{X}^i - \mathtt{First}^{i,j}$ and $\mathtt{Z}^j - \mathtt{Second}^{i,j}$ together with running $\mathfrak{C}^{i,j}$ forces $\mathscr{S}$ to satisfy $\chi(w^i)\zeta(w^j) = \zeta(w^j)\chi(w^i)$. Finally, for i,j such that $\langle w^i, w^j \rangle = 1$, the consistency checks $\mathtt{X}^i - \mathtt{var}_{11}^{i,j}$ and $\mathtt{Z}^j - \mathtt{var}_{22}^{i,j}$ together with running $\mathfrak{M}^{i,j}$ forces $\mathscr{S}$ to satisfy $\chi(w^i)\zeta(w^j) = -\zeta(w^j)\chi(w^i)$. Hence, by taking all product of images of χ and ζ, we get a representation of the Pauli group P_k, and since some of the images anti-commute, all irreducible components of this representation are copies of the unique non-commuting representation ρ defined in (45) — which was our goal.

Claim 3.82 (Completeness of the Pauli basis game). *Let m and k be positive integers with $k \geq 2$. Let $\rho: P_k \to U(\mathbb{C}^{\mathbb{F}_2^k})$ be the representation of the Pauli group acting on k qubits defined in (45), and let $\mathcal{U}$ be a unitary in $U(\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m)$. Then, there is a perfect strategy that commutes along edges $\mathscr{S} = \{\mathcal{U}\}$ for the Pauli basis game $\mathfrak{Pauli\ Basis}_k$ such that the representations $\mathcal{U}^{\mathtt{Pauli_X}}$ and $\mathcal{U}^{\mathtt{Pauli_Z}}$, which $\mathscr{S}$ associates to the vertices $\mathtt{Pauli_X}$ and $\mathtt{Pauli_Z}$, are $\mathcal{U}^{-1}\rho^{\mathbb{X}} \otimes \mathrm{Id}_m \mathcal{U}$ and $\mathcal{U}^{-1}\rho^{\mathbb{X}} \otimes \mathrm{Id}_m \mathcal{U}$ respectively (see (46) in Definition 3.68). In particular, if $\mathcal{U}$ is the identity, then this strategy is a permutation strategy, and the images $\mathcal{U}^{\mathtt{Pauli_Z}}$ are diagonal in the standard basis.*

Proof sketch. We mainly follow the restrictions of the Pauli basis game, as described in Remark 3.81. We are forced, by the claim, to let

$$\forall v, w \in \mathbb{F}_2^k: \ \mathcal{U}^{\mathtt{Pauli_X}}(v) = \mathcal{U}^{-1}(\mathbb{X}^{\otimes v} \otimes \mathrm{Id}_m)\mathcal{U} \ , \ \mathcal{U}^{\mathtt{Pauli_Z}}(w) = \mathcal{U}^{-1}(\mathbb{Z}^{\otimes w} \otimes \mathrm{Id}_m)\mathcal{U} \ .$$

By claim 3.36, for the checks incident to the $\mathtt{X}^i$ and $\mathtt{Z}^j$ vertices to perfectly be satisfied, we need

$$\mathcal{U}(\mathtt{X}^i) = \mathcal{U}(\mathsf{First}^{i,j}) = \mathcal{U}(\mathsf{Var}_{11}^{i,j}) = \mathcal{U}^{-1}(\mathbb{X}^{\otimes w^i} \otimes \mathrm{Id}_m)\mathcal{U} \ , \ \mathcal{U}(\mathtt{Z}^i) = \mathcal{U}(\mathsf{Second}^{i,j}) = \mathcal{U}(\mathsf{Var}_{22}^{i,j}) = \mathcal{U}^{-1}(\mathbb{Z}^{\otimes w^j} \otimes \mathrm{Id}_m)\mathcal{U} \ ,$$

where w^i and w^j are the ith and j^{th} vectors from the fixed set $\mathscr{B}$. Now, we can use Claim 3.77 to extend $\mathcal{U}$ to $\mathsf{Both}_1^{i,j}$ and $\mathsf{Both}_2^{i,j}$ in case $\mathcal{U}(\mathtt{X}^i)$ and $\mathcal{U}(\mathtt{Z}^j)$ commute, which is exactly the case where $\langle w^i, w^j \rangle = 0$; otherwise, we let $\mathcal{U}(\mathsf{Both}_1^{i,j}) = \mathcal{U}(\mathsf{Both}_2^{i,j}) = \mathrm{Id}$. This verifies that indeed $\mathcal{U}$ is perfect when restricted to the (null-)commutation games $\mathfrak{C}^{i,j}$. For the (null-)anti-commutation games $\mathfrak{M}^{i,j}$ we have more flexibility, as a perfect strategy for them requires a quadruple of observables and we fixed only two, namely $\mathcal{U}(\mathsf{Var}_{11}^{i,j})$ and $\mathcal{U}(\mathsf{Var}_{22}^{i,j})$. In case $\langle w^i, w^j \rangle = 1$, we fix a pair of vectors $v_1^{i,j}, v_2^{i,j} \in \mathbb{F}_2^k$ satisfying $\langle v_1^{i,j}, w^j \rangle = 0$, $\langle v_1^{i,j}, v_2^{i,j} \rangle = 1$ and $\langle w^i, v_2^{i,j} \rangle = 0$ — we leave it to the reader to check that such vectors exist whenever $k \geq 2$. Then, the quadruple

$$\mathcal{U}(\mathtt{X}^i) \ , \ \mathcal{U}^{-1}(\mathbb{X}^{v_1^{i,j}} \otimes \mathrm{Id}_m)\mathcal{U} \ , \ \mathcal{U}^{-1}(\mathbb{Z}^{v_2^{i,j}} \otimes \mathrm{Id}_m)\mathcal{U} \ , \ \mathcal{U}(\mathtt{Z}^j) \ ,$$

satisfies the conditions of Claim 3.80, and can thus be extended to a perfect strategy for $\mathfrak{M}^{i,j}$. In case $\langle w^i, w^j \rangle = 0$, $\mathfrak{M}^{i,j}$ is a null-anti-commutaion game, and we can thus extend $\mathcal{U}$ to it such that it is Id for all variables not yet defined.

In case $\mathcal{U}$ is the identity, the image of $\mathcal{U}$ consists of products and tensor products of signed permutation matrices, and is thus a permutation strategy. In particular, $\mathcal{U}^{\mathtt{Pauli_Z}} = \rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$ is a diagonal representation, as needed. $\qquad\square$

Claim 3.83 (Characterization of almost-perfect strategies of the Pauli basis game). *Let $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$ be an ordered set, and let $\mathfrak{Pauli\ Basis}_k = \mathfrak{Pauli\ Basis}_k(\mathscr{B})$ be the appropriate generalized Pauli basis game. Let $\mathscr{S}$ be an N-dimensional strategy satisfying $\mathrm{val}(\mathfrak{Pauli\ Basis}_k, \mathscr{S}) \geq 1 - \varepsilon$, and $\mathcal{U}$ be $\mathscr{S}$ in observable (and representation) form. Then, for some universal constant $C > 0$, we have that $\mathcal{U}^{\mathtt{Pauli_X}}(w^i)\mathcal{U}^{\mathtt{Pauli_Z}}(w^j)$ is $C\varepsilon$-close to $(-1)^{\langle w^i, w^j \rangle}\mathcal{U}^{\mathtt{Pauli_Z}}(w^j)\mathcal{U}^{\mathtt{Pauli_X}}(w^i)$ on average over uniform $i, j \in [n]$; namely*

$$\mathop{\mathbb{E}}_{i,j \in [n]} \left[\left\| \mathcal{U}^{\mathtt{Pauli_X}}(w^i)\mathcal{U}^{\mathtt{Pauli_Z}}(w^j) - (-1)^{\langle w^i, w^j \rangle}\mathcal{U}^{\mathtt{Pauli_Z}}(w^j)\mathcal{U}^{\mathtt{Pauli_X}}(w^i) \right\|_{hs}^2 \right] \leq C\varepsilon \ .$$

Proof sketch. By the fact that $\mathscr{S}$ passes the game with probability of at least $1 - \varepsilon$, and the collection of edges of type $\mathtt{Pauli}_{\mathbb{X}} - \mathtt{X}^i$ and $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Z}^j$ have a constant probability of being sampled under the game distribution, $\mathscr{S}$ passes a uniformly random edges of this type with probability of at least $1 - O(\varepsilon)$. Hence, $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}(w^i)$ is $O(\varepsilon)$-inconsistent (Definition 3.10) with $\mathcal{U}(\mathtt{X}^i)$ on average over $i \in [n]$, and similarly that $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}(w^j)$ is $O(\varepsilon)$-inconsistent with $\mathcal{U}(\mathtt{Z}^j)$ on average over $j \in [n]$. Hence, by Proposition 3.12 and Claim 3.19, we have[50]

$$\mathbb{E}_{i \in [n]} \left[\|\mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}(w^i) - \mathcal{U}(\mathtt{X}^i)\|_{hs}^2 \right] \;, \quad \mathbb{E}_{j \in [n]} \left[\|\mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}(w^j) - \mathcal{U}(\mathtt{Z}^j)\|_{hs}^2 \right] \;\leq O(\varepsilon) \;.$$

Since $\mathscr{S}$ passes the edges of type $\mathtt{X}^i - \mathtt{First}^{i,j}$, $\mathtt{X}^i - \mathtt{var}_{11}^{i,j}$, $\mathtt{Z}^j - \mathtt{Second}^{i,j}$ and $\mathtt{Z}^j - \mathtt{var}_{22}^{i,j}$ in $\mathfrak{Pauli\,Basis}_k$ with probability $1 - O(\varepsilon)$ (on average over uniform pairs $i, j \in [n]$), we can deduce by Claim 3.36 that

$$\mathbb{E}_{i,j \in [n]} \left[\|\mathcal{U}(\mathsf{First}^{i,j}) - \mathcal{U}(\mathtt{X}^i)\|_{hs}^2 \right] \;, \quad \mathbb{E}_{i,j \in [n]} \left[\|\mathcal{U}(\mathsf{Second}^{i,j}) - \mathcal{U}(\mathtt{Z}^j)\|_{hs}^2 \right] \;\leq O(\varepsilon) \;,$$

$$\mathbb{E}_{i,j \in [n]} \left[\|\mathcal{U}(\mathsf{Var}_{11}^{i,j}) - \mathcal{U}(\mathtt{X}^i)\|_{hs}^2 \right] \;, \quad \mathbb{E}_{i,j \in [n]} \left[\|\mathcal{U}(\mathsf{Var}_{22}^{i,j}) - \mathcal{U}(\mathtt{Z}^j)\|_{hs}^2 \right] \;\leq O(\varepsilon) \;.$$

Since $\mathscr{S}$ passes the copies of the commutation games $\mathfrak{C}^{i,j}$ and anti-commutation games $\mathfrak{M}^{i,j}$ in $\mathfrak{Pauli\,Basis}_k$ with probability $1 - O(\varepsilon)$ (on average over uniform $i, j \in [n]$), we can deduce using Facts 3.75 and 3.78 that

$$\mathbb{P}_{i,j \in [n]}[\langle w^i, w^j \rangle = 0] \cdot \mathbb{E}_{i,j \in [n]\,:\,\langle w^i,w^j\rangle=0} \left[\|\mathcal{U}(\mathsf{First}^{i,j})\mathcal{U}(\mathsf{Second}^{i,j}) - \mathcal{U}(\mathsf{Second}^{i,j})\mathcal{U}(\mathsf{First}^{i,j})\|_{hs}^2 \right] \leq O(\varepsilon) \;,$$

$$\mathbb{P}_{i,j \in [n]}[\langle w^i, w^j \rangle = 1] \cdot \mathbb{E}_{i,j \in [n]\,:\,\langle w^i,w^j\rangle=1} \left[\|\mathcal{U}(\mathsf{Var}_{11}^{i,j})\mathcal{U}(\mathsf{Var}_{22}^{i,j}) + \mathcal{U}(\mathsf{Var}_{22}^{i,j})\mathcal{U}(\mathsf{Var}_{11}^{i,j})\|_{hs}^2 \right] \leq O(\varepsilon) \;.$$

By combining all of the above observations, the claim is deduced. $\qquad\square$

Corollary 3.84 (The Pauli basis game is a (semi)-robust self test). *For every N-dimensional strategy $\mathscr{S} = \{\mathcal{U}\}$ for the Pauli basis game $\mathfrak{Pauli\,Basis}_k$ with value $1 - \varepsilon$, there is a perfect strategy $\mathscr{S}' = \{\mathcal{V}\}$ for the game such that:*

1. *The representations $\mathcal{V}^{\mathtt{Pauli}_{\mathbb{X}}}$ and $\mathcal{V}^{\mathtt{Pauli}_{\mathbb{Z}}}$ are respective direct sums of $\rho^{\mathbb{X}}$ and $\rho^{\mathbb{Z}}$ from (46). Namely, there is a positive integer m such that*

$$\forall v, w \in \mathbb{F}_2^k :\quad \mathcal{V}^{\mathtt{Pauli}_{\mathbb{X}}}(v) = \mathbb{X}^{\otimes v} \otimes \mathrm{Id}_m \quad \text{and} \quad \mathcal{V}^{\mathtt{Pauli}_{\mathbb{Z}}}(w) = \mathbb{Z}^{\otimes w} \otimes \mathrm{Id}_m \;.$$

2. *The representations $\mathcal{V}^{\mathtt{Pauli}_{\mathbb{X}}}$ and $\mathcal{V}^{\mathtt{Pauli}_{\mathbb{Z}}}$ are $O(\varepsilon \cdot k^2/d^2)$-flexibly-close to $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}$ and $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}$ respectively. Namely, there is a universal constant $C > 0$, and a $(C \cdot \varepsilon \cdot k^2/d^2)$-near bijection $\omega \colon \mathbb{C}^N \to \mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$ such that*

$$\mathbb{E}_{v \in \mathbb{F}_2^k} \left[\|\mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}(v) - \omega^* \mathcal{V}^{\mathtt{Pauli}_{\mathbb{X}}}(v)\omega\|_{hs}^2 \right] \;, \quad \mathbb{E}_{w \in \mathbb{F}_2^k} \left[\|\mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}(w) - \omega^* \mathcal{V}^{\mathtt{Pauli}_{\mathbb{Z}}}(w)\omega\|_{hs}^2 \right] \leq C \cdot \varepsilon \cdot k^2/d^2 \;.$$

Proof. By the characterization of almost perfect strategies for $\mathfrak{Pauli\,Basis}_k$ (Claim 3.83), $\chi = \mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}$ and $\zeta = \mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}$ satisfy the conditions of the semi-stability result for P_k (Fact 3.73). Applying the semi-stability result provides a near bijection ω so that conjugating $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{X}}}$ and $\mathcal{U}^{\mathtt{Pauli}_{\mathbb{Z}}}$ by it brings them close to $\rho^{\mathbb{X}} \otimes \mathrm{Id}_m$ and $\rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$ respectively. Finally, Claim 3.82 says that this representation $\rho \otimes \mathrm{Id}_m$ can be extended to a perfect strategy for $\mathfrak{Pauli\,Basis}_k$, which we denote by $\mathcal{V}$. $\quad\square$

---

[50]This can also be deduced from Claim 3.36.

4 Question reduction via introspection

The goal of this section is to devise an algorithm QuestionReduction that takes as input a tailored normal form verifier and outputs a new tailored normal form verifier whose n^{th} game *simulates* the $(2^n)^{\text{th}}$ game of the original verifier. Though this new verifier is not as time efficient as needed for compression (Theorem 2.53), its sampling procedure is. Recall the asymptotic notation from Remark 1.2.

Theorem 4.1 (Informal Question Reduction, see Theorem 4.36 for the formal version)**.** *There exists a polynomial time 2-input Turing machine* QuestionReduction *that takes as input a TNFV* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *and a positive integer* λ*, and outputs a TNFV*

$$\text{QuestionReduction}(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}', \mathcal{A}', \mathcal{L}', \mathcal{D}),$$

such that $\mathcal{S}'$ *runs in* $\text{poly}(n, \lambda)$*-time,* $\mathcal{A}'$ *and* $\mathcal{L}'$ *run in* $\exp(n, \lambda)$*-time, and given that* $\mathcal{V}$ *is* λ*-bounded, the output* $\mathcal{V}'$ *satisfies: For all* $n \geq 2$,

1. **Completeness**: *If* $\mathcal{V}_{2^n}$ *has a perfect Z-aligned permutation strategy, then so does* $\mathcal{V}'_n$.

2. **Soundness**: *For every* $\varepsilon > 0$*, if* $\mathcal{V}'_n$ *has a value* $1 - \varepsilon$ *strategy, then* $\mathcal{V}_{2^n}$ *has a value* $1 - O(\varepsilon^{1/16})$ *strategy.*

3. **Entanglement**: *For every* $\varepsilon > 0$,

$$\mathscr{E}(\mathcal{V}'_n, 1 - \varepsilon) \geq (1 - O(\varepsilon)) \cdot 2^{2^{\lambda n}} \cdot \mathscr{E}(\mathcal{V}_{2^n}, 1 - O(\varepsilon^{1/16})).$$

The combinatorial transformation underlying the question reduction algorithm stems from the straightforward idea of *introspection* — "let the provers sample their own questions". This can be done naively, by letting $\mathfrak{Intro}(\mathfrak{G})$ be as in Definition 4.2. But, for this transformation to be helpful for compression, we are going to take an augmented sum (see Definitions 3.44 and 3.45) of $\mathfrak{Intro}(\mathfrak{G})$ with the *generalized Pauli basis* game $\mathfrak{Pauli} \mathfrak{Basis}_k$ (see Section 3.8). As we previously showed, the game $\mathfrak{Pauli} \mathfrak{Basis}_k$ is robust, and has essentially one perfect strategy, which induces the non-commutative representation ρ (45) of the Pauli group acting on k qubits (Definition 3.64). By connecting the total $\mathbb{X}$ and total $\mathbb{Z}$ measurements guaranteed by $\mathfrak{Pauli} \mathfrak{Basis}_k$ — namely, the vertices $\texttt{Pauli}_{\mathbb{X}}$ and $\texttt{Pauli}_{\mathbb{Z}}$ — in a clever way to $\mathfrak{Intro}(\mathfrak{G})$ we can ensure that any almost perfect strategy for the introspection game is close to being honest (Definition 4.4), and thus induces an almost perfect strategy of $\mathfrak{G}$.

4.1 The introspection game

Throughout this section, let $\mathfrak{G}$ be a tailored game with vertex set $\mathbb{F}_2^r$, and assume the distribution μ over its edges is a pushforward of the uniform measure on $\mathbb{F}_2^k$.[51] Namely, there is a function $\mathfrak{s} \colon \mathbb{F}_2^k \to \mathbb{F}_2^r \times \mathbb{F}_2^r$ such that

$$\forall x, y \in \mathbb{F}_2^r : \quad \mu(xy) = \frac{|\mathfrak{s}^{-1}(x, y)|}{2^k}. \tag{48}$$

Given $z \in \mathbb{F}_2^k$, we use $\mathfrak{s}^A(z) = x$ to denote the first coordinate of the output of $\mathfrak{s}$, and similarly $\mathfrak{s}^B(z) = y$ for the second coordinate. Assume furthermore that the readable and unreadable answer length functions of $\mathfrak{G}$ are constant and equal $\Lambda \in \mathbb{N}$.[52] Finally, let us denote by $S_x^{\mathfrak{R}} = \{X^{\mathfrak{R}, i} \mid 1 \leq i \leq \Lambda\}$ and $S_x^{\mathfrak{L}} = \{X^{\mathfrak{L}, i} \mid 1 \leq i \leq \Lambda\}$ the formal generator sets at $x \in \mathbb{F}_2^r$, and similarly $S_y^{\cdot} = \{Y^{\cdot, i}\}$ for $y \in \mathbb{F}_2^r$.

---

[51]This is always the case for games defined via normal form verifiers, as $\mathcal{S}$ calculates a pushforward of this form. It is not clear that all the vertices have the same bit length description (r in this case), but up to some encoding it can be assumed as well.

[52]As will be seen in Section 4.5.4, this assumption is not much of a constraint.

1. **Introspection** $\mathfrak{Intro}(\cdot)$. This transformation takes a game $\mathfrak{G}$ with (possibly) a very intricate underlying graph and replaces it by a game $\mathfrak{Intro}(\mathfrak{G})$ with an underlying graph containing a single edge between two vertices. The provers in the transformed game are expected to both *sample* their own questions and then *answer* them accordingly. It is introduced in Section 4.1 (see Figure 8 for a summary). Completeness and Soundness for "honest" strategies is sketched in the same section, specifically in Claim 4.6.

2. **Commutation game** $\mathfrak{C}$. This game is described in Section 3.8.1. It is a small, independent game on 3 questions that aims to verify commutation between two observables. Its completeness and soundness are stated in Fact 3.75.

3. **Anti-commutation game** $\mathfrak{M}$. This game is described in Section 3.8.2. It is a small, independent game on 15 questions that aims to verify anti-commutation between two observables. Its completeness and soundness are stated in Fact 3.78.

4. **Pauli basis game** $\mathfrak{Pauli\ Basis}_k(\mathscr{B})$. This game depends on an integer k and a set $\mathscr{B} = \{w^1, ..., w^n\} \subseteq \mathbb{F}_2^k$. Its goal is, essentially, to verify that a subset of a strategy's observables induce a non-commutative representation of the Pauli group P_k (introduced in Section 3.7). A combinatorial description of $\mathfrak{Pauli\ Basis}_k(\mathscr{B})$ is given in Section 3.8.3, see also Figure 3 for a summary. Completeness of this game is shown in Claim 3.82 and soundness in Claim 3.83. An algorithmic implementation of $\mathfrak{Pauli\ Basis}_k(\mathscr{B})$ as a tailored normal form verifier is implicitly given in Section 4.36, where a tailored normal form verifier for the larger game $\mathfrak{QueRed}(\mathfrak{G})$ is given.

5. **"Baby" question reduction** $\mathfrak{Baby}(\cdot)$. This transformation takes as input a game $\mathfrak{G}$ whose question distribution is a pushforward of the uniform measure over $\mathbb{F}_2^k$ through a *linear* function, and returns a game $\mathfrak{Baby}(\mathfrak{G})$ that has exponentially fewer questions and yet perfect ZPC completeness and soundness are preserved. The game is introduced in Section 4.2 (see Figure 9 for a summary). Completeness and soundness are shown in Theorem 4.8. (This game is introduced for illustrative purposes and results about it are not formally used.)

6. **Question reduced** $\mathfrak{QueRed}(\cdot)$. This transformation takes as input a game $\mathfrak{G}$ whose question distribution is *conditionally linear*, a generalization of the previous case described in Section 4.3, and returns a game $\mathfrak{QueRed}(\mathfrak{G})$ that has exponentially fewer questions and yet perfect ZPC completeness and soundness are preserved. The game is introduced in Section 4.4, see also Figure 12 for a summary. Completeness and soundness are shown in Theorem 4.24. A tailored normal form verifier implementing this game (given as input a tailored normal verifier for $\mathfrak{G}$) is described in Section 4.5, resulting in the proof of the main theorem of this section, Theorem 4.36, in Section 4.6.

Figure 7: We list the main games, or transformations thereof, used and introduced in this section, and where to find the most important statements about them.

Let $\mathfrak{G}$ be a tailored game whose readable and unreadable answer length functions are constant and equal to Λ.

Question	Readable variables	Unreadable variables
Intro.	$\{\mathsf{Que}^{\cdot,i}\}_{i=1}^{r}$ $\{\mathsf{Ans}^{\cdot,\mathfrak{R},j}\}_{j=1}^{\Lambda}$	$\{\mathsf{Ans}^{\cdot,\mathfrak{L},j}\}_{j=1}^{\Lambda}$

For any $\gamma\colon S_{\mathtt{Intro}_A}\cup S_{\mathtt{Intro}_B}\to\mathbb{F}_2$, denote

$$\mathtt{x}=\gamma|_{\mathsf{Que}^A},\ \mathtt{y}=\gamma|_{\mathsf{Que}^B},\ a^{\mathfrak{R}}=\gamma|_{\mathsf{Ans}^{A,\mathfrak{R}}},\ a^{\mathfrak{L}}=\gamma|_{\mathsf{Ans}^{A,\mathfrak{L}}},\ b^{\mathfrak{R}}=\gamma|_{\mathsf{Ans}^{B,\mathfrak{R}}},\ b^{\mathfrak{L}}=\gamma|_{\mathsf{Ans}^{B,\mathfrak{L}}}\ .$$

Then $\mathfrak{Intro}(\mathfrak{G})$ accepts $\mathtt{x},(a^{\mathfrak{R}},a^{\mathfrak{L}}),\mathtt{y},(b^{\mathfrak{R}},b^{\mathfrak{L}})$, if and only if $\mathfrak{G}$ accepts $(a^{\mathfrak{R}},a^{\mathfrak{L}}),(b^{\mathfrak{R}},b^{\mathfrak{L}})$ given that $\mathtt{xy}$ were asked.

Figure 8: Questions and answers in the game $\mathfrak{Intro}(\mathfrak{G})$.

Definition 4.2 (The introspection transformation of a tailored game). Let $\mathfrak{G}$ be a (tailored) game with the above fixed properties. The *introspection game* $\mathfrak{Intro}(\mathfrak{G})$ consists of only two vertices $\mathtt{Intro}_A$ and $\mathtt{Intro}_B$, with a single edge between them (see Figure 8 for a summary). As there is only one edge, it is always chosen by the question distribution of $\mathfrak{Intro}(\mathfrak{G})$. The readable length of both $\mathtt{Intro}_A$ and $\mathtt{Intro}_B$ is $r+\Lambda$, and their unreadable length is Λ. Define

$$\mathsf{Que}^A=\{\mathsf{Que}^{A,i}\mid 1\le i\le r\}\quad,\quad \mathsf{Ans}^{A,\mathfrak{R}}=\{\mathsf{Ans}^{A,\mathfrak{R},j}\mid 1\le j\le\Lambda\}\quad\text{and}\quad \mathsf{Ans}^{A,\mathfrak{L}}=\{\mathsf{Ans}^{A,\mathfrak{L},j}\mid 1\le j\le\Lambda\},$$

and similarly $\mathsf{Que}^B,\mathsf{Ans}^{B,\mathfrak{R}}$ and $\mathsf{Ans}^{B,\mathfrak{L}}$. Then, let the formal readable variables at $\mathtt{Intro}_A$ and $\mathtt{Intro}_B$ be

$$S_{\mathtt{Intro}_A}^{\mathfrak{R}}=\mathsf{Que}^A\sqcup\mathsf{Ans}^{A,\mathfrak{R}}\quad\text{and}\quad S_{\mathtt{Intro}_B}^{\mathfrak{R}}=\mathsf{Que}^B\sqcup\mathsf{Ans}^{B,\mathfrak{R}}$$

respectively, and let the formal unreadable variables at these vertices be

$$S_{\mathtt{Intro}_A}^{\mathfrak{L}}=\mathsf{Ans}^{A,\mathfrak{L}}\quad\text{and}\quad S_{\mathtt{Intro}_B}^{\mathfrak{L}}=\mathsf{Ans}^{B,\mathfrak{L}}.$$

The naming scheme is Que for "question" and Ans for " answer". I.e., the assignment to the variable $\mathsf{Que}^{A,i}$ (respectively $\mathsf{Que}^{B,i}$) is expected to be the ith bit of a question $\mathtt{x}\in\mathbb{F}_2^r$ (respectively $\mathtt{y}\in\mathbb{F}_2^r$), the assignment to $\mathsf{Ans}^{A,\mathfrak{R},j}$ (respectively $\mathsf{Ans}^{B,\mathfrak{R},j}$) is expected to be the jth bit of the readable part of an answer in $\mathfrak{G}$ to the question $\mathtt{x}$ (respectively $\mathtt{y}$), and the assignment to $\mathsf{Ans}^{A,\mathfrak{L},j}$ (respectively $\mathsf{Ans}^{B,\mathfrak{L},j}$) is expected to be the jth bit of the unreadable part of the answer to $\mathtt{x}$ (respectively $\mathtt{y}$). The controlled linear constraint function $L_{\mathtt{Intro}_A\,\mathtt{Intro}_B}(\gamma)$ works as follows. Let $\gamma\colon S_{\mathtt{Intro}_A}\cup S_{\mathtt{Intro}_B}\to\mathbb{F}_2$, and denote its restrictions as follows

$$\mathtt{x}=\gamma|_{\mathsf{Que}^A},\ \mathtt{y}=\gamma|_{\mathsf{Que}^B},\ a^{\mathfrak{R}}=\gamma|_{\mathsf{Ans}^{A,\mathfrak{R}}},\ a^{\mathfrak{L}}=\gamma|_{\mathsf{Ans}^{A,\mathfrak{L}}},\ b^{\mathfrak{R}}=\gamma|_{\mathsf{Ans}^{B,\mathfrak{R}}},\quad\text{and}\quad b^{\mathfrak{L}}=\gamma|_{\mathsf{Ans}^{B,\mathfrak{L}}}\ .$$

Note that $\mathtt{x}$ and $\mathtt{y}$ are r-long bit strings, and can thus be viewed as vertices in the underlying graph of the original game $\mathfrak{G}$. Recall that we denoted $S_{\mathtt{x}}=\{\mathsf{X}^{\cdot,j}\}$ and $S_{\mathtt{y}}=\{\mathsf{Y}^{\cdot,j}\}$ for the formal generator sets associated with $\mathtt{x}$ and $\mathtt{y}$ in $\mathfrak{G}$. Then, if $\mathtt{xy}$ is not an edge in the underlying graph of $\mathfrak{G}$, then $L_{\mathtt{Intro}_A\,\mathtt{Intro}_B}(\gamma)$ will output the singleton $\{\mathsf{J}\}$ — which induces the linear constraint $1=0$, i.e., rejection. Otherwise, for every $c\colon S_{\mathtt{x}}\cup S_{\mathtt{y}}\cup\{\mathsf{J}\}\to\mathbb{F}_2$ in $L_{\mathtt{xy}}(a^{\mathfrak{R}},b^{\mathfrak{R}})$, we add the constraint

coefficients function $c': S_{\texttt{Intro}_A} \cup S_{\texttt{Intro}_B} \cup \{\mathsf{J}\} \to \mathbb{F}_2$ to $L_{\texttt{Intro}_A\ \texttt{Intro}_B}(\gamma)$, where

$$\forall 1 \le i \le r, 1 \le j \le \Lambda : \quad c'(\mathsf{Que}^{A,i}) = c'(\mathsf{Que}^{B,i}) = 0\,,$$
$$c'(\mathsf{Ans}^{A,\mathfrak{R},j}) = c(\mathsf{X}^{\mathfrak{R},j})\,,$$
$$c'(\mathsf{Ans}^{A,\mathfrak{L},j}) = c(\mathsf{X}^{\mathfrak{L},j})\,,$$
$$c'(\mathsf{Ans}^{B,\mathfrak{R},j}) = c(\mathsf{Y}^{\mathfrak{R},j})\,,$$
$$c'(\mathsf{Ans}^{B,\mathfrak{L},j}) = c(\mathsf{Y}^{\mathfrak{L},j})\,,$$
$$c'(\mathsf{J}) = c(\mathsf{J})\,.$$

In words, $\mathfrak{Intro}(\mathfrak{G})$ treats $\mathsf{Ans}^{A,\cdot,j}$ as if they were the generators $\mathsf{X}^{\cdot,j}$ of the sampled vertex x, and similarly for $\mathsf{Ans}^{B,\cdot,j}$ and $\mathsf{Y}^{\cdot,j}$ for the other sampled vertex y.

Remark 4.3. Though the above definition is a bit technical, it can be explained in plain words: The answer to $\texttt{Intro}_A$ is of the form $(\mathrm{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ and to $\texttt{Intro}_B$ is of the form $(\mathrm{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$. Then, $\mathfrak{Intro}(\mathfrak{G})$ accepts this pair of answers if and only if $\mathfrak{G}$ would accept $(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ given that xy was the sampled edge.

We now define the notion of an *honest* strategy for $\mathfrak{Intro}(\mathfrak{G})$. Colloquially, such a strategy $\mathscr{S}$ is derived from a strategy $\mathscr{S}'$ for $\mathfrak{G}$ as follows: First, it samples a bit string $z \in \mathbb{F}_2^k$ uniformly at random. Then, it calculates $\mathfrak{s}(z) = \mathrm{xy}$. Then, only depending on x it performs the measurements induced by $\mathscr{S}'$ given that x was asked, which yields the answers $a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$; similarly, only depending on y, using the measurements of $\mathscr{S}'$, it obtains $b^{\mathfrak{R}}, b^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$. Finally, it replies $(\mathrm{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ as the assignment to the $\texttt{Intro}_A$ variables and $(\mathrm{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ as the assignment to the $\texttt{Intro}_B$ variables. It is straightforward that the value of this honest strategy $\mathscr{S}$ is the same as the value of the associated strategy $\mathscr{S}'$ for the original game.

Definition 4.4 (Honest strategies for the introspection game). Given an N-dimensional strategy $\mathscr{S} = \{\mathcal{P}\}$ to the original game $\mathfrak{G}$, one can construct the following *honest* strategy $\mathscr{S}' = \{\mathcal{Q}\}$ to $\mathfrak{Intro}(\mathfrak{G})$ acting on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^N$: As the length functions of all the vertices in $\mathfrak{G}$ are Λ (both readable and unreadable), $\mathcal{P}^{\mathrm{x}}: \mathbb{F}_2^{\Lambda} \times \mathbb{F}_2^{\Lambda} \to M_N(\mathbb{C})$. For $\texttt{Intro}_A$ and $\texttt{Intro}_B$, their readable length is $r + \Lambda$ and unreadable length is Λ, so $\mathcal{Q}^{\texttt{Intro}_\cdot}: \mathbb{F}_2^r \times \mathbb{F}_2^{\Lambda} \times \mathbb{F}_2^{\Lambda} \to \mathrm{End}(\mathbb{C}^{\mathbb{F}_2^k}) \otimes M_N(\mathbb{C})$. Recall the notation $\mathscr{F}_z^{\mathbb{Z}} \in \mathrm{End}(\mathbb{C}^{\mathbb{F}_2^k})$ for the orthogonal projection on the subspace spanned by the indicator $\mathbf{1}_z$ in $\mathbb{C}^{\mathbb{F}_2^k}$ (Definition 3.67). Then, for every $\mathrm{x} \in \mathbb{F}_2^r, a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$, let

$$\mathcal{Q}_{\mathrm{x},a^{\mathfrak{R}},a^{\mathfrak{L}}}^{\texttt{Intro}_A} = \sum_{z \in \mathbb{F}_2^k\,:\,\mathfrak{s}^A(z) = \mathrm{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{a^{\mathfrak{R}},a^{\mathfrak{L}}}^{\mathrm{x}}\,, \tag{49}$$

and similarly for every $\mathrm{y} \in \mathbb{F}_2^r, b^{\mathfrak{R}}, b^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$, let

$$\mathcal{Q}_{\mathrm{y},b^{\mathfrak{R}},b^{\mathfrak{L}}}^{\texttt{Intro}_B} = \sum_{z \in \mathbb{F}_2^k\,:\,\mathfrak{s}^B(z) = \mathrm{y}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{b^{\mathfrak{R}},b^{\mathfrak{L}}}^{\mathrm{y}}\,. \tag{50}$$

Remark 4.5. Note that the matrix $\mathscr{F}_z^{\mathbb{Z}}$ has a single 1 on the diagonal at the position (z, z) (the matrix's rows and columns are parameterized by $\mathbb{F}_2^k$) and 0 everywhere else — this matrix is often denoted by e_{zz} or $\mathbf{1}_{zz}$. Hence, for every collection of same sized square matrices $\mathscr{A}^z$, the matrix $\sum_z \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$ is a block diagonal matrix with the $\mathscr{A}^z$'s on the diagonal. In particular, if $\mathscr{A}^z$ are diagonal then also $\sum_z \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$ is diagonal, and similarly if $\mathscr{A}^z$ are signed permutation matrices then also $\sum_z \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$ is a signed permutation matrix.

Claim 4.6 (Completeness and soundness of honest strategies for the introspection game). *Let $\mathscr{S} = \{\mathcal{P}\}$ be a strategy for $\mathfrak{G}$ and $\mathscr{S}' = \{\mathcal{Q}\}$ the honest strategy for $\mathfrak{Intro}(\mathfrak{G})$ associated with $\mathscr{S}$ (as defined in (49) and (50)). Then,*

1. $\mathrm{val}(\mathfrak{G}, \mathscr{S}) = \mathrm{val}(\mathfrak{Intro}(\mathfrak{G}), \mathscr{S}')$;

2. $\mathscr{S}$ *being* ZPC *implies* $\mathscr{S}'$ *is* ZPC.

Proof. For item 1, jointly sampling $((\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})(\mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}})) \sim (\mathcal{Q}^{\mathtt{Intro}_A}, \mathcal{Q}^{\mathtt{Intro}_B})$ (Definition 2.2) gives the same distribution on six-tuples as first sampling $\mathtt{xy} \sim \mu$ (as was defined in (48)) and then jointly sampling $((a^{\mathfrak{R}}, a^{\mathfrak{L}}), (b^{\mathfrak{R}}, b^{\mathfrak{L}})) \sim (\mathcal{P}^{\mathtt{x}}, \mathcal{P}^{\mathtt{y}})$. This means that indeed the value of the honest strategy against the introspection game is the same as that of the original strategy against the original game.

For item 2, we need to view both $\mathscr{S}$ and $\mathscr{S}'$ in their observable forms, which we denote by $\mathcal{U}$ and $\mathcal{V}$ respectively. As $\mathcal{P}^{\mathtt{x}}$ is a PVM for every $\mathtt{x} \in \mathbb{F}_2^r$, the marginalization (i.e., restriction, cf. Definition 3.32) of $\mathcal{Q}$ to the $\mathtt{Que}^A$-variables satisfies

$$\mathcal{Q}_{\mathtt{x}}^{\mathtt{Que}^A} = \sum_{a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}} \mathcal{Q}_{\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}_A} =_{(49)} \sum_{z \in \mathbb{F}_2^k : \, \mathfrak{s}^A(z) = \mathtt{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id} = \mathscr{F}_{[\mathfrak{s}^A(\cdot) = \mathtt{x}]}^{\mathbb{Z}} \, ,$$

and similarly $\mathcal{Q}_{\mathtt{y}}^{\mathtt{Que}^B} = \mathscr{F}_{[\mathfrak{s}^B(\cdot) = \mathtt{y}]}^{\mathbb{Z}}$. As $\mathscr{F}^{\mathbb{Z}}$ is a diagonal PVM, it remains diagonal under data processing, and thus the inverse Fourier transformed (Definition 2.4) representations $\mathcal{V}^{\mathtt{Que}^A}$ and $\mathcal{V}^{\mathtt{Que}^B}$ are also diagonal — this shows that the observables associated with the readable variables $\mathtt{Que}^A$ and $\mathtt{Que}^B$ are indeed Z-aligned and consist of signed permutations.

When marginalizing $\mathcal{Q}$ to the $\mathtt{Ans}^A$ variables we get

$$\mathcal{Q}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Ans}^A} = \sum_{z \in \mathbb{F}_2^k} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathfrak{s}^A(z)} \, .$$

As described in Remark 4.5, these are block diagonal matrices whose zz-block (for $z \in \mathbb{F}_2^k$) contains the PVM $\mathcal{P}^{\mathfrak{s}^A(z)}$. As the inverse Fourier transform for block diagonal matrices works block by block, we deduce that the representation $\mathcal{V}^{\mathtt{Ans}^A}$ satisfies

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda} : \quad \mathcal{V}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) = \sum_{z \in \mathbb{F}_2^k} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{U}^{\mathfrak{s}^A(z)}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}). \tag{51}$$

In particular, if $\mathcal{U}$ consists of only signed permutation matrices, then so does $\mathcal{V}$, and similarly if the marginalization to the readable variables is diagonal for $\mathcal{U}$, then it is diagonal for $\mathcal{V}$. We can thus deduce that $\mathscr{S}$ being a Z-aligned permutation strategy implies $\mathscr{S}'$ is such.

We are left to prove that $\mathscr{S}$ commuting along edges implies $\mathscr{S}'$ is also commuting along edges. By (49) and (50) we have

$$\mathcal{Q}_{\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}^A} \mathcal{Q}_{\mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{Intro}^B} = \left(\sum_{z \in \mathbb{F}_2^k : \, \mathfrak{s}^A(z) = \mathtt{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{x}} \right) \left(\sum_{z' \in \mathbb{F}_2^k : \, \mathfrak{s}^B(z') = \mathtt{y}} \mathscr{F}_{z'}^{\mathbb{Z}} \otimes \mathcal{P}_{b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{y}} \right) \, . \tag{52}$$

As $\mathscr{F}^{\mathbb{Z}}$ is projective, when one distributes the above product, the only summands that are potentially non-zero are those indexed by $z \in \mathbb{F}_2^k$ for which both $\mathfrak{s}^A(z) = \mathtt{x}$ and $\mathfrak{s}^B(z) = \mathtt{y}$; in particular, this product is zero if $\mathtt{xy}$ is not an edge in the original game $\mathfrak{G}$. This is true for the reversed product $\mathcal{Q}_{\mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{Intro}^B} \mathcal{Q}_{\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}^A}$, and thus if $\mathtt{xy}$ is not an edge, then $\mathcal{Q}_{\mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{Intro}^B}$ and $\mathcal{Q}_{\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}^A}$ commute (as their product in both orders is equal to zero). In case $\mathtt{xy}$ is an edge in the original game, the product in (52) is equal to

$$\sum_{\substack{z \in \mathbb{F}_2^k \\ \mathfrak{s}(z) = \mathtt{xy}}} \mathscr{F}_z^{\mathbb{Z}} \otimes (\mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{x}} \mathcal{P}_{b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{y}}) = \sum_{\substack{z \in \mathbb{F}_2^k \\ \mathfrak{s}(z) = \mathtt{xy}}} \mathscr{F}_z^{\mathbb{Z}} \otimes (\mathcal{P}_{b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{y}} \mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{x}}) = \mathcal{Q}_{\mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}}}^{\mathtt{Intro}^B} \mathcal{Q}_{\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}^A} \, ,$$

where the first equation is due to $\mathscr{S} = \{\mathcal{P}\}$ being commuting along edges, and the second equation is, again, from the projectivity of $\mathscr{F}^{\mathbb{Z}}$. All in all, $\mathscr{S}' = \{\mathcal{Q}\}$ commutes along edges, as needed. $\square$

Although, given that the original game has a perfect ZPC-strategy, one can extract a perfect honest ZPC-strategy for $\mathfrak{Intro}(\mathfrak{G})$ (as desecribed above), there are many perfect strategies for $\mathfrak{Intro}(\mathfrak{G})$ that are **not honest**. For example, if there is any edge $\mathtt{xy}$ and answer $(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ for it which is accepted by the decision predicate of $\mathfrak{G}$, then a strategy for $\mathfrak{Intro}(\mathfrak{G})$

can always assign the values x and y to the Que variables, and assign the accepting answer $(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ to the Ans variables. A further dismotivating observation is the following: even if $\mathscr{S}$ indeed samples a string $z \in \mathbb{F}_2^k$ as a random seed and uses it to calculate x and y appropriately — namely samples an edge in $\mathfrak{G}$ according to the correct distribution — there is no guarantee that the observables it associates with $\mathsf{Ans}^{A,\cdot,\cdot}$ depend only on x and disregard y. This means that the strategy can choose for every edge xy a fixed accepting answer $(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ and provide it given that $\mathfrak{s}(z) = $ xy. In plain words, the fact that the strategy sampled its own edge is the same as for the provers to be able to share their questions before providing their answers (in the dramatized version, Remark 2.23) — which usually collapses everything to a single prover interactive proof.

Therefore, as the above two non-honest perfect strategies suggest, $\mathfrak{Intro}(\mathfrak{G})$ is not very useful on its own. We amend this by taking the sum of $\mathfrak{Intro}(\mathfrak{G})$ and the Pauli basis game $\mathfrak{Pauli\,Basis}_k$ (defined in Section 3.8) and augment it by connecting the total $\mathbb{Z}$-measurement (vertex $\mathtt{Pauli_{\mathbb{Z}}}$) to the Que variables so that $\mathscr{S}$ is forced to sample an edge according to the suitable distribution induced by $\mathfrak{s}$. Then, we are going to use the $\mathbb{X}$-measurements (vertex $\mathtt{Pauli_{\mathbb{X}}}$) of $\mathfrak{Pauli\,Basis}_k$ to ensure that the observables $\mathscr{S}$ associates to $\mathsf{Ans}^{A,\cdot,\cdot}$ depend only on x and that the observables $\mathscr{S}$ associates to $\mathsf{Ans}^{B,\cdot,\cdot}$ depend only on y — which forces any almost perfect strategy for this augmentation to be close to an honest strategy for the introspective game. This second amendment uses the fact that "depending only on x" is the same as providing the same answer for any two seeds z_1, z_2 such that $\mathfrak{s}^A(z_1) = \mathfrak{s}^A(z_2) = $ x, and there is an $\mathbb{X}$-Pauli matrix that moves from the seed z_1 to the seed z_2 — namely, this independence boils down to certain commutation relations with $\mathbb{X}$-Pauli matrices.

4.2 Motivational interlude — Question Reduction in the linear sampler case

Note, this section provides a simpler version of the final (combinatorial) transformation of question reduction. The full version is described in Section 4.4.

Let $\mathfrak{G}$ be a tailored game with the properties fixed in the beginning of the section, namely, its vertex set is $\mathbb{F}_2^r$, the distribution on edges is induced by the pushforward of the uniform distribution on $\mathbb{F}_2^k$ through $\mathfrak{s} \colon \mathbb{F}_2^k \to \mathbb{F}_2^r \times \mathbb{F}_2^r$, and its length functions are constant and equal to Λ. In addition, assume that $\mathfrak{s}$ is **linear**. As $\mathfrak{s}^A, \mathfrak{s}^B \colon \mathbb{F}_2^k \to \mathbb{F}_2^r$ are linear, we think about them as $r \times k$ matrices (with respect to the standard basis), and we assume given two additional $k \times k$ matrices $(\mathfrak{s}^A)^\perp, (\mathfrak{s}^B)^\perp$ (also thought of as linear operators on $\mathbb{F}_2^k$) whose rows span the respective kernels $\ker \mathfrak{s}^A, \ker \mathfrak{s}^B \leq \mathbb{F}_2^k$.

The baby question reduction transformation $\mathfrak{Baby}(\mathfrak{G}) = \mathfrak{Baby}(\mathfrak{G}, k, \mathscr{B})$ (see Figure 9 for a summary): First, as hinted in the notation, this transformation depends on three inputs — a positive integer k, a subset $\mathscr{B}$ of n vectors in $\mathbb{F}_2^k$ which induces an $[n, k, d]$-code as was defined Section 3.7.2, and a game $\mathfrak{G}$ with the properties fixed in the previous paragraph. With these inputs, the Pauli basis game (Section 3.8.3) $\mathfrak{Pauli\,Basis}_k = \mathfrak{Pauli\,Basis}_k(\mathscr{B})$ and the introspection game (Definition 4.2) $\mathfrak{Intro}(\mathfrak{G})$ can be defined. The baby question reduction game $\mathfrak{Baby}(\mathfrak{G})$ is an augmentation (Definition 3.45) of the sum (Definition 3.44) of $\mathfrak{Pauli\,Basis}_k$ and $\mathfrak{Intro}(\mathfrak{G})$; the augmentation consists of two apparatuses:

1. A **Sampling apparatus** which connects the introspection game vertices to the total Z-measurement of the Pauli basis game (i.e., the vertex $\mathtt{Pauli_{\mathbb{Z}}}$). The goal of this apparatus is twofold — first, to verify that the "questions" part of the players' answers when the copy of $\mathfrak{Intro}(\mathfrak{G})$ is played is distributed according to the question distribution of $\mathfrak{G}$; second, to verify that the observables associated with the "answers" part of the players' answers in $\mathfrak{Intro}(\mathfrak{G})$ commute with the total Z-measurement.

2. A **Hiding apparatus** which connects the introspection game vertices to the total X-measurement of the Pauli basis game (i.e., the vertex $\mathtt{Pauli_{\mathbb{X}}}$). The goal of this apparatus is to verify that the "answers" part of the players' answers in $\mathfrak{Intro}(\mathfrak{G})$ commute with certain X-measurements (though not the total one).

In the sampling apparatus, two additional vertices $\mathtt{Sample}_A, \mathtt{Sample}_B$ are added, and they are connected as follows

$$\mathtt{Intro}_A - \mathtt{Sample}_A - \mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_B - \mathtt{Intro}_B \,.$$

73

Let $\mathfrak{G}$ be a tailored game with vertex set $\mathbb{F}_2^r$ and whose distribution on edges is the pushforward of the uniform distribution on $\mathbb{F}_2^k$ through a **linear map** $\mathfrak{s} = (\mathfrak{s}^A, \mathfrak{s}^B) \colon \mathbb{F}_2^k \to \mathbb{F}_2^r \times \mathbb{F}_2^r$; let $(\mathfrak{s}^A)^\perp, (\mathfrak{s}^B)^\perp$ be $k \times k$ matrices whose rows span $\ker(\mathfrak{s}^A)$ and $\ker(\mathfrak{s}^B)$. In addition, the game $\mathfrak{G}$ is assumed to have readable and unreadable answer lengths equal to some constant Λ.

Sub-Structure	Question	Readable answer	Unreadable answer
$\mathfrak{Pauli\ Basis}_k$	$\mathtt{Pauli}_{\mathbb{Z}}$		$z \in \mathbb{F}_2^k$
	$\mathtt{Pauli}_{\mathbb{X}}$		$\chi \in \mathbb{F}_2^k$
	See Figure 3 for rest		
$\mathfrak{Intro}(\mathfrak{G})$	$\mathtt{Intro}_A$	$(\mathbf{x}, a^{\mathfrak{R}}) \in \mathbb{F}_2^r \times \mathbb{F}_2^\Lambda$	$a^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda$
Sampling apparatus	$\mathtt{Sample}_A$	$(z_{sam}, a^{\mathfrak{R}}_{sam}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$	$a^{\mathfrak{L}}_{sam} \in \mathbb{F}_2^\Lambda$
Hiding apparatus	$\mathtt{Read}_A$	$(\mathbf{x}_{read}, a^{\mathfrak{R}}_{read}) \in \mathbb{F}_2^r \times \mathbb{F}_2^\Lambda$	$(\nu_{read}, a^{\mathfrak{L}}_{read}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$
	$\mathtt{Hide}_A$		$\nu \in \mathbb{F}_2^k$

The following tests are performed when the corresponding augmented edge is sampled — the checks for B are similar:

1. $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_A$: Check that $z = z_{sam}$.

2. $\mathtt{Intro}_A - \mathtt{Sample}_A$: Check that $\mathbf{x} = \mathfrak{s}^A(z_{sam})$, $a^{\mathfrak{R}} = a^{\mathfrak{R}}_{sam}$ and $a^{\mathfrak{L}} = a^{\mathfrak{L}}_{sam}$.

3. $\mathtt{Intro}_A - \mathtt{Read}_A$: Check that $\mathbf{x} = \mathbf{x}_{read}$, $a^{\mathfrak{R}} = a^{\mathfrak{R}}_{read}$ and $a^{\mathfrak{L}} = a^{\mathfrak{L}}_{read}$.

4. $\mathtt{Hide}_A - \mathtt{Read}_A$: Check that $\nu_{read} = \nu$.

5. $\mathtt{Hide}_A - \mathtt{Pauli}_{\mathbb{X}}$: Check that $\nu = (\mathfrak{s}^A)^\perp(\chi)$.

Figure 9: Questions and answers in the game $\mathfrak{Baby}(\mathfrak{G})$. Since the game is an augmentation of the sum of $\mathfrak{Pauli\ Basis}_k$ and $\mathfrak{Intro}(\mathfrak{G})$ we only list new questions and answers, and additional tests, and refer to Figure 3 and Figure 8 for questions and answers of the latter. Note that all the augmented checks are linear and independent of the values associated to readable variables — this will not be the case in the general question reduction transformation described in Section 4.4.

In the hiding apparatus, 4 additional vertices are added

$$\texttt{Read}_A \, , \, \texttt{Read}_B \, , \, \texttt{Hide}_A \, , \, \texttt{Hide}_B \, ,$$

and they are connected as follows

$$\texttt{Intro}_A - \texttt{Read}_A - \texttt{Hide}_A - \texttt{Pauli}_{\mathbb{Z}} - \texttt{Hide}_B - \texttt{Read}_B - \texttt{Intro}_B \, .$$

For a graphical view of the underlying graph of $\mathfrak{Baby}(\mathfrak{G})$, see Figure 10.

Figure 10: The underlying graph of $\mathfrak{Baby}(\mathfrak{G})$, where most of the embedded Pauli basis game is hidden.

Question distribution of the baby question reduced game: With probability $1/4$ do one of the following:

- Sample an edge from $\mathfrak{Pauli \ Basis}_k$ according to the appropriate distribution therein.

- Sample the single edge $\texttt{Intro}_A - \texttt{Intro}_B$ from $\mathfrak{Intro}(\mathfrak{G})$.

- Sample a uniformly random edge from the Sampling apparatus.

- Sample a uniformly random edge from the Hiding apparatus.

Lengths and formal generating sets for the augmented vertices of baby question reduction:

Sampling apparatus — The readable length of $\texttt{Sample}_A$ (and $\texttt{Sample}_B$) is $k + \Lambda$, and its unreadable length is Λ. We associate with it the formal generators

$$\mathsf{SamZ}^A = \{\mathsf{SamZ}^{A,i} \mid 1 \leq i \leq k\} \, ,$$

$$\mathsf{SamAns}^{A,\mathfrak{R}} = \{\mathsf{SamAns}^{A,\mathfrak{R},j} \mid, 1 \leq j \leq \Lambda\} \, , \quad \mathsf{SamAns}^{A,\mathfrak{L}} = \{\mathsf{SamAns}^{A,\mathfrak{L},j} \mid, 1 \leq j \leq \Lambda\} \, ,$$

$$S^{\mathfrak{R}}_{\texttt{Sample}_A} = \mathsf{SamZ}^A \sqcup \mathsf{SamAns}^{A,\mathfrak{L}} \, , \quad S^{\mathfrak{L}}_{\texttt{Sample}_A} = \mathsf{SamAns}^{A,\mathfrak{L}} \, .$$

and similarly for $\mathtt{Sample}_B$. Namely, answers are formatted as $(z_{sam}, a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}})$, where $z_{sam} \in \mathbb{F}_2^k$, and $a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$.

Hiding apparatus —

- The readable length of $\mathtt{Read}_A$ (respectively $\mathtt{Read}_B$) is $r + \Lambda$, and its unreadable length is $k + \Lambda$. We associate with it the formal generators

$$\mathsf{ReadQue}^A = \{\mathsf{ReadQue}^{A,i} \mid 1 \leq i \leq r\}, \quad \mathsf{ReadPerp}^A = \{\mathsf{ReadPerp}^{A,i} \mid 1 \leq i \leq k\},$$
$$\mathsf{ReadAns}^{A,\mathfrak{R}} = \{\mathsf{ReadAns}^{A,\mathfrak{R},j} \mid 1 \leq j \leq \Lambda\}, \quad \mathsf{ReadAns}^{A,\mathfrak{L}} = \{\mathsf{ReadAns}^{A,\mathfrak{L},j} \mid 1 \leq j \leq \Lambda\},$$
$$S_{\mathtt{Read}_A}^{\mathfrak{R}} = \mathsf{ReadQue}^A \sqcup \mathsf{ReadAns}^{A,\mathfrak{R}}, \quad S_{\mathtt{Read}_A}^{\mathfrak{L}} = \mathsf{ReadPerp}^A \sqcup \mathsf{ReadAns}^{A,\mathfrak{L}},$$

and similarly for $\mathtt{Read}_B$. Namely, answers are formatted as $(\mathrm{x}_{read}, a_{read}^{\mathfrak{R}}, \nu_{read}, a_{read}^{\mathfrak{L}})$, where $\mathrm{x}_{read} \in \mathbb{F}_2^r, a_{read}^{\mathfrak{R}}, a_{read}^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$ and $\nu_{read} \in \mathbb{F}_2^k$ (and for B, $(\mathrm{y}_{read}, b_{read}^{\mathfrak{R}}, \mu_{read}, b_{read}^{\mathfrak{L}})$ in the appropriate spaces).

- The readable length of $\mathtt{Hide}_A$ (respectively $\mathtt{Hide}_B$) is 0, and its unreadable length is k. We associate with it the formal generators

$$S_{\mathtt{Hide}_A}^{\mathfrak{L}} = \mathsf{Perp}^A = \{\mathsf{Perp}^{A,i} \mid 1 \leq i \leq k\},$$

and similarly for $\mathtt{Hide}_B$. Namely, the answer is formatted as $\nu \in \mathbb{F}_2^k$ (respectively $\mu \in \mathbb{F}_2^k$).

The decision process for the augmented edges of the baby question reduced game:[53]

- *Sampling apparatus —*

  (1) In case $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_A$ (respectively $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_B$) is sampled, then check that

  $$\forall 1 \leq i \leq k : \quad \gamma(\mathsf{PZ}^i) = \gamma(\mathsf{SamZ}^{A,i}).$$

  In other words, if z is the answer to $\mathtt{Pauli}_{\mathbb{Z}}$, then it checks that $z = z_{sam}$.

  (2) In case $\mathtt{Intro}_A - \mathtt{Sample}_A$ (respectively $\mathtt{Intro}_B - \mathtt{Sample}_B$) is sampled, first check that

  $$\forall 1 \leq j \leq \Lambda : \quad \gamma(\mathsf{Ans}^{A,\cdot,j}) = \gamma(\mathsf{SamAns}^{A,\cdot,j}),$$

  and then check that

  $$\forall 1 \leq i \leq r : \quad \gamma(\mathsf{Que}^{A,i}) = \sum_{j=1}^{k} \mathfrak{s}_{ij}^A \gamma(\mathsf{SamZ}^{A,j}).$$

  In other words, if $(z_{sam}, a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}})$ is the answer to $\mathtt{Sample}_A$, and $(\mathrm{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ is the answer to $\mathtt{Intro}_A$, then it checks that $\mathrm{x} = \mathfrak{s}^A(z_{sam}), a^{\mathfrak{R}} = a_{sam}^{\mathfrak{R}}$ and $a^{\mathfrak{L}} = a_{sam}^{\mathfrak{L}}$.

- *Hiding apparatus —*

  (1) In case $\mathtt{Intro}_A - \mathtt{Read}_A$ (respectively $\mathtt{Intro}_B - \mathtt{Read}_B$) is sampled, check that

  $$\forall 1 \leq i \leq r, 1 \leq j \leq \Lambda : \quad \gamma(\mathsf{Ans}^{A,\cdot,j}) = \gamma(\mathsf{ReadAns}^{A,\cdot,j}), \quad \gamma(\mathsf{Que}^{A,i}) = \gamma(\mathsf{ReadQue}^{A,i}).$$

  In other words, if $(\mathrm{x}_{read}, a_{read}^{\mathfrak{R}}, \nu_{read}, a_{read}^{\mathfrak{L}})$ is the answer to $\mathtt{Read}_A$, and $(\mathrm{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ is the answer to $\mathtt{Intro}_A$, check that $\mathrm{x}_{read} = \mathrm{x}, a_{read}^{\mathfrak{R}} = a^{\mathfrak{R}}$ and $a_{read}^{\mathfrak{L}} = a^{\mathfrak{L}}$.

---

[53]This is essentially the description of the controlled linear constraints function.

(2) In case $\texttt{Hide}_A - \texttt{Read}_A$ (respectively $\texttt{Hide}_B - \texttt{Read}_B$) is sampled, check that

$$\forall 1 \leq i \leq k : \quad \gamma(\mathsf{Perp}^{A,i}) = \gamma(\mathsf{ReadPerp}^{A,i}) \,.$$

In other words, if $\left(\mathtt{x}_{read}, a^{\mathfrak{R}}_{read}, v_{read}, a^{\mathfrak{L}}_{read}\right)$ is the answer to $\texttt{Read}_A$, and ν is the answer to $\texttt{Hide}_A$, check that $\nu = v_{read}$.

(3) In case $\texttt{Hide}_A - \texttt{Pauli}_{\mathbb{X}}$ (respectively $\texttt{Hide}_B - \texttt{Pauli}_{\mathbb{X}}$) is sampled, check that

$$\forall 1 \leq i \leq k : \quad \gamma(\mathsf{Perp}^{A,i}) = \sum_{j=1}^{k} (\mathfrak{s}^A)^{\perp}_{ij} \gamma(\mathsf{PX}^j) \,,$$

where $(\mathfrak{s}^A)^{\perp}$ was the matrix whose rows span $\ker \mathfrak{s}^A$. In other words, if ν is the answer to $\texttt{Hide}_A$ and χ is the answer to $\texttt{Pauli}_{\mathbb{X}}$, check that $\nu = (\mathfrak{s}^A)^{\perp}(\chi)$.

Remark 4.7 (Informal analysis of $\mathfrak{Baby}(\mathfrak{G})$)**.** Before proving rigorously that this game is complete and sound, let us discuss the role of the various checks described above in forcing strategies to behave appropriately in this augmented version of $\mathfrak{Pauli\ Basis}_k \oplus \mathfrak{Intro}(\mathfrak{G})$. Similarly to the way the ultimate goal of all the checks in $\mathfrak{Pauli\ Basis}_k$ was for the observables at $\texttt{Pauli}_{\mathbb{Z}}$ and $\texttt{Pauli}_{\mathbb{X}}$ to induce a (specific) representation of the Pauli group, the ultimate goal of all the above checks is to force the strategy to play *honestly* (Definition 4.4) when the copy of $\mathfrak{Intro}(\mathfrak{G})$ is played — namely, it samples a seed $z \in \mathbb{F}_2^k$ uniformly at random, calculates $\mathfrak{s}(z) = \mathtt{xy}$, calculates $(a^{\mathfrak{R}}, a^{\mathfrak{L}})$ depending only on $\mathtt{x}$ and $(b^{\mathfrak{R}}, b^{\mathfrak{L}})$ depending only on $\mathtt{y}$, and finally replies with $(\mathtt{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}, \mathtt{y}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$.

This is achieved as follows: First, the copy of the Pauli basis game forces the answer z in $\texttt{Pauli}_{\mathbb{Z}}$ to be uniformly distributed over $\mathbb{F}_2^k$, using the PVM (in representation form) $\rho^{\mathbb{Z}} \otimes \mathrm{Id}$ (and $\rho^{\mathbb{X}} \otimes \mathrm{Id}$ for $\texttt{Pauli}_{\mathbb{X}}$), where ρ is the representation from Definition 3.68. The checks $\texttt{Pauli}_{\mathbb{Z}} - \texttt{Sample}_A$ and $\texttt{Intro}_A - \texttt{Sample}_A$ force $\mathtt{x}$ to be $\mathfrak{s}^A(z)$ and similarly $\texttt{Pauli}_{\mathbb{Z}} - \texttt{Sample}_B$ and $\texttt{Intro}_B - \texttt{Sample}_B$ force $\mathtt{y}$ to be $\mathfrak{s}^B(z)$. Now, $\texttt{Intro}_A - \texttt{Sample}_A$ and $\texttt{Intro}_A - \texttt{Read}_A$ force $a^{\cdot} = a^{\cdot}_{sam} = a^{\cdot}_{read}$, and similarly $\texttt{Sample}_B - \texttt{Intro}_B - \texttt{Read}_B$ force $b^{\cdot} = b^{\cdot}_{sam} = b^{\cdot}_{read}$. Furthermore, since they are mutually measured with the seed z (in $\texttt{Sample}_A$) and with v_{read} (in $\texttt{Read}_A$), they are forced to commute with the observables associated with them. The checks $\texttt{Read}_A - \texttt{Hide}_A$ and $\texttt{Hide}_A - \texttt{Pauli}_{\mathbb{X}}$, force $v_{read} = \nu = (\mathfrak{s}^A)^{\perp}(\chi)$. Hence, the observables of $\mathsf{Ans}^{A,\cdot,\cdot}$ are forced to commute with $\mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}$ for every $\alpha \in \ker \mathfrak{s}^A$, and with $\mathbb{Z}^{\otimes v} \otimes \mathrm{Id}$ for every $v \in \mathbb{F}_2^k$. The second commutation means that the mutual orthonormal eigenbasis for the observables of $\mathsf{Ans}^{A,\cdot,\cdot}$ is of the form $\{\mathbf{1}_z \otimes u_{z,t} \mid z \in \mathbb{F}_2^k, t \in \mathbb{F}_2^{2\Lambda}\}$, and the first commutation means that they act the same on $\mathbf{1}_z \otimes u$ and $\mathbb{X}^{\alpha} \otimes \mathrm{Id} \cdot \mathbf{1}_z \otimes u = \mathbf{1}_{z+\alpha} \otimes u$. This means that the response $a^{\mathfrak{R}}, a^{\mathfrak{L}}$ is the same for every two z's that differ by an element of $\ker \mathfrak{s}^A$, which implies that they depend only on their $\mathfrak{s}^A$-image, and this is what we wanted!

A reader may notice that to achieve the above goal, we could have dropped the $\texttt{Sample}_{\cdot}$ and $\texttt{Hide}_{\cdot}$ vertices and applied a more direct check (simplifying the augmentation). Though this is true, it will hinder the perfect completeness case, as we seek perfect ZPC strategies, in particular strategies that commute along edges, which is problematic without these "buffer" questions.

Theorem 4.8. *Let $k \geq 2$ be an integer, $\mathscr{B}$ a tuple of n vectors in $\mathbb{F}_2^k$ that induce an $[n,k,d]$-code, and $\mathfrak{G}$ a game with the properties fixed in the beginning of this subsection. Then, the baby question reduction game $\mathfrak{Baby}(k, \mathscr{B}, \mathfrak{G}) = \mathfrak{Baby}(\mathfrak{G})$ has the following properties:*

(1) Perfect ZPC Completeness*: If $\mathfrak{G}$ has a perfect* ZPC *strategy, then so does $\mathfrak{Baby}(\mathfrak{G})$.*

(2) Soundness*: If $\mathfrak{Baby}(\mathfrak{G})$ has a strategy with value $1 - \varepsilon$, then $\mathfrak{G}$ has a strategy with value of at least $1 - O(\sqrt{(1 + k^2/d^2)\varepsilon})$.*[54]

(3) Entanglement*: For every $\varepsilon > 0$,*

$$\mathscr{E}(\mathfrak{Baby}(\mathfrak{G}), 1 - \varepsilon) \geq 2^k \cdot (1 - O((1 + k^2/d^2)\varepsilon)) \cdot \mathscr{E}\left(\mathfrak{G}, 1 - O\left(\sqrt{(1 + k^2/d^2)\varepsilon}\right)\right).$$

---

[54]The O-notation is genuinely some universal constant that can be extracted from all the approximations we are doing.

Proof of perfect completeness (1)

Recall the notation $\mathcal{V}^{S'}$ for the restriction to $\mathbb{F}_2^{S'}$ of $\mathcal{V}$ with outcomes in $\mathbb{F}_2^{S' \sqcup S''}$ (Definition 3.32).

Assume $\mathfrak{G}$ has a perfect m-dimensional ZPC strategy $\mathscr{S} = \{\mathcal{U}\}$. Then, it induces an honest ZPC strategy $\mathscr{S}' = \{\mathcal{V}\}$ which is perfect for $\mathfrak{Intro}(\mathfrak{G})$ (Definition 4.4 and Claim 4.6). Let us extend $\mathcal{V}$ to the other vertices so it becomes a perfect ZPC strategy for $\mathfrak{Baby}(\mathfrak{G})$. First, let $\mathcal{V}^{\mathtt{Pauli}\mathbb{X}}$ and $\mathcal{V}^{\mathtt{Pauli}\mathbb{Z}}$ be the appropriate restrictions of $\rho \otimes \mathrm{Id}_m$, where ρ is the representation from Definition 3.68, to the X and Z subgroups of P_k, namely

$$\forall \alpha \in \mathbb{F}_2^k : \quad \mathcal{V}^{\mathtt{Pauli}\mathbb{X}}(\alpha) = \rho^{\mathbb{X}}(\alpha) \otimes \mathrm{Id}_m = \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m \quad , \quad \mathcal{V}^{\mathtt{Pauli}\mathbb{Z}}(\alpha) = \rho^{\mathbb{Z}}(\alpha) \otimes \mathrm{Id}_m = \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m \, .$$

By Claim 3.82, it can be extended to the rest of the $\mathfrak{Pauli\ Basis}_k$ vertices in a ZPC manner such that on the copy of $\mathfrak{Pauli\ Basis}_k$ in $\mathfrak{Baby}(\mathfrak{G})$ it has value 1. As described in Corollary 3.39, by calculating the inverse Fourier transform of the data processed $\mathscr{F}_{[\mathfrak{s}^\cdot(\cdot)=\cdot]}^{\mathbb{Z}}$, and by denoting $\alpha \cdot \mathfrak{s}^\cdot$ for the multiplication from the left of (the row vector) $\alpha \in \mathbb{F}_2^r$ with the matrix $\mathfrak{s}^\cdot \in M_{r \times k}(\mathbb{F}_2)$, we have

$$\forall \alpha, \beta \in \mathbb{F}_2^r : \quad \mathcal{V}^{\mathsf{Que}^A}(\alpha) = \mathbb{Z}^{\otimes \alpha \cdot \mathfrak{s}^A} \otimes \mathrm{Id}_m \, , \, \mathcal{V}^{\mathsf{Que}^B}(\beta) = \mathbb{Z}^{\otimes \beta \cdot \mathfrak{s}^B} \otimes \mathrm{Id}_m \, \in \, U(\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m) \, .$$

Note that in particular we have the following relationship through $\mathfrak{s}^\cdot$-evaluation (data processing, Definition 3.32) in representation form

$$\mathcal{V}^{\mathsf{Que}^A} = \mathcal{V}_{[\mathfrak{s}^A]}^{\mathtt{Pauli}\mathbb{Z}} \quad , \quad \mathcal{V}^{\mathsf{Que}^B} = \mathcal{V}_{[\mathfrak{s}^B]}^{\mathtt{Pauli}\mathbb{Z}} \, .$$

As the rest of the checks in $\mathfrak{Baby}(\mathfrak{G})$ are linear consistency checks, by Claim 3.36, they are forcing us to choose the following PVMs (in representation form) to be the same —

$$\mathcal{V}^{\mathsf{Ans}^A} = \mathcal{V}^{\mathsf{ReadAns}^A} = \mathcal{V}^{\mathsf{SamAns}^A} \, , \quad \mathcal{V}^{\mathsf{Ans}^B} = \mathcal{V}^{\mathsf{ReadAns}^B} = \mathcal{V}^{\mathsf{SamAns}^B} \, ,$$

$$\mathcal{V}^{\mathtt{Pauli}\mathbb{Z}} = \mathcal{V}^{\mathsf{SamZ}^A} = \mathcal{V}^{\mathsf{SamZ}^B} \, ,$$

$$\mathcal{V}^{\mathsf{Que}^A} = \mathcal{V}^{\mathsf{ReadQue}^A} = \mathcal{V}_{[\mathfrak{s}^A]}^{\mathtt{Pauli}\mathbb{Z}} \, , \quad \mathcal{V}^{\mathsf{Que}^B} = \mathcal{V}^{\mathsf{ReadQue}^B} = \mathcal{V}_{[\mathfrak{s}^B]}^{\mathtt{Pauli}\mathbb{Z}} \, ,$$

$$\mathcal{V}^{\mathsf{Perp}^A} = \mathcal{V}^{\mathsf{ReadPerp}^A} = \mathcal{V}_{[(\mathfrak{s}^A)^\perp]}^{\mathtt{Pauli}\mathbb{X}} \, , \quad \mathcal{V}^{\mathsf{Perp}^B} = \mathcal{V}^{\mathsf{ReadPerp}^B} = \mathcal{V}_{[(\mathfrak{s}^B)^\perp]}^{\mathtt{Pauli}\mathbb{X}} \, .$$

By choosing the above extension of $\mathcal{V}$, we are guaranteed that it passes all the augmented edges perfectly (both from the Sampling apparatus and the Hiding apparatus). We already described why $\mathcal{V}$ passes the copies of $\mathfrak{Pauli\ Basis}_k$ and $\mathfrak{Intro}(\mathfrak{G})$ perfectly, so, if it is well defined, then it is a perfect strategy for the game $\mathfrak{Baby}(\mathfrak{G})$. Furthermore, it is straightforward to check that all the observables we chose are signed permutation matrices, and the readable ones are diagonal; hence, if this strategy is well defined, it is a perfect Z-aligned permutation strategy.

We are left to argue why $\mathcal{V}$ is well defined — we chose restrictions of the PVMs at each vertex in a well defined manner, but it may be that these restricted PVMs do not amount to a single global one at the vertex, as they may be non-commuting. In addition, we need to check that $\mathcal{V}$ is commuting along edges. These are all quite straightforward checks (or, are corollaries of Claims 3.82 and 4.6), except for the Read.-variables. — as the images of $\mathcal{V}^{\mathsf{ReadPerp}^\cdot}$ are of the form $\mathbb{X}^\cdot \otimes \mathrm{Id}_m$, they may not commute with the images of $\mathcal{V}^{\mathsf{ReadQue}^\cdot}$ which are of the form $\mathbb{Z}^\cdot \otimes \mathrm{Id}_m$, and the images of $\mathcal{V}^{\mathsf{ReadAns}^\cdot}$ which are of the form $\sum \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$. Let us demonstrate why they are commuting nonetheless — we focus on Read_A, but the proof for Read_B is almost identical. For every $\alpha \in \mathbb{F}_2^r$ and $\beta \in \mathbb{F}_2^k$, we have

$$\mathcal{V}^{\mathsf{ReadQue}^A}(\alpha) = \mathbb{Z}^{\otimes \alpha \cdot \mathfrak{s}^A} \otimes \mathrm{Id}_m \, , \quad \mathcal{V}^{\mathsf{ReadPerp}^A}(\beta) = \mathbb{X}^{\otimes \beta \cdot (\mathfrak{s}^A)^\perp} \otimes \mathrm{Id}_m \, , \tag{53}$$

and thus they commute if and only if $\langle \alpha \cdot \mathfrak{s}^A, \beta \cdot (\mathfrak{s}^A)^\perp \rangle = 0$, where we think of both as row vectors. By recalling the notation $*$ for transposition of matrices with coefficients in $\mathbb{F}_2$ (Section 3.7), and by the choice of $(\mathfrak{s}^A)^\perp$ having rows in $\ker \mathfrak{s}^A$, we have

$$\langle \alpha \cdot \mathfrak{s}^A, \beta \cdot (\mathfrak{s}^A)^\perp \rangle = \alpha \cdot \mathfrak{s}^A \cdot ((\mathfrak{s}^A)^\perp)^* \cdot \beta^* = 0 \, ,$$

as $\mathfrak{s}^A \cdot ((\mathfrak{s}^A)^\perp)^*$ is the matrix whose columns are the $\mathfrak{s}^A$-evaluation of the rows of $(\mathfrak{s}^A)^\perp$. Now, for every pair $\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda$, by (51) and the fact we chose $\mathcal{V}^{\mathsf{Ans}^A} = \mathcal{V}^{\mathsf{ReadAns}^A}$, we have

$$\mathcal{V}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) = \sum_{z \in \mathbb{F}_2^k} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{U}^{\mathfrak{s}^A(z)}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) .$$

Hence,

$$
\begin{aligned}
\mathcal{V}^{\mathsf{ReadPerp}^A}(\beta) \cdot \mathcal{V}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \cdot \mathcal{V}^{\mathsf{ReadPerp}^A}(\beta) &= \sum_{z \in \mathbb{F}_2^k} \left(\mathbb{X}^{\otimes \beta \cdot (\mathfrak{s}^A)^\perp} \cdot \mathscr{F}_z^{\mathbb{Z}} \cdot \mathbb{X}^{\otimes \beta \cdot (\mathfrak{s}^A)^\perp} \right) \otimes \mathcal{U}^{\mathfrak{s}^A(z)}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \\
&= \sum_{z \in \mathbb{F}_2^k} \mathscr{F}_{z + \beta \cdot (\mathfrak{s}^A)^\perp}^{\mathbb{Z}} \otimes \mathcal{U}^{\mathfrak{s}^A(z)}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \\
&= \sum_{z \in \mathbb{F}_2^k} \mathscr{F}_{z + \beta \cdot (\mathfrak{s}^A)^\perp}^{\mathbb{Z}} \otimes \mathcal{U}^{\mathfrak{s}^A(z + \beta \cdot (\mathfrak{s}^A)^\perp)}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \\
&= \mathcal{V}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) ,
\end{aligned}
$$

where the first and last equations are by definition, the second equation is due to $\mathbb{X}^{\otimes \gamma} \mathscr{F}_z^{\mathbb{Z}} \mathbb{X}^{\otimes \gamma} = \mathscr{F}_{z+\gamma}^{\mathbb{Z}}$, and the third equation is since $\beta \cdot (\mathfrak{s}^A)^\perp$ is a linear combination of rows of $(\mathfrak{s}^A)^\perp$, which means it is in the kernel of $\mathfrak{s}^A$ and thus satisfies $\mathfrak{s}^A(z + \beta \cdot (\mathfrak{s}^A)^\perp) = \mathfrak{s}^A(z)$ for every $z \in \mathbb{F}_2^k$. $\qquad \square$

Proof of soundness (2) and entanglement lower bound (3)

The proof idea is as follows: Given an almost perfect strategy of $\mathfrak{Baby}(\mathfrak{G})$, we are going to perturb it so that it passes perfectly all non-$\mathfrak{Intro}(\mathfrak{G})$ edges. We are then going to show that the resulting strategy is honest (Definition 4.4) when restricted to the copy of $\mathfrak{Intro}(\mathfrak{G})$, and thus has the same value as some strategy of $\mathfrak{G}$ (in a much smaller dimension). As we did not perturb the strategy too much, its value did not change too much, and we can deduce the soundness and entanglement lower bound claims.

Let us move to the formal proof. Assume $\mathscr{S} = \{\mathcal{U}\}$ is an N-dimensional strategy in observable form for $\mathfrak{Baby}(\mathfrak{G})$ with value $1 - \varepsilon$. For notational simplicity, let $\varepsilon' = (1 + k^2/d^2)\varepsilon$. All O-notations in this proof are universal constants, and in particular are independent of $k, \mathcal{B}, \mathfrak{G}$ or the strategy $\mathscr{S}$ — genuinely universal. We repeatedly use Claim 3.22, replacing previous bounds on expectations by the same bounds on the maxima (up to a constant factor that is absorbed into the O and $\approx$ notations).

As the copy of $\mathfrak{Pauli\,Basis}_k$ is played with probability $1/4$ when running $\mathfrak{Baby}(\mathfrak{G})$, the restriction of $\mathscr{S}$ to the vertices of $\mathfrak{Pauli\,Basis}_k$ passes it with value of at least $1 - 4\varepsilon$. Hence, by Claim 3.83 and Fact 3.73, there is a $k^2\varepsilon/d^2$-near bijection (Definition 3.4) $\omega \colon \mathbb{C}^N \to \mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$ for which the corner POVM $\omega \mathcal{U}^{\mathsf{Pauli\,x}} \omega^*$ is $k^2\varepsilon/d^2$-close to $\rho^{\mathbb{X}} \otimes \mathrm{Id}_m$, and the corner POVM $\omega \mathcal{U}^{\mathsf{Pauli\,z}} \omega^*$ is $k^2\varepsilon/d^2$-close to $\rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$; namely, using Claim 3.22, we have

$$\forall \alpha \in \mathbb{F}_2^k : \quad \left\| \omega \mathcal{U}^{\mathsf{Pauli\,x}}(\alpha) \omega^* - \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m \right\|_{hs}^2 , \quad \left\| \omega \mathcal{U}^{\mathsf{Pauli\,z}}(\alpha) \omega^* - \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m \right\|_{hs}^2 \leq O(k^2\varepsilon/d^2) ,$$

and

$$1 - \tau(\omega^* \omega) , \quad 1 - \tau(\omega \omega^*) \leq O(k^2\varepsilon/d^2) . \tag{54}$$

Let $\varepsilon' = (1 + k^2/d^2)\varepsilon$. So, the above quantities are all $O(\varepsilon')$. Moreover, by orthonormalization (Fact 3.21), the corner PVMs $\omega \mathcal{U}^{\mathsf{Ans}^A} \omega^*$ and $\omega \mathcal{U}^{\mathsf{Ans}^B} \omega^*$ are $O(\varepsilon')$-close to genuine representations that we denote by θ^A and θ^B; namely, using Claim 3.22, we have

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda : \quad \left\| \omega \left(\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \right) \omega^* - \theta^A(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \right\|_{hs}^2 \leq O(\varepsilon') , \tag{55}$$

$$\forall \beta^{\mathfrak{R}}, \beta^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda : \quad \left\| \omega \left(\mathcal{U}^{\mathsf{Ans}^B}(\beta^{\mathfrak{R}}, \beta^{\mathfrak{L}}) \right) \omega^* - \theta^B(\beta^{\mathfrak{R}}, \beta^{\mathfrak{L}}) \right\|_{hs}^2 \leq O(\varepsilon') . \tag{56}$$

79

The following few calculations aim to show that $\theta^{\cdot}$ almost commutes with $\mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m$ for every $z \in \mathbb{F}_2^k$ and with $\mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m$ for every $\alpha \in \ker \mathfrak{s}^{\cdot}$. As $\mathscr{S}$ has value $1 - \varepsilon$, and the augmented edges are sampled with probability of at least $1/24$, we can deduce from the definition of inconsistency (Definition 3.10) and the equivalence of inconsistency and distance for projective measurements (item 1 of Proposition 3.12) various results:

1. By the comparison along $\mathtt{Intro}_A - \mathtt{Sample}_A$, we can deduce that $\mathcal{U}^{\mathsf{Que}^A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{SamZ}^A}_{[\mathfrak{s}^A]}$ and $\mathcal{U}^{\mathsf{Ans}^A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{SamAns}^A}$; namely, using Claim 3.22, Claim 3.38 and the notation $\alpha \cdot \mathfrak{s}^A$ for the product from the left of the row vector $\alpha \in \mathbb{F}_2^r$ with the $r \times k$ matrix $\mathfrak{s}^A$,

$$\forall \alpha \in \mathbb{F}_2^r : \quad \|\mathcal{U}^{\mathsf{Que}^A}(\alpha) - \mathcal{U}^{\mathsf{SamZ}^A}(\alpha \cdot \mathfrak{s}^A)\|_{hs}^2 \leq O(\varepsilon) , \tag{57}$$

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda} : \quad \|\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) - \mathcal{U}^{\mathsf{SamAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\|_{hs}^2 \leq O(\varepsilon) . \tag{58}$$

2. By the comparison along $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_A$, we can deduce that $\mathcal{U}^{\mathtt{Pauli}\mathbb{Z}} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{SamZ}^A}$, and using Claim 3.22 this implies

$$\forall \alpha \in \mathbb{F}_2^k : \quad \|\mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(\alpha) - \mathcal{U}^{\mathsf{SamZ}^A}(\alpha)\|_{hs}^2 \leq O(\varepsilon) . \tag{59}$$

3. By the comparison $\mathtt{Intro}_A - \mathtt{Read}_A$, we can deduce that $\mathcal{U}^{\mathsf{Que}^A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{ReadQue}^A}$ and $\mathcal{U}^{\mathsf{Ans}^A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{ReadAns}^A}$, and using Claim 3.22,

$$\forall \alpha \in \mathbb{F}_2^r : \quad \|\mathcal{U}^{\mathsf{Que}^A}(\alpha) - \mathcal{U}^{\mathsf{ReadQue}^A}(\alpha)\|_{hs}^2 \leq O(\varepsilon) , \tag{60}$$

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda} : \quad \|\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) - \mathcal{U}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\|_{hs}^2 \leq O(\varepsilon) . \tag{61}$$

4. By the comparison $\mathtt{Hide}_A - \mathtt{Read}_A$, we can deduce that $\mathcal{U}^{\mathtt{Hide}_A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathsf{ReadPerp}^A}$, and using Claim 3.22,

$$\forall \alpha \in \mathbb{F}_2^k : \quad \|\mathcal{U}^{\mathtt{Hide}_A}(\alpha) - \mathcal{U}^{\mathsf{ReadPerp}^A}(\alpha)\|_{hs}^2 \leq O(\varepsilon) . \tag{62}$$

5. By the comparison $\mathtt{Hide}_A - \mathtt{Pauli}_{\mathbb{X}}$, we can deduce that $\mathcal{U}^{\mathtt{Hide}_A} \simeq_{O(\varepsilon)} \mathcal{U}^{\mathtt{Pauli}\mathbb{X}}_{[(\mathfrak{s}^A)^{\perp}]}$, and using Claim 3.22 and Claim 3.38,

$$\forall \alpha \in \mathbb{F}_2^k : \quad \left\|\mathcal{U}^{\mathtt{Hide}_A}(\alpha) - \mathcal{U}^{\mathtt{Pauli}\mathbb{X}}\left(\alpha \cdot (\mathfrak{s}^A)^{\perp}\right)\right\|_{hs}^2 \leq O(\varepsilon) . \tag{63}$$

Thus, using the notation $\square \approx_{\varepsilon} \heartsuit$ whenever $\|\square - \heartsuit\|_{hs}^2 \leq \varepsilon$ (similar to the distance notation, Definition 3.10), we have that

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}, z \in \mathbb{F}_2^k : \quad \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z) \approx_{O(\varepsilon)} \mathcal{U}^{\mathsf{SamAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathsf{SamZ}}(z)$$
$$= \mathcal{U}^{\mathsf{SamZ}}(z)\mathcal{U}^{\mathsf{SamAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \tag{64}$$
$$\approx_{O(\varepsilon)} \mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z)\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) ,$$

where both approximations use (58) and (59). Therefore,

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}, z \in \mathbb{F}_2^k : \quad \theta^A(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \cdot \mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m \approx_{O(\varepsilon')} \omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^* \omega \mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z)\omega^*$$
$$\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z)\omega^*$$
$$\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z)\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^* \tag{65}$$
$$\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathtt{Pauli}\mathbb{Z}}(z)\omega^* \omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^*$$
$$\approx_{O(\varepsilon')} \mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m \cdot \theta^A(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) ,$$

where the first and last approximations are by (55) and (54), and the middle one is by (64). For the second and fourth approximations, note that $1 - \tau(\omega^*\omega), 1 - \tau(\omega\omega^*) \leq O(\varepsilon)$ from (54) and that $\|\square\heartsuit\|_{hs} \leq \|\square\|_{op}\|\heartsuit\|_{hs}$ for square matrices, therefore using Claim 3.5, we have

$$
\begin{aligned}
\|\omega \mathcal{U}_{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})(\mathrm{Id} - \omega^*\omega)\mathcal{U}_{\mathsf{Pauli}_{\mathbb{Z}}}(z)\omega^*\|_{hs}^2 &\leq \|\mathcal{U}_{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})(\mathrm{Id} - \omega^*\omega)\mathcal{U}_{\mathsf{Pauli}_{\mathbb{Z}}}(z)\|_{hs}^2 + 4\varepsilon \\
&\leq \underbrace{\|\mathcal{U}_{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\|_{op}^2}_{=1}\underbrace{\|\mathrm{Id} - \omega^*\omega\|_{hs}^2}_{=1-\tau(\omega^*\omega)}\underbrace{\|\mathcal{U}_{\mathsf{Pauli}_{\mathbb{Z}}}(z)\|_{op}^2}_{=1} + 4\varepsilon \\
&\leq 5\varepsilon .
\end{aligned}
\tag{66}
$$

As the rows of $(\mathfrak{s}^A)^\perp$ span the kernel of $\mathfrak{s}^A$, for every $\alpha \in \ker \mathfrak{s}^A$ there is a $\beta \in \mathbb{F}_2^k$ such that $\alpha = \beta \cdot (\mathfrak{s}^A)^\perp$. Hence,

$$
\begin{aligned}
\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda \; : \; \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha) &\approx_{O(\varepsilon)} \mathcal{U}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathsf{Hide}_A}(\beta) \\
&\approx_{O(\varepsilon)} \mathcal{U}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathsf{ReadPerp}^A}(\beta) \\
&= \mathcal{U}^{\mathsf{ReadPerp}^A}(\beta)\mathcal{U}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \\
&\approx_{O(\varepsilon)} \mathcal{U}^{\mathsf{Hide}_A}(\beta)\mathcal{U}^{\mathsf{ReadAns}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \\
&\approx_{O(\varepsilon)} \mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha)\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) ,
\end{aligned}
\tag{67}
$$

where the first and last approximations use (61) and (63), and the middle ones use (62). Therefore,

$$
\begin{aligned}
\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda, \alpha \in \ker \mathfrak{s}^A \; : \; \theta^A(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \cdot \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m &\approx_{O(\varepsilon')} \omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^*\omega \mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha)\omega^* \\
&\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha)\omega^* \\
&\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha)\mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^* \\
&\approx_{O(\varepsilon)} \omega \mathcal{U}^{\mathsf{Pauli}_{\mathbb{X}}}(\alpha)\omega^*\omega \mathcal{U}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})\omega^* \\
&\approx_{O(\varepsilon')} \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m \cdot \theta^A(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) ,
\end{aligned}
\tag{68}
$$

where the first and last approximations are due to (55) and (54), the second and fourth are using (66), and the middle approximation is by (67).

All in all, we deduced that the images of θ^A (and similarly for θ^B) almost commute with all the $\mathbb{Z}$-matrices and certain $\mathbb{X}$-matrices. This is (essentially) the end of the proof, in a similar manner to that of the Pauli basis game, in which getting to an approximate relations situation allows one to apply a group stability result to finish the argument. Here we also need to analyze why commuting with these specific matrices completes the proof, but this is quite straightforward.

Claim 4.9. *Let $\rho\colon G \to U(N)$ be a (unitary) representation of a finite group G, and $\phi\colon A \to U(N)$ be a representation of a finite abelian group. Assume*
$$
\forall g \in G, a \in A \; : \; \|\rho(g)\phi(a) - \phi(a)\rho(g)\|_{hs}^2 \leq \varepsilon.
$$

Then, there is another representation $\xi\colon A \to U(N)$, such that $\xi(a)\phi(g) = \phi(g)\xi(a)$ for every $a \in A$ and $g \in G$, and

$$
\forall a \in A \; : \; \|\xi(a) - \phi(a)\|_{hs}^2 \leq O(\varepsilon).
$$

Proof. The proof is a combination of an averaging trick common in the study of property (T) groups (cf. [Ioa, dlS22b]), and a strict version of the Gowers–Hatami theorem due to Akhtiamov–Dogon [AD22]. It can also be deduced directly from orthonormalization (Fact 3.21), but we show a different argument.

First, let $\tilde{\phi}(a) = \mathbb{E}_{g \in G}[\rho(g)\phi(a)\rho(g)^{-1}] \in M_N(\mathbb{C})$. By our assumption, $\|\tilde{\phi}(a) - \phi(a)\|_{hs} \leq \mathbb{E}_{g \in G} \|\rho(g)\phi(a)\rho(g)^{-1} - \phi(a)\| \leq \sqrt{\varepsilon}$. Moreover, $\tilde{\phi}$ commutes with ρ. Now, denote by $\mathcal{M}$ the commutant of $\rho(G)$, namely the collection of matrices

that commute with all the ρ-images of G. Then, $\mathrm{Im}\tilde\phi$ is in $\mathscr{M}$, and we can apply the rest of our arguments in this von-Neumann algebra. Note that

$$\|\tilde\phi(a)\tilde\phi^*(a) - \mathrm{Id}\|_{hs} \leq \mathop{\mathbb{E}}_{g,h\in G}[\|\rho(g)\phi(a)\rho(g)^{-1}\rho(h)\phi(a)^{-1}\rho(h)^{-1} - \underbrace{\mathrm{Id}}_{\rho(g)\rho(g)^{-1}\rho(h)\phi(a)\phi(a)^{-1}\rho(h)^{-1}}\|_{hs}]$$

$$= \mathop{\mathbb{E}}_{g,h\in G}[\|\phi(a)\rho(g^{-1}h) - \rho(g^{-1}h)\phi(a)\|_{hs}]$$

$$\leq \sqrt{\varepsilon}.$$

Then, by Lemma 2.2 in [AD22], there is a map $\zeta\colon A \to U(\mathscr{M})$, namely to unitaries in the von-Neumann algebra $\mathscr{M}$ such that $\|\zeta(a) - \tilde\phi(a)\|_{hs} \leq \|\tilde\phi(a)\tilde\phi^*(a) - \mathrm{Id}\|_{hs}$ (this is quite straightforward from the SVD decomposition). Note in addition that

$$\forall a,b \in A: \quad \|\zeta(a)\zeta(b) - \zeta(ab)\|_{hs} \leq \|\zeta(a) - \tilde\phi(a)\|_{hs} + \|\zeta(b) - \tilde\phi(b)\|_{hs}$$
$$+ \|\zeta(ab) - \tilde\phi(ab)\|_{hs} + \|\tilde\phi(a)\tilde\phi(b) - \tilde\phi(ab)\|_{hs}$$

$$\forall a,b \in A: \quad \|\tilde\phi(a)\tilde\phi(b) - \tilde\phi(ab)\|_{hs} \leq \|\phi(a) - \tilde\phi(a)\|_{hs} + \|\phi(b) - \tilde\phi(b)\|_{hs}$$
$$+ \|\phi(ab) - \tilde\phi(ab)\|_{hs} + \|\phi(a)\phi(b) - \phi(ab)\|_{hs}$$

Now, $\phi(a)\phi(b) = \phi(ab)$ since ϕ is a representation, and all other summands are bounded by $\sqrt{\varepsilon}$. Hence,

$$\forall a,b \in A: \quad \|\zeta(a)\zeta(b) - \zeta(ab)\|_{hs} \leq 6\sqrt{\varepsilon}.$$

By [AD22, Corollary 1.7 and Claim 3.3], there is a unitary representation $\xi\colon A \to U(\mathscr{M})$[55] such that

$$\forall a \in A: \quad \|\xi(a) - \zeta(a)\|_{hs} \leq O(\sqrt{\varepsilon}).$$

Applying several triangle inequalities shows that

$$\forall a \in A: \quad \|\xi(a) - \phi(a)\|_{hs} \leq O(\sqrt{\varepsilon}),$$

which in turn finishes the proof. $\qquad\square$

By applying Claim 4.9 where G is the group generated by $\mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m$ and $\mathbb{X}^{\otimes\alpha} \otimes \mathrm{Id}_m$ for all $z \in \mathbb{F}_2^k, \alpha \in \ker\mathfrak{s}^A$, ρ is the identity map, and with $A = \mathbb{F}_2^\Lambda \times \mathbb{F}_2^\Lambda$ and $\phi = \theta^A$, we deduce that there is a representation $\xi^A\colon \mathbb{F}_2^\Lambda \times \mathbb{F}_2^\Lambda \to U(\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m)$ such that ξ^A commutes with all $\mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m$ and $\mathbb{X}^{\otimes\alpha} \otimes \mathrm{Id}_m$ and also

$$\forall a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda: \quad \|\theta^A(a^{\mathfrak{R}}, a^{\mathfrak{L}}) - \xi^A(a^{\mathfrak{R}}, a^{\mathfrak{L}})\|_{hs}^2 \leq O(\varepsilon'). \tag{69}$$

Moreover, everything can be done similarly for B resulting with a ξ^B that commutes with all $\mathbb{Z}^z \otimes \mathrm{Id}_m$ and $\mathbb{X}^\beta \otimes \mathrm{Id}_m$ for every $\beta \in \ker\mathfrak{s}^B$.

Now, we can define an $2^k \times m$-dimensional strategy $\mathscr{S}' = \{\mathcal{V}\}$ almost as we did in the completeness proof:

- We let $\mathcal{V}^{\mathtt{Paulix}}(\alpha) = \mathbb{X}^{\otimes\alpha} \otimes \mathrm{Id}_m, \mathcal{V}^{\mathtt{Pauliz}}(\alpha) = \mathbb{Z}^{\otimes\alpha} \otimes \mathrm{Id}_m$, and extend it to a perfect strategy of the Pauli basis game using Claim 3.82.

- For the other variables, let

$$\mathcal{V}^{\mathrm{Ans}^A} = \mathcal{V}^{\mathrm{ReadAns}^A} = \mathcal{V}^{\mathrm{SamAns}^A} = \xi^A, \quad \mathcal{V}^{\mathrm{Ans}^B} = \mathcal{V}^{\mathrm{ReadAns}^B} = \mathcal{V}^{\mathrm{SamAns}^B} = \xi^B,$$

$$\mathcal{V}^{\mathtt{Pauliz}} = \mathcal{V}^{\mathrm{SamZ}^A} = \mathcal{V}^{\mathrm{SamZ}^B},$$

$$\mathcal{V}^{\mathrm{Que}^A} = \mathcal{V}^{\mathrm{ReadQue}^A} = \mathcal{V}^{\mathtt{Pauliz}}_{[\mathfrak{s}^A]}, \quad \mathcal{V}^{\mathrm{Que}^B} = \mathcal{V}^{\mathrm{ReadQue}^B} = \mathcal{V}^{\mathtt{Pauliz}}_{[\mathfrak{s}^B]},$$

$$\mathcal{V}^{\mathrm{Perp}^A} = \mathcal{V}^{\mathrm{ReadPerp}^A} = \mathcal{V}^{\mathtt{Paulix}}_{[(\mathfrak{s}^A)^\perp]}, \quad \mathcal{V}^{\mathrm{Perp}^B} = \mathcal{V}^{\mathrm{ReadPerp}^B} = \mathcal{V}^{\mathtt{Paulix}}_{[(\mathfrak{s}^B)^\perp]}.$$

---

[55]Note that $\mathscr{M}$ is the same! This is the main difference between Akhtiamov–Dogon to the standard Gowers–Hatami. This is possible thanks to the additional assumption that A is abelian.

Now, $\mathscr{S}'$ is indeed a strategy — namely, all images are order 2 unitaries that commute for every fixed vertex — and it passes by design all checks in $\mathfrak{Babx}(\mathfrak{G})$ with probability 1, except for maybe $\mathtt{Intro}_A - \mathtt{Intro}_B$. For that edge, we note that by (69), (59), (57), (55) and (56) the strategy $\mathscr{S}' = \{\mathcal{V}\}$ is $O(\varepsilon')$-close on this edge to the original strategy $\{\mathcal{U}\}$, and Claim 3.29 states that this means they produce $\sqrt{\varepsilon'}$-close correlations, and thus $\mathcal{V}$ passes this edge with probability close to that of $\mathcal{U}$. As this edge is sampled with probability $1/4$ in $\mathfrak{Babx}(\mathfrak{G})$, $\mathcal{U}$ passes it with probability $1 - O(\varepsilon)$, which means $\mathcal{V}$ passes it with probability $1 - O(\sqrt{\varepsilon'})$.

We are left to show that $\mathcal{V}$ is an honest strategy, and thus (by definition) a strategy with the same value can be extracted for $\mathfrak{G}$. This is immediate by analyzing the commutant of $\{\mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m, \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m\}_{z \in \mathbb{F}_2^k, \alpha \in \ker \mathfrak{s}^A}$. A matrix that commutes with all $\mathbb{Z}^{\otimes z} \otimes \mathrm{Id}_m$ is of the form $\sum_{z \in \mathbb{F}_2^k} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$ for $\mathscr{A}^z \in M_m(\mathbb{C})$ and $\mathscr{F}_z^{\mathbb{Z}}$ the projections on the indicators $\mathbf{1}_z$ in $\mathbb{C}^{\mathbb{F}_2^k}$ (Definition 3.67). For such matrices, commuting with $\mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m$ is the same as requiring $\mathscr{A}^z = \mathscr{A}^{z+\alpha}$ for every $z \in \mathbb{F}_2^k$. Hence, the commutant consists of all matrices of the form $\sum_z (\sum_{\alpha \in \ker \mathfrak{s}^A} \mathscr{F}_{z+\alpha}^{\mathbb{Z}}) \otimes \mathscr{A}^z$ where the sum over z's takes a representative from every coset of $\ker \mathfrak{s}^A$ in $\mathbb{F}_2^k$. But, this is the same as writing every matrix as $\sum_{\mathbf{x} \in \mathbb{F}_2^r} (\sum_{z: \, \mathfrak{s}^A(z)=\mathbf{x}} \mathscr{F}_z^{\mathbb{Z}}) \otimes \mathscr{A}^{\mathbf{x}}$. As this is true, in particular, for any projection in the commutant, we can write the PVM associated with the images of $\mathcal{V}$ at $\mathtt{Intro}_A$ as $\mathcal{Q}_{\mathbf{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathtt{Intro}_A} = \sum_{z \in \mathbb{F}_2^k: \mathfrak{s}^A(z)=\mathbf{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathbf{x}}$, and similarly for $\mathtt{Intro}_B$. The resulting $\mathcal{P}: \mathbb{F}_2^r \times \mathbb{F}_2^{\Lambda} \times \mathbb{F}_2^{\Lambda} \to M_m(\mathbb{C})$ is a PVM strategy for $\mathfrak{G}$ that passes it with the same probability as $\mathcal{V}$ passes $\mathtt{Intro}_A - \mathtt{Intro}_B$ (as in Definition 4.4 on honest strategies), which is $1 - O(\sqrt{\varepsilon'})$. This finishes the proof of soundness.

Note that by (54), the normalized dimension difference $1 - \frac{N}{2^k \cdot m} \leq 1 - \tau(\omega\omega^*) \leq O(\varepsilon')$. Furthermore, as we extracted from the honest $\mathcal{V}$ an m-dimensional strategy $\mathcal{P}$ for $\mathfrak{G}$ with value $1 - O(\sqrt{\varepsilon'})$, we deduce that $m \geq \mathscr{E}(\mathfrak{G}, 1 - O(\sqrt{\varepsilon'}))$, which proves the entanglement lower bound (3).

4.3   Conditionally linear maps

Figure 11: An illustration of an h-level CLM $\mathfrak{s}$ (adapted from [JNV$^+$21, Figure 1]). Let $z \in \mathbb{F}_2^k$ be the input. First a register subspace V_1 and a linear map $\mathfrak{s}_1$ are chosen and applied on the restriction of z to V_1 to obtain $\mathbf{x}_1 = \mathfrak{s}_1(z^{V_1}) \in V_1$. Then depending the value of $\mathbf{x}_1$, a register subspace $V_2 = V_2^{\mathbf{x}_1}$ and a linear map $\mathfrak{s}_2 = \mathfrak{s}_2^{\mathbf{x}_1}$ are chosen and applied on the restriction of z to V_2 to obtain $\mathbf{x}_2 = \mathfrak{s}_2(z^{V_2}) \in V_2$ and so on. Finally, $\mathfrak{s}(z)$ is defined to be $\sum_{j=1}^h \mathbf{x}_j$.

The collection of linear maps $\mathfrak{s}: \mathbb{F}_2^k \to \mathbb{F}_2^r \times \mathbb{F}_2^r$ induces a family of samplers which is too restrictive for us to prove

compression with. But, a certain generalization of linear maps, called *conditionally linear maps* [JNV$^+$21, Definition 4.1], are rich enough to deduce compression. This section is devoted to this generalized setup.

Intuitively, conditionally linear maps that act on $\mathbb{F}_2^k$ apply a sequence of linear maps on subspaces of it, where each map in the sequence depends on the value which the previous linear maps produced. These maps are in a sweet spot, being rich enough so that all the samplers that we will need can be described as pushforwards of the uniform measure along them, while being amenable to a construction similar to $\mathfrak{Baby}(\mathfrak{G})$ from Section 4.2.

Conditionally linear maps are not complicated objects, yet the notation associated with them can take some time getting used to. We recommend that the reader attempt to follow the visual explanation given in Figure 11 first, to form their own intuition; which can then be matched to the formal definitions that follow.

Definition 4.10. A *register subspace* of $\mathbb{F}_2^k$ is one which is spanned by some subset of the standard basis $\{e_1, ..., e_k\}$. As there is a bijection between register subspaces and subsets of $[k]$, we often associate with such a subspace the appropriate subset of indices $I \subseteq [k]$. Given a register subspace $V = \mathrm{Span}\{e_{i_1}, ..., e_{i_m}\}$ (in which case $I = \{i_1, ..., i_m\}$) and a vector $z \in \mathbb{F}_2^k$ we denote by z^V the restriction of z to the coordinates of V, namely $z^V = \sum_{j=1}^m \langle z, e_{i_j} \rangle e_{i_j}$. As V is canonically isomorphic to $\mathbb{F}_2^I$, we often treat vectors in V as parameterized by I instead of $[k]$.

Two register subspaces are said to be *disjoint* if their intersection is trivial. Two register subspaces are said to be *complementary* if they are disjoint and sum up to the whole space $\mathbb{F}_2^k$.

Definition 4.11 (Conditionally linear map — recursive definition). Let $k \geq 1$ and $h \geq 0$ be integers. The collection of h-level conditionally linear maps (CLMs) on $\mathbb{F}_2^k$ is defined inductively on h as follows.

- A 0-level CLM over $\mathbb{F}_2^k$ is the zero map.

- Assume $(h-1)$-level CLMs were already defined. An h-level CLM $\mathfrak{s}$ over $\mathbb{F}_2^k$ consists of the following data:

    - a register subspace (Definition 4.10) $V_1 \subseteq \mathbb{F}_2^k$, whose complement is denoted by $V_{>1} \subseteq \mathbb{F}_2^k$;
    - a linear map $\mathfrak{s}_1 : V_1 \to V_1$;
    - for every $u \in V_1$, an $(h-1)$-level CLM $\mathfrak{s}_{>1}^u$ on $V_{>1}$.

The data of an h-level CLM defines a function $\mathfrak{s} : \mathbb{F}_2^k \to \mathbb{F}_2^k$ as follows:

- For the 0-level case the function is the zero function.

- Assuming we defined evaluation along $(h-1)$-level CLMs, the evaluation along an h-level CLM is:

    - Given $z \in \mathbb{F}_2^k$, recall that $z^{V_1} \in V_1$ is its restriction to V_1. One can thus use the linear map $\mathfrak{s}_1 : V_1 \to V_1$ to evaluate $\mathbf{x}_1 = \mathfrak{s}_1(z^{V_1})$.
    - As there is an $(h-1)$-level CLM associated to $\mathbf{x}_1$, $\mathfrak{s}_{>1}^{\mathbf{x}_1} : V_{>1} \to V_{>1}$, and as we assumed evaluation along $(h-1)$-level CLMs was already defined, we let $\mathbf{x}_{>1} = \mathfrak{s}_{>1}^{\mathbf{x}_1}(z^{V_{>1}}) \in V_{>1}$, where again $z^{V_{>1}}$ is the restriction of the input z to the complementary subspace $V_{>1}$.
    - Finally, the $\mathfrak{s}$-evaluation of z is defined to be $\mathfrak{s}(z) = \mathbf{x}_1 + \mathbf{x}_{>1} \in V_1 \oplus V_{>1} = \mathbb{F}_2^k$.

Remark 4.12. Note that 1-level CLMs are exactly linear functions from $\mathbb{F}_2^k$ to itself. An example of a 2-level CLM is $(x_1, x_2, x_3) \mapsto (0, x_1 x_3 + x_2 x_3 + x_1, x_3)$ — this example can be used as a sanity check, and a proof of it being a 2-level CLM appears in [JNV$^+$21, Example 4.3].

The following is a more intricate definition of h-level CLMs, which avoids the recursive nature of Definition 4.11. A proof that the two definitions are equivalent is given in [JNV$^+$21, Lemma 4.6] (with somewhat different notation).

Definition 4.13 (Conditionally linear map — direct definition). Let $h \geq 0$ and $k \geq 1$ be integer. To describe an h-level *conditionally linear map* (CLM) $\mathfrak{s} : \mathbb{F}_2^k \to \mathbb{F}_2^k$ we need the following structure. First, there is a collection of register subspaces defined inductively:

- $V_1 \subseteq \mathbb{F}_2^k$ is a fixed register subspace, and we denote by $V_{>1}$ its complement.

- For every $u_1 \in V_1$, there is a register subspace $V_2^{u_1} \subseteq V_{>1}$, and we denote the sum $V_1 \oplus V_2^{u_1}$ by $V_{\leq 2}^{u_1}$, and its complement by $V_{>2}^{u_1}$.

- Then, for every $u_2 \in V_2^{u_1}$, there is a register subspace $V_3^{u_1,u_2} \subseteq V_{>2}^{u_1}$, which gives rise to the subspaces $V_{\leq 3}^{u_1,u_2} = V_{\leq 2}^{u_1} \oplus V_3^{u_1,u_2}$ and its complement $V_{>3}^{u_1,u_2}$.

- This keeps on, so that in the jth step, for every $u_1 \in V_1, u_2 \in V_2^{u_1}, \ldots, u_{j-1} \in V_{j-1}^{u_1,\ldots,u_{j-2}}$ there is a register subspace $V_j^{u_1,\ldots,u_{j-1}}$ disjoint of $V_1 \oplus V_2^{u_1} \oplus \cdots \oplus V_{j-1}^{u_1,\ldots,u_{j-2}}$, giving rise to the appropriate $V_{\leq j} = V_{\leq j}^{u_1,\ldots,u_{j-1}}$ and complement $V_{>j}^{u_1,\ldots,u_{j-1}}$.

- No matter the process, we are guaranteed to reach $\mathbb{F}_2^k$ after h steps, namely $V_1 \oplus V_2^{u_1} \oplus \cdots \oplus V_h^{u_1,\ldots,u_{h-1}} = \mathbb{F}_2^k$ for every $u_1 \in V_1, u_2 \in V_2^{u_1}, \ldots, u_{h-1} \in V_{h-1}^{u_1,\ldots,u_{h-1}}$.

Now, in addition to the above collections of register subspaces, there is a collection of linear maps on them:

- On V_1 there is a fixed $\mathfrak{s}_1 \colon V_1 \to V_1$.

- For every $u_1 \in V_1$ there is a linear map $\mathfrak{s}_2^{u_1} \colon V_2^{u_1} \to V_2^{u_1}$.

- This continues in a similar manner to before, where in the jth step, for every $u_1 \in V_1, u_2 \in V_2^{u_1}, \ldots, u_{j-1} \in V_{j-1}^{u_1,\ldots,u_{j-2}}$ there is a linear map $\mathfrak{s}_j = \mathfrak{s}_j^{u_1,\ldots,u_{j-1}} \colon V_j^{u_1,\ldots,u_{j-1}} \to V_j^{u_1,\ldots,u_{j-1}}$.

Finally, the function $\mathfrak{s}$ is calculated as follows:

- Let $z \in \mathbb{F}_2^k$ be the input to $\mathfrak{s}$.

- Calculate $\mathrm{x}_1 = \mathfrak{s}_1(z^{V_1})$, and let $V_2 = V_2^{\mathrm{x}_1}$.

- Calculate $\mathrm{x}_2 = \mathfrak{s}_2^{\mathrm{x}_1}(z^{V_2})$, and let $V_3 = V_3^{\mathrm{x}_1,\mathrm{x}_2}$.

- In the jth step, calculate $\mathrm{x}_j = \mathfrak{s}_j^{\mathrm{x}_1,\ldots,\mathrm{x}_{j-1}}(z^{V_j})$, and let $V_{j+1} = V_{j+1}^{\mathrm{x}_1,\ldots,\mathrm{x}_j}$.

- After h steps are completed, resulting in $\mathrm{x}_1, \ldots, \mathrm{x}_h$, output $\mathfrak{s}(z) = \mathrm{x} = \mathrm{x}_1 + \mathrm{x}_2 + \ldots + \mathrm{x}_h \in V_1 \oplus \cdots \oplus V_h = \mathbb{F}_2^k$.

We denote by $\mathfrak{s}_j(z)$ the value of x_j in the above computation of $\mathfrak{s}(z)$, and let

$$\mathfrak{s}_{\leq j}(z) = \mathrm{x}_{\leq j} = \mathrm{x}_1 + \ldots + \mathrm{x}_j \tag{70}$$

be the cumulative output up to the jth computation.

Definition 4.14 (Seeded conditionally linear maps)**.** Let $\mathfrak{s}$ be an h-level CLM with all the data from Definition 4.13. Let $u \in \mathbb{F}_2^k$ be a vector which plays the role of a *seed*. Then, u induces a decomposition of $\mathbb{F}_2^k$ into h disjoint register subspaces, which we call the *u-seeded register subspaces*, as follows:

- Regardless of u, we let $W_1^u = V_1$. As W_1^u is a register subspace, it has an associated subset of indices from $[k]$, which we denote by $I_1^u \subseteq [k]$. Furthermore, denote the restriction of u to this register subspace by $u_1 = u^{W_1^u}$.

- Then, we let $W_2^u = V_2^{u_1}$ with $I_2^u \subseteq [k]$ the associated subset of indices, and denote the restriction of u to it by $u_2 = u^{W_2^u}$.

85

- More generally, given that we have already defined $W_1^u, ..., W_{j-1}^u$ and thus the respective restrictions $u_1, ..., u_{j-1}$ of u, we let

$$W_j^u = V_j^{u_1,...,u_{j-1}} \quad \text{and} \quad u_j = u^{W_j^u} , \tag{71}$$

with $I_j^u \subseteq [k]$ being again the associated subset of indices.

We use $W_{\leq j}^u$, $W_{<j}^u$, $W_{\geq j}^u$ and $W_{>j}^u$ in a similar way to before, and $I_{\leq j}^u$, $I_{<j}^u$, $I_{\geq j}^u$, $I_{>j}^u$ for the indices supporting each of these register subspaces. We call

$$u_{\leq j} = u_1 + ... + u_j \in W_1^u \oplus ... \oplus W_j^u \tag{72}$$

the jth *prefix of* u, and note that $W_{\leq j+1}^u$, $W_{>j+1}^u$ depend only on this j^{th} prefix and not all of u. Note also that $\mathfrak{s}_{\leq j}(z)$ as in (70) is **equal** to the jth prefix of $\mathfrak{s}(z)$, namely to $(\mathfrak{s}(z))_{\leq j}$, since $\mathfrak{s}$ uses its partial computation as the seed to the rest of it. We also have the notions of *u-seeded j^{th}-position, prefix and suffix* of any vector, which are defined by

$$\forall z \in \mathbb{F}_2^k : \quad z_j^u = z^{W_j^u} , \quad z_{\leq j}^u = z^{W_{\leq j}^u} , \quad z_{\geq j}^u = z^{W_{\geq j}^u} . \tag{73}$$

The *u-seeded CLM*, denoted by $\mathfrak{s}^u$, is the following linear map: Letting $u_i = u^{W_i^u}$ be as in (71), we define the jth u-seeded linear map $\mathfrak{s}_j^u : W_j^u \to W_j^u$ by

$$\forall z \in W_j^u : \quad \mathfrak{s}_j^u(z) = \mathfrak{s}_j^{u_1,...,u_{j-1}}(z) . \tag{74}$$

Namely, $\mathfrak{s}_j^u$ uses the $(j-1)^{\text{th}}$ prefix of u as a seed, which defines both the appropriate j^{th} subspace W_j^u (and thus the restriction of z to this subspace) as well as the jth linear map $\mathfrak{s}_j^u = \mathfrak{s}_j^{u_{<j}}$ acting on this subspace. Then, we let $\mathfrak{s}_{\leq j}^u : W_{\leq j}^u \to W_{\leq j}^u$ be

$$\mathfrak{s}_{\leq j}^u = \mathfrak{s}_1^u \oplus \mathfrak{s}_2^u \oplus ... \oplus \mathfrak{s}_j^u \quad \text{and} \quad \mathfrak{s}^u = \mathfrak{s}_{\leq h}^u : \mathbb{F}_2^k \to \mathbb{F}_2^k . \tag{75}$$

As $\mathfrak{s}_j^u : W_j^u \to W_j^u$ are linear maps, and I_j^u is the set of indices associated with W_j^u, we think of them as matrices represented in the standard basis supported on I_j^u, i.e., for every $i, t \in I_j^u$ we have $(\mathfrak{s}_j^u)_{it} = \langle e_i, \mathfrak{s}_j^u(e_t) \rangle$.

We often use the following natural extension of the jth u-seeded linear map to all of $\mathbb{F}_2^k$: As W_j^u has a complement register subspace $W_{\neq j}^u$, which is the sum of all W_i^u such that $i \neq j$, we can let the *extended j^{th} u-seeded linear map* $\mathfrak{S}_j^u : \mathbb{F}_2^k \to \mathbb{F}_2^k$ by letting

$$\forall z \in \mathbb{F}_2^k : \quad \mathfrak{S}_j^u(z) = \mathfrak{s}_j^u(z_j^u) , \tag{76}$$

namely, it acts the same way as $\mathfrak{s}_j^u$ on W_j^u, and sends everything else to zero. We use a similar notation as before, $\mathfrak{S}_{\leq j}^u = \bigoplus_{i=1}^j \mathfrak{S}_i^u$, and note that $\mathfrak{S}_{\leq h}^u = \mathfrak{s}_{\leq h}^u = \mathfrak{s}^u$.

Corollary 4.15 (Seeded versus unseeded CLMs). *Let h and k be positive integers, and let $\mathfrak{s}$ be an h-level CLM on $\mathbb{F}_2^k$. Let $j \in [h]$, $u \in \mathbb{F}_2^k$ a seed and $z \in \mathbb{F}_2^k$ a vector. Then, $\mathfrak{s}_{\leq j}(z) = u_{\leq j}$ as defined in (70) if and only if $\mathfrak{s}_{\leq j}^u(z_{\leq j}^u) = u_{\leq j}$ as defined in (75). In particular, $\mathfrak{s}(z) = u$ if and only if $\mathfrak{s}^u(z) = u$.*

Definition 4.16 (Sampling scheme induced by h-level CLMs). A tailored game $\mathfrak{G}$ is said to have a sampling scheme induced by h-level CLMs, if there exist a pair of h-level CLMs $\mathfrak{s} = (\mathfrak{s}^A, \mathfrak{s}^B) : \mathbb{F}_2^k \to \mathbb{F}_2^k \times \mathbb{F}_2^k$ (Definition 4.13), such that the vertex set of the underlying graph of $\mathfrak{G}$ is $\mathbb{F}_2^k$, and the distribution over edges is the pushforward of the uniform distribution over $\mathbb{F}_2^k$ through $\mathfrak{s}$. Namely, for $x, y \in \mathbb{F}_2^k$,

$$\mu(xy) = \frac{|\{z \in \mathbb{F}_2^k \mid \mathfrak{s}^A(z) = x, \mathfrak{s}^B(z) = y\}|}{2^k} .$$

Perpendicular maps

As seen in the baby question reduction transformation from Section 4.2, we need a notion of a "perpendicular map". Specifically for the full question reduction transformation, we will need a perpendicular map for every seeded CLM (Definition 4.14).

Definition 4.17. Let $f\colon \mathbb{F}_2^k \to \mathbb{F}_2^k$ be a linear map. A *perpendicular map* to f, is a linear map $f^\perp\colon \mathbb{F}_2^k \to \mathbb{F}_2^k$ whose rows span $\ker(f)$, namely $\mathrm{Im}((f^\perp)^*) = \ker(f)$, where $*$ is the dual map with respect to the bilinear form $\langle \cdot, \cdot \rangle$ — see the beginning of Section 3.7.

Remark 4.18. As defined, the perpendicular map is not unique. There is an efficient algorithmic way of extracting a perpendicular map given the matrix representation of a linear map using Gaussian elimination — see, e.g., [JNV$^+$21, Definition 3.11].

Claim 4.19. *Let $f\colon \mathbb{F}_2^k \to \mathbb{F}_2^k$ be a linear map, and let $f^\perp\colon \mathbb{F}_2^k \to \mathbb{F}_2^k$ be a perpendicular map to f (Definition 4.17), namely a matrix whose rows span the kernel of f. Then, the f-evaluated (Definition 3.32) PVM $\mathscr{F}^{\mathbb{Z}}$ commutes with the $f^\perp$-evaluated PVM $\mathscr{F}^{\mathbb{X}}$. Namely,*

$$\forall \nu, \mathrm{x} \in \mathbb{F}_2^k\colon \quad \mathscr{F}^{\mathbb{Z}}_{[f(\cdot)=\mathrm{x}]} \cdot \mathscr{F}^{\mathbb{X}}_{[f^\perp(\cdot)=\nu]} = \mathscr{F}^{\mathbb{X}}_{[f^\perp(\cdot)=\nu]} \cdot \mathscr{F}^{\mathbb{Z}}_{[f(\cdot)=\mathrm{x}]} \,.$$

In words, one can measure the f-evaluation according to the $\mathbb{Z}$-basis simultaneously with the $f^\perp$-evaluation according to the $\mathbb{X}$-basis.

Proof. These two PVMs commute in projective form if and only if they commute in representation form. Let $\rho^{\mathbb{Z}}$ and $\rho^{\mathbb{X}}$ be the representation forms of $\mathscr{F}^{\mathbb{Z}}$ and $\mathscr{F}^{\mathbb{X}}$ respectively (as they were defined in (46)), namely $\rho^{\mathbb{Z}}(\alpha) = \mathbb{Z}^{\otimes \alpha}$ and $\rho^{\mathbb{X}}(\beta) = \mathbb{X}^{\otimes \beta}$. Then, by Corollary 3.39,

$$\rho^{\mathbb{Z}}_{[f]}(\alpha) = \mathbb{Z}^{\otimes \alpha \cdot f} \quad , \quad \rho^{\mathbb{X}}_{[f^\perp]}(\beta) = \mathbb{X}^{\otimes \beta \cdot f^\perp} \,.$$

Now, for every $\alpha, \beta \in \mathbb{F}_2^k$, thought of as row vectors, we have

$$\mathbb{Z}^{\otimes \alpha \cdot f} \mathbb{X}^{\otimes \beta \cdot f^\perp} = (-1)^{\langle \alpha \cdot f, \beta \cdot f^\perp \rangle} \mathbb{X}^{\otimes \beta \cdot f^\perp} \mathbb{Z}^{\otimes \alpha \cdot f} \,.$$

But,

$$\langle \alpha \cdot f, \beta \cdot f^\perp \rangle = \alpha \underbrace{f(f^\perp)^*}_{=0} \beta^* = 0 \,,$$

where $*$ is transposition in the above calculation. Hence the PVMs commute as claimed. $\square$

Remark 4.20. In the next section, we assume to be given in addition to a CLM $\mathfrak{s}$ acting on $\mathbb{F}_2^k$, a collection of perpendicular maps $(\mathfrak{s}_j^u)^\perp\colon W_j^u \to W_j^u$ for each jth u-seeded CLM. In the same spirit as before, we use the notation $(\mathfrak{s}_{\leq j}^u)^\perp$ for $\bigoplus_{i=1}^j (\mathfrak{s}_i^u)^\perp$, and it is indeed perpendicular to the map $\mathfrak{s}_{\leq j}^u$.

By extending $(\mathfrak{s}_j^u)^\perp$ to $(\mathfrak{S}_j^u)^\perp\colon \mathbb{F}_2^k \to \mathbb{F}_2^k$ to be the identity on $W_{\neq j}^u$ we indeed obtain a function which is perpendicular to the extended j^{th} u-seeded CLM defined in (76) — so the notation is fitting. This extension satisfies that

$$(\mathfrak{S}_{\leq j}^u)^\perp = (\mathfrak{S}_j^u)^\perp \circ ... \circ (\mathfrak{S}_2^u)^\perp \circ (\mathfrak{S}_1^u)^\perp \,. \tag{77}$$

4.4 Question Reduction in the conditionally linear sampler case

Note, this is the proper augmentation that is used in compression, as opposed to the simplified case described and analyzed in Section 4.2. The sections are structured in a very similar manner, in the hope that by first reading Section 4.2, the following description and analysis of the proper augmentation become clear.

Let $\mathfrak{G}$ be a tailored game with the following properties:

(1) Its sampling scheme is induced by h-level CLMs (Definition 4.16), for some positive integer h. Namely, its vertex set is $\mathbb{F}_2^k$, and the distribution on edges is induced by the pushforward of the uniform distribution on $\mathbb{F}_2^k$ through a pair of h-level CLMs (Definition 4.13) $\mathfrak{s} = (\mathfrak{s}^A, \mathfrak{s}^B) \colon \mathbb{F}_2^k \to \mathbb{F}_2^k \times \mathbb{F}_2^k$.

(2) Its length functions are constant and equal to some positive integer Λ.

(3) In a similar manner to Section 4.2, we need a basis for the kernel of the jth u-seeded CLM $\mathfrak{s}_j^{A,u} \colon W_j^{A,u} \to W_j^{A,u}$ (Definition 4.14), which is a linear map, for every j and u. So, we assume to be given perpendicular maps (Definition 4.17) $(\mathfrak{s}_j^{A,u})^\perp \colon W_j^{A,u} \to W_j^{A,u}$, namely their rows, parametrized by the indices in $I_j^{A,u} \subset [k]$, span $\ker \mathfrak{s}_j^{A,u}$. We mainly use the extensions $(\mathfrak{S}_j^{A,u})^\perp$ of these maps to all of $\mathbb{F}_2^k$ as in Remark 4.20, and specifically those defined in (77).

Let $\mathfrak{G}$ be a tailored game with vertex set $\mathbb{F}_2^k$ and whose distribution on edges is induced by the pushforward of the uniform distribution on $\mathbb{F}_2^k$ through a pair of h-level CLMs $\mathfrak{s} = (\mathfrak{s}^A, \mathfrak{s}^B) \colon \mathbb{F}_2^k \to \mathbb{F}_2^k \times \mathbb{F}_2^k$. As usual, it is assumed that $\mathfrak{G}$ has constant readable and unreadable answer lengths both equal to Λ. In addition, a collection of perpendicular maps $(\mathfrak{s}_j^{A,\mathbf{x}})^\perp$ to the seeded CLMs $\mathfrak{s}_j^{A,\mathbf{x}}$ are assumed to be provided, and we use the notation $(\mathfrak{S}_j^{A,\mathbf{x}})^\perp$ for their extensions as in Remark 4.20.

Sub-Structure	Question	Readable answers	Unreadable answers
$\mathfrak{Pauli\ Basis}_k$	$\texttt{Pauli}_{\mathbb{Z}}$		$z \in \mathbb{F}_2^k$
	$\texttt{Pauli}_{\mathbb{X}}$		$\chi \in \mathbb{F}_2^k$
	See Figure 3 for rest		
$\mathfrak{Intro}(\mathfrak{G})$	$\texttt{Intro}_A$	$(\mathbf{x}, a^{\mathfrak{R}}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$	$a^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda$
Sampling apparatus	$\texttt{Sample}_A$	$(z_{sam}, a_{sam}^{\mathfrak{R}}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$	$a_{sam}^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda$
Hiding apparatus	$\texttt{Read}_A$	$(\mathbf{x}_{read}, a_{read}^{\mathfrak{R}}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$	$(v_{read}, a_{read}^{\mathfrak{L}}) \in \mathbb{F}_2^k \times \mathbb{F}_2^\Lambda$
	$\texttt{Hide}_A^j$	$\mathbf{x}_{hide\ j} \in \mathbb{F}_2^k$	$v_{hide\ j} \in \mathbb{F}_2^k$

The following tests are performed when the corresponding augmented edge is sampled:

1. $\texttt{Pauli}_{\mathbb{Z}} - \texttt{Sample}$: Check that $z = z_{sam}$.

2. $\texttt{Intro} - \texttt{Sample}$: Check that $\mathbf{x} = \mathfrak{s}^A(z_{sam})$, $a^{\mathfrak{R}} = a_{sam}^{\mathfrak{R}}$ and $a^{\mathfrak{L}} = a_{sam}^{\mathfrak{L}}$.

3. $\texttt{Intro} - \texttt{Read}$: Check that $\mathbf{x} = \mathbf{x}_{read}$, $a^{\mathfrak{R}} = a_{read}^{\mathfrak{R}}$ and $a^{\mathfrak{L}} = a_{read}^{\mathfrak{L}}$.

4. $\texttt{Hide}^h - \texttt{Read}$: Check that $v_{hide\ h} = v_{read}$ and that $\mathbf{x}_{hide\ h} = (\mathbf{x}_{read})_{<h}$, where $(\cdot)_{<h}$ is the $(h-1)^{\text{th}}$ prefix (72).

5. $\texttt{Hide}^1 - \texttt{Pauli}_{\mathbb{X}}$: Check that $\mathbf{x}_{hide\ 1} = 0$ and that $v_{hide\ 1} = (\mathfrak{S}_1^{A,\mathbf{x}_{hide\ 1}})^\perp(\chi)$.

6. $\texttt{Hide}^j - \texttt{Hide}^{j-1}$: Fixing $\mathbf{x} = \mathbf{x}_{hide\ j}$, we check two things. First, that $\mathbf{x}_{hide\ j-1} = \mathbf{x}_{<j-1}$, where $(\cdot)_{<j-1}$ is the $(j-2)^{\text{th}}$ prefix (72). Second, that $(\mathfrak{S}_j^{A,\mathbf{x}})^\perp(v_{hide\ j-1}) = v_{hide\ j}$.

Figure 12: Questions and answers in the game $\mathfrak{QueRed}_h(\mathfrak{G}, k, \mathscr{B})$. Since the game is an augmentation of the sum of $\mathfrak{Pauli\ Basis}_k(\mathscr{B})$ and $\mathfrak{Intro}(\mathfrak{G})$ we only list new questions and answers, and additional tests, and refer to Figure 3 and Figure 8 for questions and answers of the latter.

The question reduction transformation $\mathfrak{QueRed}(\mathfrak{G}) = \mathfrak{QueRed}_h(\mathfrak{G}, k, \mathscr{B})$ (See Figure 12 for an overview): The inputs are expected to be a positive integer k, a tuple $\mathscr{B}$ of n vectors in $\mathbb{F}_2^k$ that induce and $[n, k, d]$-code, and a tailored game

$\mathfrak{G}$ satisfying (1), (2) and (3) from the beginning of the section. The game $\mathfrak{QueRed}(\mathfrak{G})$ is then an augmented (Definition 3.45) sum (Definition 3.44) of the Pauli basis game $\mathfrak{Pauli\,Basis}_k = \mathfrak{Pauli\,Basis}_k(\mathscr{B})$ (Section 3.8.3) and the introspection game $\mathfrak{Intro}(\mathfrak{G})$ (Definition 4.2). The augmentation consists of two apparatuses:

1. A **Sampling apparatus** which connects the introspection game vertices to the total Z-measurement of the Pauli basis game (i.e., the vertex $\mathtt{Pauli_Z}$). The goal of this apparatus is two-fold — first, to verify that the "questions" part of the players' answers when the copy of $\mathfrak{Intro}(\mathfrak{G})$ is played is distributed according to the question distribution of $\mathfrak{G}$ — namely the pushforward of the uniform distribution along the CLMs $(\mathfrak{s}^A, \mathfrak{s}^B)$; second, to verify that the observables associated with the "answers" part of the players' answers in $\mathfrak{Intro}(\mathfrak{G})$ commute with the total Z-measurement.

2. A **Hiding apparatus** which connects the introspection game vertices to the total X-measurement of the Pauli basis game (i.e., the vertex $\mathtt{Pauli_X}$). The goal of this apparatus is to verify that the "answers" part of the players' answers in $\mathfrak{Intro}(\mathfrak{G})$ commute with a certain data processing of the X-measurements, specifically through the map $(\mathfrak{s}^{\cdot\cdot})^{\perp}$.

For the sampling apparatus, two vertices $\mathtt{Sample}_A, \mathtt{Sample}_B$ are added and are connected as follows

$$\mathtt{Intro}_A - \mathtt{Sample}_A - \mathtt{Pauli_Z} - \mathtt{Sample}_B - \mathtt{Intro}_B \,.$$

For the hiding apparatus, $2h+2$ vertices are added — $\mathtt{Read}_A, \mathtt{Read}_B$, and for every $1 \le j \le h$ the vertices $\mathtt{Hide}_A^j, \mathtt{Hide}_B^j$ — and are connected as follows

$$\mathtt{Intro}_A - \mathtt{Read}_A - \mathtt{Hide}_A^h - ... - \mathtt{Hide}_A^1 - \mathtt{Pauli_X} - \mathtt{Hide}_B^1 - ... - \mathtt{Hide}_B^h - \mathtt{Read}_B - \mathtt{Intro}_B \,. \tag{78}$$

See Figure 13 for a graphical view of the underlying graph of $\mathfrak{QueRed}(\mathfrak{G})$.

Question distribution of the question reduced game:[56] With probability $1/4$ do one of the following —

- Sample an edge from $\mathfrak{Pauli\,Basis}_k$ according to the appropriate distribution therein.

- Sample the single edge $\mathtt{Intro}_A - \mathtt{Intro}_B$ from $\mathfrak{Intro}(\mathfrak{G})$.

- Sample a uniformly random edge from the Sampling apparatus.

- Sample a uniformly random edge from the Hiding apparatus.

Lengths and formal generating sets for the augmented vertices of question reduction (which are almost the same as in Section 4.2):

Sampling apparatus — The readable length of $\mathtt{Sample}_A$ (and $\mathtt{Sample}_B$) is $k + \Lambda$, and its unreadable length is Λ. We associate with it the formal generators

$$S_{\mathtt{Sample}_A}^{\mathfrak{R}} = \mathsf{SamZ}^A \sqcup \mathsf{SamAns}^{A,\mathfrak{R}} = \{\mathsf{SamZ}^{A,i}, \mathsf{SamAns}^{A,\mathfrak{R},j} \mid 1 \le i \le k, 1 \le j \le \Lambda\},$$
$$S_{\mathtt{Sample}_A}^{\mathfrak{L}} = \mathsf{SamAns}^{A,\mathfrak{L}} = \{\mathsf{SamAns}^{A,\mathfrak{L},j} \mid 1 \le j \le \Lambda\}.$$

and similarly for $\mathtt{Sample}_B$. Namely, answers are formatted as $(z_{sam}, a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}})$, where $z_{sam} \in \mathbb{F}_2^k$, and $a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$.

Hiding apparatus —

- The readable length of $\mathtt{Read}_A$ (respectively $\mathtt{Read}_B$) is $k + \Lambda$, and its unreadable length is $k + \Lambda$ as well. We associate with it the formal generators

$$S_{\mathtt{Read}_A}^{\mathfrak{R}} = \mathsf{ReadQue}^A \sqcup \mathsf{ReadAns}^{A,\mathfrak{R}} = \{\mathsf{ReadQue}^{A,i}, \mathsf{ReadAns}^{A,\mathfrak{R},j} \mid 1 \le i \le k, 1 \le j \le \Lambda\},$$
$$S_{\mathtt{Read}_A}^{\mathfrak{L}} = \mathsf{ReadPerp}^A \sqcup \mathsf{ReadAns}^{A,\mathfrak{L}} = \{\mathsf{ReadPerp}^{A,i}, \mathsf{ReadAns}^{A,\mathfrak{L},j} \mid 1 \le i \le k, 1 \le j \le \Lambda\},$$

and similarly for $\mathtt{Read}_B$. Namely, answers are formatted as $(\mathtt{x}_{read}, a_{read}^{\mathfrak{R}}, \nu_{read}, a_{read}^{\mathfrak{L}})$, where $\mathtt{x}_{read} \in \mathbb{F}_2^k, a_{read}^{\mathfrak{R}}, a_{read}^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda}$ and $\nu_{read} \in \mathbb{F}_2^k$ (and for B, $(\mathtt{y}_{read}, b_{read}^{\mathfrak{R}}, \mu_{read}, b_{read}^{\mathfrak{L}})$ in the appropriate spaces).

---

[56] The distribution that we eventually use is slightly different, as the sampler of this game needs to be induced by CLMs so that we can iterate compression. This is handled in Section 4.5.2, where we provide a distribution such that for every edge the probability is the same as this one up to some global constant factor independent of $k, \mathscr{B}$ or $\mathfrak{G}$, though it may depend on h.

Figure 13: The underlying graph of $\mathfrak{QueRed}(\mathfrak{G})$, where most of the embedded Pauli basis game is hidden. Also, there are $h - 2$ extra vertices between $\mathtt{Hide}^1$ and $\mathtt{Hide}^h$.

- The readable length of $\mathtt{Hide}_A^j$ (respectively $\mathtt{Hide}_B^j$) is k, and its unreadable length is k. We associate with it the formal generators

$$S_{\mathtt{Hide}_A^j}^{\mathfrak{R}} = \mathsf{Hide}^j\mathsf{Que}^A = \{\mathsf{Hide}^j\mathsf{Que}^{A,i} \mid 1 \leq i \leq k\},$$

$$S_{\mathtt{Hide}_A^j}^{\mathfrak{L}} = \mathsf{Hide}^j\mathsf{Perp}^A = \{\mathsf{Hide}^j\mathsf{Perp}^{A,i} \mid 1 \leq i \leq k\},$$

and similarly for $\mathtt{Hide}_B^j$. Namely, the answer is formatted as $(x_{hide\ j}, v_{hide\ j}) \in \mathbb{F}_2^{2k}$ (respectively $(y_{hide\ j}, \mu_{hide\ j}) \in \mathbb{F}_2^{2k}$).

Decision procedure of the augmented edges in the question reduced game: This is essentially the description of the controlled linear constraints function L_{xy} of $\mathfrak{QueRed}(\mathfrak{G})$, but we use the phrase "check that" repeatedly, by which we mean

"add this sequence of linear constraints to the image of L_{xy} and the canonical verifier will check them".

Sampling apparatus —

(1) In case $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_A$ (respectively $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_B$) is sampled, check that

$$\forall 1 \leq i \leq k: \quad \gamma(\mathsf{PZ}^i) = \gamma(\mathsf{SamZ}^{A,i}). \tag{79}$$

In other words, if z is the answer to $\mathtt{Pauli}_{\mathbb{Z}}$, then check that $z = z_{sam}$.

(2) In case $\mathtt{Intro}_A - \mathtt{Sample}_A$ (respectively $\mathtt{Intro}_B - \mathtt{Sample}_B$) is sampled, first check that

$$\forall 1 \leq j \leq \Lambda: \quad \gamma(\mathsf{Ans}^{A,\cdot,j}) = \gamma(\mathsf{SamAns}^{A,\cdot,j}) \tag{80}$$

and then check that
$$\mathfrak{s}^A(\gamma(\mathsf{SamZ}^{A,1}), ..., \gamma(\mathsf{SamZ}^{A,k})) = (\gamma(\mathsf{Que}^{A,1}), ..., \gamma(\mathsf{Que}^{A,k})). \tag{81}$$

In other words, if $(z_{sam}, a_{sam}^{\mathfrak{R}}, a_{sam}^{\mathfrak{L}})$ is the answer to $\mathtt{Sample}_A$, and $(\mathsf{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ is the answer to $\mathtt{Intro}_A$, then check that $\mathsf{x} = \mathfrak{s}^A(z_{sam}), a^{\mathfrak{R}} = a_{sam}^{\mathfrak{R}}$ and $a^{\mathfrak{L}} = a_{sam}^{\mathfrak{L}}$.

Note that the check (81) is not linear (because $\mathfrak{s}^A$ is only *conditionally* linear, not linear). But, as both $\mathsf{SamZ}^{\cdot\cdot}$ and $\mathsf{Que}^{\cdot\cdot}$ are readable variables, it can be tailored as follows: $L_{\mathtt{Intro}_A\,\mathtt{Sample}_A}(\mathsf{x}, a^{\mathfrak{R}}, z_{sam}, a_{sam}^{\mathfrak{R}})$ contains all constraints induced by (80), adds no additional linear constraints if (81) is satisfied, and adds $\{\mathsf{J}\}$ as a constraint if (81) is not satisfied (which translates to definite rejection).

Hiding apparatus —

(1) In case $\mathtt{Intro}_A - \mathtt{Read}_A$ (respectively $\mathtt{Intro}_B - \mathtt{Read}_B$) is sampled, check that

$$\forall 1 \leq i \leq k, 1 \leq j \leq \Lambda: \quad \gamma(\mathsf{Ans}^{A,\cdot,j}) = \gamma(\mathsf{ReadAns}^{A,\cdot,j}), \quad \gamma(\mathsf{Que}^{A,i}) = \gamma(\mathsf{ReadQue}^{A,i}). \tag{82}$$

In other words, if $(\mathsf{x}_{read}, a_{read}^{\mathfrak{R}}, v_{read}, a_{read}^{\mathfrak{L}})$ is the answer to $\mathtt{Read}_A$, and $(\mathsf{x}, a^{\mathfrak{R}}, a^{\mathfrak{L}})$ is the answer to $\mathtt{Intro}_A$, check that $\mathsf{x}_{read} = \mathsf{x}, a_{read}^{\mathfrak{R}} = a^{\mathfrak{R}}$ and $a_{read}^{\mathfrak{L}} = a^{\mathfrak{L}}$.

(2) In case $\mathtt{Hide}_A^h - \mathtt{Read}_A$ (respectively $\mathtt{Hide}_B^h - \mathtt{Read}_B$) is sampled: First check that

$$\forall 1 \leq i \leq k: \quad \gamma(\mathsf{Hide}^h\mathsf{Perp}^{A,i}) = \gamma(\mathsf{ReadPerp}^{A,i}). \tag{83}$$

Namely, if we denote by $(\mathsf{x}_{read}, a_{read}^{\mathfrak{R}}, v_{read}, a_{read}^{\mathfrak{L}})$ the answer to $\mathtt{Read}_A$, and by $(\mathsf{x}_{hide\ h}, v_{hide\ h})$ the answer to $\mathtt{Hide}_A^h$, (83) checks that $v_{hide\ h} = v_{read}$. In addition, check that $\mathsf{x}_{hide\ h}$ is the $(h-1)$-prefix (72) of x_{read}, namely that $(\mathsf{x}_{read})_{\leq h-1} = \mathsf{x}_{hide\ h}$, or equivalently

$$\begin{aligned} \forall i \in I_{<h}^{\mathsf{x}_{read}}: \quad &\gamma(\mathsf{Hide}^h\mathsf{Que}^{A,i}) = \gamma(\mathsf{Read}^h\mathsf{Que}^{A,i}), \\ \forall i \in I_h^{\mathsf{x}_{read}}: \quad &\gamma(\mathsf{Hide}^h\mathsf{Que}^{A,i}) = 0. \end{aligned} \tag{84}$$

Note that although these are linear checks, they depend on the value of x_{read}. But, as $\mathsf{ReadQue}^{\cdot\cdot}$ are readable, this is allowed in the tailored category.

(3) In case $\mathtt{Hide}_A^1 - \mathtt{Pauli}_{\mathbb{X}}$ (respectively $\mathtt{Hide}_B^1 - \mathtt{Pauli}_{\mathbb{X}}$) is sampled, let $\mathsf{x} := \mathsf{x}_{hide\ 1} = (\gamma(\mathsf{Hide}^1\mathsf{Que}^{A,i}))_{i=1}^k$, and check that

$$(\mathfrak{S}_1^{A,\mathsf{x}})^{\perp}(\gamma(\mathsf{PX}^i))_{i=1}^k = (\gamma(\mathsf{Hide}^1\mathsf{Perp}^i))_{i=1}^k, \tag{85}$$

where $(\mathfrak{S}_1^{A,\mathsf{x}})^{\perp}$ is the extended perpendicular 1st x-seeded CLM defined in Item (3). In addition, check that

$$\forall i \in [k]: \quad \gamma(\mathsf{Hide}^1\mathsf{Que}^{A,i}) = 0. \tag{86}$$

In other words, if $(\mathtt{x}_{hide\ 1}, v_{hide\ 1})$ is the answer to $\mathtt{Hide}_A^1$ and χ is the answer to $\mathtt{Pauli}_{\mathbb{X}}$, check that $\mathtt{x}_{hide\ 1} = \vec{0}$, and that

$$(\mathfrak{S}_1^{A,\mathtt{x}_{hide\ 1}})^{\perp}(\chi) = v_{hide\ 1} \,. \tag{87}$$

This is the same as for $\chi^{V_{>1}} = v_{hide\ 1}^{V_{>1}}$ and $(\mathfrak{s}_1^A)^{\perp}(\chi^{V_1}) = v_{hide\ 1}^{V_1}$. Note that, as $(\mathfrak{S}_1^{A,\mathtt{x}})^{\perp} : \mathbb{F}_2^k \to \mathbb{F}_2^k$ is a linear map, this check can be tailored appropriately.

(4) In case $\mathtt{Hide}_A^j - \mathtt{Hide}_A^{j-1}$ (respectively $\mathtt{Hide}_B^j - \mathtt{Hide}_B^{j-1}$) is sampled for $2 \leq j \leq h$: Let $(\mathtt{x}_{hide\ j}, v_{hide\ j})$ be the answer to $\mathtt{Hide}_A^j$, and $(\mathtt{x}_{hide\ j-1}, v_{hide\ j-1})$ the answer to $\mathtt{Hide}_A^{j-1}$. Fix $\mathtt{x} := \mathtt{x}_{hide\ j}$ as the seed, and note that $\mathtt{Hide}^j \mathtt{Que}^{A,\cdot}$ are readable variables so we may perform checks that depend non-linearly on them. First check that the $(j-2)^{\text{th}}$ prefix of $\mathtt{x}$ (see (72)) is equal to $\mathtt{x}_{hide\ j-1}$, namely that $\mathtt{x}_{\leq j-2} = \mathtt{x}_{hide\ j-1}$, or equivalently

$$
\begin{aligned}
\forall i \in I_{<j-1}^{\mathtt{x}} : \quad &\gamma(\mathtt{Hide}^{j-1}\mathtt{Que}^{A,i}) = \gamma(\mathtt{Hide}^j \mathtt{Que}^{A,i}) \,, \\
\forall i \in I_{\geq j-1}^{\mathtt{x}} : \quad &\gamma(\mathtt{Hide}^{j-1}\mathtt{Que}^{A,i}) = 0 \,,
\end{aligned}
\tag{88}
$$

where $I^{\mathtt{x}}$ is the set of indices associated with the seeded register subspace $W^{\mathtt{x}}$ which was defined in (71). In addition, check that $(\mathfrak{S}_j^{A,\mathtt{x}})^{\perp}(v_{hide\ j-1}) = v_{hide\ j}$, namely that

$$\forall i \in I_{\neq j}^{\mathtt{x}} : \quad \gamma(\mathtt{Hide}^j \mathtt{Perp}^{A,i}) = \gamma(\mathtt{Hide}^{j-1}\mathtt{Perp}^{A,i}) \,, \tag{89}$$

and

$$\forall i \in I_j^{\mathtt{x}} : \quad \gamma(\mathtt{Hide}^j \mathtt{Perp}^{A,i}) = \sum_{t \in I_j} (\mathfrak{s}_j^{A,\mathtt{x}})_{it}^{\perp} \gamma(\mathtt{Hide}^{j-1}\mathtt{Perp}^{A,t}) \,. \tag{90}$$

Again, as $(\mathfrak{S}_j^{A,\mathtt{x}})^{\perp}$ is linear for every seed $\mathtt{x}$, and $\mathtt{x}$ is decided by readable variables, these checks can be tailored appropriately.

Remark 4.21 (Restriction notation). Recall the notation $\mathcal{V}^{S'}$ (in observable form) and $\mathcal{Q}^{S'}$ (in projective form) for the restriction to $\mathbb{F}_2^{S'}$ of the respective PVMs $\mathcal{V}$ and $\mathcal{Q}$ with outcomes in $\mathbb{F}_2^{S' \sqcup S''}$ (Definition 3.32). For example, in the case of $\mathfrak{QueRed}(\mathfrak{G})$, recall that

$$\mathsf{Ans}^A = \mathsf{Ans}^{A,\mathfrak{R}} \sqcup \mathsf{Ans}^{A,\mathfrak{L}} = \{\mathsf{Ans}^{A,\mathfrak{R},j}, \mathsf{Ans}^{A,\mathfrak{L},j}\}_{j=1}^{\Lambda}$$

is the set of Ans^A-variables, which is a subset of $S_{\mathtt{Intro}_A}$. Then, the representation $\mathcal{V}^{\mathsf{Ans}^A} : \mathbb{F}_2^{\mathsf{Ans}^A} \to U(N)$ is induced by the observable form $\mathcal{V}$ of some strategy $\mathscr{S}$ by letting

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda} : \quad \mathcal{V}^{\mathsf{Ans}^A}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) = \prod_i \mathcal{V}(\mathsf{Ans}^{A,\mathfrak{R},i})^{\alpha_i^{\mathfrak{R}}} \prod_j \mathcal{V}(\mathsf{Ans}^{A,\mathfrak{L},j})^{\alpha_j^{\mathfrak{L}}} \,,$$

and the PVM $\mathcal{Q}^{\mathsf{Ans}^A}$ is induced by the projective form $\mathcal{Q}$ by letting

$$\mathcal{Q}_{a^{\mathfrak{R}},a^{\mathfrak{L}}}^{\mathsf{Ans}^A} = \sum_{\mathtt{x} \in \mathbb{F}_2^r} \mathcal{Q}_{\mathtt{x},a^{\mathfrak{R}},a^{\mathfrak{L}}}^{\mathtt{Intro}_A} \,.$$

We use a similar notation for restrictions to various subsets of generators at different vertices, such as

$$\mathsf{ReadQue}^{\cdot}, \mathsf{ReadPerp}^{\cdot}, \mathsf{SamZ}^{\cdot} \,,$$

and so on.

Remark 4.22 (Analysis of the question reduced game $\mathfrak{QueRed}(\mathfrak{G})$). Let us briefly analyze the properties of a strategy $\mathscr{S}$, with $\mathcal{U}$ being its observable form and $\mathcal{P}$ its projective form, that passes all edges but $\mathtt{Intro}_A - \mathtt{Intro}_B$ in $\mathfrak{QueRed}(\mathfrak{G})$ perfectly:

(1) Since it passes the copy of $\mathfrak{Pauli\ Basis}_k$ perfectly, we can deduce by Claim 3.83 and Fact 3.73 (in the case $\varepsilon = 0$) that the observables $\mathscr{S}$ associates with the generators at the vertices $\mathtt{Pauli_X}$ and $\mathtt{Pauli_Z}$ induce the unique (up to direct sums) representation of the Pauli group P_k defined in (45). Namely, there is some natural number $m \in \mathbb{N}$ such that $\mathscr{S}$ acts on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$, and

$$\forall 1 \le i \le k : \quad \mathcal{U}(\mathsf{PX}^i) = \mathbb{X}^{\otimes e_i} \otimes \mathrm{Id}_m , \quad \mathcal{U}(\mathsf{PZ}^i) = \mathbb{Z}^{\otimes e_i} \otimes \mathrm{Id}_m , \tag{91}$$

or equivalently in representation form

$$\forall \alpha \in \mathbb{F}_2^k : \quad \mathcal{U}^{\mathtt{Pauli_X}}(\alpha) = \rho^{\mathbb{X}}(\alpha) \otimes \mathrm{Id}_m = \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m , \quad \mathcal{U}^{\mathtt{Pauli_Z}}(\alpha) = \rho^{\mathbb{Z}}(\alpha) \otimes \mathrm{Id}_m = \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m .$$

(2) As $\mathscr{S}$ passes the check along the edges $\mathtt{Pauli_Z} - \mathtt{Sample}$. perfectly, the assignments to the PZ variables and SamZ variables are consistent. Namely,

$$\forall 1 \le i \le k : \quad \mathcal{U}(\mathsf{SamZ}^{\cdot,i}) = \mathcal{U}(\mathsf{PZ}^i) = \mathbb{Z}^{\otimes e_i} \otimes \mathrm{Id}_m ,$$

and equivalently in projective form

$$\forall z \in \mathbb{F}_2^k : \quad \mathcal{P}_z^{\mathsf{SamZ}^\cdot} = \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m . \tag{92}$$

(3) As $\mathscr{S}$ passes the checks $\mathtt{Sample}_A - \mathtt{Intro}_A - \mathtt{Read}_A$ perfectly, we can deduce that the assignments to the Ans, ReadAns and SamAns variables are consistent:

$$\mathcal{U}^{\mathsf{SamAns}^A} = \mathcal{U}^{\mathsf{Ans}^A} = \mathcal{U}^{\mathsf{ReadAns}^A}, \tag{93}$$

namely that for every $1 \le j \le \Lambda$, $\mathcal{U}(\mathsf{SamAns}^{A,\cdot,j}) = \mathcal{U}(\mathsf{Ans}^{A,\cdot,j}) = \mathcal{U}(\mathsf{ReadAns}^{A,\cdot,j})$ (and similarly for B). In addition, as the Que^A-variables are checked to be the $\mathfrak{s}^A$ image of the SamZ^A-variables, and $\mathsf{ReadQue}^A$ are checked to be consistent with Que^A, we can deduce that

$$\forall \mathbf{x} \in \mathbb{F}_2^k : \quad \mathcal{P}_{\mathbf{x}}^{\mathsf{ReadQue}^A} = \mathcal{P}_{\mathbf{x}}^{\mathsf{Que}^A} = \mathcal{P}_{[\mathfrak{s}^A(\cdot) = \mathbf{x}]}^{\mathsf{SamZ}^A} = \sum_{z:\, \mathfrak{s}^A(z) = \mathbf{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m ,$$

where $\mathcal{P}_{[\mathfrak{s}^A(\cdot) = \mathbf{x}]}^{\mathsf{SamZ}^A}$ is the $\mathfrak{s}^A$-evaluated PVM (Definition 3.32) associated to $\mathcal{P}_z^{\mathsf{SamZ}^A}$, and the last equality is due to (92).

(4) As $\mathscr{S}$ passes the checks $\mathtt{Read}_A - \mathtt{Hide}_A^h - ... - \mathtt{Hide}_A^2 - \mathtt{Hide}_A^1$ perfectly, we can deduce that for every $1 \le r \le h - 1$ and $\mathbf{x} \in \mathbb{F}_2^k$,

$$\mathcal{P}_{\mathbf{x}}^{\mathsf{Hide}^{r+1}\mathsf{Que}^A} = \mathcal{P}_{[(\cdot)_{\le r} = \mathbf{x}]}^{\mathsf{ReadQue}^A} = \sum_{z:\, \mathfrak{s}_{\le r}^A(z) = \mathbf{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m , \tag{94}$$

where $(\cdot)_{\le r}$ is the r-prefix function (72) — which is part of the data of the CLM $\mathfrak{s}^A$ — and $\mathfrak{s}_{\le r}^A(z)$ is as in (70).

(5) As $\mathscr{S}$ passes the checks $\mathtt{Pauli_X} - \mathtt{Hide}_A^1 - ... - \mathtt{Hide}_A^h - \mathtt{Read}$ perfectly, and using (91) and (94), we get for every $1 \le r \le h$ and $\mathbf{x}, \nu \in \mathbb{F}_2^k$ that

$$\mathcal{P}_{\mathbf{x},\nu}^{\mathsf{Hide}_A^r} = \mathscr{F}_{[\mathfrak{s}_{<r}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}} \cdot \mathscr{F}_{[(\mathfrak{S}_{\le r}^{A,\mathbf{x}})^\perp(\cdot) = \nu]}^{\mathbb{X}} = \sum_{\substack{z \in \mathbb{F}_2^k \\ \mathfrak{s}_{<r}^A(z) = \mathbf{x}}} \sum_{\substack{w \in \mathbb{F}_2^k \\ (\mathfrak{S}_{\le r}^{A,\mathbf{x}})^\perp(w) = \nu}} \mathscr{F}_z^{\mathbb{Z}} \mathscr{F}_w^{\mathbb{X}} \otimes \mathrm{Id}_m ,$$

where $\mathfrak{S}_{\le j}^{A,\mathbf{x}}$ are again the extensions of the perpendicular maps that were defined in Item (3); also

$$\forall \mathbf{x}, \nu \in \mathbb{F}_2^k : \quad \mathcal{P}_{\mathbf{x},\nu}^{\mathsf{ReadQue}^A \sqcup \mathsf{ReadPerp}^A} = \mathscr{F}_{[\mathfrak{s}(\cdot) = \mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{s}^{A,\mathbf{x}})^\perp(\cdot) = \nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m .$$

93

(6) Finally, according to (93), the Ans^A-observables commute with the $\mathsf{ReadPerp}^A$-observables and the SamZ^A-observables. Commuting with the SamZ^A-observables translates to the Ans^A-observables being of the form $\sum \mathscr{F}_z^{\mathbb{Z}} \otimes \mathscr{A}^z$. Commuting with the $\mathsf{ReadPerp}^A$-observables implies that $\mathscr{A}^z$ is equal to $\mathscr{A}^{z'}$ whenever $\mathfrak{s}(z) = \mathfrak{s}(z')$. Namely (by repeating all these arguments for B and $\mathfrak{s}^B$ as well), the strategy $\mathscr{S}$ is honest (Definition 4.4) when restricted to the copy of $\mathfrak{Intro}(\mathfrak{G})$ in $\mathfrak{QueRed}(\mathfrak{G})$. In particular, its value on $\mathfrak{Intro}(\mathfrak{G})$ is the same as some quantum strategy for $\mathfrak{G}$ itself.

Remark 4.23. A reader may notice that to achieve the above goals, we could have dropped the $\mathtt{Sample}$ and $\mathtt{Hide}$ vertices altogether, and applied a more direct check (simplifying the augmentation). Though this is true, it will hinder the perfect completeness case, as we seek perfect ZPC strategies, in particular strategies that commute along edges, which is problematic without these buffer questions.

Theorem 4.24 (Completeness and Soundness of Question Reduction). *Let k be a positive integer, $\mathscr{B}$ a tuple of n vectors in $\mathbb{F}_2^k$ that induce an $[n,k,d]$-code (Section 3.7.2), and $\mathfrak{G}$ a tailored game satisfying (1),(2) and (3) from the beginning of this section. Then, the question reduced game $\mathfrak{QueRed}(\mathfrak{G}) = \mathfrak{QueRed}_h(\mathfrak{G},k,\mathscr{B})$ has the following properties:*

(1) Completeness: *If $\mathfrak{G}$ has a perfect ZPC strategy, then so does $\mathfrak{QueRed}(\mathfrak{G})$.*

(2) Soundness: *If $\mathfrak{QueRed}(\mathfrak{G})$ has a strategy with value $1 - \varepsilon$, then $\mathfrak{G}$ has a strategy with value at least*

$$1 - O(h^2 \cdot 2^h \cdot (1 + k^2/d^2) \cdot \varepsilon^{1/8}) \ .$$

(3) Entanglement: *For every $\varepsilon > 0$,*

$$\mathscr{E}(\mathfrak{QueRed}(\mathfrak{G}), 1 - \varepsilon) \geq 2^k \cdot (1 - O((1 + k^2/d^2)\varepsilon)) \cdot \mathscr{E}\big(\mathfrak{G}, 1 - O\big(h^2 \cdot 2^h \cdot (1 + k^2/d^2) \cdot \varepsilon^{1/8}\big)\big) \ .$$

Remark 4.25. The underlying combinatorial game (Definition 2.25) of $\mathfrak{QueRed}(\mathfrak{G})$ is the same (when $\mathscr{B}$ for $\mathfrak{Pauli\,Basis}_k$ is chosen appropriately) as the question reduction applied in [JNV$^+$21]. This means that the soundness proof therein already covers the soundness of Theorem 4.24. Although this is true, and the reader familiar with [JNV$^+$21] may even prefer their soundness proof, we include our own proof here. They are essentially the same, up to our proof leaning more on the observable perspective, which may be easier for readers who approach this result from the group stability community.

Although we construct a perfect ZPC strategy φ in our completeness proof, which is not something the authors of [JNV$^+$21] were concerned about, this strategy is (essentially) the same as their perfect complete strategy (Section 8.3.2 therein). So, if one seeks more details regarding the perfect value of our strategy, then they can seek there as well.

Proof of perfect completeness (1)

Assume that $\mathfrak{G}$ has a perfect m-dimensional ZPC strategy $\mathscr{S}$, and let $\mathcal{U}$ be its observable (and representation) form and $\mathcal{P}$ be its projective form. Then, we can induce from it a perfect honest ZPC strategy $\mathscr{S}'$ for $\mathfrak{Intro}(\mathfrak{G})$, with $\mathcal{V}$ being its observable form and $\mathcal{Q}$ its projective form, acting on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$ (see (49) and (50) in Definition 4.4, and Claim 4.6). Let us extend $\mathscr{S}'$ to the other vertices so it becomes a perfect ZPC strategy for $\mathfrak{QueRed}(\mathfrak{G})$. First, let $\mathcal{V}^{\mathtt{Pauli\mathbb{X}}}$ and $\mathcal{V}^{\mathtt{Pauli\mathbb{Z}}}$ be $\rho^{\mathbb{X}} \otimes \mathrm{Id}_m$ and $\rho^{\mathbb{X}} \otimes \mathrm{Id}_m$, namely

$$\forall \alpha \in \mathbb{F}_2^k : \quad \mathcal{V}^{\mathtt{Pauli\mathbb{X}}}(\alpha) = \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m \quad , \quad \mathcal{V}^{\mathtt{Pauli\mathbb{Z}}}(\alpha) = \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m \ .$$

By Claim 3.82, it can be extended to the rest of the $\mathfrak{Pauli\,Basis}_k$ vertices in a ZPC-manner such that on the copy of $\mathfrak{Pauli\,Basis}_k$ in $\mathfrak{QueRed}(\mathfrak{G})$ it has value 1. The rest of the PVMs are forced on us via the consistency checks along the augmented edges (see the analysis in Remark 4.22); recall the restriction notations from Remark 4.21 and the data processing notation from Definition 3.32. For the sampling apparatus, we have

$$\mathcal{Q}^{\mathsf{Ans}^A} = \mathcal{Q}^{\mathsf{SamAns}^A} , \ \mathcal{Q}^{\mathsf{Ans}^B} = \mathcal{Q}^{\mathsf{SamAns}^B} , \tag{95}$$

$$\mathcal{Q}^{\mathtt{Pauli\mathbb{Z}}} = \mathcal{Q}^{\mathsf{SamZ}^A} = \mathcal{Q}^{\mathsf{SamZ}^B} , \tag{96}$$

$$\mathcal{Q}^{\mathsf{Que}^A} = \mathcal{Q}^{\mathsf{SamZ}^A}_{[\mathfrak{s}^A(\cdot)=\cdot]} , \ \mathcal{Q}^{\mathsf{Que}^B} = \mathcal{Q}^{\mathsf{SzmZ}^B}_{[\mathfrak{s}^B(\cdot)=\cdot]} . \tag{97}$$

94

For the PVMs in the hiding apparatus, we elaborate more as the notation may be confusing. For the $\mathtt{Read}.$ vertices, recalling the linear map $(\mathfrak{s}^{A,\mathbf{x}})^{\perp} = (\mathfrak{s}_{\leq h}^{A,\mathbf{x}})^{\perp}$ from Item (3) and Remark 4.20, we have

$$\forall \mathbf{x}, \nu \in \mathbb{F}_2^k, \; a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^{\Lambda} : \quad \mathcal{Q}_{\mathbf{x}, a^{\mathfrak{R}}, \nu, a^{\mathfrak{L}}}^{\mathtt{Read}_A} = \sum_{z \in \mathbb{F}_2^k : \, \mathfrak{s}^A(z) = \mathbf{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathcal{P}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathbf{x}} \cdot \sum_{\chi \in \mathbb{F}_2^k : \, (\mathfrak{s}^{A,\mathbf{x}})^{\perp}(\chi) = \nu} \mathscr{F}_{\chi}^{\mathbb{X}} \otimes \mathrm{Id}_m , \qquad (98)$$

where $\mathcal{P}$ is the projective form of the original perfect strategy for $\mathfrak{G}$ (and similarly for B). Following the definition of an honest strategy, (49) and (50), it is straightforward to check that the restrictions to the $\mathsf{ReadQue}^{\cdot}$ and $\mathsf{ReadAns}^{\cdot}$ variables satisfy

$$\mathcal{Q}^{\mathsf{ReadQue}^A} = \mathcal{Q}^{\mathsf{Que}^A}, \; \mathcal{Q}^{\mathsf{ReadQue}^B} = \mathcal{Q}^{\mathsf{Que}^B},$$
$$\mathcal{Q}^{\mathsf{ReadAns}^A} = \mathcal{Q}^{\mathsf{Ans}^A}, \; \mathcal{Q}^{\mathsf{ReadAns}^B} = \mathcal{Q}^{\mathsf{Ans}^B}.$$

Though we claim that (98) is a PVM, this is not obvious from its definition — one needs to be convinced that the right sum commutes with the left sum for it to be an orthogonal projection. Let us prove that. By Corollary 4.15, $\sum_{z: \, \mathfrak{s}^A(z) = \mathbf{x}} = \sum_{z: \, \mathfrak{s}^{A,\mathbf{x}}(z) = \mathbf{x}}$, and thus by Claim 4.19 the product of the two sums does commute. Moreover, as the readable variables are data processed versions of $\mathcal{V}^{\mathtt{Pauli}\mathbb{Z}} = \rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$, which is diagonal, we deduce that this PVM is readably Z-aligned. For the $\mathsf{ReadAns}$ and $\mathsf{ReadQue}$ observables, as they are consistent with Ans and Que at $\mathtt{Intro}$, they agree with an honest strategy induced by a ZPC one and hence consist of signed permutations. For the $\mathsf{ReadPerp}$ observables, taking the inverse Fourier transform of (98), we have

$$\mathcal{V}^{\mathsf{ReadPerp}^A}(\alpha) = \sum_{\mathbf{x}} \mathscr{F}_{[\mathfrak{s}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}} \mathbb{X}^{\otimes \alpha \cdot (\mathfrak{s}^{A,\mathbf{x}})^{\perp}} \mathscr{F}_{[\mathfrak{s}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}} \otimes \mathrm{Id}_m ,$$

where the sum is over all $\mathbf{x}$ in the image of $\mathfrak{s}^A$ — namely, this is a block diagonal matrix whose blocks are corners of the permutation matrices $\mathbb{X}^{\otimes \cdot}$. Hence, if this matrix is invertible, it is a permutation matrix. As $\mathscr{F}_{[\mathfrak{s}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}}$ commutes with $\mathbb{X}^{\otimes \alpha \cdot (\mathfrak{s}^{A,\mathbf{x}})^{\perp}}$ (by Corollary 4.15 and Claim 4.19), it is its own inverse, which proves this is indeed a signed permutation PVM.

We are left to define the PVMs at the $\mathtt{Hide}^j$ vertices for $1 \leq j \leq h$. This is done in a similar way to the $\mathtt{Read}.$ vertices, and is forced on us by the consistency checks along the augmented edges. In projective form,

$$\forall \mathbf{x}, \nu \in \mathbb{F}_2^k : \quad \mathcal{Q}_{\mathbf{x}, \nu}^{\mathtt{Hide}^j_A} = \mathscr{F}_{[\mathfrak{s}_{<j}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}} \cdot \mathscr{F}_{[(\mathfrak{S}_{\leq j}^{A,\mathbf{x}})^{\perp}(\cdot) = \nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m$$
$$= \sum_{\substack{z \in \mathbb{F}_2^k \\ \mathfrak{s}_{<j}^A(z) = \mathbf{x}}} \sum_{\substack{\alpha \in \mathbb{F}_2^k \\ (\mathfrak{S}_{\leq j}^{A,\mathbf{x}})^{\perp}(\alpha) = \nu}} \mathscr{F}_z^{\mathbb{Z}} \mathscr{F}_{\alpha}^{\mathbb{X}} \otimes \mathrm{Id}_m . \qquad (99)$$

By Corollary 4.15, $\mathfrak{s}_{<j}^A(z) = \mathbf{x}$ if and only if $\mathfrak{S}_{<j}^{A,\mathbf{x}}(x) = \mathbf{x}$. Hence, by Claim 4.19, this is indeed a PVM. Actually, Claim 4.19 shows that this strategy commutes along all $\mathtt{Hide}^j - \mathtt{Hide}^{j+1}$ edges, as $\mathscr{F}_{[\mathfrak{s}_{\leq j}^A(\cdot) = \mathbf{x}]}^{\mathbb{Z}} = \mathscr{F}_{[\mathfrak{S}_{\leq j}^{A,\mathbf{x}}(\cdot) = \mathbf{x}]}^{\mathbb{Z}}$ commutes with $\mathscr{F}_{[(\mathfrak{S}_{\leq j}^{A,\mathbf{x}})^{\perp}(\cdot) = \nu]}^{\mathbb{X}}$ according to it. Furthermore, in a similar manner to the $\mathtt{Read}$-vertex PVM, we can deduce that this is a readably Z-aligned signed permutation PVM. All in all, $\mathscr{S}' = \{\mathcal{Q}\}$ is indeed a ZPC-strategy (all non-hide to non-hide edges are clearly commuting). The fact that this strategy is perfect can be checked by the reader, with the help of the analysis of perfect strategies in Remark 4.22. This finishes the perfect completeness proof.

Proof of soundness (2) and entanglement lower bound (3)

The idea in the soundness proof is to perturb an almost perfect strategy for $\mathfrak{QueRed}(\mathfrak{G})$ to become a strategy that passes all edges of $\mathfrak{QueRed}(\mathfrak{G})$ perfectly, except for the single edge of $\mathfrak{Intro}(\mathfrak{G})$. The way we perturb $\mathscr{S}$ to be perfect on all edges (except for $\mathtt{Intro}_A - \mathtt{Intro}_B$), roughly follows the analysis of such strategies in Remark 4.22.

95

Claim 4.26. *Let $\mathscr{S} = \{\mathcal{U}\}$ be an N-dimensional strategy for $\mathfrak{QueReo}(\mathfrak{G})$ with value $1 - \varepsilon$. Let $\varepsilon' = (1 + k^2/d^2)\varepsilon$. Then, there is a strategy $\mathscr{S}' = \{\mathcal{W}\}$, acting on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$, such that:*

1. *(Almost Perfect) $\mathscr{S}'$ has value of at least $1 - O(\sqrt{\varepsilon'})$;*

2. *(Perfect on Pauli basis) $\mathscr{S}'$ passes the edges from the copy of $\mathfrak{Pauli\ Basis}_k$ perfectly;*

3. *(Uses a specific representation of the Pauli group) $\mathscr{S}'$ satisfies*

$$\mathcal{W}^{\mathtt{PauliX}} = \rho^{\mathbb{X}} \otimes \mathrm{Id}_m \quad \textit{and} \quad \mathcal{W}^{\mathtt{PauliZ}} = \rho^{\mathbb{Z}} \otimes \mathrm{Id}_m \,,$$

   where ρ is the representation specified in Definition 3.68.

Proof. As the copy of $\mathfrak{Pauli\ Basis}_k$ is played with probability $1/4$ when running $\mathfrak{QueReo}(\mathfrak{G})$, the restriction of $\mathscr{S}$ to the vertices of $\mathfrak{Pauli\ Basis}_k$ passes it with probability of at least $1 - 4\varepsilon$. Hence, by Claim 3.83, Fact 3.73 and Claim 3.22, there is a partial isometry $\omega \colon \mathbb{C}^N \to \mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$ such that

$$\forall \alpha \in \mathbb{F}_2^k : \quad \left\| \omega \mathcal{U}^{\mathtt{PauliX}}(\alpha) \omega^* - \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m \right\|_{hs}^2 , \quad \left\| \omega \mathcal{U}^{\mathtt{PauliZ}}(\alpha) \omega^* - \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m \right\|_{hs}^2 \leq O(k^2 \varepsilon / d^2)$$

$$\text{and} \tag{100}$$

$$1 - \tau(\omega^* \omega) \,, \quad 1 - \tau(\omega \omega^*) \leq O(k^2 \varepsilon / d^2)$$

As $\varepsilon' = (1 + k^2/d^2)\varepsilon$, the above quantities are all $O(\varepsilon')$. Letting $\mathcal{W}^{\mathtt{PauliZ}}(\alpha) = \mathbb{Z}^{\otimes \alpha} \otimes \mathrm{Id}_m$ and $\mathcal{W}^{\mathtt{PauliX}}(\alpha) = \mathbb{X}^{\otimes \alpha} \otimes \mathrm{Id}_m$, we can use Claim 3.82 to extend $\mathcal{W}$ to a perfect strategy for $\mathfrak{Pauli\ Basis}_k$. As $1 - \tau(\omega^* \omega), 1 - \tau(\omega \omega^*) \leq O(\varepsilon')$, we can use orthogonalization (Fact 3.21) and extend $\mathcal{W}$ such that for every non-$\mathfrak{Pauli\ Basis}_k$ vertex x,

$$\forall \alpha \colon S_{\mathtt{x}} \to \mathbb{F}_2 : \quad \left\| \omega \mathcal{U}^{\mathtt{x}}(\alpha) \omega^* - \mathcal{W}^{\mathtt{x}}(\alpha) \right\|_{hs}^2 \leq O(\varepsilon'). \tag{101}$$

Now $\mathcal{W}$ induces a representation on all vertices of $\mathfrak{QueReo}(\mathfrak{G})$, and is thus a strategy for it. We already assured that $\mathscr{S}' = \{\mathcal{W}\}$ passes the copy of $\mathfrak{Pauli\ Basis}_k$ perfectly, and we chose it such that $\mathcal{W}^{\mathtt{PauliX}} = \rho^{\mathbb{X}} \otimes \mathrm{Id}$ and $\mathcal{W}^{\mathtt{PauliZ}} = \rho^{\mathbb{Z}} \otimes \mathrm{Id}$. Hence, conditions 2. and 3. are satisfied. Finally, equations (100) and (101) imply that across any non-$\mathfrak{Pauli\ Basis}_k$ edge, the observables of $\mathcal{W}$ are $O(\varepsilon')$-close to those of $\mathcal{U}$, hence they pass all these edges with probability at most $O(\sqrt{\varepsilon'})$ worse than $\mathcal{U}$ (Claim 3.29). This implies $\mathrm{val}(\mathscr{S}', \mathfrak{QueReo}(\mathfrak{G})) \geq 1 - O(\sqrt{\varepsilon'})$, finishing the proof. $\square$

Claim 4.27. *Let $\mathscr{S}$ be a strategy for $\mathfrak{QueReo}(\mathfrak{G})$ with value $1 - \varepsilon$, where $\mathcal{U}$ is its observable form and $\mathcal{P}$ its projective from, that acts on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$. Moreover, assume it passes the copy of $\mathfrak{Pauli\ Basis}_k$ with probability 1, and satisfies $\mathcal{U}^{\mathtt{PauliX}} = \rho^{\mathbb{X}} \otimes \mathrm{Id}_m$ and $\mathcal{U}^{\mathtt{PauliZ}} = \rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$, or equivalently in projective form $\mathcal{P}^{\mathtt{PauliZ}} = \mathscr{F}^{\mathbb{Z}} \otimes \mathrm{Id}_m$ and $\mathcal{P}^{\mathtt{PauliX}} = \mathscr{F}^{\mathbb{X}} \otimes \mathrm{Id}_m$. Then, there is a strategy $\mathscr{S}'$, with $\mathcal{Q}$ being its projective form and $\mathcal{W}$ its observable form, such that:*

1. *(Almost Perfect) $\mathscr{S}'$ has value $1 - O(\sqrt{h^3 \cdot \varepsilon})$;*

2. *(Agrees on Pauli basis) $\mathscr{S}'$ agrees with $\mathscr{S}$ on the copy of $\mathfrak{Pauli\ Basis}_k$;*

3. *(Readable variables are consistent with $\mathbb{Z}$-measurements) $\mathscr{S}'$ satisfies*

$$\forall z, \mathtt{x}, \mathtt{y} \in \mathbb{F}_2^k \,, \ r \in [h] : \quad \mathcal{Q}_z^{\mathtt{SamZ}^A} = \mathcal{Q}_z^{\mathtt{SamZ}^B} = \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \,,$$

$$\mathcal{Q}_{\mathtt{x}}^{\mathtt{ReadQue}^A} = \mathcal{Q}_{\mathtt{x}}^{\mathtt{Que}^A} = \mathscr{F}_{[\mathfrak{s}^A(\cdot) = \mathtt{x}]}^{\mathbb{Z}} \otimes \mathrm{Id}_m = \sum_{z \,:\, \mathfrak{s}^A(z) = \mathtt{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \,,$$

$$\mathcal{Q}_{\mathtt{y}}^{\mathtt{ReadQue}^B} = \mathcal{Q}_{\mathtt{y}}^{\mathtt{Que}^B} = \mathscr{F}_{[\mathfrak{s}^B(\cdot) = \mathtt{y}]}^{\mathbb{Z}} \otimes \mathrm{Id}_m = \sum_{z \,:\, \mathfrak{s}^B(z) = \mathtt{y}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \,,$$

$$\mathcal{Q}_{\mathtt{x}}^{\mathtt{Hide}^r \mathtt{Que}^A} = \mathscr{F}_{[\mathfrak{s}_{<r}^A(\cdot) = \mathtt{x}]}^{\mathbb{Z}} = \sum_{z \,:\, \mathfrak{s}_{<r}^A(z) = \mathtt{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \,,$$

$$\mathcal{Q}_{\mathtt{y}}^{\mathtt{Hide}^r \mathtt{Que}^B} = \mathscr{F}_{[\mathfrak{s}_{<r}^B(\cdot) = \mathtt{y}]}^{\mathbb{Z}} = \sum_{z \,:\, \mathfrak{s}_{<r}^B(z) = \mathtt{y}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \,.$$

   Namely, all consistency checks on the readable variables in $\mathfrak{QueReo}(\mathfrak{G})$ are satisfied.

Proof. As $\mathscr{S}$ has value $1 - \varepsilon$, and each edge in the sequence $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_{\cdot} - \mathtt{Intro}_{\cdot} - \mathtt{Read}_{\cdot} - \mathtt{Hide}^h_{\cdot} - \ldots - \mathtt{Hide}^1_{\cdot}$ is sampled with probability $\Omega(\frac{1}{h})$, we can deduce that $\mathscr{S}$ passes each such edge with probability of at least $1 - O(h\varepsilon)$. Specifically, from the consistency check along $\mathtt{Pauli}_{\mathbb{Z}} - \mathtt{Sample}_{\cdot}$ together with the fact that small inconsistency implies closeness of PVMs (Proposition 3.12), we can deduce that

$$\mathcal{P}^{\mathtt{SamZ}_{\cdot}} \approx_{O(h\varepsilon)} \mathcal{P}^{\mathtt{Pauli}_{\mathbb{Z}}} = \mathscr{F}^{\mathbb{Z}} \otimes \mathrm{Id}_m \,. \tag{102}$$

By the comparison along $\mathtt{Sample}_{\cdot} - \mathtt{Intro}_{\cdot}$ together with Observation 3.34, we can deduce that

$$\mathcal{P}^{\mathtt{SamZ}_{\cdot}}_{[\mathfrak{s}^{\cdot}(\cdot)=\cdot]} \approx_{O(h\varepsilon)} \mathcal{P}^{\mathtt{Que}_{\cdot}} \,. \tag{103}$$

In combination with (102) and the semi-triangle inequality for PVMs (Item 2. in Proposition 3.12), we get

$$\mathcal{P}^{\mathtt{Que}_{\cdot}} \approx_{O(h\varepsilon)} \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}(\cdot)=\cdot]} \otimes \mathrm{Id}_m \,. \tag{104}$$

Just as a sanity check for the reader, the closeness claim on PVMs in (104) means, for the A vertices, that

$$\sum_{\mathbf{x} \in \mathbb{F}_2^k} \left\| \mathcal{P}^{\mathtt{Que}^A}_{\mathbf{x}} - \sum_{\substack{z \in \mathbb{F}_2^k \\ \mathfrak{s}^A(z)=\mathbf{x}}} \mathscr{F}^{\mathbb{Z}}_z \otimes \mathrm{Id}_m \right\|^2_{hs} \leq O(h\varepsilon) \,.$$

By the comparison along $\mathtt{Intro}_{\cdot} - \mathtt{Read}_{\cdot}$, the $\mathtt{Que}_{\cdot}$ and $\mathtt{ReadQue}_{\cdot}$ observables are highly consistent with each other, and as high consistency implies closeness, we have

$$\mathcal{P}^{\mathtt{ReadQue}_{\cdot}} \approx_{O(h\varepsilon)} \mathcal{P}^{\mathtt{Que}_{\cdot}} \,. \tag{105}$$

In combination with (104) and the semi-triangle inequality, we get

$$\mathcal{P}^{\mathtt{ReadQue}_{\cdot}} \approx_{O(h\varepsilon)} \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}(\cdot)=\cdot]} \otimes \mathrm{Id}_m \,. \tag{106}$$

By the consistency checks along $\mathtt{Read}_{\cdot} - \mathtt{Hide}^h_{\cdot} - \ldots - \mathtt{Hide}^2_{\cdot} - \mathtt{Hide}^1_{\cdot}$ combined with (106), and using h-many times the semi-triangle inequality, we get that

$$\forall 1 \leq r \leq h : \quad \mathcal{P}^{\mathtt{Hide}^r\mathtt{Que}_{\cdot}} \approx_{O(h^3 \cdot \varepsilon)} \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}_{<r}(\cdot)=\cdot]} \otimes \mathrm{Id}_m \,. \tag{107}$$

We can now describe the perturbed strategy $\mathscr{S}'$, with projective form $\mathcal{Q}$ and observable form $\mathcal{W}$. First, it agrees with $\mathscr{S}$ on $\mathfrak{Pauli\ Basis}_k$ vertices (and thus satisfies condition 2.). Then, let

$$\forall z \in \mathbb{F}_2^k : \quad \mathcal{Q}^{\mathtt{SamZ}_{\cdot}}_z = \mathscr{F}^{\mathbb{Z}}_z \otimes \mathrm{Id}_m \,, \tag{108}$$

$$\forall \mathbf{x} \in \mathbb{F}_2^k : \quad \mathcal{Q}^{\mathtt{Que}_{\cdot}}_{\mathbf{x}} = \mathcal{Q}^{\mathtt{ReadQue}_{\cdot}}_{\mathbf{x}} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}(\cdot)=\mathbf{x}]} \otimes \mathrm{Id}_m = \sum_{z \,:\, \mathfrak{s}^{\cdot}(z)=\mathbf{x}} \mathscr{F}^{\mathbb{Z}}_z \otimes \mathrm{Id}_m \,, \tag{109}$$

$$\forall 1 \leq r \leq h, \, \forall \mathbf{x} \in \mathbb{F}_2^k : \quad \mathcal{Q}^{\mathtt{Hide}^r\mathtt{Que}_{\cdot}}_{\mathbf{x}} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}_{<r}(\cdot)=\mathbf{x}]} \otimes \mathrm{Id}_m = \sum_{z \,:\, \mathfrak{s}^{\cdot}_{<r}(z)=\mathbf{x}} \mathscr{F}^{\mathbb{Z}}_z \otimes \mathrm{Id}_m \,. \tag{110}$$

Note that if we extend this choice of $\mathscr{S}'$ to a quantum strategy, then condition 3. is satisfied, which means the only condition left to be verified (after $\mathscr{S}'$ is fully defined) is that it has value at least $1 - O(\sqrt{h^3 \cdot \varepsilon})$. Let us complete the definition of $\mathscr{S}'$: For the rest of the $\mathcal{W}$-observables in the vertices where changes were made, we are going to use Claim 4.9 to change the $\mathcal{U}$-observables so they commute with our choices in (108), (109) and (110). Let us demonstrate this analysis for two vertices, $\mathtt{Sample}_{\cdot}$ and $\mathtt{Read}_{\cdot}$, as for the rest it is essentially the same type of argument. For $\mathtt{Sample}_{\cdot}$, as closeness in L^1 for representations implies closeness in L^∞ (Claim 3.22), we have

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda, \, \beta \in \mathbb{F}_2^k : \quad \mathbb{Z}^{\otimes \beta} \otimes \mathrm{Id}_m \cdot \mathcal{U}^{\mathtt{SamAns}_{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \approx_{O(h\varepsilon)} \mathcal{U}^{\mathtt{SamZ}_{\cdot}}(\beta) \cdot \mathcal{U}^{\mathtt{SamAns}_{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}})$$

$$= \mathcal{U}^{\mathtt{SamAns}_{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \cdot \mathcal{U}^{\mathtt{SamZ}_{\cdot}}(\beta)$$

$$\approx_{O(h\varepsilon)} \mathcal{U}^{\mathtt{SamAns}_{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \cdot \mathbb{Z}^{\otimes \beta} \otimes \mathrm{Id}_m \,,$$

97

where the approximations are due to (102) and Claim 3.22, while the middle equality is due to the fact $\mathcal{U}$ is a quantum strategy (and thus the observables at the same vertex commute). Hence, by Claim 4.9 (or by Orthonormaliztion 3.21), there are observables

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda : \quad \mathcal{W}^{\mathsf{SamAns}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}),$$

which induce a representation of $\mathbb{F}_2^{2\Lambda}$ that commutes with $\mathbb{Z}^\beta \otimes \mathrm{Id}_m$ for every $\beta \in \mathbb{F}_2^k$, and

$$\mathcal{W}^{\mathsf{SamAns}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}) \approx_{O(h\varepsilon)} \mathcal{U}^{\mathsf{SamAns}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}).$$

For Read., we have that for every $\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda$ and $\beta, \gamma \in \mathbb{F}_2^k$,

$$\rho^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}]}(\gamma) \otimes \mathrm{Id}_m \cdot \mathcal{U}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta) \approx_{O(h\varepsilon)} \mathcal{U}^{\mathsf{ReadQue}^{\cdot}}(\gamma) \cdot \mathcal{U}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta)$$

$$= \mathcal{U}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta) \cdot \mathcal{U}^{\mathsf{ReadQue}^{\cdot}}(\gamma)$$

$$\approx_{O(h\varepsilon)} \mathcal{U}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta) \cdot \rho^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}]}(\gamma) \otimes \mathrm{Id}_m.$$

Thus, we can apply Claim 4.9 again to obtain a representation $\mathcal{W}$ on $\mathbb{F}_2^{2\Lambda+k}$ that commutes with the PVM $\mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}(\cdot)=\cdot]} \otimes \mathrm{Id}_m$ and satisfies

$$\mathcal{U}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta) \approx_{O(h\varepsilon)} \mathcal{W}^{\mathsf{ReadAns}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}}(\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}, \beta)$$

for every $\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}$ and β. The change in the Hide vertices depends on the closeness parameter achieved in (107), which is $O(h^3 \cdot \varepsilon)$. All in all, the constructed strategy $\mathscr{S}'$ is $O(h^3 \cdot \varepsilon)$-close to the original one $\mathscr{S}$, and thus by Claim 3.29 it has value of at least $1 - O(\sqrt{h^3 \cdot \varepsilon})$ against $\mathfrak{QueRed}(\mathfrak{G})$, proving clause 1. and completing the proof. $\qquad \square$

Claim 4.28. *Let $\mathscr{S} = \{\mathcal{P}\}$ be a strategy for $\mathfrak{QueRed}(\mathfrak{G})$, acting on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$, satisfying:*

1. *(Almost Perfect)* $\mathrm{val}(\mathscr{S}; \mathfrak{QueRed}(\mathfrak{G})) \geq 1 - \varepsilon$;

2. *(Perfect on Pauli basis and all readable variables)* $\mathscr{S}$ *passes all $\mathfrak{Pauli\,Basis}_k$ edges perfectly, and in addition satisfies*

$$\forall z, \chi \in \mathbb{F}_2^k : \quad \mathcal{P}_z^{\mathtt{Pauli\,z}} = \mathcal{P}_z^{\mathsf{SamZ}^{\cdot}} = \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m \quad , \quad \mathcal{P}_\chi^{\mathtt{Pauli\,x}} = \mathscr{F}_\chi^{\mathbb{X}} \otimes \mathrm{Id}_m,$$

$$\forall \mathbf{x} \in \mathbb{F}_2^k : \quad \mathcal{P}_{\mathbf{x}}^{\mathsf{ReadQue}^{\cdot}} = \mathcal{P}_{\mathbf{x}}^{\mathsf{Que}^{\cdot}} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}(\cdot)=\mathbf{x}]} \otimes \mathrm{Id}_m = \sum_{v: \, \mathfrak{s}^{\cdot}(v)=\mathbf{x}} \mathscr{F}_v^{\mathbb{Z}} \otimes \mathrm{Id}_m,$$

$$\forall 1 \leq r \leq h, \, \forall \mathbf{x} \in \mathbb{F}_2^k : \quad \mathcal{P}_{\mathbf{x}}^{\mathsf{Hide}^r \mathsf{Que}^{\cdot}} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}_{<r}(\cdot)=\mathbf{x}]} = \sum_{z: \, \mathfrak{s}^{\cdot}_{<r}(z)=\mathbf{x}} \mathscr{F}_z^{\mathbb{Z}} \otimes \mathrm{Id}_m. \tag{111}$$

Then, there is another strategy $\mathscr{S}' = \{\mathcal{Q}\}$ for $\mathfrak{QueRed}(\mathfrak{G})$ acting on the same space $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$, satisfying:

1. *(Almost Perfect)* $\mathrm{val}(\mathscr{S}', \mathfrak{QueRed}(\mathfrak{G})) \geq 1 - O(h \cdot 2^h \cdot \sqrt{\varepsilon})$;

2. *(Agrees with $\mathscr{S}$ on $\mathfrak{Pauli\,Basis}_k$ vertices as well as SamZ and Que-variables) For all $\mathbf{x} \in \mathfrak{Pauli\,Basis}_k$, $\mathcal{P}^{\mathbf{x}} = \mathcal{Q}^{\mathbf{x}}$, and in addition the PVM $\mathcal{Q}^{\cdot}$ satisfy the same conditions as $\mathcal{P}^{\cdot}$ in (111).*

3. *(Passes all non-$\mathtt{Intro}_A - \mathtt{Intro}_B$ edges perfectly)*

$$\forall a^{\mathfrak{R}}, a^{\mathfrak{L}} \in \mathbb{F}_2^\Lambda : \quad \mathcal{Q}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathsf{Ans}^{\cdot}} = \mathcal{Q}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathsf{SamAns}^{\cdot}} = \mathcal{Q}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}^{\mathsf{ReadAns}^{\cdot}},$$

$$\forall \mathbf{x}, \nu \in \mathbb{F}_2^k : \quad \mathcal{Q}_{\mathbf{x}, \nu}^{\mathsf{ReadQue}^{\cdot} \sqcup \mathsf{ReadPerp}^{\cdot}} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}(\cdot)=\mathbf{x}]} \cdot \mathscr{F}^{\mathbb{X}}_{[(\mathfrak{s}^{\cdot,\mathbf{x}})^\perp(\cdot)=\mathbf{x}]} \otimes \mathrm{Id}_m$$

$$= \sum_{\substack{z, \chi \in \mathbb{F}_2^k : \\ \mathfrak{s}^{\cdot}(z)=\mathbf{x}, \, (\mathfrak{s}^{\cdot,\mathbf{x}})^\perp(\chi)=\nu}} \mathscr{F}_z^{\mathbb{Z}} \mathscr{F}_\chi^{\mathbb{X}} \otimes \mathrm{Id}_m,$$

$$\forall r, \, \forall \mathbf{x}, \nu \in \mathbb{F}_2^k : \mathcal{Q}_{\mathbf{x}, \nu}^{\mathsf{Hide}^r} = \mathscr{F}^{\mathbb{Z}}_{[\mathfrak{s}^{\cdot}_{<r}(\cdot)=\mathbf{x}]} \cdot \mathscr{F}^{\mathbb{X}}_{[(\mathfrak{G}^{\cdot,\mathbf{x}}_{\leq r})^\perp(\cdot)=\nu]} \otimes \mathrm{Id}_m$$

$$= \sum_{\substack{z, \chi \in \mathbb{F}_2^k : \\ \mathfrak{s}^{\cdot}_{<r}(z)=\mathbf{x}, \, (\mathfrak{G}^{\cdot,\mathbf{x}}_{\leq r})^\perp(\chi)=\nu}} \mathscr{F}_z^{\mathbb{Z}} \mathscr{F}_\chi^{\mathbb{X}} \otimes \mathrm{Id}_m, \tag{112}$$

where $(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^{\perp}$ are the extended perpendicular maps to the seeded CLMs as defined in Item (3) and Remark 4.20.

Proof. Recall that every augmented edge in $\mathfrak{QueReo}(\mathfrak{G})$ is sampled with probability $\Omega(\frac{1}{h})$. For ease of following the proof, let us denote by $C > 0$ the universal constant induced by $\Omega(\frac{1}{h})$, namely the probability of sampling any augmented edge is at least $\frac{1}{Ch}$ and hence $\mathscr{S}$ passes every such edge with probability of at least $1 - Ch\varepsilon$. The reader can check that for the probability distribution on edges we fixed for $\mathfrak{QueReo}(\mathfrak{G})$, each augmented edge is sampled with probability of at least $\frac{1}{2h+4}$, so $C = 6$ is enough. The reason we use an abstract constant instead of 6, is that we later change the distribution over edges in $\mathfrak{QueReo}(\mathfrak{G})$ (see Example 4.39 and specifically Figure 14) so that every augmented edge is sampled with probability of at least $\frac{1}{4h+63}$, in which case $C = 67$ is enough. In any case, let us treat C as an unknown constant.

First, we have

$$
\begin{aligned}
\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^1} &= \mathcal{P}_{\mathbf{x}}^{\mathtt{Hide}^1\mathtt{Que}^{\cdot}}\mathcal{P}_{\nu}^{\mathtt{Hide}^1\mathtt{Perp}^{\cdot}} \\
&= \mathscr{F}_{[(\cdot)_{<1}=\mathbf{x}]}^{\mathbb{Z}} \otimes \mathrm{Id}_m \cdot \mathcal{P}_{\nu}^{\mathtt{Hide}^1\mathtt{Perp}^{\cdot}} \\
&\simeq_{Ch\varepsilon} \mathscr{F}_{[(\cdot)_{<1}=\mathbf{x}]}^{\mathbb{Z}} \otimes \mathrm{Id}_m \cdot \mathcal{P}_{[(\mathfrak{S}_{\leq 1}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathtt{Pauli}_{\mathbb{X}}} \\
&= \mathscr{F}_{[(\cdot)_{<1}=\mathbf{x}]}^{\mathbb{Z}}\mathscr{F}_{[(\mathfrak{S}_{\leq 1}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m \,,
\end{aligned}
$$

where the first equation is by definition for a projective measurement, the second and last equations use the assumptions from (111) on the PVMs at $\mathtt{Pauli}_{\mathbb{X}}$ and $\mathtt{Hide}^1$, and the inconsistency in the middle is due to $\mathscr{S}$ passing $\mathtt{Pauli}_{\mathbb{X}} - \mathtt{Hide}^1$ with probability $1 - Ch\varepsilon$. Hence, by the translation of consistency to closeness (Proposition 3.12), we deduce that

$$
\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^1} \approx_{2Ch\varepsilon} \mathscr{F}_{[(\cdot)_{<1}=\mathbf{x}]}^{\mathbb{Z}}\mathscr{F}_{[(\mathfrak{S}_{\leq 1}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m \,. \tag{113}
$$

We now establish the following inductive step. Assume that for some $1 \leq j < r$ it holds that

$$
\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^{j-1}} \approx_{\delta} \mathscr{F}_{[\mathfrak{s}_{<j-1}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}\mathscr{F}_{[(\mathfrak{S}_{\leq j-1}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m \,, \tag{114}
$$

for some $\delta \geq 2Ch\varepsilon$. Then it follows that

$$
\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^{j}} \approx_{4\delta} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}\mathscr{F}_{[(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathbb{X}} \otimes \mathrm{Id}_m \,. \tag{115}
$$

To show the implication (114) $\implies$ (115), we first use the fact that $\mathscr{S}$ passes $\mathtt{Hide}^{j-1} - \mathtt{Hide}^{j}$ with probability of at least $1 - Ch\varepsilon$ to write

$$
\sum_{\substack{\mathbf{x},\mathbf{x}',\nu,\nu' \\ (\mathbf{x})_{<j-1}=\mathbf{x}' \\ \nu=(\mathfrak{S}_{j}^{\cdot,\mathbf{x}})^{\perp}(\nu')}} \tau\left(\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^{j}}\mathcal{P}_{\mathbf{x}',\nu'}^{\mathtt{Hide}^{j-1}}\right) \geq 1 - Ch\varepsilon \,. \tag{116}
$$

Using the projectivity of $\mathcal{P}^{\mathtt{Hide}^{r}}$ (for every $r \in [h]$) and our assumptions in (111),

$$
\mathcal{P}_{\mathbf{x},\nu}^{\mathtt{Hide}^{r}} = \mathcal{P}_{\mathbf{x}}^{\mathtt{Hide}^r\mathtt{Que}^{\cdot}}\mathcal{P}_{\nu}^{\mathtt{Hide}^r\mathtt{Perp}^{\cdot}} = \mathscr{F}_{[\mathfrak{s}_{<r}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}\mathcal{P}_{\nu}^{\mathtt{Hide}^r\mathtt{Perp}^{\cdot}} \,.
$$

For every $\mathbf{x} \in \mathbb{F}_2^k$, let

$$
A_{\nu}^{\mathbf{x}} = \mathcal{P}_{[(\mathfrak{S}_{j}^{\cdot,\mathbf{x}})^{\perp}(\cdot)=\nu]}^{\mathtt{Hide}^{j-1}\mathtt{Perp}^{\cdot}} = \sum_{\nu':\, (\mathfrak{S}_{j}^{\cdot,\mathbf{x}})^{\perp}(\nu')=\nu} \mathcal{P}_{\nu}^{\mathtt{Hide}^{j-1}\mathtt{Perp}^{\cdot}} \,.
$$

This is a data processed version of $\mathcal{P}^{\mathtt{Hide}^{j-1}\mathtt{Perp}^{\cdot}}$, but the exact function through which we are evaluating depends on the seed $\mathbf{x}$. Then, (116) can be rewritten as

$$
\sum_{\mathbf{x},\nu} \tau\left(\mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}\mathcal{P}_{\nu}^{\mathtt{Hide}^{j}\mathtt{Perp}^{\cdot}} A_{\nu}^{\mathbf{x}}\right) \geq 1 - Ch\varepsilon \,,
$$

which means that $\mathcal{P}_{\mathbf{x},\nu}^{\text{Hide}^j}$ is $Ch\varepsilon$-inconsistent with $\mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} A_\nu^\mathbf{x} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}$ — note that this is a POVM but not necessarily a projective one. In general, if one has three PVMs, B and C with outcomes in X, and D with outcomes in Y, then $B \approx_\delta C$ implies $DBD \approx_\delta DCD$. Hence, using (114), we can deduce that

$$\mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} A_\nu^\mathbf{x} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \approx_\delta \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{S}_j^{\cdot,\mathbf{x}})^\perp \circ (\mathfrak{S}_{\leq j-1}^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} = \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}} ,$$

where the equation uses both that, as defined in Remark 4.20, $(\mathfrak{S}_j^{\cdot,\mathbf{x}})^\perp \circ (\mathfrak{S}_{\leq j-1}^{\cdot,\mathbf{x}})^\perp = (\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^\perp$, as well as the fact that $\mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}}$ and $\mathscr{F}_{[(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}}$ commute. Combined with the above, we have

$$\mathcal{P}_{\mathbf{x},\nu}^{\text{Hide}^j} \approx_{2Ch\varepsilon} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} A_\nu^\mathbf{x} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \approx_\delta \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}} ,$$

which implies (115) using the semi-triangle inequality for closeness of POVMs and the fact $\delta \geq 2Ch\varepsilon$.

This establishes the desired implication (114) $\implies$ (115). Together with the base case (113), we deduce that

$$\forall j \in [h] : \quad \mathcal{P}_{\mathbf{x},\nu}^{\text{Hide}^j} \approx_{4^j Ch\varepsilon} \mathscr{F}_{[\mathfrak{s}_{<j}(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{S}_{\leq j}^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}} \otimes \text{Id}_m . \tag{117}$$

By the consistency check along $\text{Hide}^h - \text{Read}.$, we deduce from (117) that

$$\mathcal{P}_{\mathbf{x},\nu}^{\text{ReadQue} \sqcup \text{ReadPerp}} \approx_{4^{h+1} Ch\varepsilon} \mathscr{F}_{[\mathfrak{s}.(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathscr{F}_{[(\mathfrak{s}.^{\cdot,\mathbf{x}})^\perp(\cdot)=\nu]}^{\mathbb{X}} \otimes \text{Id}_m . \tag{118}$$

Let us choose the following signed permutation representations of $\mathbb{F}_2^k$,

$$B^\cdot(\alpha) = \sum_{\mathbf{x} \in \mathbb{F}_2^k} \mathscr{F}_{[\mathfrak{s}.(\cdot)=\mathbf{x}]}^{\mathbb{Z}} \mathbb{X}^{\otimes \alpha \cdot (\mathfrak{s}.^{\cdot,\mathbf{x}})^\perp} ,$$

where, as usual, $\alpha \cdot (\mathfrak{s}.^{\cdot,\mathbf{x}})^\perp$ is the left multiplication of the row vector α with the matrix $(\mathfrak{s}.^{\cdot,\mathbf{x}})^\perp$ — this is indeed a matrix as $(\mathfrak{s}.^{\cdot,\mathbf{x}})^\perp$ is a linear map for every fixed $\mathbf{x}$. By applying the inverse Fourier transform to (118), one can deduce that

$$\mathcal{U}^{\text{ReadPerp}} \approx_{4^{h+1} Ch\varepsilon} B^\cdot . \tag{119}$$

By the consistency checks along $\text{Sample}. - \text{Intro}. - \text{Read}.$, we deduce that

$$\mathcal{U}^{\text{SamAns}} \approx_{Ch\varepsilon} \mathcal{U}^{\text{Ans}} \approx_{Ch\varepsilon} \mathcal{U}^{\text{ReadAns}} . \tag{120}$$

Hence, using the "small L^1-distance between representations implies small L^∞ distance" proved in Claim 3.22 twice, combined with (111) and (120), we have

$$\forall \alpha^\mathfrak{R}, \alpha^\mathfrak{L} \in \mathbb{F}_2^\Lambda , \beta \in \mathbb{F}_2^k : \quad \mathbb{Z}^{\otimes \beta} \otimes \text{Id}_m \cdot \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \approx_{6Ch\varepsilon} \mathcal{U}^{\text{SamZ}}(\beta) \cdot \mathcal{U}^{\text{SamAns}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L})$$
$$= \mathcal{U}^{\text{SamAns}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \cdot \mathcal{U}^{\text{SamZ}}(\beta)$$
$$\approx_{6Ch\varepsilon} \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \cdot \mathbb{Z}^{\otimes \beta} \otimes \text{Id}_m .$$

Similarly, using (119), (120) and Claim 3.22, we have

$$\forall \alpha^\mathfrak{R}, \alpha^\mathfrak{L} \in \mathbb{F}_2^\Lambda , \beta \in \mathbb{F}_2^k : \quad B^\cdot(\beta) \cdot \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \approx_{6 \cdot 4^{h+1} Ch\varepsilon} \mathcal{U}^{\text{ReadPerp}}(\beta) \cdot \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L})$$
$$\approx_{6Ch\varepsilon} \mathcal{U}^{\text{ReadPerp}}(\beta) \cdot \mathcal{U}^{\text{ReadAns}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L})$$
$$= \mathcal{U}^{\text{ReadAns}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \cdot \mathcal{U}^{\text{ReadPerp}}(\beta)$$
$$\approx_{6Ch\varepsilon} \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \cdot \mathcal{U}^{\text{ReadPerp}}(\beta)$$
$$\approx_{6 \cdot 4^{h+1} Ch\varepsilon} \mathcal{U}^{\text{Ans}}(\alpha^\mathfrak{R}, \alpha^\mathfrak{L}) \cdot B^\cdot(\beta) .$$

Note that, as $B^{\cdot}$ and $\rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$ are both signed permutation representations, the group generated by their images is finite. We can thus apply Claim 4.9 where G is the group generated by the images of $B^{\cdot}$ and $\rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$ (which also fixes the representation of G in the claim), and A is $\mathbb{F}_2^{2\Lambda}$ with representation $\psi = \mathcal{U}^{\mathsf{Ans}^{\cdot}}$; this gives us two representations θ^A, θ^B of $\mathbb{F}_2^{2\Lambda}$ such that $\theta^A \approx_{O(4^h h \varepsilon)} \mathcal{U}^{\mathsf{Ans}^A}$, $\theta^B \approx_{O(4^h h \varepsilon)} \mathcal{U}^{\mathsf{Ans}^B}$ and $\theta^{\cdot}$ perfectly commutes with $B^{\cdot}$ and $\rho^{\mathbb{Z}} \otimes \mathrm{Id}_m$.

Combining all of the above, if we let $\mathscr{S}' = \{\mathcal{V}\} = \{\mathcal{Q}\}$ satisfy (111), (112) and

$$\mathcal{V}^{\mathsf{Ans}^A} = \theta^A \quad , \quad \mathcal{V}^{\mathsf{Ans}^B} = \theta^B \;,$$

then $\mathscr{S}'$ satisfies clause 2. and 3. from the requirements of this claim, as well as being $O(4^h \cdot h \cdot \varepsilon)$-close to $\mathscr{S}$. By applying Claim 3.29, we can deduce clause 1. as well. $\qquad \square$

To conclude the soundness proof, we need to combine the three preceding claims:

1. Given a strategy $\mathscr{S}$ acting on $\mathbb{C}^N$ that has value $1 - \varepsilon$, we can apply Claim 4.26 on it to get a strategy $\mathscr{S}'$ that acts on $\mathbb{C}^{\mathbb{F}_2^k} \otimes \mathbb{C}^m$, passes the $\mathfrak{Pauli\,Basis}_k$-vertices perfectly and has value of $1 - O(\sqrt{(1 + k^2/d^2)\varepsilon})$. Moreover, the strategies are $O((1 + k^2/d^2)\varepsilon)$-close on $\mathtt{Pauli_X}$ and $\mathtt{Pauli_Z}$ observables, which means in particular that $1 - \frac{N}{2^k \cdot m} \leq O((1 + k^2/d^2)\varepsilon)$ and thus $N \geq (1 - O((1 + k^2/d^2)\varepsilon))2^k \cdot m$.

2. The strategy $\mathscr{S}'$ satisfies the assumptions of Claim 4.27, and thus there is a strategy $\mathscr{S}''$ for $\mathfrak{QueRed}(\mathfrak{G})$ that behaves well on all Que-variables and has value $1 - O\left(h^{3/2} \cdot ((1 + k^2/d^2)\varepsilon)^{1/4}\right)$.

3. The strategy $\mathscr{S}''$ satisfies the assumptions of Claim 4.28, and thus there is a strategy $\mathscr{S}'''$ which passes all edges of $\mathfrak{QueRed}(\mathfrak{G})$ perfectly (except for maybe $\mathtt{Intro}_A - \mathtt{Intro}_B$), and has value of at least $1 - O(h^2 \cdot 2^h \cdot (1 + k^2/d^2) \cdot \varepsilon^{1/8})$, which proves the soundness in Item (2). In addition, the resulting strategy is honest (Definition 4.4). Hence, by Claim 4.6, such a strategy induces a strategy for $\mathfrak{G}$ with the same value which acts on $\mathbb{C}^m$. Hence, $m \geq \mathscr{E}(\mathfrak{G}, 1 - O(h^2 \cdot 2^h \cdot (1 + k^2/d^2) \cdot \varepsilon^{1/8}))$, and we can conclude that

$$\mathscr{E}(\mathfrak{QueRed}(\mathfrak{G}), 1 - \varepsilon) \geq 2^k \cdot (1 - O((1 + k^2/d^2)\varepsilon)) \cdot \mathscr{E}(\mathfrak{G}, 1 - O(h^2 \cdot 2^h \cdot (1 + k^2/d^2) \cdot \varepsilon^{1/8})) \;,$$

which proves the entanglement lower bound Item (3).

4.5 Applying question reduction to a tailored normal form verifier

Up until now, we discussed a certain combinatorial transformation that takes as input an integer k, a set $\mathscr{B} = \{w_1, ..., w_N\} \subseteq \mathbb{F}_2^k$, and a game $\mathfrak{G}$ (with certain assumptions on its sampling mechanism and answer length functions), and outputs a new game $\mathfrak{QueRed}(\mathfrak{G})$ (defined in Section 4.4), which is a specific augmented sum of $\mathfrak{Pauli\,Basis}_k(\mathscr{B})$ from Section 3.8.3 and $\mathfrak{Intro}(\mathfrak{G})$ from Section 4.1.

For the proof of Compression (Theorem 2.53), one needs a way of applying this combinatorial transformation on the level of tailored normal form verifiers (TNFVs), as was described in Theorem 4.1. Namely, we seek a transformation on a pair consisting of an integer λ and a TNFV $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ that outputs a new TNFV $\mathsf{QuestionReduction}(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}_{\mathrm{QR}}^\lambda, \mathcal{A}_{\mathrm{QR}}^\lambda, \mathcal{L}', \mathcal{D})$, such that on the combinatorial level the nth game of $\mathcal{V}'$ is the question reduced 2^n-th game of $\mathcal{V}$. So,

$$\mathcal{V}'_n = \mathfrak{QueRed}(\mathcal{V}_{2^n}, k(n, \lambda), \mathscr{B}(n, \lambda))$$

for some integer-valued function $k(n, \lambda)$ and function $\mathscr{B}(n, \lambda)$ valued in tuples of vectors in $\mathbb{F}_2^{k(n, \lambda)}$.

Recall that for $\mathfrak{QueRed}_h(\mathfrak{G}, k, \mathscr{B})$ to have the desired properties of Theorem 4.24, several non-trivial assumptions about the inputs need to be satisfied:

(1) The game $\mathfrak{G}$ is tailored, with an underlying h-level CL sampling scheme (Definition 4.16); namely, its underlying graph's vertex set is $\mathbb{F}_2^r$ and its distribution over edges μ is the pushforward of the uniform distribution over $\mathbb{F}_2^r$ through a fixed pair of h-level conditionally linear maps (Definition 4.13) $\mathfrak{s} = (\mathfrak{s}^A, \mathfrak{s}^B) \colon \mathbb{F}_2^r \to \mathbb{F}_2^r \times \mathbb{F}_2^r$.

Note that in the definition of $\mathfrak{QueRed}(\mathfrak{G})$ we always insisted that this parameter r, controlling the number of vertices in $\mathfrak{G}$, to be equal to k, which is the length of answers at the $\texttt{Pauli}_{\mathbb{X}}$ and $\texttt{Pauli}_{\mathbb{Z}}$ vertices in the generalized Pauli basis game $\mathfrak{Pauli\,Basis}_k(\mathscr{B})$. But, and it is straightforward to check, we only needed k to be larger or equal to r — that is because we can pre-compose $(\mathfrak{s}^A, \mathfrak{s}^B)$ with the restriction to the first r coordinates rest: $\mathbb{F}_2^k \to \mathbb{F}_2^r$ and proceed accordingly. All in all, we need the sampling procedure of $\mathfrak{G}$ to be induced by two CLMs $\mathfrak{s}^A, \mathfrak{s}^B \colon \mathbb{F}_2^r \to \mathbb{F}_2^r$ and for $r \leq k$.

As the input game $\mathfrak{G}$ is assumed to have a sampling procedure induced by an h-level CLM, we need the TNFV $\mathcal{V}$ that we manipulate to be such that for every $n \in \mathbb{N}$, the nth game $\mathcal{V}_n$ has an underlying h-level sampling scheme. In addition, every transformation that we apply on $\mathcal{V}$, e.g. QuestionReduction, should retain this property. Hence, we are going to move to a subcategory of TNFVs that have this property.

(2) In addition to the sampling procedure, we needed $\mathfrak{G}$ to have constant length functions $\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}} \colon \mathbb{F}_2^r \to \mathbb{N}$. Namely, there is some integer Λ such that for all $\mathbf{x} \in \mathbb{F}_2^r$ the functions satisfy $\ell^{\mathfrak{R}}(\mathbf{x}) = \ell^{\mathfrak{L}}(\mathbf{x}) = \Lambda$. This turns out to be an easy condition to satisfy, and even if the normal form verifier does not satisfy it, a *padding* transformation can be applied on it so that it does (see Section 4.5.4).

(3) We aim to "question reduce", which translates to making the underlying graph of $\mathfrak{QueRed}(\mathfrak{G})$ exponentially smaller than that of $\mathfrak{G}$, i.e., the number of questions in it needs to be $\mathrm{poly}(r)$.[57] As $\mathfrak{Intro}(\mathfrak{G})$ contributes 2 vertices, and the augmentation adds on $2h + 4$ more[58], most of the vertices in the underlying graph of $\mathfrak{QueRed}(\mathfrak{G})$ come from $\mathfrak{Pauli\,Basis}_k$. In $\mathfrak{Pauli\,Basis}_k$, there are $2 + 2N + 18N^2$ vertices, where $N = |\mathscr{B}|$. In addition, for the soundness and entanglement lower bounds proved before (Item (2) and Item (3) of Theorem 4.24) to have any meaning, we need the parameter k/d to be bounded, where d is the distance of the code induced by the set $\mathscr{B}$. A tradeoff arises: For the distance of the code defined by $\mathscr{B}$ to be large enough, the set itself needs to be large enough (in particular larger than k). But, on the other hand, the larger $\mathscr{B}$ is the larger the underlying graph of $\mathfrak{QueRed}(\mathfrak{G})$ is. Finally, there should be an efficient way of calculating, given $i \in [N]$, the vector $w_i \in \mathscr{B}$. This point is solved completely by the existence of good error correcting codes with an efficient algorithm to calculate their encoding matrix (Fact 3.72).

(4) As part of the application of $\mathfrak{QueRed}$, we used quite a lot of structure regarding the CLMs $\mathfrak{s}^A$ and $\mathfrak{s}^B$. We assumed to have access to the spaces $W_r^{\mathbf{x}}$, the jth step $\mathfrak{s}_j^{\cdot, \mathbf{x}}$ of the CLMs calculation for every $1 \leq j \leq h$ and seed $\mathbf{x}$, and also to the *perpendicular* functions $(\mathfrak{s}_j^{\cdot, \mathbf{x}})^{\perp}$. To handle this intricacy, we are going to change the definition of a sampler (Definition 2.39) so that it can describe further its "inner working".

4.5.1 The category of h-level tailored normal form verifiers

Definition 4.29 (h-level conditionally linear sampler). Let h be a positive integer. An h-level conditionally linear sampler (CL sampler) is a 6-input deterministic Turing machine $\mathcal{S}$ that satisfies the following restrictions. First, the input to $\mathcal{S}$ is expected to be

$$(n, \text{Action}, \text{Player}, j, \mathbf{x}, z) \,,$$

where n and j are positive integers in binary, Action is taken from the set

$$\{\text{Dimension}, \text{Register}, \text{Marginal}, \text{Evaluate}, \text{Perpendicular}\} \,,$$

Player is taken from the set $\{A, B\}$, and $\mathbf{x}, z$ are bit strings (interpreted as vectors in some finite vector space). Second, for every positive integer n, there must exist an integer $r = r(n)$ and two h-level conditionally linear maps (CLMs, Definition 4.13) $\mathfrak{s}^A(n), \mathfrak{s}^B(n) = \mathfrak{s}^A, \mathfrak{s}^B \colon \mathbb{F}_2^r \to \mathbb{F}_2^r$, such that $\mathcal{S}$ *encodes* this pair of functions:

1. If $\mathcal{S}$ gets as input $(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$, then it outputs (the binary encoding of) $r(n)$.

---

[57]Actually, it is enough to be quasi-polynomial in r, which is the parameter setup used in [JNV$^+$21].

[58]For the proof of Compression, h can be upper bounded by 5 (see Remark 4.35). So this can be thought of as a constant number of vertices.

2. If $\mathcal{S}$ gets as input $(n, \text{Register}, \text{Player}, j, \mathbf{x}, \cdot)$, then it outputs the jth register subspace with respect to the seed $\mathbf{x}$, namely $W_j^{\mathbf{x}}$ (71) with respect to $\mathfrak{s}^{\text{Player}}$. As $W_j^{\mathbf{x}}$ is a register subspace, the way it is encoded is by providing the indicator of the set $I_j^{\mathbf{x}} \subseteq [r]$, which says what are the standard basis vectors that span $W_j^{\mathbf{x}}$. Note that the indicator of $I_j^{\mathbf{x}}$ is just a bit string of length r.

3. If $\mathcal{S}$ gets as input $(n, \text{Marginal}, \text{Player}, j, \cdot, z)$, then it outputs the jth prefix of $\mathfrak{s}^{\text{Player}}$'s evaluation of z, namely $\mathfrak{s}_{\leq j}^{\text{Player}}(z)$.

4. If $\mathcal{S}$ gets as input $(n, \text{Evaluate}, \text{Player}, j, \mathbf{x}, z)$, then it outputs the jth-register output of $\mathfrak{s}^{\text{Player}}$ evaluated on z given the seed $\mathbf{x}$, namely $\mathfrak{s}_j^{\text{Player}, \mathbf{x}}(z^{W_j^{\mathbf{x}}})$. Recall that $\mathfrak{s}_j^{\text{Player}, \mathbf{x}} : W_j^{\mathbf{x}} \to W_j^{\mathbf{x}}$ is the linear function which controls the jth step in the calculation of $\mathfrak{s}^{\text{Player}}$, given that the calculation up to this point produced $\mathbf{x}_{<j}$.

5. If $\mathcal{S}$ gets as input $(n, \text{Perpendicular}, \text{Player}, j, \mathbf{x}, z)$, then it outputs the jth-register output of $(\mathfrak{s}^{\text{Player}, \mathbf{x}})^{\perp}$ evaluated on z given the seed $\mathbf{x}$, namely $(\mathfrak{s}_j^{\text{Player}, \mathbf{x}})^{\perp}(z^{W_j^{\mathbf{x}}})$. Recall that the maps $(\mathfrak{s}_j^{\text{Player}, \mathbf{x}})^{\perp} : W_j^{\mathbf{x}} \to W_j^{\mathbf{x}}$ are some fixed linear function whose rows are spanning the subspace perpendicular to the rows of $\mathfrak{s}_j^{\text{Player}, \mathbf{x}}$.[59]

Remark 4.30. A few things to note about the differences between the above definition and [JNV$^+$21, Definition 4.14]: They use the Action name "Linear" instead of "Evaluate". Furthermore, they do not include the Perpendicular action — This is because there is a canonical (and efficient) way of calculating linear maps $(\mathfrak{s}_j^{\text{Player}, \mathbf{x}})^{\perp}$ from the rest of the possible outputs of the sampler, as described in [JNV$^+$21, Section 8.2] clause 6. in page 94, which is the detailed description of the decider of their introspective verifier.

Remark 4.31 (Dimension of CL sampler bounded by running time)**.** Since, given inputs such as $(n, \text{Marginal}, \text{Player}, j, \cdot, z)$, the CL sampler $\mathcal{S}$ needs to output an $r(n)$-long bit string, where $r(n) = \mathcal{S}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$, it is immediate that $r(n) \leq \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$.

Remark 4.32. Our original sampler from Definition 2.39 is a randomized TM, while the h-level sampler is a deterministic one. To extract the output of the original sampler out of an h-level sampler, we can — though it is not part of the definition of a CL sampler — attach a Sample action to the list. Given $(n, \text{Sample}, \cdot, \cdot, \cdot, \cdot)$, $\mathcal{S}$ runs as follows:

1. First, it calls $\mathcal{S}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ to obtain r.

2. Then, it samples r random bits to obtain $z \in \mathbb{F}_2^r$ (which makes $\mathcal{S}$ back to a randomized TM, but only with respect to this Sample action).

3. It then calls $\mathcal{S}(n, \text{Marginal}, A, h, \cdot, z)$ to obtain $\mathbf{x} = \mathfrak{s}_{\leq h}^A(z) = \mathfrak{s}^A(z)$, and $\mathcal{S}(n, \text{Marginal}, B, h, \cdot, z)$ to obtain $\mathbf{y} = \mathfrak{s}_{\leq h}^B(z) = \mathfrak{s}^B(z)$.

4. Finally, it outputs (the encoding of) $\mathbf{x} \sqcup \mathbf{y}$.

Note that, indeed, this is what we expect a TM to do to be able to sample from the distribution induced by the pair $(\mathfrak{s}^A, \mathfrak{s}^B)$.

Definition 4.33 (Tailored h-level normal form verifier)**.** An *h-level tailored normal form verifier* (h-level TNFV) is a quadruple of Turing machines $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$, where $\mathcal{S}$ is an h-level conditionally linear sampler as in Definition 4.29, $\mathcal{A}$ is an answer length calculator as in Definition 2.41, $\mathcal{L}$ is a linear constraint processor as in Definition 2.43, and $\mathcal{D}$ is the canonical decider as in Definition 2.45.

Such a TNFV is *λ-bounded*, for a positive integer λ, if

---

[59]Note that these maps were not part of the definition of a CLM, but were assumed to be part of the data needed for question reduction in the beginning of Section 4.4. As Remark 4.30 notes, there is a canonical way of extracting such maps from the rest of the CLMs data.

- The running times (Definition 2.36) of $\mathcal{S}, \mathcal{A}$ and $\mathcal{L}$ are all bounded by n^λ , namely

$$\forall n \in \{0,1\}^* : \ \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \, , \, \mathbb{T}(\mathcal{A}; n, \cdot, \cdot) \, , \, \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \ \le \ n^\lambda.$$

- The description length (Definition 2.37) of $\mathcal{V}$ is bounded by λ, namely $|\mathcal{V}| \le \lambda$.

Similar to Definition 2.48, when $\mathcal{V}$ is a λ-bounded h-level TNFV, then there is an associated n^{th} game to it for every $n \ge 2$. Similar to Remark 2.49, the nth game of an h-level TNFV may be well defined even if it is not λ-bounded. Actually, all we need is for

- $\mathcal{A}(n, \mathrm{x}, \kappa)$ to halt whenever x is of length $r(n) = \mathcal{S}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$;

- $\mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ needs to halt whenever x, y are of length $r(n)$, and $a^{\mathfrak{R}}$ and $b^{\mathfrak{R}}$ are of length $|\text{dec}(\mathcal{A}(n, \mathrm{x}, \mathfrak{R}))|$ and $|\text{dec}(\mathcal{A}(n, \mathrm{y}, \mathfrak{R}))|$ respectively.

Note that by assuming $\mathcal{S}$ is an h-level CL sampler (Definition 4.29), we are guaranteed that it behaves well, in particular it always halts (on relevant inputs), and there are associated CLMs underlying it. So, no additional assumptions on $\mathcal{S}$ are needed.

We now have all the definitions required to formulate the version of compression (Theorem 2.53) which is proved in this paper. Recall the asymptotic notation from Remark 1.2.

Theorem 4.34 (Compression of h-level tailored normal form verifiers). *For every positive integer h, there exist two positive integers*

$$c = c(h) \quad \text{and} \quad C = C(h)$$

that depend only on h, and a 2-input Turing machine $\mathsf{Compress}_h$, *that takes as input a h-level TNFV $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ and a positive integer λ (in binary), and outputs a **5-level** TNFV* $\mathsf{Compress}_h(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}^\lambda, \mathcal{A}^\lambda, \mathcal{L}', \mathcal{D})$, *such that:*

- *Sampler properties: The 5-level CL sampler $\mathcal{S}^\lambda$ depends only on λ and h, (but not the specific $\mathcal{V}$), and* $\mathsf{Compress}_h$ *can calculate its description in time* $\text{polylog}_h(\lambda)$;[60] *in particular, $|\mathcal{S}^\lambda| \le c \log^c \lambda$. In addition, $\mathcal{S}^\lambda$ runs in $\text{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \ \mathbb{T}(\mathcal{S}^\lambda; n) \le c \cdot (n^c + \lambda^c) \, .$$

- *Answer length calculator properties: $\mathcal{A}^\lambda$ depends only on λ and h, and* $\mathsf{Compress}_h$ *can calculate its description in time* $\text{polylog}_h(\lambda)$; *in particular $|\mathcal{A}^\lambda| \le c \log^c \lambda$. In addition, $\mathcal{A}^\lambda$ runs in $\text{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \ \mathbb{T}(\mathcal{A}^\lambda; n, \cdot, \cdot) \le c \cdot (n^c + \lambda^c) \, .$$

Finally, given that $\mathrm{x} \in \mathbb{F}_2^{r(n)}$, where $r(n) = \mathcal{S}^\lambda(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and that $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the output of $\mathcal{A}^\lambda(n, \mathrm{x}, \kappa)$ never decodes (Definition 2.34) to an $\mathfrak{error}$ sign.

- *Linear constraints process properties: $\mathcal{L}'$ depends on both λ and $\mathcal{V}$, and* $\mathsf{Compress}_h$ *can calculate its description in time* $\text{poly}_h(\log \lambda, |\mathcal{V}|)$; *in particular, $|\mathcal{L}'| \le c \cdot (\log^c \lambda + |\mathcal{V}|^c)$. In addition, $\mathcal{L}'$ runs in $\text{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \ \mathbb{T}(\mathcal{L}'; n, \cdot, \cdot, \cdot, \cdot) \ \le \ c \cdot (n^c + \lambda^c) \, .$$

- *Decider properties: The canonical decider $\mathcal{D}$ (Definition 2.45) is fixed and runs in time which is linear in its input length.*

- *Value properties: If $\mathcal{V}$ is λ-bounded, then $\mathcal{V}'$, the output of* $\mathsf{Compress}_h$, *satisfies: For all $n \ge C$,*

---

[60]Recall our asymptotic notation from Remark 1.2 to parse polylog_h.

1. **Completeness**: If $\mathcal{V}_{2^n}$ has a perfect Z-aligned permutation strategy that commutes along edges (ZPC strategy), then so does $\mathcal{V}'_n$.

2. **Soundness**: $\mathscr{E}(\mathcal{V}'_n, \frac{1}{2}) \geq \max\left\{\mathscr{E}(\mathcal{V}_{2^n}, \frac{1}{2}), 2^{2^{\lambda n}-1}\right\}$.

Remark 4.35. The above version of compression, Theorem 4.34, is the one proven in this paper. By choosing $h = 5$, and using Compress$_5$ from the above theorem instead of Compress from Theorem 2.53, TailoredMIP* = RE (Theorem 2.31) can still be deduced exactly as in Section 2.6.

In a similar way, we now describe the version of Theorem 4.1 which is proved in this section. Recall again the asymptotic notation from Remark 1.2.

Theorem 4.36 (*h*-level Question Reduction). *Let h be a positive integer. There exists a positive integer*

$$c = c_{\text{QR}}(h) \tag{121}$$

that depends only on h, and a 2-input Turing machine QuestionReduction$_h$ *that takes as input an h-level TNFV* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *and a positive integer λ (in binary), and outputs a new* 3-**level** *TNFV*

$$\text{QuestionReduction}_h(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}^\lambda_{\text{QR}}, \mathcal{A}^\lambda_{\text{QR}}, \mathcal{L}', \mathcal{D})$$

such that:

- *Sampler properties:* $\mathcal{S}^\lambda_{\text{QR}}$ *depends only on λ and h (and not the specific $\mathcal{V}$), and* QuestionReduction$_h$ *can calculate its description in time* $\text{polylog}_h(\lambda)$*; in particular, $|\mathcal{S}^\lambda_{\text{QR}}| \leq c \log^c \lambda$. In addition, $\mathcal{S}^\lambda$ runs in $\text{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{S}^\lambda_{\text{QR}}; n) \leq c \cdot (n^c + \lambda^c) \, .$$

- *Answer length calculator properties:* $\mathcal{A}^\lambda_{\text{QR}}$ *depends only on λ and h, and* QuestionReduction$_h$ *can calculate its description in time* $\text{polylog}_h(\lambda)$*; in particular $|\mathcal{A}^\lambda_{\text{QR}}| \leq c \log^c \lambda$. In addition, $\mathcal{A}^\lambda$ runs in $\exp_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{A}^\lambda; n, \cdot, \cdot) \leq 2^{c \cdot (n^c + \lambda^c)} \, .$$

Finally, given that $\mathrm{x} \in \mathbb{F}_2^{r(n)}$*, where* $r(n) = \mathcal{S}^\lambda_{\text{QR}}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$*, and that* $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$*, the output of* $\mathcal{A}^\lambda_{\text{QR}}(n, \mathrm{x}, \kappa)$ *never decodes (Definition 2.34) to an* error *sign.*

- *Linear constraints process properties:* $\mathcal{L}'$ *depends on λ, h and $\mathcal{V}$, and* QuestionReduction$_h$ *can calculate its description in time* $\text{poly}_h(\log \lambda, |\mathcal{V}|)$*; in particular, $|\mathcal{L}'| \leq c \cdot (\log^c \lambda + |\mathcal{V}|^c)$. In addition, $\mathcal{L}'$ runs in $\exp_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{L}'; n, \cdot, \cdot, \cdot, \cdot) \leq 2^{c \cdot (n^c + \lambda^c)} \, .$$

Note that the running time upper bound itself is independent of the specific $\mathcal{V}$.

- *Value properties: If $\mathcal{V}$ is λ-bounded, then $\mathcal{V}'$, the output of* QuestionReduction$_h$*, satisfies: For all $n \geq 2$,*

    1. **Completeness**: *If $\mathcal{V}_{2^n}$ has a perfect Z-aligned permutation strategy, then so does $\mathcal{V}'_n$.*

    2. **Soundness**: *For every $\varepsilon > 0$, if $\mathcal{V}'_n$ has a value $1 - \varepsilon$ strategy, then $\mathcal{V}_{2^n}$ has a value $1 - c \cdot \varepsilon^{1/16}$ strategy.*

    3. **Entanglement**: *For every $\varepsilon > 0$,*

$$\mathscr{E}(\mathcal{V}'_n, 1-\varepsilon) \geq (1 - c \cdot \varepsilon) \cdot 2^{2^{\lambda n}} \cdot \mathscr{E}(\mathcal{V}_{2^n}, 1 - c \cdot \varepsilon^{1/16}) \, .$$

Remark 4.37. As claimed in the theorem, the output of QuestionReduction$_h$ is always a 3-**level** normal form verifier, regardless of what h was. The original level h plays a role in the underlying graph of $\mathcal{V}'$, as well as in the exact soundness guarantees, governed by $c_{\text{QR}}(h)$ (121).

4.5.2 Typed conditionally linear sampling schemes

Though we focus on the category of games induced by (two) h-level CLMs — and respectively on h-level TNFVs — it will often be easier to describe the underlying sampling scheme of our games in a slightly different manner.

Definition 4.38. A game $\mathfrak{G}$ is said to have a *h-level typed conditionally linear sampling scheme* if its underlying graph (V, E) and measure μ on edges are defined as follows: There is a positive integer r and a set $\mathcal{T}$ — which we call the *type set*, and its elements are called *types* — such that $V = \mathcal{T} \times \mathbb{F}_2^r$. In addition, there is a subset $\mathcal{E}$ of $\mathcal{T} \times \mathcal{T}$ which induces a graph structure $(\mathcal{T}, \mathcal{E})$, and an h-level CLM $\mathfrak{s}^t : \mathbb{F}_2^r \to \mathbb{F}_2^r$ associated to every type $t \in \mathcal{T}$. Then, a pair in $V = \mathcal{T} \times \mathbb{F}_2^r$ is sampled by taking a uniformly random seed $z \in \mathbb{F}_2^r$ and a uniformly random edge $tt' \in \mathcal{E}$ and outputting the pair $(t, \mathfrak{s}^t(z)), (t', \mathfrak{s}^{t'}(z))$ — namely, the pair of types are chosen uniformly from $\mathcal{E}$, and the seed defines the right coordinates by evaluating the CLM of the chosen types on it.

Example 4.39. Let $\mathfrak{G}$ be a game with h-level CLMs acting on $\mathbb{F}_2^k$ controlling its sampling scheme, and let $\mathscr{B} \subseteq \mathbb{F}_2^k$ be a set of size 2^m for some $m \geq \log k$. Then, the sampling scheme of $\mathfrak{QueRed}(\mathfrak{G}) = \mathfrak{QueRed}_h(\mathfrak{G}, k, \mathscr{B})$ which was described in Section 4.4, is (up to some constant factor that depends only on h) a 1-level typed conditionally linear sampling scheme. To see that, the type set $\mathcal{T}$ consists of

$$\forall 1 \leq j \leq h : \quad \texttt{Hide}_A^j \,,\, \texttt{Hide}_B^j \,,$$
$$\texttt{Intro}_A \,,\, \texttt{Intro}_B \,,\, \texttt{Read}_A \,,\, \texttt{Read}_B \,,\, \texttt{Sample}_A \,,\, \texttt{Sample}_B \,,$$
$$\texttt{Pauli}_\mathsf{Z} \,,\, \texttt{Pauli}_\mathsf{X} \,,\, \mathsf{X} \,,\, \mathsf{Z} \,,\, \texttt{First} \,,\, \texttt{Second} \,,\, \texttt{Both} \,,$$
$$\forall 1 \leq a \leq 3 \,,\, 1 \leq b \leq 3 : \quad \texttt{var}_{ab} \,,\, \texttt{row}_a \,,\, \texttt{col}_b \,.$$

All in all, $2h + 28$ types. They are connected according to Figure 14 — though not drawn in the figure, we assume **all loops** appear in the type graph, namely for any type t of the above $2h + 28$ types, in addition to the edges in the figure, the edge tt is also present.

Now, the 1-level CLMs (i.e., linear maps) associated to the types act on the space $\mathbb{F}_2^{2m}$, which has (not surprisingly) the same cardinality as $\mathscr{B} \times \mathscr{B}$. For all non Pauli basis types, i.e.,

$$\texttt{Hide}_A^j \,,\, \texttt{Hide}_B^j \,,\, \texttt{Intro}_A \,,\, \texttt{Intro}_B \,,\, \texttt{Read}_A \,,\, \texttt{Read}_B \,,\, \texttt{Sample}_A \,,\, \texttt{Sample}_B \,,$$

and for $\texttt{Pauli}_\mathsf{X}$ and $\texttt{Pauli}_\mathsf{Z}$, the associated CLM is the 0-function. Namely, for every such type t,

$$\forall u, v \in \mathbb{F}_2^m : \quad \mathfrak{s}^t(u, v) = (\vec{0}, \vec{0}) \,.$$

For all other types t except for X, Z, i.e., types from the list

$$\texttt{row}_a \,,\, \texttt{col}_b \,,\, \texttt{var}_{ab} \,,\, \texttt{First} \,,\, \texttt{Second} \,,\, \texttt{Both} \,,$$

the associated CLM is the identity map. Namely,

$$\forall u, v \in \mathbb{F}_2^m : \quad \mathfrak{s}^t(u, v) = (u, v).$$

Finally, for X we have

$$\forall u, v \in \mathbb{F}_2^m : \quad \mathfrak{s}^\mathsf{X}(u, v) = (u, \vec{0}).$$

and for Z we have

$$\forall u, v \in \mathbb{F}_2^m : \quad \mathfrak{s}^\mathsf{Z}(u, v) = (\vec{0}, v).$$

Figure 14: The type graph of $\mathfrak{Que}\mathfrak{Red}(\mathfrak{G})$. Though not drawn, all self loops are also edges in this type graph. So, in total, there are $2h + 28$ vertices and $4h + 63$ edges in this type graph.

In this perspective, the vertices of $\mathfrak{Que}\mathfrak{Red}(\mathfrak{G})$ are from $\mathcal{T} \times \mathbb{F}_2^m \times \mathbb{F}_2^m$. But the vertices with a positive probability to be sampled belong to the following strict subset:

$$
\begin{aligned}
\forall 1 \leq j \leq h : \quad & (\text{Hide}_A^j, \vec{0}, \vec{0}), (\text{Hide}_B^j, \vec{0}, \vec{0}), \\
& (\text{Intro}_A, \vec{0}, \vec{0}), (\text{Intro}_B, \vec{0}, \vec{0}), (\text{Read}_A, \vec{0}, \vec{0}), (\text{Read}_B, \vec{0}, \vec{0}), \\
& (\text{Sample}_A, \vec{0}, \vec{0}), (\text{Sample}_B, \vec{0}, \vec{0}), (\text{Pauli}_{\mathbb{Z}}, \vec{0}, \vec{0}), (\text{Pauli}_{\mathbb{X}}, \vec{0}, \vec{0}) \\
\forall u, v \in \mathbb{F}_2^m : \quad & (\text{X}, u, \vec{0}), (\text{Z}, \vec{0}, v), \\
\forall 1 \leq a, b \leq 3, \, u, v \in \mathbb{F}_2^m : \quad & (\text{var}_{ab}, u, v), (\text{row}_a, u, v), (\text{col}_b, u, v), \\
\forall u, v \in \mathbb{F}_2^m : \quad & (\text{First}, u, v), (\text{Second}, u, v), (\text{Both}, u, v).
\end{aligned}
$$

It is straightforward to compare these vertices to the ones we originally had when defining $\mathfrak{Que}\mathfrak{Red}(\mathfrak{G})$:

1. The vertices

$$\forall 1 \leq j \leq h : \; \left(\mathtt{Hide}_A^j, \vec{0}, \vec{0}\right), \left(\mathtt{Hide}_B^j, \vec{0}, \vec{0}\right), \left(\mathtt{Intro}_A, \vec{0}, \vec{0}\right), \left(\mathtt{Intro}_B, \vec{0}, \vec{0}\right), \left(\mathtt{Read}_A, \vec{0}, \vec{0}\right)$$
$$\left(\mathtt{Read}_B, \vec{0}, \vec{0}\right), \left(\mathtt{Sample}_A, \vec{0}, \vec{0}\right), \left(\mathtt{Sample}_B, \vec{0}, \vec{0}\right), \left(\mathtt{Pauli}_{\mathbb{Z}}, \vec{0}, \vec{0}\right), \left(\mathtt{Pauli}_{\mathbb{X}}, \vec{0}, \vec{0}\right),$$

are the same vertices as

$$\forall 1 \leq j \leq h : \; \mathtt{Hide}_A^j, \mathtt{Hide}_B^j, \mathtt{Intro}_A, \mathtt{Intro}_B, \mathtt{Read}_A,$$
$$\mathtt{Read}_B, \mathtt{Sample}_A, \mathtt{Sample}_B, \mathtt{Pauli}_{\mathbb{Z}}, \mathtt{Pauli}_{\mathbb{X}},$$

in the original description.

2. The vertices

$$\forall u, v \in \mathbb{F}_2^m : \; \left(\mathtt{X}, u, \vec{0}\right), \left(\mathtt{Z}, \vec{0}, v\right),$$

are the same as the vertices

$$\forall u, v \in \mathbb{F}_2^m : \; \mathtt{X}^u, \mathtt{Z}^v,$$

in the original description of $\mathfrak{Pauli\,Basis}_k$ in Section 3.8.3. This makes sense as the set $\mathscr{B}$ is of size $N = 2^m$ which can be parametrized by $\mathbb{F}_2^m$.[61]

3. The vertices

$$\forall 1 \leq a, b \leq 3, \; u, v \in \mathbb{F}_2^m : \; \left(\mathtt{var}_{ab}, u, v\right), \left(\mathtt{row}_a, u, v\right), \left(\mathtt{col}_b, u, v\right),$$

are the same as the vertices

$$\forall 1 \leq a, b \leq 3, \; u, v \in \mathbb{F}_2^m : \; \mathtt{var}_{ab}^{u,v}, \mathtt{row}_a^{u,v}, \mathtt{col}_b^{u,v},$$

from the (u, v)-th copy of the anti-commutation game (or its nullified version) $\mathfrak{M}^{u,v}$ used in the definition of $\mathfrak{Pauli\,Basis}_k$.

4. The vertices

$$\forall u, v \in \mathbb{F}_2^m : \; \left(\mathtt{First}, u, v\right), \left(\mathtt{Second}, u, v\right), \left(\mathtt{Both}, u, v\right),$$

are the same as the vertices

$$\forall u, v \in \mathbb{F}_2^m : \; \mathtt{First}^{u,v}, \mathtt{Second}^{u,v}, \mathtt{Both}^{u,v},$$

from the (u, v)-th copy of the commutation game (or its nullified version) $\mathfrak{C}^{u,v}$ used in the definition of $\mathfrak{Pauli\,Basis}_k$.

The distribution on edges of $\mathfrak{QueRed}(\mathfrak{G})$ induced by this typed 1-level CL sampling scheme is not *exactly* the one we used before, but the probabilities are the same up to some constant factor (which depends on h). For example, the probability of sampling $\mathtt{Intro}_A - \mathtt{Intro}_B$ in this setup is $\frac{1}{|\mathcal{E}|} = \frac{1}{63+4h}$, which is lower than the $1/4$ we had before, while the probability of sampling $\mathtt{var}_{11}^{u,v} - \mathtt{row}_1^{u,v}$ is $\frac{1}{(63+4h)\cdot 2^{2m}}$, which may be lower or higher than the $\frac{1}{72\cdot 2^{2m}}$ it was before (depending on h). Though these distributions differ, the new distribution samples an edge with probability of at least $\frac{1}{63+4h}$ times the old probability. Thus, all of our soundness arguments (in which the distribution played a role) are the same up to a constant depending on h.

---

[61] When we defined $\mathfrak{Pauli\,Basis}_k$ we used i, j to parametrize the elements of $\mathscr{B}$ instead of u, v.

4.5.3 Detyping

As Compression (Theorem 4.34) begins and ends with an h-level normal form verifier, and not with typed ones, we ought to have a method of *detyping* a sampling scheme in a way that preserves most of the properties of the original game. To that end we make the following definition.

Definition 4.40 (Combinatorial Detyping). Let $\mathfrak{G}$ be a game with a h-level typed CL sampling scheme (Definition 4.38), where the type graph is $(\mathcal{T}, \mathcal{E})$, and where the CLMs $\mathfrak{s}^t$ act on $\mathbb{F}_2^r$. The *detyped version* of $\mathfrak{G}$, which we denote by $\mathfrak{DeType}(\mathfrak{G})$, is a game with an $(h+2)$-level non-typed CL sampling scheme (Definition 4.16), with CLMs $\mathfrak{s}^A, \mathfrak{s}^B$ acting on $\mathbb{F}_2^{4|\mathcal{T}|+r}$, and which are defined as follows: First, the type $t \in \mathcal{T}$ is encoded as a string

$$\text{enc}(t) = (\mathbf{1}_t, \sum_{t' \sim t} \mathbf{1}_{t'}) \in \mathbb{F}_2^{\mathcal{T}} \times \mathbb{F}_2^{\mathcal{T}},$$

namely the first vector is the indicator of t, and the second vector is the indicator of neighbors of t in the graph $(\mathcal{T}, \mathcal{E})$. Given

$$(u_{type}, u_{neighbors}, v_{type}, v_{neighbors}, z) \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^r,$$

$\mathfrak{s}^A$ operates as follows:

1. First, $\mathfrak{s}^A$ applies the identity on $u_{type}, u_{neighbors}$.

2. Then, there are two options — either there exists a $t \in \mathcal{T}$ such that $(u_{type}, u_{neighbors}) = \text{enc}(t)$, or not. If not, then $\mathfrak{s}^A$ zeros out the rest of the registers, namely outputs $(u_{type}, u_{neighbors}, \vec{0}, \vec{0}, \vec{0})$. If there is such a $t \in \mathcal{T}$, then it zeroes out all v_{type} and all coordinates of $v_{neighbors}$ except the tth one.

3. Lastly, if the partial computation of the first two steps resulted in $(\text{enc}(t), \vec{0}, \mathbf{1}_t) \in (\mathbb{F}_2^{\mathcal{T}})^4$, then it applies $\mathfrak{s}^t$ on the seed $z \in \mathbb{F}_2^r$, and otherwise zeroes z out.

The CLM $\mathfrak{s}^B$ acts similarly with the roles of u and v swapped. Namely:

1. First, $\mathfrak{s}^B$ applies the identity on $v_{type}, v_{neighbors}$.

2. Then, there are two options — either there exists a $t' \in \mathcal{T}$ such that $(v_{type}, v_{neighbors}) = \text{enc}(t')$, or not. If not, then $\mathfrak{s}^B$ zeros out the rest of the registers, namely outputs $(\vec{0}, \vec{0}, v_{type}, v_{neighbors}, \vec{0})$. If there is such a $t' \in \mathcal{T}$, then it zeroes out all u_{type} and all coordinates of $u_{neighbors}$ except the tth one.

3. Lastly, if the partial computation of the first two steps resulted in $(\vec{0}, \mathbf{1}_{t'}, \text{enc}(t')) \in (\mathbb{F}_2^{\mathcal{T}})^4$, then it applies $\mathfrak{s}^{t'}$ on the seed $z \in \mathbb{F}_2^r$, and otherwise zeroes z out.

Given $t \in \mathcal{T}$, the vector $(\text{enc}(t), \vec{0}, \mathbf{1}_t, \mathbf{x})$ is called *the A-copy of* $(t, \mathbf{x})$, and similarly $(\vec{0}, \mathbf{1}_{t'}, \text{enc}(t'), \mathbf{y})$ is called *the B-copy of* $(t', \mathbf{y})$. A vector of the form $(u_{type}, u_{neighbors}, \vec{0}, \vec{0}, \vec{0})$ is called an *A-player anchor vertex*, and a vector of the form $(\vec{0}, \vec{0}, v_{type}, v_{neighbors}, \vec{0})$ is called a *B-player anchor vertex*.[62] Note that the above sampling procedure produces a pair with at least one anchor vertex, unless there is an edge $tt' \in \mathcal{E}$ in the type graph such that $(u_{type}, u_{neighbors}, v_{type}, v_{neighbors}) = (\text{enc}(t), \text{enc}(t'))$, in which case

$$\mathfrak{s}^A(\text{enc}(t), \text{enc}(t'), z) = (\text{enc}(t), \vec{0}, \mathbf{1}_t, \mathfrak{s}^t(z)) \quad , \quad \mathfrak{s}^B(\text{enc}(t), \text{enc}(t'), z) = (\vec{0}, \mathbf{1}_{t'}, \text{enc}(t'), \mathfrak{s}^{t'}(z)),$$

namely this is an edge between the *A-copy of* $(t, \mathfrak{s}^t(z))$ and the *B-copy of* $(t', \mathfrak{s}^{t'}(z))$. All in all, the only vertices that have a positive probability of being sampled are those $\cdot$-player "anchor" and "copy" vertices.

---

[62]As will soon be described, whenever an anchor vertex is sampled, the game always accepts. In the case of detyping, these vertices are added just for the CL sampling structure to be attained. Later in this paper, the idea of anchoring a game is used for parallel repetition — see Section 6.

For the length functions, if $\mathtt{w}$ is a vertex of $\mathfrak{DeType}(\mathfrak{G})$ which is the A or B copy of a vertex $(t, \mathtt{x})$ in $\mathfrak{G}$, then $\ell_{\mathfrak{DeType}(\mathfrak{G})}(\mathtt{w}) = \ell_{\mathfrak{G}}(t, \mathtt{x})$. Namely, the length of the copies of any vertex in the detyped game is the same as in the original game. For the rest of the vertices, the length functions are zero.

For the decision procedure: If one of the vertices of the sampled edge is an anchor vertex, then the game accepts no matter what the answers are. Otherwise, the seed was of the form $(\mathrm{enc}(t), \mathrm{enc}(t'), z)$, and thus the sampled edge is between the A-copy of $(t, \mathfrak{s}^z(t))$ and the B-copy of $(t', \mathfrak{s}^{t'}(z))$; in this case, $\mathfrak{DeType}(\mathfrak{G})$ checks the answers at these vertices as if they were from $\mathfrak{G}$, and decides accordingly. In addition, if $t = t'$, then the detyped game will also check consistency, namely that the answers at the A-copy of $(t, \mathfrak{s}^z(t))$ and the B-copy of the same vertex $(t, \mathfrak{s}^z(t))$ are the same.

Remark 4.41 (The double cover embeds in the detyped game). Note that the game $\mathfrak{G}$ does not embed in $\mathfrak{DeType}(\mathfrak{G})$ — every vertex $(t, \mathfrak{s}^t(z)) \in \mathcal{T} \times \mathbb{F}_2^r$ is mapped to two vertices,

$$(\mathrm{enc}(t), \vec{0}, \mathbf{1}_t, \mathfrak{s}^t(z)) \quad \text{and} \quad (\vec{0}, \mathbf{1}_t, \mathrm{enc}(t), \mathfrak{s}^t(z)) .$$

But, by restricting the detyped game $\mathfrak{DeType}(\mathfrak{G})$ only to such vertices (i.e., A-copies and B-copies of vertices from $\mathfrak{G}$), we get a copy of the **double cover** $\mathfrak{DoubleCover}(\mathfrak{G})$ (Definition 3.52) of the game $\mathfrak{G}$. Moreover, this copy is played with probability of at least $2^{-4|\mathcal{T}|}$ (and otherwise, one of the sampled vertices is an anchor one, which means it automatically accepts).

The above remark leads us to the following immediate corollary.

Corollary 4.42 (Completeness and soundness of the detyped game. Cf. Lemma 6.18 in [JNV$^+$21]). *Let $\mathfrak{G}$ be a tailored game with a h-level conditionally linear sampling scheme, type graph $(\mathcal{T}, \mathcal{E})$, and CLMs which act on a space of dimension r. Then, letting $\mathfrak{DeType}(\mathfrak{G})$ be the detyping of $\mathfrak{G}$ as in Definition 4.40, we have the following:*

- *$\mathfrak{DeType}(\mathfrak{G})$ is a tailored game with an $(h+2)$-level conditionally linear sampling scheme.*

- *(Completeness) If $\mathfrak{G}$ has a perfect ZPC strategy, then so does its detyping $\mathfrak{DeType}(\mathfrak{G})$.*

- *(Soundness) If the detyped game $\mathfrak{DeType}(\mathfrak{G})$ has a value $1 - \varepsilon$ strategy, then the double cover $\mathfrak{DoubleCover}(\mathfrak{G})$ (Definition 3.52) of the original game $\mathfrak{G}$ has a value $1 - O(2^{4|\mathcal{T}|}\varepsilon)$ strategy.*

- *(Entanglement) In addition,*

$$\mathscr{E}(\mathfrak{G}', 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{DoubleCover}(\mathfrak{G}), 1 - O(2^{4|\mathcal{T}|}\varepsilon)) .$$

Proof sketch. The claim item is clear and follows by inspection. In particular, the description of $\mathfrak{s}^A$ and $\mathfrak{s}^B$ in Definition 4.40 make them clearly $(h+2)$-level. The soundness and entanglement lower bound are immediate from Remark 4.41. For completeness, recall that the double cover has a perfect ZPC strategy given that $\mathfrak{G}$ has one (Claim 3.54). Every such strategy extends, in a ZPC manner, to the rest of the vertices of $\mathfrak{DeType}(\mathfrak{G})$ — as the lengths at all the other vertices is 0. As the decision in $\mathfrak{DeType}(\mathfrak{G})$ always accepts when one of the endpoints of the sampled edges is an anchor, the resulting ZPC strategy for $\mathfrak{DeType}(\mathfrak{G})$ is indeed perfect. $\square$

Corollary 4.43. *If in the type graph $(\mathcal{T}, \mathcal{E})$ underlying the sampling scheme of $\mathfrak{G}$, all self loops tt are edges in $\mathcal{E}$, then the detyped game $\mathfrak{DeType}(\mathfrak{G})$ satisfies the following strengthened soundness and entanglement lower bound conditions: If $\mathfrak{DeType}(\mathfrak{G})$ has a value $1 - \varepsilon$ strategy, then the original game $\mathfrak{G}$ (and not its double cover) has a value $1 - O(|\mathcal{T}| \cdot 2^{2|\mathcal{T}|} \cdot \sqrt{\varepsilon})$ strategy of the same dimension, which implies*

$$\mathscr{E}(\mathfrak{G}', 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{G}, 1 - O(|\mathcal{T}| \cdot 2^{2|\mathcal{T}|} \cdot \sqrt{\varepsilon})) .$$

Proof. Recall that if $\mathfrak{G}$ has a typed h-level sampling scheme (Definition 4.38), then it samples edges as follows: It chooses a uniform edge of types $tt' \in \mathcal{E}$, and a uniform $z \in \mathbb{F}_2^r$, and returns the edge $(t, \mathfrak{s}^t(z))(t', \mathfrak{s}^{t'}(z))$. This means that the marginal distribution $\mu(t, \mathfrak{s}^t(z))$ is

$$\mathop{\mathbb{P}}_{z' \in \mathbb{F}_2^r}[\mathfrak{s}^t(z') = \mathfrak{s}^t(z)] \cdot \frac{|\{t' \neq t \mid tt' \in \mathcal{E}\}| + |\{t' \neq t \mid t't \in \mathcal{E}\}| + 1}{|\mathcal{E}|} ,$$

while the probability $\mu((t, \mathfrak{s}^t(z))(t, \mathfrak{s}^t(z)))$ of choosing this loop is

$$\mathop{\mathbb{P}}_{z' \in \mathbb{F}_2^r}[\mathfrak{s}^t(z') = \mathfrak{s}^t(z)] \cdot \frac{1}{|\mathcal{E}|} .$$

Hence, for every $\mathbf{x} \in \mathcal{T} \times \mathbb{F}_2^r$ (with positive probability of being sampled) we have

$$\frac{\mu(\mathbf{xx})}{\mu(\mathbf{x})} \geq \frac{1}{2|\mathcal{T}| + 1} .$$

Combining Corollary 4.42 and Claim 3.54 finishes the proof. $\qquad\square$

As for all of our combinatorial transformations, we need to implement them on the level of normal form verifiers to prove compression. Thus, we need to define the *typed* version of an h-level CL sampler, which will underlie some normal form verifier instead of the *usual h-level sampler* (Definition 4.29). The definitions are very similar. The main difference is that the following encodes a sequence of CLMs parametrized by the vertices of a fixed finite type graph $(\mathcal{T}, \mathcal{E})$, instead of just two sequences of CLMs. Actually, the following can be seen as a generalization of the non-typed h-level sampler, where in that case the type graph consists of two vertices A and B with a single oriented edge AB between them.

Definition 4.44 (Typed h-level conditionally linear sampler)**.** Let h be a positive integer. Colloquially, a *h-level typed conditionally linear sampler* (typed CL sampler) $\mathcal{S}$ with underlying type graph $(\mathcal{T}, \mathcal{E})$ is, essentially, an h-level conditionally linear sampler (Definition 4.29), but instead of the Player action having only 2 possible inputs, it has $|\mathcal{T}|$ inputs.

Formally, $\mathcal{S}$ is a 6-input deterministic Turing machine that satisfies some additional properties. First, its input is expected to be

$$(n, \mathrm{Action}, \mathrm{Type}, j, \mathbf{x}, z) ,$$

where n and j are positive integers in binary, Action is taken from the set

$$\{\mathrm{Graph},\ \mathrm{Dimension},\ \mathrm{Register},\ \mathrm{Marginal},\ \mathrm{Evaluate},\ \mathrm{Perpendicular}\} ,$$

Type is taken from the set $\mathcal{T}$, and $\mathbf{x}, z$ are bit strings (interpreted as vectors in some finite vector space). Second, for every positive integer n, there exist an integer $r = r(n)$, and for every type $t \in \mathcal{T}$ there is an h-level conditionally linear map (CLM, Definition 4.13) $\mathfrak{s}^t(n) = \mathfrak{s}^t \colon \mathbb{F}_2^r \to \mathbb{F}_2^r$, such that $\mathcal{S}$ *encodes* the appropriate typed h-level sampling scheme (Definition 4.38):

1. If $\mathcal{S}$ gets as input $(\cdot, \mathrm{Graph}, \cdot, \cdot, \cdot, \cdot)$, then it outputs the graph $(\mathcal{T}, \mathcal{E})$ in the following way: It provides a list of all the types in $\mathcal{T}$ according to some order, and then the adjacency matrix associated to $\mathcal{E}$ with respect to the order induced on $\mathcal{T}$.

2. If $\mathcal{S}$ gets as input $(\overline{n}, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$, then it outputs (the binary encoding of) $r(n)$.

3. If $\mathcal{S}$ gets as input $(\overline{n}, \mathrm{Register}, \mathrm{Type}, j, \mathbf{x}, \cdot)$, then it outputs the jth register subspace with respect to the seed $\mathbf{x}$, namely $W_j^{\mathbf{x}}$ (71) with respect to $\mathfrak{s}^{\mathrm{Type}}$.

4. If $\mathcal{S}$ gets as input $(\overline{n}, \mathrm{Marginal}, \mathrm{Type}, j, \cdot, z)$, then it outputs the jth prefix of $\mathfrak{s}^{\mathrm{Type}}$'s evaluation of z, namely $\mathfrak{s}_{\leq j}^{\mathrm{Type}}(z)$.

5. If $\mathcal{S}$ gets as input $(\overline{n}, \text{Evaluate}, \text{Type}, j, \mathrm{x}, z)$, then it outputs the jth-register output of $\mathfrak{s}^{\text{Type}}$ evaluated on z given the seed x, namely $\mathfrak{s}_j^{\text{Type},\mathrm{x}}(z^{W_j^{\mathrm{x}}})$. Recall that $\mathfrak{s}_j^{\text{Type},\mathrm{x}} : W_j^{\mathrm{x}} \to W_j^{\mathrm{x}}$ is the linear function which controls the jth step in the calculation of $\mathfrak{s}^{\text{Type}}$, given that the calculation up to this point produced $\mathrm{x}_{<j}$.

6. If $\mathcal{S}$ gets as input $(\overline{n}, \text{Perpendicular}, \text{Type}, j, \mathrm{x}, z)$, then it outputs the jth-register output of $(\mathfrak{s}^{\text{Type}})^{\perp}$ evaluated on z given the seed x, namely $(\mathfrak{s}_j^{\text{Type},\mathrm{x}})^{\perp}(z^{W_j^{\mathrm{x}}})$. Recall that the maps $(\mathfrak{s}_j^{\text{Type},\mathrm{x}})^{\perp} : W_j^{\mathrm{x}} \to W_j^{\mathrm{x}}$ are some fixed linear function whose rows are spanning the subspace perpendicular to the rows of $\mathfrak{s}_j^{\text{Type},\mathrm{x}}$.

Conditions 3. to 6. above are essentially identical to conditions 2. to 5. in Definition Definition 4.29, except that the "Player" input is replaced by the "Type" input, which may have a bigger range (i.e. the set $\mathcal{T}$).

Definition 4.45 (Typed h-level tailored normal form verifier)**.** A *typed h-level TNFV* is a quadruple of Turing machines $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$, where $\mathcal{S}$ is a typed h-level CL sampler as in Definition 4.44, $\mathcal{A}$ is an answer length calculator as in Definition 2.41, $\mathcal{L}$ is a linear constraint processor as in Definition 2.43, and $\mathcal{D}$ is the canonical decider as in Definition 2.45.
Such a typed normal form verifier is said λ-*bounded*, for a positive integer λ, if

- The running times (Definition 2.36) of $\mathcal{S}, \mathcal{A}$ and $\mathcal{L}$ are all bounded by n^{λ}, namely

$$\forall \overline{n} \in \{0,1\}^* : \ \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot) , \ \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot) , \ \mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \ \leq \ n^{\lambda} .$$

- The description length of $\mathcal{V}$ is bounded by λ, namely $|\mathcal{V}| \leq \lambda$ (Definition 2.37).

Similar to Definition 2.48, Remark 2.49 and Definition 4.33, when $\mathcal{V}$ is a λ-bounded tailored typed h-level normal form verifier, then there is an associated n^{th} game to it for every $n \geq 2$. Furthermore, the nth game of such a normal form verifier is *well defined*, even if it is not λ-bounded, if the normal form verifier satisfies the following conditions:

- $\mathcal{A}(\overline{n}, (t, \mathrm{x}), \kappa)$ halts whenever $t \in \mathcal{T}$, where $(\mathcal{T}, \mathcal{E})$ is the appropriate decoding of $\mathcal{S}(\cdot, \text{Graph}, \cdot, \cdot, \cdot, \cdot)$, x is of length $r(n) = \mathcal{S}(\overline{n}, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$.

- $\mathcal{L}(\overline{n}, (t, \mathrm{x}), (t', \mathrm{y}), a^{\mathfrak{R}}, b^{\mathfrak{R}})$ halts whenever $t, t' \in \mathcal{T}$, x, y are of length $r(n)$, and $a^{\mathfrak{R}}$ and $b^{\mathfrak{R}}$ are of length $|\text{dec}(\mathcal{A}(\overline{n}, \mathrm{x}, \mathfrak{R}))|$ and $|\text{dec}(\mathcal{A}(\overline{n}, \mathrm{y}, \mathfrak{R}))|$ respectively.

Note that by assuming $\mathcal{S}$ is a typed h-level CL sampler (Definition 4.44), we are guaranteed that it behaves well, in particular it always halts (on relevant inputs), and there is a type graph and associated CLMs underlying it. So, no additional assumptions on $\mathcal{S}$ are needed.

Claim 4.46 (Algorithmic detyping of normal form verifiers)**.** *There exists a Turing machine* DeType_h *that takes as input a* **typed** *h-level TNFV* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *and outputs a* (**non-typed**) $(h+2)$*-level TNFV*

$$\text{DeType}_h(\mathcal{V}) = \mathcal{V}' = (\mathcal{S}', \mathcal{A}', \mathcal{L}', \mathcal{D})$$

such that:

- *(Combinatorial DeTyping) For every* $n \in \mathbb{N}$*, if* $\mathcal{V}_n$ *is well defined (Definition 4.45), then* $\mathcal{V}'_n$ *is well defined (Definition 4.33) and satisfies* $\mathcal{V}'_n = \mathfrak{DeType}(\mathcal{V}_n)$ *with respect to the underlying type graph* $(\mathcal{T}, \mathcal{E})$ *decoded from* $\mathcal{S}(\cdot, \text{Graph}, \cdot, \cdot, \cdot, \cdot)$*.*

- *(Sampler properties) The* $(h+2)$*-level sampler* $\mathcal{S}'(\overline{n}, \cdot, \cdot, \cdot, \cdot, \cdot)$ *runs in time which is polynomial (with constants that may depend on h) in:*
    - *the number of types* $|\mathcal{T}|$*, where* $(\mathcal{T}, \mathcal{E})$ *is the type graph decoded from* $\mathcal{S}(\cdot, \text{Graph}, \cdot, \cdot, \cdot, \cdot)$*;*
    - *the running time* $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$*.*

    Moreover, DeType_h *calculates the description of* $\mathcal{S}'$ *in polynomial time from the description of* $\mathcal{S}$*, and in particular* $|\mathcal{S}'| = \text{poly}_h(|\mathcal{S}|)$*.*

- *(Answer length properties) The output answer length calculator $\mathcal{A}'(\overline{n}, \mathrm{x}, \kappa)$ runs in time which is polynomial (with constants that may depend on h) in:*

    - *the number of types $|\mathcal{T}|$;*
    - *the running time $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$;*
    - *the running time $\mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot)$.*

    Moreover, DeType_h calculates the description of $\mathcal{A}'$ in polynomial time in that of $\mathcal{S}$ and $\mathcal{A}$; in particular, $|\mathcal{A}'| = \mathrm{poly}_h(|\mathcal{S}|, |\mathcal{A}|)$.

- *(Linear constraints processor properties) The ouput linear constraints processor TM $\mathcal{L}'(\overline{n}, \cdot, \cdot, \cdot, \cdot)$ runs in time polynomial (with constants that may depend on h) in:*

    - *the number of types $|\mathcal{T}|$;*
    - *the running time $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$;*
    - *the running time $\mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot)$;*
    - *the running time $\mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$.*

    Moreover, DeType_h calculates the description of $\mathcal{L}'$ is polynomial time from the descriptions of $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{L}$; in particular $|\mathcal{L}'| = \mathrm{poly}_h(|\mathcal{S}|, |\mathcal{A}|, |\mathcal{L}|)$.

Proof. Throughout this proof, we use the notation $\mathrm{enc}(t) = (\mathbf{1}_t, \sum_{t' \sim t} \mathbf{1}_{t'}) \in \mathbb{F}_2^{\mathcal{T}} \times \mathbb{F}_2^{\mathcal{T}}$ from Definition 4.40. Let us describe the sampler $\mathcal{S}'$, answer length calculator $\mathcal{A}'$ and linear constraints processor $\mathcal{L}'$ of $\mathcal{V}' = \mathsf{DeType}_h(\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$.

The Sampler: We start with a detailed description of the operations of the detyped sampler $\mathcal{S}'$. The operation of the sampler follows Definition 4.40. The details of the implementation are straightforward, but we include them for completeness.

1. $\mathcal{S}'(\overline{n}, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ runs as follows: First, it calls $\mathcal{S}(\cdot, \mathrm{Graph}, \cdot, \cdot, \cdot, \cdot)$ to extract the set of types $\mathcal{T}$, and thus the size of this set. Then, it calls $\mathcal{S}(\overline{n}, \mathrm{dimension}, \cdot, \cdot, \cdot, \cdot)$ to extract $r(n)$. Finally, it outputs $r'(n) = 4|\mathcal{T}| + r(n)$.

2. $\mathcal{S}'(\overline{n}, \mathrm{Register}, \mathrm{Player}, j, \mathrm{x}, \cdot)$ runs as follows: First, it runs as $\mathcal{S}'(\overline{n}, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ — which was defined in the previous clause — to extract the type set $\mathcal{T}$, the edge set $\mathcal{E}$, and the dimension $r'(n) = 4|\mathcal{T}| + r(n)$. Then, it checks that x is a bit string of length $4|\mathcal{T}| + r(n)$ and that $1 \leq j \leq h + 2$, and outputs an error sign otherwise.

    Now, if $\mathrm{Player} = A$ and $j = 1$, then it outputs a bit string of length $4|\mathcal{T}| + r(n)$ whose first $2|\mathcal{T}|$ entries are 1 and the rest are 0, namely

    $$(\sum_{t \in \mathcal{T}} \mathbf{1}_t, \sum_{t \in \mathcal{T}} \mathbf{1}_t, \vec{0}, \vec{0}, \vec{0}) \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}.$$

    Recall that such a bit string is interpreted as the registers which define W_1^{x} with respect to the CLM $\mathfrak{s}^A$, and in this case it means that regardless of x, this space consists of the first and second copies of $\mathbb{F}_2^{\mathcal{T}}$.

    If $\mathrm{Player} = A$ and $j = 2$, then it outputs a bit string of length $4|\mathcal{T}| + r(n)$ whose first $2|\mathcal{T}|$ entries are 0, the following $2|\mathcal{T}|$ bits are 1, and the rest are 0, namely

    $$(\vec{0}, \vec{0}, \sum_{t \in \mathcal{T}} \mathbf{1}_t, \sum_{t \in \mathcal{T}} \mathbf{1}_t, \vec{0}) \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}.$$

    This means that regardless of x, the register subspace W_2^{x} with respect to the CLM $\mathfrak{s}^A$ is always spanned by the third and fourth copies of $\mathbb{F}_2^{\mathcal{T}}$.

    If $\mathrm{Player} = A$ and $j \geq 3$, then it reads the first $4|\mathcal{T}|$ bits of x. If there is no type $t \in \mathcal{T}$ such that these bits are equal to $(\mathrm{enc}(t), \vec{0}, \mathbf{1}_t)$, then it splits into two cases — if $j = 3$, then it outputs a bit string of length $4|\mathcal{T}| + r(n)$ whose first $4|\mathcal{T}|$ entries are 0, and the rest are 1, namely

    $$(\vec{0}, \vec{0}, \vec{0}, \vec{0}, \vec{1}) \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}.$$

This is interpreted as W_x^3 being a copy of $\mathbb{F}_2^{r(n)}$. And, if $j > 3$, then it outputs a bit string of length $4|\mathcal{T}| + r(n)$ consisting of only zeros, namely $\vec{0} \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}$. This is interpreted as $W_x^j = \{\vec{0}\}$ being the trivial space for any $3 < j \leq h + 2$.

Otherwise, there is a type $t \in \mathcal{T}$ such that $x|_{\mathbb{F}_2^{4\mathcal{T}}} = (\text{enc}(t), \vec{0}, \mathbf{1}_t)$. In this case, $\mathcal{S}'$ calls

$$\mathcal{S}(n, \text{Register}, t, j - 2, x|_{\mathbb{F}_2^{r(n)}}, \cdot),$$

whose output is denoted by $\vec{i} \in \mathbb{F}_2^{r(n)}$; then, it outputs a bit string of length $4|\mathcal{T}| + r(n)$ whose first $4|\mathcal{T}|$ entries are 0, and the rest are $\vec{i}$, namely

$$(\vec{0}, \vec{0}, \vec{0}, \vec{0}, \vec{i}) \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}.$$

This is interpreted as W_x^j with respect to $\mathfrak{s}^A$ being $\{(\vec{0}, \vec{0}, \vec{0}, \vec{0})\} \times W_{x|_{\mathbb{F}_2^{r(n)}}}^{j-2}$, where $W_{x|_{\mathbb{F}_2^{r(n)}}}^{j-2}$ is the $(j-2)^{\text{th}}$-register subspace given the seed $x|_{\mathbb{F}_2^{r(n)}}$ with respect to the CLM $\mathfrak{s}^t$.

The case where Player $= B$ is similar, and we omit its description.

3. $\mathcal{S}'(\overline{n}, \text{Marginal}, \text{Player}, j, \cdot, z)$ runs as follows: First, it runs $\mathcal{S}'(\overline{n}, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ — which was defined in the first clause — to extract the type set $\mathcal{T}$, the edge set $\mathcal{E}$, and the dimension $r'(n) = 4|\mathcal{T}| + r(n)$. Then, it checks that z is a bit string of length $4|\mathcal{T}| + r(n)$ and that $1 \leq j \leq h + 2$, and returns an error sign otherwise.

Now, if Player $= A$ and $j = 1$, then $\mathcal{S}'$ outputs the first $2|\mathcal{T}|$ bits of z. If Player $= A$ and $j = 2$, then $\mathcal{S}'$ reads the first $2|\mathcal{T}|$ bits of z. If $(z_1, ..., z_{2|\mathcal{T}|}) = \text{enc}(t)$ for some $t \in \mathcal{T}$, then it zeroes out all coordinates in the third copy of $\mathbb{F}_2^{\mathcal{T}}$, and all coordinates but the tth one in the fourth copy of $\mathbb{F}_2^{\mathcal{T}}$; otherwise, it zeroes out the third and fourth copy of $\mathbb{F}_2^{\mathcal{T}}$ completely. If Player $= A$ and $j \geq 3$, then it does the first two steps as described above, resulting in a vector x in $(\mathbb{F}_2^{\mathcal{T}})^4$; if there is no $t \in \mathcal{T}$ such that $x = (\text{enc}(t), \vec{0}, \mathbf{1}_t)$, then it zeros out all coordinates of $\mathbb{F}_2^{r(n)}$, resulting in $(x, \vec{0})$. Otherwise, there is some $t \in \mathcal{T}$ such that $x = (\text{enc}(t), \vec{0}, \mathbf{1}_t)$, in which case it calls $\mathcal{S}(n, \text{Marginal}, t, j - 2, \cdot, z|_{\mathbb{F}_2^{r(n)}})$, whose output we denote by x', and it outputs $(x, x') \in (\mathbb{F}_2^{\mathcal{T}})^4 \times \mathbb{F}_2^{r(n)}$.

Again, the case Player $= B$ is similar, and we omit it.

4. $\mathcal{S}'(\overline{n}, \text{Evaluate}, \text{Player}, j, x, z)$ runs as follows: If $j = 1$, then it runs $\mathcal{S}'(\overline{n}, \text{Marginal}, \text{Player}, j, \cdot, z)$ as defined in the previous clause. If Player $= A$ and $j = 2$, then it checks whether the restriction of x to the first and second copies of $\mathbb{F}_2^{\mathcal{T}}$ agrees with $\text{enc}(t)$ for some $t \in \mathcal{T}$; if it does, then it outputs the vector $(\vec{0}, x') \in \mathbb{F}_2^{\mathcal{T}} \times \mathbb{F}_2^{\mathcal{T}}$, where x' is the vector whose all coordinates are zero except for the tth coordinate, which is the tth coordinate in the fourth copy of $\mathbb{F}_2^{\mathcal{T}}$ in z; otherwise, it outputs $\vec{0} \in \mathbb{F}_2^{\mathcal{T}} \times \mathbb{F}_2^{\mathcal{T}}$. If Player $= A$ and $j \geq 3$, it first checks whether the restriction of $x|_{\mathbb{F}_2^{4|\mathcal{T}|}} = (\text{enc}(t), \vec{0}, \mathbf{1}_t)$ for some $t \in \mathcal{T}$; if so, it outputs the same output as $\mathcal{S}(\overline{n}, \text{Evaluate}, t, j - 2, x|_{\mathbb{F}_2^{r(n)}}, z|_{\mathbb{F}_2^{r(n)}})$; otherwise, if $j = 3$, then it outputs $\vec{0} \in \mathbb{F}_2^{r(n)}$, and if $j > 3$ it outputs the empty string. For Player $= B$, it acts similarly with the first and second copies of $\mathbb{F}_2^{\mathcal{T}}$ swapping roles with the third and fourth copies.

5. There is a *canonical* way of extracting the perpendicular action out of the others. See Clause 6 on page 94 of [JNV$^+$21], which explains how step 3c in Figure 10 is implemented.

Before describing the rest of the TMs, we note that indeed the sampler satisfies the conditions of the proof:

For the first condition, which says that this transformed sampler is indeed the detyping transformation on the combinatorial level, we leave for the reader to compare this algorithm to the description in Definition 4.40.

For efficient runtime, note that all the operations are either calling the original sampler on the same index n namely $\mathcal{S}(\overline{n}, \cdot, \ldots, \cdot)$, or a previous subroutine which was already defined (again on the same index n), or is some polynomial time operation on $\mathbb{F}_2^{r(n)+4|\mathcal{T}|}$ which translates to $\mathrm{poly}(|\mathcal{T}|, r(n))$ number of operations. As both $|\mathcal{T}|$ and $r(n)$ are bounded by $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$ (Remark 4.31), the runtime bound is deduced.

To deduce that $\mathcal{S}'$ can be described in length that is polynomial in that of $\mathcal{S}$, note that the (constant length) natural language description we provided above can be translated to a constant length code in some programming language (or more precisely, a formal description of a TM according to the encoding fixed in Section 2.5.1). Therefore, the description length of $\mathcal{S}'$ is some constant, up to the appending of the description of $\mathcal{S}$ (for it to run the appropriate subroutines). As the exact effect of appending $\mathcal{S}$ to $\mathcal{S}'$ on the level of the description of $\mathcal{S}'$ depends on the specific choice of encodings of TMs, we use Item (3) of Fact 2.32, and deduce that the description of $\mathcal{S}'$ can be calculated from $\mathcal{S}$ in polynomial time and thus $|\mathcal{S}'| = \mathrm{poly}_h(|\mathcal{S}|)$.[63]

The Answer length calculator: Recall that the input to $\mathcal{A}'$ is expected to be $(\overline{n}, \mathrm{x}, \kappa)$, where $\mathrm{x} \in \mathbb{F}_2^{4|\mathcal{T}|+r(n)}$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$. On the other hand, $\mathcal{A}$ is expecting an input of the form $(\overline{n}, (t, \mathrm{y}), \kappa)$, where $t \in \mathcal{T}$ and $\mathrm{y} \in \mathbb{F}_2^{r(n)}$.

So, $\mathcal{A}'(\overline{n}, \mathrm{x}, \kappa)$ runs as follows: It calls $\mathcal{S}$ to extract $\mathcal{T}$ and $r(n)$, and then if the conditions on the inputs are not satisfied, it outputs an $\mathfrak{error}$ sign. Otherwise, it checks whether there is a $t \in \mathcal{T}$ such that $\mathrm{x}|_{\mathbb{F}_2^{4|\mathcal{T}|}} = (\mathrm{enc}(t), \vec{0}, \mathbf{1}_t)$ or $\mathrm{x}|_{\mathbb{F}_2^{4|\mathcal{T}|}} = (\vec{0}, \mathbf{1}_t, \mathrm{enc}(t))$. If not, it outputs the empty string (which is the unary representation of 0). If there is such a t, it provides the output of $\mathcal{A}(\overline{n}, (t, \mathrm{x}|_{\mathbb{F}_2^{r(n)}}), \kappa)$.

Again, it is straightforward to check that this induces a length function which is compatible with the description of $\mathfrak{DeType}(\mathfrak{G})$ in Definition 4.40. For description length, again the above description is constant up to fixing the appropriate inputs to DeType_h. Lastly, $\mathcal{A}'$ calls $\mathcal{S}$ and $\mathcal{A}$, and does some polynomial time manipulations on vectors in $\mathbb{F}_2^{4|\mathcal{T}|+r(n)}$. Hence, it runs in time polynomial in the above.

The Linear constraints processor: $\mathcal{L}'(\overline{n}, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ runs as follows. First, it calls $\mathcal{S}$ to extract $\mathcal{T}$ and $r(n)$. Then, it checks that $\mathrm{x}, \mathrm{y} \in \mathbb{F}_2^{4|\mathcal{T}|+r(n)}$, and if not outputs the single constraint $\{\mathsf{J}\}$. Otherwise, it calls $\mathcal{A}'(\overline{n}, \mathrm{x}, \mathfrak{R})$ and $\mathcal{A}'(\overline{n}, \mathrm{y}, \mathfrak{R})$ and compares the lengths of $a^{\mathfrak{R}}$ and $b^{\mathfrak{R}}$ to their decoded outputs respectively. If they do not match, it outputs the single constraint $\{\mathsf{J}\}$. Finally, if the input passed all the above checks, then $\mathcal{L}'$ checks the following: If there are no $t, t' \in \mathcal{T}$ such that $\mathrm{x} = (\mathrm{enc}(t), \vec{0}, \mathbf{1}_t)$ and $\mathrm{y} = (\vec{0}, \mathbf{1}_{t'}, \mathrm{enc}(t'))$, then it outputs the empty string (interpreted as no constraints, which is immediate acceptance). Otherwise, it returns the output of $\mathcal{L}(\overline{n}, (t, \mathrm{x}|_{\mathbb{F}_2^{r(n)}}), (t', \mathrm{y}|_{\mathbb{F}_2^{r(n)}}), a^{\mathfrak{R}}, b^{\mathfrak{R}})$.

As for the sampler and answer length calculator, checking that this $\mathcal{L}'$ satisfies what we need is straightforward, and we leave it to the reader. This finishes the proof. □

4.5.4 Padding

The idea of padding is enlarging the length of answers artificially in a way that *essentially* does not change the game. There are many ways of doing that, and the following is a simple version which is *restriction free* — namely, we enlarge the answer length and do not require the appended bits to satisfy any requirements.

Definition 4.47 (Restriction free combinatorial padding). Let $\mathfrak{G}$ be a tailored game, and Λ a positive integer. The tailored game
$$\widetilde{\mathfrak{G}} = \mathfrak{Padding}(\mathfrak{G}, \Lambda)$$
has the same underlying graph as $\mathfrak{G}$, and the same edge sampling distribution μ. For every vertex $\mathrm{x} \in V$, both the readable length and unreadable length of x are defined to be Λ, namely $\widetilde{\ell^{\mathfrak{R}}}(\mathrm{x}) = \widetilde{\ell^{\mathfrak{L}}}(\mathrm{x}) = \Lambda$. For the controlled linear constraints function $\widetilde{L_{\mathrm{xy}}}$:

---

[63]Actually, the description is fixed up to appending $\mathcal{S}$ and h, which means the dependence is $\mathrm{poly}(\log h, |\mathcal{S}|)$. This is better than $\mathrm{poly}_h(|\mathcal{S}|)$, but we do not need this better bound.

- If the readable or unreadable length of either x or y are larger than Λ, namely

$$\Lambda < \max\{\ell^{\mathfrak{R}}(x), \ell^{\mathfrak{R}}(y), \ell^{\mathfrak{L}}(x), \ell^{\mathfrak{L}}(y)\},$$

then $\widetilde{L_{xy}}$ outputs no constraints regardless of what $\gamma^{\mathfrak{R}}$ is (which translates to automatic acceptance).

- Otherwise, for every $z \in V$ we let $\widetilde{S_z^{\mathfrak{R}}} = S_z^{\mathfrak{R}} \sqcup T_z^{\mathfrak{R}}$, where $S_z^{\mathfrak{R}}$ is the original formal set of generators at z and $T_z^{\mathfrak{R}}$ is a set of $\Lambda - \ell^{\mathfrak{R}}(z)$ many additional variables (and similarly for $\widetilde{S_z^{\mathfrak{L}}}$). Namely,

$$\widetilde{S_{\cdot}} = S_{\cdot} \sqcup T_{\cdot}$$

where the subscripts can be x, y or xy (which indicates union of x and y variables) and the superscripts can be $\mathfrak{R}, \mathfrak{L}$ or none (which indicates the union of readable and unreadable variables). Recall that for $\gamma^{\mathfrak{R}} \colon S_{xy}^{\mathfrak{R}} \to \mathbb{F}_2$, the output of $L_{xy}(\gamma^{\mathfrak{R}})$ is a collection of indicators on the set $S_{xy} \sqcup \{J\}$, representing linear constraints that should be checked on $\gamma \colon S_{xy} \to \mathbb{F}_2$. So, for $\widetilde{\gamma^{\mathfrak{R}}} \colon \widetilde{S_{xy}^{\mathfrak{R}}} \to \mathbb{F}_2$, letting $\gamma^{\mathfrak{R}} = \widetilde{\gamma^{\mathfrak{R}}}|_{S_{xy}^{\mathfrak{R}}}$, we can define $\widetilde{L_{xy}}(\widetilde{\gamma^{\mathfrak{R}}})$ to be the extension by zeros of the outputs of $L_{xy}(\gamma^{\mathfrak{R}})$ to the T_{xy} variables. Namely, for every $c \colon S_{xy} \sqcup \{J\} \to \mathbb{F}_2$ in the output of $L_{xy}(\gamma^{\mathfrak{R}})$, we let $\tilde{c} \colon \widetilde{S_{xy}} \sqcup \{J\} \to \mathbb{F}_2$ be defined by

$$\tilde{c}(X) = \begin{cases} c(X) & X \in S_{xy} \sqcup \{J\}, \\ 0 & X \in T_{xy}. \end{cases}$$

On the combinatorial level, the padded game $\widetilde{\mathfrak{G}}$ samples an edge from the original graph (according to the same distribution), and as long as Λ is large enough, it disregards the added variables T and plays the original game $\mathfrak{G}$ only according to the assignments to S.

The following is a straightforward fact to check.

Fact 4.48. *Assume* $\Lambda \geq \max(\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}})$. *Then,*

- *(Completeness) If* $\mathfrak{G}$ *has a perfect* ZPC *strategy, then so does* $\mathfrak{Padding}(\mathfrak{G}, \Lambda)$.

- *(Soundness and entanglement) If* $\mathfrak{Padding}(\mathfrak{G}, \Lambda)$ *has a value* $1 - \varepsilon$ *strategy, then so does* $\mathfrak{G}$, *and furthermore*

$$\mathscr{E}(\mathfrak{Padding}(\mathfrak{G}, \Lambda), 1 - \varepsilon) = \mathscr{E}(\mathfrak{G}, 1 - \varepsilon).$$

Claim 4.49. *There is a polynomial-time TM* Padding *that takes as input a tailored h-level normal form verifier* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *and a 1-input TM* Λ, *and outputs a new tailored h-level normal form verifier* Padding$(\mathcal{V}, \Lambda) = \mathcal{V}' = (\mathcal{S}, \mathcal{A}^{\Lambda}, \mathcal{L}', \mathcal{D})$ *satisfying:*

- *(Combinatorial Padding) For every* $n \in \mathbb{N}$, *if* $\mathcal{V}_n$ *is well defined (Definition 4.33), then* $\mathcal{V}'_n$ *is well defined, and* $\mathcal{V}'_n = \mathfrak{Padding}(\mathcal{V}_n, |\Lambda(n)|)$, *where* $|\cdot|$ *is the length of words function.*[64]

- *(Sampler properties) The output sampler is the same as the original one, and thus its running time and description lengths stay the same.*

- *(Answer length properties) The output answer length TM* $\mathcal{A}^{\Lambda}$ *depends only on* Λ. *Furthermore,* $\mathcal{A}^{\Lambda}(n, \cdot, \cdot)$ *runs in time which is linear in* $\mathbb{T}(\Lambda; n)$. *Finally, the description length of* $\mathcal{A}^{\Lambda}$ *is linear in that of* Λ.

- *(Linear constraints processor properties) The output linear constraints processor* $\mathcal{L}'$ *runs in time which is polynomial in:*

---

[64]This is the same as thinking of the output of $\Lambda(n)$ as representing a natural number in unary.

- *the running time $\mathbb{T}(\Lambda; \overline{n})$;*
- *the running time $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot, \cdot)$;*
- *the running time $\mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot)$;*
- *the running time $\mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$.*

Moreover, the description length of $\mathcal{L}'$ is linear in that of $\Lambda, \mathcal{S}, \mathcal{A}$ and $\mathcal{L}$.

Proof.

The Sampler: We keep $\mathcal{S}$ as the sampler. So, running time and description length stay the same.

The answer length calculator: For every $n \in \mathbb{N}$, $\mathcal{A}^\Lambda(\overline{n}, x, \kappa) = \text{enc}(\Lambda(n))$ (Definition 2.34), regardless of x or κ. Hence, $|\text{dec}(\mathcal{A}^\Lambda(\overline{n}, x, \kappa))| = |\Lambda(n)|$, as is needed for $\mathcal{V}'_n$ to be equal to $\mathfrak{Padding}(\mathcal{V}_n, |\Lambda(n)|)$. It is immediate that the running time and description length are linear in Λ's.

The linear constraints processor: $\mathcal{L}'(\overline{n}, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ runs as follows. First it calls $\mathcal{S}(\overline{n}, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ to retrieve $r(n)$ and checks that $x, y \in \mathbb{F}_2^{r(n)}$; if not, it outputs $\{J\}$[65] (which is instant rejection); if they do satisfy this condition, then it checks whether $|a^{\mathfrak{R}}| = |b^{\mathfrak{R}}| = |\Lambda(n)|$; if not, it outputs $\{J\}$; otherwise, the inputs are well structured and $\mathcal{L}'$ can proceed.

Now, $\mathcal{L}'$ calls

$$\ell_a^{\mathfrak{R}} = |\text{dec}(\mathcal{A}(\overline{n}, x, \mathfrak{R}))|, \; \ell_b^{\mathfrak{R}} = |\text{dec}(\mathcal{A}(\overline{n}, y, \mathfrak{R}))|, \; \ell_a^{\mathfrak{L}} = |\text{dec}(\mathcal{A}(\overline{n}, x, \mathfrak{L}))| \quad \text{and} \quad \ell_b^{\mathfrak{L}} = |\text{dec}(\mathcal{A}(\overline{n}, y, \mathfrak{L}))|.$$

If $|\Lambda(n)|$ is strictly smaller than either of the lengths of these outputs, then $\mathcal{L}'$ outputs the empty string (i.e., no constraints). Otherwise, let $a_0^{\mathfrak{R}}$ be the restriction of $a^{\mathfrak{R}}$ to its first $|\ell_a^{\mathfrak{R}}|$ bits and $b_0^{\mathfrak{R}}$ be the restriction of $b^{\mathfrak{R}}$ to its first $|\ell_b^{\mathfrak{R}}|$ bits. Then, $\mathcal{L}'$ calls $\mathcal{L}(\overline{n}, x, y, a_0^{\mathfrak{R}}, b_0^{\mathfrak{R}})$ and gets as output a bit string. If this bit string is not (the encoding) of bit strings $(c^1, ..., c^k)$, where every c^i is of length $|\ell_a^{\mathfrak{R}}| + |\ell_b^{\mathfrak{R}}| + |\ell_a^{\mathfrak{L}}| + |\ell_b^{\mathfrak{L}}| + 1$, then it outputs $\{J\}$. Otherwise, it does the following operation on each c^i: First, it splits it to 5 bit strings $c_{a,\mathfrak{R}}^i, c_{b,\mathfrak{R}}^i, c_{a,\mathfrak{L}}^i, c_{b,\mathfrak{L}}^i, c_J^i$ of lengths $\ell_a^{\mathfrak{R}}, \ell_b^{\mathfrak{R}}, \ell_a^{\mathfrak{L}}, \ell_b^{\mathfrak{L}}$ and 1 respectively. Then, it appends each of $c_{a,\mathfrak{R}}^i, c_{b,\mathfrak{R}}^i, c_{a,\mathfrak{L}}^i, c_{b,\mathfrak{L}}^i$ with zeros until they are of length $|\Lambda(n)|$ — we denote the resulting strings by $\tilde{c}_{\cdot,\cdot}^i$. Finally, the bit string $\tilde{c}^i$ of length $4|\Lambda(n)| + 1$ is defined to be the concatenation of $\tilde{c}_{a,\mathfrak{R}}^i, \tilde{c}_{b,\mathfrak{R}}^i, \tilde{c}_{a,\mathfrak{L}}^i, \tilde{c}_{b,\mathfrak{L}}^i$ and c_J^i. After this operation was done for each string c^i, resulting with new strings $\tilde{c}^i$, $\mathcal{L}'$ outputs (the encoding of) $(\tilde{c}^1, ..., \tilde{c}^k)$.

The description is again just the above finite one, with the inputs $\mathcal{V}$ and Λ fixed. Hence, by Item (3) in Fact 2.32, the description can be calculated in polynomial time from them and is thus of polynomial length. For running time, $\mathcal{L}'$ either calls $\mathcal{A}, \mathcal{S}$ or $\mathcal{L}$, or is applying polynomial time operations on bit strings of length at most $O(|\Lambda(n)|)$ — where $|\Lambda(n)|$ is a quantity smaller than the running time of $\Lambda(n)$ — or bit strings of length at most $r(n)$. Recall that $r(n)$ is bounded by $\mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)$ by Remark 4.31, which explains the time bounds of $\mathcal{L}'$. $\qquad\square$

4.6 Proving the main theorem of Question Reduction: Theorem 4.36

Let $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ be a tailored h-level normal form verifier, and λ a positive integer. The goal is to describe the question reduced verifier $\text{QuestionReduction}_h(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}_{\text{QR}}^\lambda, \mathcal{A}_{\text{QR}}^\lambda, \mathcal{L}', \mathcal{D})$ which proves the theorem. The idea is, under the assumption that $\mathcal{V}$ is λ-bounded, to first choose for every n: an appropriate integer k which will be larger than the dimension of the CLMs used in $\mathcal{V}_{2^n}$, namely larger than $\mathcal{S}(2^n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$; an appropriate Λ which will be larger than the lengths used in $\mathcal{V}_{2^n}$, namely larger than every possible $\mathcal{A}(2^n, \cdot, \cdot)$; an appropriate $\mathscr{B} \subseteq \mathbb{F}_2^k$ that would have size a power of 2, and induce a good error correcting code with some predetermined parameters. After these choices are made, the goal of $\mathcal{V}'$ is for its nth game to be, combinatorially,

$$\mathcal{V}'_n = \mathfrak{DeType}(\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, \Lambda), k, \mathscr{B})).$$

---
[65]By outputting $\{J\}$, we mean the encoding (as in Definition 2.43) of the bit string of length $4|\Lambda(n)| + 1$ where all of its bits are zero except the last one which is 1.

We already described how to detype and to pad on the level of verifiers in Claims 4.46 and 4.49. So, we are left to describe a Turing machine that assumes the input is already padded, and outputs a **typed** normal form verifier that implements combinatorial question reduction. In the next claim, we let $\mathscr{B}$ be a (fixed) TM that takes as input a bit string x of length m and outputs a list of $2 \cdot 2^m$ vectors in $\mathbb{F}_2^{2^m}$ that induces an encoding matrix whose associated code has normalized distance δ for some universal constant $\delta > 0$, and such that $\mathbb{T}(\mathscr{B}; \mathrm{x}) = 2^{O(|\mathrm{x}|)}$ (the existence of such a TM $\mathscr{B}$ and such a universal constant δ is guaranteed by Fact 3.72).

Claim 4.50. *There is a polynomial time TM* TypedQuestionReduction$_h$ *that takes two inputs — an h-level normal form verifier* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$; *a 1-input TM* $\mathcal{K}$ — *and outputs a typed 1-level normal form verifier*

$$\mathsf{TypedQuestionReduction}_h(\mathcal{V}, \mathcal{K}) = \widetilde{\mathcal{V}} = (\mathcal{S}^{\mathcal{K}}, \widetilde{\mathcal{A}}, \widetilde{\mathcal{L}}, \mathcal{D})$$

such that

- *(Combinatorial Question Reduction) For every $n \in \mathbb{N}$, if $\mathcal{V}_{2^n}$ is well defined (Definition 4.33), and there is a function $\Delta \colon \mathbb{N} \to \mathbb{N}$ such that $|\mathrm{dec}(\mathcal{A}(2^n, \mathrm{x}, \kappa))| = \Delta(n)$ regardless of x and κ, and $2^{|\mathcal{K}(n)|}$ is larger than*

$$\mathcal{S}(2^n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot) \,,$$

  then $\widetilde{\mathcal{V}}_n$ is well defined (Definition 4.45), and

$$\widetilde{\mathcal{V}}_n = \mathfrak{QueRed}(\mathcal{V}_{2^n}, 2^{|\mathcal{K}(n)|}, \mathscr{B}(\mathcal{K}(n))) \,.$$

- *(Sampler properties) The sampler $\mathcal{S}^{\mathcal{K}}$ depends only on $\mathcal{K}$ (and h), but not on $\mathcal{V}$. Furthermore, it runs in time which is polynomial in that of $\mathcal{K}$.[66] Finally, its description can be calculated from that of $\mathcal{K}$ in polynomial time, which means in particular $|\mathcal{S}^{\mathcal{K}}| = \mathrm{poly}_h(|\mathcal{K}|)$.*

- *(Answer length calculator properties) The TM $\widetilde{\mathcal{A}}$ depends only on $\mathcal{K}$ and on $\mathcal{A}$, and not on $\mathcal{S}$ or $\mathcal{L}$. In addition, it runs in time*

$$\mathbb{T}(\widetilde{\mathcal{A}}; n, \cdot, \cdot) = \mathrm{poly}_h(2^{|\mathcal{K}(n)|}, \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot)) \,.$$

  Finally, its description can be calculated in polynomial time from the relevant inputs, and in particular $|\widetilde{\mathcal{A}}| \le \mathrm{poly}_h(|\mathcal{K}|, |\mathcal{A}|)$.

- *(Linear constraints processor properties) $\widetilde{\mathcal{L}}$ runs in time which is polynomial in:*
    - *the integer $2^{|\mathcal{K}(n)|}$;*
    - *the running time $\mathbb{T}(\mathcal{K}; n)$;*
    - *the running time $\mathbb{T}(\mathcal{S}; 2^n, \cdot, \cdot, \cdot, \cdot, \cdot)$;*
    - *the running time $\mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot)$;*
    - *the running time $\mathbb{T}(\mathcal{L}; 2^n, \cdot, \cdot, \cdot, \cdot, \cdot)$;*

  Furthermore, its description can be calculated from $\mathcal{K}, \Lambda, \mathcal{S}$ and $\mathcal{L}$ in polynomial time, and in particular $|\widetilde{\mathcal{L}}| \le \mathrm{poly}_h(|\mathcal{K}|, |\Lambda|, |\mathcal{S}|, |\mathcal{L}|)$.

Proof.

The typed CL sampler: Recall Example 4.39, and specifically that we aim to define a typed 1-level sampler. This means that, in particular, in the expected input $(n, \mathrm{Action}, \mathrm{Type}, j, \mathrm{x}, z)$, we can assume that $j = 1$ always.

---

[66]Here, the constants depend on h, namely this is $\mathrm{poly}_h(\mathbb{T}(\mathcal{K}; n))$.

1. $\mathcal{S}^{\mathcal{K}}(\cdot, \text{Graph}, \cdot, \cdot, \cdot, \cdot)$ outputs the list

$$\forall 1 \leq j \leq h : \quad \texttt{Hide}_A^j \,, \texttt{Hide}_B^j \,,$$
$$\texttt{Intro}_A \,, \texttt{Intro}_B \,, \texttt{Read}_A \,, \texttt{Read}_B \,, \texttt{Sample}_A \,, \texttt{Sample}_B \,,$$
$$\texttt{Pauli}_{\mathbb{Z}} \,, \texttt{Pauli}_{\mathbb{X}} \,, \texttt{X} \,, \texttt{Z} \,, \texttt{First} \,, \texttt{Second} \,, \texttt{Both} \,,$$
$$\forall 1 \leq i \leq 3 \,, 1 \leq j \leq 3 : \quad \texttt{var}_{ij} \,, \texttt{row}_i \,, \texttt{col}_j \,,$$

   followed by the adjacency matrix of the graph depicted in Figure 14.

2. $\mathcal{S}^{\mathcal{K}}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ calls $\mathcal{K}(n)$ and outputs $2|\mathcal{K}(n)| + 2 = 2 \log |\mathscr{B}(\mathcal{K}(n))|$.

3. $\mathcal{S}^{\mathcal{K}}(n, \text{Register}, \text{Type}, 1, \texttt{x}, \cdot)$ outputs a bit string consisiting of only 1's of length $2|\mathcal{K}(n)| + 2$, which indicates that the whole space is the first (and only) register subspace of $\mathfrak{s}^{\text{Type}}$ — note that this requires it to first call $\mathcal{K}(n)$ as a subroutine.

4. $\mathcal{S}^{\mathcal{K}}(n, \text{Marginal}, \text{Type}, 1, \cdot, z)$ runs as follows:

   - If Type is one of

     $$\texttt{Hide}_A^j \,, \texttt{Hide}_B^j \,, \texttt{Intro}_A \,, \texttt{Intro}_B \,, \texttt{Read}_A \,, \texttt{Read}_B \,, \texttt{Sample}_A \,, \texttt{Sample}_B \,, \texttt{Pauli}_{\mathbb{Z}} \,, \texttt{Pauli}_{\mathbb{X}} \,,$$

     then it zeroes out z and outputs $\vec{0}$ (which is a concatenation of $2|\mathcal{K}(n)| + 2$ zeros in this case).
   - If Type is one of
     $$\texttt{row}_i \,, \texttt{col}_j \,, \texttt{var}_{ij} \,, \texttt{First} \,, \texttt{Second} \,, \texttt{Both} \,,$$
     then it acts as the identity on z, namely outputs z.
   - If Type is either X or Z, let us split z into z_1, z_2, where z_1 is the first $|\mathcal{K}(n)| + 1$ bits of z and z_2 are its last $|\mathcal{K}(n)| + 1$ bits. Then, given Type $=$ X, it outputs $(z_1, \vec{0})$, and given Type $=$ Z, it outputs $(\vec{0}, z_2)$ (here $\vec{0}$ is a concatenation of $|\mathcal{K}(n)| + 1$ zeros).

5. As, again, we can assume $j = 1$, $\mathcal{S}^{\mathcal{K}}(n, \text{Evaluate}, \text{Type}, 1, \texttt{x}, z)$ runs exactly the same as $\mathcal{S}^{\mathcal{K}}(n, \text{Marginal}, \text{Type}, 1, \cdot, z)$.

6. There is a *canonical* way of extracting the perpendicular action out of the others. See Clause 6 on page 94 of [JNV$^+$21], which explains how step 3c in Figure 10 is implemented. Albeit, in this case it is straightforward what the perpendicular maps are. $\mathcal{S}^{\mathcal{K}}(n, \text{Perpendicular}, \text{Type}, 1, \texttt{x}, z)$ runs as follows:

   - If Type is one of

     $$\texttt{Hide}_A^j \,, \texttt{Hide}_B^j \,, \texttt{Intro}_A \,, \texttt{Intro}_B \,, \texttt{Read}_A \,, \texttt{Read}_B \,, \texttt{Sample}_A \,, \texttt{Sample}_B \,, \texttt{Pauli}_{\mathbb{Z}} \,, \texttt{Pauli}_{\mathbb{X}} \,,$$

     then $(\mathfrak{s}^{\text{Type}})^{\perp}$ should act as the identity, and $\mathcal{S}^{\mathcal{K}}$ outputs z.
   - If Type is one of
     $$\texttt{row}_i \,, \texttt{col}_j \,, \texttt{var}_{ij} \,, \texttt{First} \,, \texttt{Second} \,, \texttt{Both} \,,$$
     then $(\mathfrak{s}^{\text{Type}})^{\perp}$ should act as the zero map, and $\mathcal{S}^{\mathcal{K}}$ outputs $\vec{0}$.
   - Finally, $(\mathfrak{s}^{\texttt{X}})^{\perp}(z) = (\vec{0}, z_2)$ and $(\mathfrak{s}^{\texttt{Z}})^{\perp}(z) = (z_1, \vec{0})$, which means $\mathcal{S}^{\mathcal{K}}$ outputs $(\vec{0}, z_2)$ in case Type $=$ X and $(z_1, \vec{0})$ in case Type $=$ Z.

We verify the required properties of the typed sampler $\mathcal{S}^{\mathcal{K}}$. Note that the above description is constant, and the only thing that actually needs to be appended is the description length of $\mathcal{K}$. By Item (3) of Fact 2.32, this shows that the description of $\mathcal{S}^{\mathcal{K}}$ can be calculated from that of $\mathcal{K}$ in polynomial time, which in particular implies the description length bound. For runtime, note that all the operations done by $\mathcal{S}^{\mathcal{K}}$ are either writing down the type graph (which takes $\mathrm{poly}(h)$-time), calls to $\mathcal{K}(n)$, or manipulations of vectors in $\mathbb{F}_2^{2|\mathcal{K}(n)|+2}$ — that take time at most $\mathrm{poly}(|\mathcal{K}(n)|)$, which is polynomial in the running time of $\mathcal{K}$. All in all, the running time is polynomial in that of h and $\mathcal{K}$.

The Answer length calculator: Recall that the readable and unreadable lengths of a vertex in $\mathfrak{QueReD}$ (Section 4.4) depend only on its type. Hence, $\widetilde{\mathcal{A}}(n, (t, \mathrm{x}), \kappa)$ runs as follows: First, it calls $\mathcal{K}(n)$ and $\mathcal{S}^{\mathcal{K}}(\cdot, \mathrm{Graph}, \cdot, \cdot, \cdot, \cdot)$ to retrieve the type set $\mathcal{T}$ underlying the typed 1-level CL sampling scheme. If $t \notin \mathcal{T}$ or $\mathrm{x} \notin \mathbb{F}_2^{2|\mathcal{K}(n)|+2}$ or $\kappa \notin \{\mathfrak{R}, \mathfrak{L}\}$, then $\widetilde{\mathcal{A}}$ outputs an $\mathfrak{error}$ sign. Otherwise, it lets $\Delta(n) = |\mathrm{dec}(\mathcal{A}(2^n, 0, \mathfrak{R}))|$,[67] and follows the table:

Type t	Decoded output if $\kappa = \mathfrak{R}$	Decoded output if $\kappa = \mathfrak{L}$
$\mathtt{Hide}^j$	$2^{|\mathcal{K}(n)|}$ ones	$2 \cdot 2^{|\mathcal{K}(n)|}$ ones
$\mathtt{Intro.}$	$2^{|\mathcal{K}(n)|} + \Delta(n)$ ones	$\Delta(n)$ ones
$\mathtt{Read.}$	$2^{|\mathcal{K}(n)|} + \Delta(n)$ ones	$2^{|\mathcal{K}(n)|} + \Delta(n)$ ones
$\mathtt{Sample.}$	$2^{|\mathcal{K}(n)|} + \Delta(n)$ ones	$\Delta(n)$ ones
$\mathtt{Pauli.}$	empty string	$2^{|\mathcal{K}(n)|}$ ones
$\mathtt{X}$	empty string	single one
$\mathtt{Z}$	empty string	single one
$\mathtt{First}$	empty string	single one
$\mathtt{Second}$	empty string	single one
$\mathtt{Both}$	empty string	two ones
$\mathtt{var}_{ij}$	empty string	single one
$\mathtt{row}_i$	empty string	three ones
$\mathtt{col}_j$	empty string	three ones

We verify the required properties of $\widetilde{\mathcal{A}}$. The above description is constant, up to appending the descriptions of $\mathcal{K}$ and $\mathcal{A}$. For running time, note that:

- it calls $\mathcal{S}^{\mathcal{K}}(\cdot, \mathrm{Graph}, \cdot, \cdot, \cdot, \cdot)$ which takes $\mathrm{poly}(h)$ time;

- it calls $\mathcal{K}(n)$ which takes $\mathbb{T}(\mathcal{K}; n)$ time;

- it verifies certain properties on bit strings of length $O(|\mathcal{K}(n)|)$, which takes $\mathrm{poly}(|\mathcal{K}(n)|)$ time;

- it calls $\mathcal{A}(2^n, \cdot, \cdot)$ which takes $\mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot)$ time, and its output is of length $\Delta(n)$ which by definition is smaller or equal to $\mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot)$;

- it outputs bit strings of length $O(2^{|\mathcal{K}(n)|} + \Delta(n))$, which takes $\mathrm{poly}(2^{|\mathcal{K}(n)|}, \mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot))$ time.

All in all, it runs in time which is $\mathrm{poly}(2^{|\mathcal{K}(n)|}, \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{A}; 2^n, \cdot, \cdot))$, as claimed.

The Linear constraints processor: $\widetilde{\mathcal{L}}(n, (t, \mathrm{y}), (t', \mathrm{y}'), a^{\mathfrak{R}}, b^{\mathfrak{R}})$ runs as follows. First it calls $\mathcal{K}(n)$, and $\mathcal{S}^{\mathcal{K}}(\cdot, \mathrm{Graph}, \cdot, \cdot, \cdot, \cdot)$ — to get the type graph $(\mathcal{T}, \mathcal{E})$. If $tt' \notin \mathcal{E}$ or $\mathrm{y}, \mathrm{y}' \notin \mathbb{F}_2^{2|\mathcal{K}(n)|+2}$, then it outputs $\mathfrak{error}$ (note that in this case, the canonical decider will reject as this sign is not a proper encoding of a sequence of bit strings). Then, it checks that $|a^{\mathfrak{R}}| = |\mathrm{dec}(\widetilde{\mathcal{A}}(n, (t, \mathrm{y}), \mathfrak{R}))|$ and that $|b^{\mathfrak{R}}| = |\mathrm{dec}(\widetilde{\mathcal{A}}(n, (t', \mathrm{y}'), \mathfrak{R}))|$, and outputs $\mathfrak{error}$ otherwise. Given that the input was well structured, it runs $\mathscr{B}(\mathcal{K}(n))$, which outputs a sequence $\mathscr{B}$ of $2^{|\mathcal{K}(n)|+1}$-many vectors in $\mathbb{F}_2^{2|\mathcal{K}(n)|}$ — this can be thought

---

[67]Here, $\widetilde{\mathcal{A}}$ will work as expected only if $\mathcal{A}$ is indeed padded and disregards its second and third inputs altogether.

of as a matrix over $\mathbb{F}_2$ with $2^{|\mathcal{K}(n)|}$ columns and $2^{|\mathcal{K}(n)|+1}$ rows — and thus the elements of $\mathscr{B}$ (which are the rows of the afore-mentioned matrix) can be parameterized by vectors in $\mathbb{F}_2^{|\mathcal{K}(n)|+1}$. Finally, it recovers the value $\Delta(n) = |\mathrm{dec}(\mathcal{A}(2^n, 0, \mathfrak{R}))|$. Then, $\widetilde{\mathcal{L}}$ acts as follows:[68]

1. **Pauli Basis Test — Consistency checks of X-variables**:

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{Pauli}_{\mathsf{X}}, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathsf{X}, u, \vec{0})$.
     Operation: The bit string $u \in \mathbb{F}_2^{|\mathcal{K}(n)|+1}$ is the index of some vector $w^u \in \mathscr{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(w^u, 1, 0) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}+2}$.
     Interpretation: In this case $S^{\mathcal{L}}_{\mathtt{Pauli}_{\mathsf{X}}} = \{\mathsf{PX}^i\}_{i=1}^{2^{|\mathcal{K}(n)|}}$ and $S^{\mathcal{L}}_{\mathsf{X}^u} = \{\mathsf{X}^u\}$, and the above bit string encodes the linear constraint
     $$\left(\sum_{i=1}^{2^{|\mathcal{K}(n)|}} w_i^u \gamma(\mathsf{PX}^i) \right) + \gamma(\mathsf{X}^u) = 0 \,.$$

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{var}_{11}, u, v)$, $(t', \mathrm{y}') = (\mathsf{X}, u, \vec{0})$.
     Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(1, 1, 0) \in \mathbb{F}_2^3$.
     Interpretation: In this case $S^{\mathcal{L}}_{\mathtt{var}_{11}^{u,v}} = \{\mathsf{Var}_{11}^{u,v}\}$ and $S^{\mathcal{L}}_{\mathsf{X}^u} = \{\mathsf{X}^u\}$, and the above bit string encodes the linear constraint
     $$\gamma(\mathsf{Var}_{11}^{u,v}) + \gamma(\mathsf{X}^u) = 0 \,.$$

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{First}, u, v)$, $(t', \mathrm{y}') = (\mathsf{X}, u, \vec{0})$.
     Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(1, 1, 0) \in \mathbb{F}_2^3$.
     Interpretation: In this case $S^{\mathcal{L}}_{\mathtt{First}^{u,v}} = \{\mathsf{First}^{u,v}\}$ and $S^{\mathcal{L}}_{\mathsf{X}^u} = \{\mathsf{X}^u\}$, and the above bit string encodes the linear constraint
     $$\gamma(\mathsf{First}^{u,v}) + \gamma(\mathsf{X}^u) = 0.$$

2. **Pauli Basis Test — Consistency checks of Z-variables:**
   This is similar to the previous case, with the obvious modifications. For completeness, we give the details:

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{Pauli}_{\mathsf{Z}}, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathsf{Z}, \vec{0}, v)$.
     Operation: The bit string $v \in \mathbb{F}_2^{|\mathcal{K}(n)|+1}$ is the parameter of some vector $w^v \in \mathscr{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(w^v, 1, 0) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}+2}$.
     Interpretation: In this case $S^{\mathcal{L}}_{\mathtt{Pauli}_{\mathsf{Z}}} = \{\mathsf{PZ}^i\}_{i=1}^{2^{|\mathcal{K}(n)|}}$ and $S^{\mathcal{L}}_{\mathsf{Z}^v} = \{\mathsf{Z}^v\}$, and the above bit string encodes the linear constraint
     $$\left(\sum_{i=1}^{2^{|\mathcal{K}(n)|}} w_i^v \gamma(\mathsf{PZ}^i) \right) + \gamma(\mathsf{Z}^v) = 0 \,.$$

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{var}_{22}, u, v)$, $(t', \mathrm{y}') = (\mathsf{Z}, \vec{0}, v)$.
     Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(1, 1, 0) \in \mathbb{F}_2^3$.
     Interpretation: In this case $S^{\mathcal{L}}_{\mathtt{var}_{22}^{u,v}} = \{\mathsf{Var}_{22}^{u,v}\}$ and $S^{\mathcal{L}}_{\mathsf{Z}^v} = \{\mathsf{Z}^v\}$, and the above bit string encodes the linear constraint
     $$\gamma(\mathsf{Var}_{22}^{u,v}) + \gamma(\mathsf{Z}^v) = 0 \,.$$

---

[68]The format is the following: Each enumerated clause is some sub graph of the typed graph of $\mathfrak{QueRed}$, which should help navigate the checks more easily. The actual operation of $\widetilde{\mathcal{L}}$ is to check what is the relevant question format, and acting according to the appropriate bullet. We first go over edges from $\mathfrak{Pauli\ Basis}_{2^{|\mathcal{K}(n)|}}(\mathscr{B})$ which was described in Section 3.8.3, then the single edge from $\mathfrak{Intro}(\mathcal{V}_{2^n})$ which was described in Section 4.1, and finally the augmented edges of $\mathfrak{QueRed}(\mathcal{V}_{2^{|\mathcal{K}(n)|}})$ described in Section 4.4.

- *Question format*: $(t, \mathbf{y}) = (\texttt{Second}, u, v)$, $(t', \mathbf{y}') = (\texttt{Z}, \vec{0}, v)$.

  Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding) of the single bit string $(1, 1, 0) \in \mathbb{F}_2^3$.

  Interpretation: In this case $S_{\texttt{Second}^{u,v}}^{\mathfrak{L}} = \{\texttt{Second}^{u,v}\}$ and $S_{\texttt{Z}^v}^{\mathfrak{L}} = \{\texttt{Z}^v\}$, and the above bit string encodes the linear constraint

  $$\gamma(\texttt{Second}^{u,v}) + \gamma(\texttt{Z}^v) = 0 \ .$$

3. **Pauli Basis Test — (null-)Commutation game** (Section 3.8.1):

   - *Question format*: $(t, \mathbf{y}) = (\texttt{First}, u, v)$, $(t', \mathbf{y}') = (\texttt{Both}, u, v)$.

     Operation: $\widetilde{\mathcal{L}}$ reads $w^u, w^v \in \mathcal{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, it calculates $\langle w^u, w^v \rangle$. If the result is 1, it outputs the empty string (which translates to immediate acceptance). Otherwise, it outputs $(1, 1, 0, 0) \in \mathbb{F}_2^4$.

     Interpretation: In this case $S_{\texttt{First}^{u,v}}^{\mathfrak{L}} = \{\texttt{First}^{u,v}\}$ and $S_{\texttt{Both}^{u,v}}^{\mathfrak{L}} = \{\texttt{Both}_1^{u,v}, \texttt{Both}_2^{u,v}\}$. If $\langle w^u, w^v \rangle = 1$, then this is a copy of the null-commutation game $\mathfrak{C}_{null}$, which always accepts. Otherwise, $\langle w^u, w^v \rangle = 0$ and the game is the commutation game $\mathfrak{C}$, in which case the single bit string encodes the linear constraint

     $$\gamma(\texttt{First}^{u,v}) + \gamma(\texttt{Both}_1^{u,v}) = 0 \ .$$

   - *Question format*: $(t, \mathbf{y}) = (\texttt{Second}, u, v)$, $(t', \mathbf{y}') = (\texttt{Both}, u, v)$.

     Operation: $\widetilde{\mathcal{L}}$ reads $w^u, w^v \in \mathcal{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, it calculates $\langle w^u, w^v \rangle$. If the result is 1, it outputs the empty string (which translates to immediate acceptance). Otherwise, it outputs $(1, 0, 1, 0) \in \mathbb{F}_2^4$.

     Interpretation: In this case $S_{\texttt{Second}^{u,v}}^{\mathfrak{L}} = \{\texttt{Second}^{u,v}\}$ and $S_{\texttt{Both}^{u,v}}^{\mathfrak{L}} = \{\texttt{Both}_1^{u,v}, \texttt{Both}_2^{u,v}\}$. If $\langle w^u, w^v \rangle = 1$, then this is a copy of the null-commutation game $\mathfrak{C}_{null}$, which always accepts. Otherwise, $\langle w^u, w^v \rangle = 0$ and the game is the commutation game $\mathfrak{C}$, in which case the single bit string encodes the linear constraint

     $$\gamma(\texttt{Second}^{u,v}) + \gamma(\texttt{Both}_2^{u,v}) = 0 \ .$$

4. **Pauli Basis Test — (null-)Anti-Commutation game** (Section 3.8.2): For every $1 \leq i, j \leq 3$,

   - *Question format*: $(t, \mathbf{y}) = (\texttt{var}_{ab}, u, v)$, $(t', \mathbf{y}') = (\texttt{row}_a, u, v)$.

     Operation: $\widetilde{\mathcal{L}}$ reads $w^u, w^v \in \mathcal{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, it calculates $\langle w^u, w^v \rangle$. If the result is 0, it outputs the empty string (which translates to immediate acceptance). Otherwise, given that $e_b \in \mathbb{F}_2^3$ is the bth vector of the standard basis (i.e., the indicator of b), $\widetilde{\mathcal{L}}$ outputs (the encoding) of the two bit strings $(1, e_b, 0), (0, 1, 1, 1, 0) \in \mathbb{F}_2^5$.[69]

     Interpretation: In this case $S_{\texttt{var}_{ab}^{u,v}}^{\mathfrak{L}} = \{\texttt{Var}_{ab}^{u,v}\}$ and $S_{\texttt{row}_a^{u,v}}^{\mathfrak{L}} = \{\texttt{Row}_{a1}^{u,v}, \texttt{Row}_{a2}^{u,v}, \texttt{Row}_{a3}^{u,v}\}$. If $\langle w^u, w^v \rangle = 0$, then this is a copy of the null-anti-commutation game $\mathfrak{M}_{null}$, which always accepts. Otherwise, $\langle w^u, w^v \rangle = 1$ and the game is the anti-commutation game $\mathfrak{M}$ — i.e., the magic square game (Example 2.30) — in which case the two bit strings encode the linear constraints

     $$\gamma(\texttt{Var}_{ab}^{u,v}) + \gamma(\texttt{Row}_{ab}^{u,v}) = 0 \quad \text{and} \quad \gamma(\texttt{Row}_{a1}^{u,v}) + \gamma(\texttt{Row}_{a2}^{u,v}) + \gamma(\texttt{Row}_{a3}^{u,v}) = 0 \ .$$

   - *Question format*: $(t, \mathbf{y}) = (\texttt{var}_{ab}, u, v)$, $(t', \mathbf{y}') = (\texttt{col}_j, u, v)$.

     Operation: (This is very similar to the previous case, with rows and columns swapped, and with the sum along columns needing to be 1 instead of 0.) $\widetilde{\mathcal{L}}$ reads $w^u, w^v \in \mathcal{B} \subseteq \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$. Then, it calculates $\langle w^u, w^v \rangle$. If the result is 0, it outputs the empty string (which translates to immediate acceptance). Otherwise, given that $e_a \in \mathbb{F}_2^3$ is the ath vector of the standard basis, $\widetilde{\mathcal{L}}$ outputs (the encoding) of the two bit strings $(1, e_a, 0), (0, 1, 1, 1, 1) \in \mathbb{F}_2^5$.

     Interpretation: In this case $S_{\texttt{var}_{ab}^{u,v}}^{\mathfrak{L}} = \{\texttt{Var}_{ab}^{u,v}\}$ and $S_{\texttt{col}_b^{u,v}}^{\mathfrak{L}} = \{\texttt{Col}_{1b}^{u,v}, \texttt{Col}_{2b}^{u,v}, \texttt{Col}_{3b}^{u,v}\}$. If $\langle w^u, w^v \rangle = 0$, then this is a copy of the null-anti-commutation game $\mathfrak{M}_{null}$, which always accepts. Otherwise, $\langle w^u, w^v \rangle = 1$ and the

---

[69] Just as a sanity check, e.g. if $b = 1$, we seek the encoding of $11000 \sqcup 01110$, which is 0101000000100001010100 according to Definition 2.34.

game is the anti-commutation game $\mathfrak{M}$ — i.e., the magic square game (Example 2.30) — in which case the two bit strings encode the linear constraints

$$\gamma(\mathsf{Var}_{ab}^{u,v}) + \gamma(\mathsf{Col}_{ab}^{u,v}) = 0 \quad \text{and} \quad \gamma(\mathsf{Col}_{1b}^{u,v}) + \gamma(\mathsf{Col}_{2b}^{u,v}) + \gamma(\mathsf{Col}_{3b}^{u,v}) = 1 \ .$$

5. **Introspection Game** (Section 4.1):

   Question format: $(t, \mathrm{y}) = (\mathtt{Intro}_A, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathtt{Intro}_B, \vec{0}, \vec{0})$.

   Operation: Recall that $\widetilde{\mathcal{L}}$ gets as input $a^{\mathfrak{R}}, b^{\mathfrak{R}}$, which in this case are in $\mathbb{F}_2^{2^{|\mathcal{K}(n)|} + \Delta(n)}$, namely bit strings of length $2^{|\mathcal{K}(n)|} + \Delta(n)$. Denote by $que_A \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$ (resp. que_B) the restriction of $a^{\mathfrak{R}}$ (resp. $b^{\mathfrak{R}}$) to its first $2^{|\mathcal{K}(n)|}$ bits, and $ans_A^{\mathfrak{R}} \in \mathbb{F}_2^{\Delta(n)}$ (resp. $ans_B^{\mathfrak{R}}$) the restriction to its last $\Delta(n)$ bits. Now, $\widetilde{\mathcal{L}}$ calls the original linear constraints processor $\mathcal{L}(n, que_A, que_B, ans_A^{\mathfrak{R}}, ans_B^{\mathfrak{R}})$. If the decoding of $\mathcal{L}$'s output is not a sequence $(c^1, ..., c^m)$ of bit strings of length $4\Delta(n) + 1$, then $\widetilde{\mathcal{L}}$ outputs $\mathfrak{error}$. Otherwise, each c^i is of the form $c^i = (c_{A,\mathfrak{R}}^i, c_{A,\mathfrak{L}}^i, c_{B,\mathfrak{R}}^i, c_{B,\mathfrak{L}}^i, c_\mathsf{J}^i)$, where the first four (sub-)bit strings are of length $\Delta(n)$, and c_J^i is of length 1. Then, $\widetilde{\mathcal{L}}$ outputs (the encoding) of the sequence of bit strings $(\tilde{c}^1, ..., \tilde{c}^m)$, where each $\tilde{c}^i = (\vec{0}, c_{A,\mathfrak{R}}^i, c_{A,\mathfrak{L}}^i, \vec{0}, c_{B,\mathfrak{R}}^i, c_{B,\mathfrak{L}}^i, c_\mathsf{J}^i)$ and $\vec{0} \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$.

   Interpretation: In this case $S_{\mathtt{Intro}}^{\mathfrak{R}} = \{\mathsf{Que}^{\cdot, i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\mathsf{Ans}^{\cdot, \mathfrak{R}, j}\}_{j=1}^{\Delta(n)}$ and $S_{\mathtt{Intro}}^{\mathfrak{L}} = \{\mathsf{Ans}^{\cdot, \mathfrak{L}, j}\}_{j=1}^{\Delta(n)}$. Then, $que_A = \gamma(\mathsf{Que}^{A, i})_{i=1}^{2^{|\mathcal{K}(n)|}}, que_B = \gamma(\mathsf{Que}^{B, i})_{i=1}^{2^{|\mathcal{K}(n)|}}$ are treated as a pair of questions in the original game, and $ans_\cdot^{\mathfrak{R}} = \gamma(\mathsf{Ans}^{\cdot, \mathfrak{R}, j})_{j=1}^{\Delta(n)}$ as the respective readable parts of answers. Thus $L_{que_A, que_B}(ans_A^{\mathfrak{R}}, ans_B^{\mathfrak{R}})$ induces linear constraints on the $4\Delta(n)$ variables at the vertices que_A and que_B in the original game, which are checked instead on $\{\mathsf{Ans}^{\cdot, \cdot, j}\}_{j=1}^{\Delta(n)}$ by

   $$\widetilde{L}_{\mathtt{Intro}_A, \mathtt{Intro}_B}(que_A, ans_A^{\mathfrak{R}}, que_B, ans_B^{\mathfrak{R}}) \ .$$

6. **Augmentation — Sampling apparatus**: For $\circ \in \{A, B\}$,[70]

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{Pauli}_{\mathbb{Z}}, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathtt{Sample}_\circ, \vec{0}, \vec{0})$.
     Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding of) the following $2^{|\mathcal{K}(n)|}$ strings

     $$\forall 1 \leq i \leq 2^{|\mathcal{K}(n)|} : \quad (e_i, e_i, \vec{0}, \vec{0}, 0) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2 \ ,$$

     where e_i is the ith standard basis vectors (i.e., indicator of i) of $\mathbb{F}_2^{2^{|\mathcal{K}(n)|}}$.
     Interpretation (Compare to (79)): In this case $S_{\mathtt{Pauli}_{\mathbb{Z}}}^{\mathfrak{L}} = \{\mathsf{PZ}^i\}_{i=1}^{2^{|\mathcal{K}(n)|}}$ and $S_{\mathtt{Sample}_\circ}^{\mathfrak{R}}$ contains $\{\mathsf{SamZ}^{\circ, i}\}_{i=1}^{2^{|\mathcal{K}(n)|}}$. Then, the sequence of bit strings induce the checks

     $$\forall 1 \leq i \leq 2^{|\mathcal{K}(n)|} : \quad \gamma(\mathsf{PZ}^i) = \gamma(\mathsf{SamZ}^{\circ, i}) \ .$$

   - *Question format*: $(t, \mathrm{y}) = (\mathtt{Intro}_\circ, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathtt{Sample}_\circ, \vec{0}, \vec{0})$.
     Operation: Recall that $\widetilde{\mathcal{L}}$ gets as input $a^{\mathfrak{R}}, b^{\mathfrak{R}}$, and in this case

     $$a^{\mathfrak{R}} = (que_\circ, ans_\circ^{\mathfrak{R}}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \quad \text{and} \quad b^{\mathfrak{R}} = (seed, ans_{sam\circ}^{\mathfrak{R}}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)}.$$

     Then, $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \mathtt{Dimension}, \cdot, \cdot, \cdot, \cdot)$ and denotes its length by r. If $r > 2^{|\mathcal{K}(n)|}$, then it outputs the empty string (which translates to acceptance). Otherwise, it takes $que_\circ'$ (resp. $seed'$) to be the restriction of $que_\circ$ (resp. $seed$) to its first r bits, and calls $\mathcal{S}(2^n, \mathtt{Marginal}, \circ, h, \cdot, seed')$ and compares it to $que_\circ'$. If they disagree, $\widetilde{\mathcal{L}}$ returns (the encoding of) the single string $(\vec{0}, 1) \in \mathbb{F}_2^{2 \cdot 2^{|\mathcal{K}(n)|} + 4\Delta(n)} \times \mathbb{F}_2$ (which is the equation associated

---

[70]To be able to distinguish between a blank spot for a player (that needs to be consistent with A or B) and inputs to Turing machines that are disregarded, both of which were denoted by $\cdot$ in the text, we use $\circ$ for the player notation.

with the singleton $\{J\}$, implying rejection). Otherwise, it outputs (the encoding of) the sequence of bit strings $(c^{1,\mathfrak{R}}, c^{1,\mathfrak{L}}, ..., c^{2^{|\mathcal{K}(n)|},\mathfrak{R}}, c^{2^{|\mathcal{K}(n)|},\mathfrak{L}})$, where

$$\forall 1 \le i \le \Delta(n): \quad c^{i,\mathfrak{R}} = (\vec{0}, e_i, \vec{0}, \vec{0}, e_i, \vec{0}, 0) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2,$$
$$c^{i,\mathfrak{L}} = (\vec{0}, \vec{0}, e_i, \vec{0}, \vec{0}, e_i, 0) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2.$$

Interpretation (Compare to (80) and (81)): In this case

$$S_{\text{Intro}_\circ}^{\mathfrak{R}} = \text{Que}^\circ \bigcup \text{Ans}^{\circ,\mathfrak{R}} = \{\text{Que}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\text{Ans}^{\circ,\mathfrak{R},j}\}_{j=1}^{\Delta(n)},$$
$$S_{\text{Intro}_\circ}^{\mathfrak{L}} = \text{Ans}^{\circ,\mathfrak{L}} = \{\text{Ans}^{\circ,\mathfrak{L},j}\}_{j=1}^{\Delta(n)},$$
$$S_{\text{Sample}_\circ}^{\mathfrak{R}} = \text{SamZ}^\circ \bigcup \text{SamAns}^{\circ,\mathfrak{R}} = \{\text{SamZ}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\text{SamAns}^{\circ,\mathfrak{R},j}\}_{j=1}^{\Delta(n)},$$
$$S_{\text{Sample}_\circ}^{\mathfrak{L}} = \text{SamAns}^{\circ,\mathfrak{L}} = \{\text{SamAns}^{\circ,\mathfrak{L},j}\}_{j=1}^{\Delta(n)}.$$

First, if $2^{|\mathcal{K}(n)|}$ is smaller than the dimension of the CLM $\mathfrak{s}^\circ$ — which is the CLM induced by the sampler $\mathcal{S}$ when fixing the Player input to $\circ$ — then no constraints are checked. Otherwise, we denote $\gamma|_{\text{Que}^\circ} = \text{x}$ and $\gamma|_{\text{SamZ}^\circ} = z$, and $\widetilde{\mathcal{L}}$ verifies that $\mathfrak{s}^\circ(z) = \text{x}$, and outputs the certain rejection linear constraint $\gamma(J) = 0$ if not (recall that $\gamma(J) = 1$ by Definition 2.24). If the above condition was held, it then outputs the linear constraints

$$\forall 1 \le j \le \Delta(n): \quad \gamma(\text{Ans}^{\circ,\mathfrak{R},j}) = \gamma(\text{SamAns}^{\circ,\mathfrak{R},j}) \quad, \quad \gamma(\text{Ans}^{\circ,\mathfrak{L},j}) = \gamma(\text{SamAns}^{\circ,\mathfrak{L},j}).$$

7. **Augmentation — Hiding apparatus**: For $\circ \in \{A, B\}$,

- *Question format*: $(t, \text{y}) = (\text{Intro}_\circ, \vec{0}, \vec{0})$, $(t', \text{y}') = (\text{Read}_\circ, \vec{0}, \vec{0})$.
  Operation: $\widetilde{\mathcal{L}}$ outputs (the encoding of) the sequence of bit strings consisting of

  $$\forall 1 \le i \le 2^{|\mathcal{K}(n)|}: \quad c^{i,\text{Que}} = (e_i, \vec{0}, \vec{0}, e_i, \vec{0}, \vec{0}, \vec{0}, 0),$$

  and

  $$\forall 1 \le j \le \Delta(n): \quad c^{j,\mathfrak{R}} = (\vec{0}, e_j, \vec{0}, \vec{0}, e_j, \vec{0}, \vec{0}, 0) \quad, \quad c^{j,\mathfrak{L}} = (\vec{0}, \vec{0}, e_j, \vec{0}, \vec{0}, \vec{0}, e_j, 0)$$

  all of which are in

  $$\mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2,$$

  with e_i being the ith standard basis in the respective space.
  Interpretation (Compare to (82)): In this case

  $$S_{\text{Intro}_\circ}^{\mathfrak{R}} = \text{Que}^\circ \bigcup \text{Ans}^{\circ,\mathfrak{R}} = \{\text{Que}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\text{Ans}^{\circ,\mathfrak{R},j}\}_{j=1}^{\Delta(n)},$$
  $$S_{\text{Intro}_\circ}^{\mathfrak{L}} = \text{Ans}^{\circ,\mathfrak{L}} = \{\text{Ans}^{\circ,\mathfrak{L},j}\}_{j=1}^{\Delta(n)},$$
  $$S_{\text{Read}\circ}^{\mathfrak{R}} = \text{ReadQue}^\circ \bigcup \text{ReadAns}^{\circ,\mathfrak{R}} = \{\text{ReadQue}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\text{ReadAns}^{\circ,\mathfrak{R},j}\}_{j=1}^{\Delta(n)},$$
  $$S_{\text{Read}_\circ}^{\mathfrak{L}} = \text{ReadPerp}^\circ \bigcup \text{ReadAns}^{\circ,\mathfrak{L}} = \{\text{ReadPerp}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\text{ReadAns}^{\circ,\mathfrak{L},j}\}_{j=1}^{\Delta(n)}.$$

  The linear constraints induced by the above are

  $$\forall 1 \le i \le 2^{|\mathcal{K}(n)|}: \quad \gamma(\text{Que}^{\circ,i}) = \gamma(\text{ReadQue}^{\circ,i}),$$
  $$\forall 1 \le j \le \Delta(n): \quad \gamma(\text{Ans}^{\circ,\mathfrak{R},j}) = \gamma(\text{ReadAns}^{\circ,\mathfrak{R},j}) \quad, \quad \gamma(\text{Ans}^{\circ,\mathfrak{L},j}) = \gamma(\text{ReadAns}^{\circ,\mathfrak{L},j}).$$

- *Question format*: $(t, \mathrm{y}) = (\mathtt{Hide}_\circ^h, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathtt{Read}_\circ, \vec{0}, \vec{0})$.

  Operation: Recall that $\widetilde{\mathcal{L}}$ gets as input $a^{\mathfrak{R}}, b^{\mathfrak{R}}$, and in this case

$$a^{\mathfrak{R}} = (que_{hide\ h\ \circ}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \quad \text{and} \quad b^{\mathfrak{R}} = (que_{read\circ}, ans_{read\circ}^{\mathfrak{R}}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)}.$$

Then, $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ and denotes its length by r. If $r > 2^{|\mathcal{K}(n)|}$, then it outputs the empty string (which translates to acceptance). Otherwise, it lets $que'_{read\circ}$ be the restriction of $que_{read\circ}$ to its first r bits, and calls $\mathcal{S}(2^n, \mathrm{Register}, \circ, j, que'_{read\circ}, \cdot)$ for every $1 \leq j \leq h - 1$. The output of each of these runs should be a vector in $\mathbb{F}_2^r$, and $\widetilde{\mathcal{L}}$ adds these outputs and get a bit string $I_{<h} \in \mathbb{F}_2^r$. Finally, $\widetilde{\mathcal{L}}$ outputs (the encoding of) the following sequence of bit strings: For every $1 \leq i \leq r$, if $I_{<h}(i) = 1$, then

$$c^{i,que} = (e_i, \vec{0}, e_i, \vec{0}, \vec{0}, \vec{0}, 0),$$

and if $I_{<h}(i) = 0$, then

$$c^{i,que} = (e_i, \vec{0}, \vec{0}, \vec{0}, \vec{0}, \vec{0}, 0).$$

For every $1 \leq j \leq r$ it adds

$$c^{j,perp} = (\vec{0}, e_j, \vec{0}, \vec{0}, e_j, \vec{0}, 0).$$

All the constraints are in

$$\mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{\Delta(n)} \times \mathbb{F}_2,$$

with e_i being the ith standard basis in the respective space.

  Interpretation (Compare to (83) and (84)): In this case

$$S_{\mathtt{Hide}_\circ^h}^{\mathfrak{R}} = \mathsf{Hide}^h\mathsf{Que}^\circ = \{\mathsf{Hide}^h\mathsf{Que}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}},$$

$$S_{\mathtt{Hide}_\circ^h}^{\mathfrak{L}} = \mathsf{Hide}^h\mathsf{Perp}^\circ = \{\mathsf{Hide}^h\mathsf{Perp}^{\circ,j}\}_{j=1}^{2^{|\mathcal{K}(n)|}},$$

$$S_{\mathtt{Read}_\circ}^{\mathfrak{R}} = \mathsf{ReadQue}^\circ \bigcup \mathsf{ReadAns}^{\circ,\mathfrak{R}} = \{\mathsf{ReadQue}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\mathsf{ReadAns}^{\circ,\mathfrak{R},j}\}_{j=1}^{\Delta(n)},$$

$$S_{\mathtt{Read}_\circ}^{\mathfrak{L}} = \mathsf{ReadPerp}^\circ \bigcup \mathsf{ReadAns}^{\circ,\mathfrak{L}} = \{\mathsf{ReadPerp}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \bigcup \{\mathsf{ReadAns}^{\circ,\mathfrak{L},j}\}_{j=1}^{\Delta(n)}.$$

The vector $I_{<h}$ calculated by $\widetilde{\mathcal{L}}$ is indeed the indicator of coordinates active in the register subspace $W_{<h}^{que'_{read\circ}}$ (see (71)) associated with the CLM $\mathfrak{s}^\circ$ induced by $\mathcal{S}$ when fixing the Player input to be $\circ$. So, the linear constraints induced by the above are

$$\forall 1 \leq i \leq r \text{ s.t. } I_{<h}(i) = 1 : \quad \gamma(\mathsf{Hide}^h\mathsf{Que}^{\circ,i}) = \gamma(\mathsf{ReadQue}^{\circ,i}),$$

$$I_{<h}(i) = 0 : \quad \gamma(\mathsf{Hide}^h\mathsf{Que}^{\circ,i}) = 0,$$

$$\forall 1 \leq j \leq 2^{|\mathcal{K}(n)|} : \quad \gamma(\mathsf{Hide}^h\mathsf{Perp}^{\circ,j}) = \gamma(\mathsf{ReadPerp}^{\circ,j}).$$

- *Question format*: $(t, \mathrm{y}) = (\mathtt{Hide}_\circ^j, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\mathtt{Hide}_\circ^{j-1}, \vec{0}, \vec{0})$.

  Operation: Recall that $\widetilde{\mathcal{L}}$ gets as input $a^{\mathfrak{R}}, b^{\mathfrak{R}}$, and in this case

$$a^{\mathfrak{R}} = (que_{hide\ j\ \circ}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \quad \text{and} \quad b^{\mathfrak{R}} = (que_{hide\ j-1\ \circ}) \in \mathbb{F}_2^{2^{|\mathcal{K}(n)|}}.$$

Then, $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ and denotes its length by r. If $r > 2^{|\mathcal{K}(n)|}$, then it outputs the empty string (which translates to acceptance). Otherwise, it lets $que'_{hide\ j\ \circ}$ be the restriction of $que_{hide\ j\ \circ}$ to its first r bits, and calls $\mathcal{S}(2^n, \mathrm{Register}, \circ, t, que'_{hide\ j\ \circ}, \cdot)$ for every $1 \leq t \leq j$. The output of each of these runs should be a vector in $\mathbb{F}_2^r$, and $\widetilde{\mathcal{L}}$:

♡ sums the first $j-2$ of these outputs to get a bit string $I_{<j-1} \in \mathbb{F}_2^r$;

♡ denotes the output when $t = j$ as $I_j \in \mathbb{F}_2^r$.

Now, $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \text{Perpendicular}, \circ, j, que'_{hide\ j\ \circ}, e_i)$ for every i in the support $\text{Supp}(I_j)$ of I_j (i.e., such that $I_j(i) = 1$) — where e_i is the ith standard basis vector (i.e., indicator of i) — and denotes their output as $Col_i \in \mathbb{F}_2^r$. By the definition of a CL sampler (Definition 4.29), the support of each Col_i is contained in $\text{Supp}(I_j)$. Let $Col_i = \vec{0} \in \mathbb{F}_2^r$ for every $i \notin \text{Supp}(I_j)$. Then, collecting all of these Col_i's as columns in a matrix gives an $r \times r$ matrix Ψ which is supported on $\text{Supp}(I_j) \times \text{Supp}(I_j)$. Let $Row_i \in \mathbb{F}_2^r$ be the ith row of Ψ.

Finally, $\widetilde{\mathcal{L}}$ outputs (the encoding of) the following sequence of bit strings:

$$\forall i \in \text{Supp}(I_{<j-1}) : \quad c^{i,que} = (e_i, \vec{0}, e_i, \vec{0}, 0) \,,$$
$$\forall i \notin \text{Supp}(I_{<j-1}) : \quad c^{i,que} = (\vec{0}, \vec{0}, e_i, \vec{0}, 0) \,,$$
$$\forall i \notin \text{Supp}(I_j) : \quad c^{i,perp} = (\vec{0}, e_i, \vec{0}, e_i, 0) \,,$$
$$\forall i \in \text{Supp}(I_j) : \quad c^{i,perp} = (\vec{0}, e_i, \vec{0}, Row_i, 0) \,.$$

All of the above are in

$$\mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2 \,.$$

Interpretation (Compare to (88), (89) and (90)): In this case

$$S^{\mathfrak{R}}_{\text{Hide}_\circ} = \text{Hide}^{\cdot}\text{Que}^{\circ} = \{\text{Hide}^{\cdot}\text{Que}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \,,$$
$$S^{\mathfrak{L}}_{\text{Hide}_\circ} = \text{Hide}^{\cdot}\text{Perp}^{\circ} = \{\text{Hide}^{\cdot}\text{Perp}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \,.$$

The vectors $I_{<j-1}, I_j$ calculated by $\widetilde{\mathcal{L}}$ are indeed the indicators of coordinates active in the respective register subspaces $W^{que'_{hide\ j\ \circ}}_{<j-1}$ and $W^{que'_{hide\ j\ \circ}}_{j}$ associated with the CLM $\mathfrak{s}^\circ$ induced by $\mathcal{S}$ when fixing the Player input to be $\circ$. So, the linear constraints induced by the above are

$$\forall i \in \text{Supp}(I_{<j-1}) : \quad \gamma(\text{Hide}^j\text{Que}^{\circ,i}) = \gamma(\text{Hide}^{j-1}\text{Que}^{\circ,i}) \,,$$
$$\forall i \notin \text{Supp}(I_{<j-1}) : \quad 0 = \gamma(\text{Hide}^{j-1}\text{Que}^{\circ,i}) \,,$$
$$\forall i \notin \text{Supp}(I_j) : \quad \gamma(\text{Hide}^j\text{Perp}^{\circ,i}) = \gamma(\text{Hide}^{j-1}\text{Perp}^{\circ,i}) \,,$$
$$\forall i \in \text{Supp}(I_j) : \quad \gamma(\text{Hide}^j\text{Perp}^{\circ,i}) = \sum_{t=1}^r Row_i(t)\gamma(\text{Hide}^{j-1}\text{Perp}^{\circ,t}) \,.$$

This makes sense, as $Row_i(t)$ is the it entry of Ψ, whose $\text{Supp}(I_j) \times \text{Supp}(I_j)$ block is exactly $(\mathfrak{s}_j^{\circ, que'_{hide\ j\ \circ}})^\perp$.

- *Question format*: $(t, \mathrm{y}) = (\text{Hide}_\circ^1, \vec{0}, \vec{0})$, $(t', \mathrm{y}') = (\text{Pauli}_{\mathbb{X}}, \vec{0}, \vec{0})$.
  Operation: $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$ and denotes its length by r. If $r > 2^{|\mathcal{K}(n)|}$, then it outputs the empty string (which translates to acceptance). Otherwise, it calls $\mathcal{S}(2^n, \text{Register}, \circ, 1, \vec{0}, \cdot)$ and denotes its output by $I_1 \in \mathbb{F}_2^r$ and its complement by $I_{>1} \in \mathbb{F}_2^r$. Now, $\widetilde{\mathcal{L}}$ calls $\mathcal{S}(2^n, \text{Perpendicular}, \circ, 1, \vec{0}, e_i)$ for every i in $\text{Supp}(I_1)$, and denotes their output as $Col_i \in \mathbb{F}_2^r$. By the definition of a CL sampler (Definition 4.29), the support of each Col_i is contained in $\text{Supp}(I_1)$. Let $Col_i = \vec{0} \in \mathbb{F}_2^r$ for every $i \notin \text{Supp}(I_1)$. Then, collecting all of these Col_i's as columns in a matrix gives an $r \times r$ matrix Ψ which is supported on $\text{Supp}(I_1) \times \text{Supp}(I_1)$. Let $Row_i \in \mathbb{F}_2^r$ be the ith row of Ψ.

Finally, $\widetilde{\mathcal{L}}$ outputs (the encoding of) the following sequence of bit strings:

$$\forall i \in [2^{\mathcal{K}(n)}] : \quad c^{i,que} = (e_i, \vec{0}, \vec{0}, 0) \,,$$
$$\forall i \in \mathrm{Supp}(I_{>1}) : \quad c^{i,perp} = (\vec{0}, e_i, e_i, 0) \,,$$
$$\forall i \in \mathrm{Supp}(I_1) : \quad c^{i,perp} = (\vec{0}, e_i, Row_i, 0) \,,$$

all of which are in

$$\mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2^{2^{|\mathcal{K}(n)|}} \times \mathbb{F}_2 \,.$$

Interpretation (Compare to (85), (86) and (87)): In this case

$$S^{\mathfrak{R}}_{\mathtt{Hide}^1_\circ} = \mathsf{Hide}^1 \mathsf{Que}^\circ = \{\mathsf{Hide}^1 \mathsf{Que}^{\circ,i}\}_{i=1}^{2^{|\mathcal{K}(n)|}} \,,$$
$$S^{\mathfrak{L}}_{\mathtt{Hide}^1_\circ} = \mathsf{Hide}^1 \mathsf{Perp}^\circ = \bigcup \{\mathsf{Hide}^1 \mathsf{Perp}^{\circ,j}\}_{j=1}^{2^{|\mathcal{K}(n)|}} \,,$$
$$S^{\mathfrak{L}}_{\mathtt{Pauli}_{\mathbb{X}}} = \mathsf{PX} = \{\mathsf{PX}^i\}_{i=1}^{2^{|\mathcal{K}(n)|}} \,.$$

The vectors I_1 and $I_{>1}$ calculated by $\widetilde{\mathcal{L}}$ are indeed the indicators of coordinates active in the register subspaces $W_1^{\vec{0}} = V_1$ and $W_{>1}^{\vec{0}} = V_{>1}$ (see (71)) associated with the CLM $\mathfrak{s}^\circ$ induced by $\mathcal{S}$ when fixing the Player input to be $\circ$. So, the linear constraints induced by the above are

$$\forall i \in [2^{\mathcal{K}(n)}] : \quad \gamma(\mathsf{Hide}^1 \mathsf{Que}^{\circ,i}) = 0 \,,$$
$$\forall i \in \mathrm{Supp}(I_{>1}) : \quad \gamma(\mathsf{Hide}^1 \mathsf{Perp}^{\circ,i}) = \gamma(\mathsf{PX}^i) \,,$$
$$\forall i \in \mathrm{Supp}(I_1) : \quad \gamma(\mathsf{Hide}^1 \mathsf{Perp}^{\circ,i}) = \sum_{t=1}^{r} Row_i(t) \gamma(\mathsf{PX}^t) \,.$$

For properties of $\widetilde{\mathcal{L}}$, note that all of its operations are either calls to $\mathcal{K}(n), \mathcal{S}(2^n, \cdot, \cdot, \cdot, \cdot), \mathcal{A}(2^n, \cdot, \cdot)$ and $\mathcal{L}(2^n, \cdot, \cdot, \cdot, \cdot)$, or manipulations of vectors in a space $\mathbb{F}_2^m$ where $m = O(2^{|\mathcal{K}(n)|} + \Delta(n))$. This proves the running time argument. Regarding description length, the above description is fixed up to appending the descriptions of $\mathscr{B}$ (which is constant length), $\mathcal{K}, \mathcal{S}, \mathcal{A}$ and $\mathcal{L}$. □

4.6.1 Proof of Theorem 4.36

The input to $\mathsf{QuestionReduction}_h$ is an h-level verifier $\mathcal{V}$ and an integer λ in binary. First, the TM $\mathsf{QuestionReduction}_h$ uses λ to define two other TMs, $\mathcal{K}^\lambda$ and Λ^λ, which act as follows:

1. $\mathcal{K}^\lambda$ gets as input n in binary (i.e., $\overline{n}$), and outputs a string of $n \cdot \lambda$ ones (i.e., $1^{*n \cdot \lambda}$).

2. Λ^λ gets as input n in binary, and outputs a string of n^λ ones (i.e., 1^{*n^λ}).

Note that the description lengths of both of these TMs is fixed up to appending λ, which requires $\log \lambda$-bits. Hence, by Item (3) of Fact 2.32, their description length is $\mathrm{polylog}(\lambda)$. Furthermore, $\mathbb{T}(\mathcal{K}; \overline{n}) = O(\lambda \cdot n)$ while $\mathbb{T}(\Lambda; \overline{n}) = O(n^\lambda)$. Now:

1. Let $\mathcal{V}^{(1)} = (\mathcal{S}^{(1)}, \mathcal{A}^{(1)}, \mathcal{L}^{(1)}, \mathcal{D})$ be the output of $\mathsf{Padding}(\mathcal{V}, \Lambda^\lambda)$ (Claim 4.49).

2. Let $\mathcal{V}^{(2)} = (\mathcal{S}^{(2)}, \mathcal{A}^{(2)}, \mathcal{L}^{(2)}, \mathcal{D})$ be the output of $\mathsf{TypedQuestionReduction}_h(\mathcal{V}^{(1)}, \mathcal{K}^\lambda)$ (Claim 4.50).

3. Let $\mathcal{V}^{(3)} = (\mathcal{S}^{(3)}, \mathcal{A}^{(3)}, \mathcal{L}^{(3)}, \mathcal{D})$ be the output of $\mathsf{DeType}_1(\mathcal{V}^{(2)})$ (Claim 4.46).[71]

---

[71] The subscript 1 in the DeType_1 is not a mistake, as the output of $\mathsf{TypedQuestionReduction}_h$ is a typed tailored 1-level normal form verifier regardless of h.

The Sampler: By Claim 4.50, $\mathcal{S}^{(2)}$ depends only on $\mathcal{K}^\lambda$ (which itself depends only on λ) and h, and is a typed 1-level CL sampler. Specifically, its description length is polynomial in that of $\mathcal{K}^\lambda$ (with the constants depending on h), i.e. it is $\mathrm{poly}_h(\log \lambda)$, and $\mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}_h(\mathbb{T}(\mathcal{K}^\lambda; n)) = \mathrm{poly}_h(\lambda, n)$. By Claim 4.46, $\mathcal{S}^{(3)}$ is a 3-level CL sampler. The description length of $\mathcal{S}^{(3)}$ is polynomial in that of $\mathcal{S}^{(2)}$ (and depends only on it), which means it is $\mathrm{poly}_h(\log \lambda)$ as well. Furthermore, the running time of $\mathcal{S}^{(3)}$ is polynomial in $|\mathcal{T}| = 2h + 28$, and $\mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}_h(\lambda, n)$. All in all, $\mathbb{T}(\mathcal{S}^{(3)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}_h(\lambda, n)$. Let $\mathcal{S}^\lambda_{\mathrm{QR}} = \mathcal{S}^{(3)}$, and note that it satisfies all the required conditions.

The Answer length function: By Claim 4.49, $\mathcal{A}^{(1)}$ depends only on Λ^λ, which itself only depends on λ. Specifically, its description length is polynomial in that of Λ^λ, namely $\mathrm{poly}(\log \lambda)$, and its runtime satisfies $\mathbb{T}(\mathcal{A}^{(1)}; n, \cdot, \cdot) = O(\mathbb{T}(\Lambda^\lambda; n)) = O(n^\lambda)$. By Claim 4.50, $\mathcal{A}^{(2)}$ depends only on $\mathcal{K}^\lambda$, h and $\mathcal{A}^{(1)}$. Specifically, its description length is polynomial in theirs (up to constants that depend on h), which is $\mathrm{poly}_h(\log \lambda)$. For running time, we have

$$\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot) = \mathrm{poly}_h(2^{\overbrace{|\mathcal{K}^\lambda(n)|}^{=\lambda n}}, \overbrace{\mathbb{T}(\mathcal{K}^\lambda; n)}^{=O(\lambda n)}, \overbrace{\mathbb{T}(\mathcal{A}^{(1)}; 2^n, \cdot, \cdot)}^{=O(2^{\lambda n})}) = \mathrm{poly}_h(2^{\lambda n}).$$

By Claim 4.46, $\mathcal{A}^{(3)}$ runs in time which is polynomial in $|\mathcal{T}| = 2h + 28$, $\mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}_h(n, \lambda)$ and $\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot) = \mathrm{poly}_h(2^{\lambda n})$. Namely, $\mathbb{T}(\mathcal{A}^{(3)}; n, \cdot, \cdot) = \exp_h(\lambda, n)$. Letting $\mathcal{A}^\lambda_{\mathrm{QR}} = \mathcal{A}^{(3)}$ satisfies the required conditions. We leave it to the reader to verify that for well structured inputs, the output of $\mathcal{A}^\lambda_{\mathrm{QR}}$ never decodes to $\mathfrak{error}$.

The Linear constraints processor: By Claim 4.49,

$$\mathbb{T}(\mathcal{L}^{(1)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(\overbrace{\mathbb{T}(\Lambda^\lambda; n)}^{=O(n^\lambda)}, \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; n, \cdot, \cdot), \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)) .$$

By Claim 4.50,

$$\mathbb{T}(\mathcal{L}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(2^{\overbrace{|\mathcal{K}^\lambda(n)|}^{\lambda n}}, \overbrace{\mathbb{T}(\mathcal{K}^\lambda; n)}^{=\lambda n}, \overbrace{\mathbb{T}(\mathcal{S}^{(1)}; 2^n, \cdot, \cdot, \cdot, \cdot, \cdot)}^{=\mathcal{S}}, \overbrace{\mathbb{T}(\mathcal{A}^{(1)}; 2^n, \cdot, \cdot)}^{=O(2^{\lambda n})}, \mathbb{T}(\mathcal{L}^{(1)}; 2^n, \cdot, \cdot, \cdot, \cdot)) .$$

By Claim 4.46,

$$\mathbb{T}(\mathcal{L}^{(3)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(\overbrace{|\mathcal{T}|}^{=2h+28}, \overbrace{\mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot, \cdot)}^{=\mathrm{poly}(\lambda, n)}, \overbrace{\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)}^{=\mathrm{poly}(2^{\lambda n})}, \mathbb{T}(\mathcal{L}^{(2)}; n, \cdot, \cdot, \cdot, \cdot)) .$$

If $\mathcal{V}$ is λ-bounded (Definition 4.33), then

$$\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; n, \cdot, \cdot), \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \leq n^\lambda ,$$

which means

$$\mathbb{T}(\mathcal{L}^{(1)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(n^\lambda) ,$$
$$\mathbb{T}(\mathcal{L}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(2^{\lambda n}) ,$$
$$\mathbb{T}(\mathcal{L}^{(3)}; n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(2^{\lambda n}) .$$

Namely, given that $\mathcal{V}$ is λ-bounded, there is a constant $c = c(h) > 0$ such that $\mathbb{T}(\mathcal{L}^{(3)}; n, \cdot, \cdot, \cdot, \cdot) \leq c2^{c\lambda n}$.

Let $\mathcal{L}'$ be the following 5-input TM: Given that $(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ was its input, it runs $\mathcal{L}^{(3)}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ for $c2^{c\lambda n}$ time steps. If it halted, $\mathcal{L}'$ outputs the same output as $\mathcal{L}^{(3)}$ did. Otherwise, it outputs $\mathfrak{error}$. Note that when $\mathcal{V}$ is λ-bounded, $\mathcal{L}'$ and $\mathcal{L}^{(3)}$ always operate in the same way (they produce the same outputs). Furthermore, the running time of $\mathcal{L}^{(3)}$ is $\exp_h(\lambda, n)$,

which was required.

Completeness, Soundness and Entanglement lower bound: Here, we can assume that $\mathcal{V}$ is λ-bounded. Thus, by Claim 4.49, $\mathcal{V}_n^{(1)} = \mathfrak{Padding}(\mathcal{V}_n, n^\lambda)$. Then, by Claim 4.50, $\mathcal{V}_n^{(2)} = \mathfrak{QueRed}(\mathcal{V}_{2^n}^{(1)}, 2^{\lambda n}, \mathscr{B}(\lambda n))$. Finally, by Claim 4.46, $\mathcal{V}_n^{(3)} = \mathfrak{DeType}(\mathcal{V}_n^{(2)})$. Now, as $\mathcal{V}$ is λ-bounded, $\mathcal{V}' = (\mathcal{S}_{QR}^\lambda, \mathcal{A}_{QR}^\lambda, \mathcal{L}', \mathcal{D})$ defines the same games as $\mathcal{V}^{(3)}$, which means that

$$\mathcal{V}'_n = \mathfrak{DeType}(\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n))) \,.$$

By Fact 4.48, if $\mathcal{V}_{2^n}$ has a perfect ZPC strategy, then so does $\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n})$. Since $\mathcal{V}$ is λ-bounded, the length of questions in the 2^n-th game $\mathcal{V}_{2^n}$ (as well as in the padded version) is at most $(2^n)^\lambda = 2^{\lambda n}$, which is the length of vectors returned by $\mathscr{B}(\lambda n)$. Therefore, we may apply Theorem 4.24 to deduce that

$$\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n))$$

has a perfect ZPC strategy as well. Finally, by Corollary 4.42, the detyping of the above has a perfect ZPC strategy, but this is exactly $\mathcal{V}'_n$, proving the completeness requirements.

In the other direction, assume $\mathrm{val}^*(\mathcal{V}'_n) \geq 1 - \varepsilon$. Then, as the typed graph of $\mathfrak{QueRed}$, described in Example 4.39 and in Figure 14, contains all self loops, using Corollary 4.43 we can deduce that the value of

$$\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n))$$

is at least $1 - O((2h + 28) \cdot 2^{4h+56} \cdot \sqrt{\varepsilon}) = 1 - O_h(\sqrt{\varepsilon})$. Furthermore, for entanglement lower bound we have

$$\mathscr{E}(\mathcal{V}'_n, 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n)), 1 - O_h(\sqrt{\varepsilon})) \,.$$

By Theorem 4.24, if $\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n))$ has value $1 - O_h(\sqrt{\varepsilon})$, then $\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n})$ has value $1 - O_h(h^2 \cdot 2^h \cdot (1 + 2^{2\lambda n}/d^2) \cdot \varepsilon^{1/16}) = 1 - O_h(\varepsilon^{1/16})$, and

$$1 + 2^{2\lambda n}/d^2 \leq 1 + |\mathscr{B}|^2/d^2 \leq 1 + \delta^{-2} = O(1) \,,$$

as d was the (un-normalized) distance of the error correcting code of dimension $2^{\lambda n}$ induced by $\mathscr{B} = \mathscr{B}(\lambda n)$, and we chose $\mathscr{B}$ such that its distance is at least $\delta|\mathscr{B}|$ for a universal constant $\delta > 0$ (all of this was guaranteed by Fact 3.72). Furthermore, for entanglement lower bounds we have

$$\mathscr{E}(\mathfrak{QueRed}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 2^{\lambda n}, \mathscr{B}(\lambda n)), 1 - O_h(\sqrt{\varepsilon})) \geq \mathscr{E}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 1 - O_h(\varepsilon^{1/16})) \,.$$

Since $\mathcal{V}$ is λ-bounded, $2^{\lambda n}$ is an upper bound on $|\mathcal{A}(2^n, \cdot, \cdot)|$ which is $\max\{\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}}\}$ in $\mathcal{V}_{2^n}$. Hence, By Fact 4.48, $\mathcal{V}_{2^n}$ has value $1 - O_h(\varepsilon^{1/16})$, and

$$\mathscr{E}(\mathfrak{Padding}(\mathcal{V}_{2^n}, 2^{\lambda n}), 1 - O_h(\varepsilon^{1/16})) \geq \mathscr{E}(\mathcal{V}_{2^n}, 1 - O_h(\varepsilon^{1/16})) \,.$$

Combining all of the above, gives the required soundness and entanglement lower bounds.

By choosing $c_{QR}(h)$ (121) to be large enough to bound all the constants along this proof, we conclude the theorem. $\qquad\square$

5 Answer reduction using probabilistically checkable proofs

The goal of this section is to devise an algorithm $\mathsf{AnswerReduction}_{h,h'}$ that takes as input a tailored h-level normal form verifier, whose sampler is efficient but answer length calculator and linear constraints processors run in exponential time (with the constants in the bounds depending on h'),[72] and transforms it in a complete and sound way to a normal form verifier with all components running efficiently. Recall the asymptotic notation from Remark 1.2.

Theorem 5.1 (Answer Reduction, proved in Section 5.6)**.** *Let h and h' be positive integers. There is a positive integer constant*

$$c = c_{\mathrm{AR}}(h, h') \tag{122}$$

*depending only on h and h', and a 2-input TM $\mathsf{AnswerReduction}_{h,h'}$, that takes as input a tailored h-level normal form verifier $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$, and a positive integer λ, and outputs a tailored, **typed** $\max(3,h)$-level normal form verifier*

$$\mathcal{V}_{\mathrm{AR}} = \mathsf{AnswerReduction}_{h,h'}(\mathcal{V}, \lambda) = (\mathcal{S}_{\mathrm{AR}}, \mathcal{A}_{\mathrm{AR}}, \mathcal{L}_{\mathrm{AR}}, \mathcal{D})$$

with 9 types and type graph depicted in Figure 17, and the following properties:

- *Sampler properties: $\mathcal{S}_{\mathrm{AR}}$ depends only on λ, h, h' and the original sampler $\mathcal{S}$ (but not on $\mathcal{A}$ or $\mathcal{L}$), and $\mathsf{AnswerReduction}_{h,h'}$ can calculate its description in time $\mathrm{poly}_{h,h'}(\log \lambda, |\mathcal{S}|)$ from them; in particular, $|S_{\mathrm{AR}}| \leq c(\log^c \lambda + |\mathcal{S}|^c)$. In addition, $\mathcal{S}_{\mathrm{AR}}$ runs in time $\mathrm{poly}_{h,h'}(n, \lambda, \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot))$, namely*

$$\forall n \in \mathbb{N}: \quad \mathbb{T}(\mathcal{S}_{\mathrm{AR}}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \leq c \cdot (n^c + \lambda^c + \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)^c),$$

  where c is from (122).

- *Answer length calculator properties: $\mathcal{A}_{\mathrm{AR}}$ depends only on h, h' and λ, and $\mathsf{AnswerReduction}_{h,h'}$ can calculate its description in time $\mathrm{polylog}_{h,h'}(\lambda)$; in particular $|\mathcal{A}_{\mathrm{AR}}| \leq c \log^c \lambda$. In addition, $\mathcal{A}_{\mathrm{AR}}$ runs in $\mathrm{poly}_{h,h'}(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N}: \quad \mathbb{T}(\mathcal{A}_{\mathrm{AR}}; n, \cdot, \cdot) \leq c \cdot (n^c + \lambda^c).$$

  Finally, given that $\mathrm{x} \in \mathbb{F}_2^{r(n)}$, where $r(n) = \mathcal{S}_{\mathrm{AR}}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and that $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the output of $\mathcal{A}_{\mathrm{AR}}(n, \mathrm{x}, \kappa)$ never decodes (Definition 2.34) to an $\mathfrak{error}$ sign.

- *Linear constraints process properties: $\mathcal{L}_{\mathrm{AR}}$ depends on all inputs, namely $h, h', \mathcal{V}$ and λ. Also, $\mathsf{AnswerReduction}_{h,h'}$ can calculate its description in time $\mathrm{poly}_{h,h'}(\log \lambda, |\mathcal{V}|)$; in particular, $|\mathcal{L}_{\mathrm{AR}}| \leq c \cdot (\log^c \lambda + |\mathcal{V}|^c)$. In addition, $\mathcal{L}_{\mathrm{AR}}$ runs in $\mathrm{poly}_{h,h'}(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N}: \quad \mathbb{T}(\mathcal{L}_{\mathrm{AR}}; n, \cdot, \cdot, \cdot, \cdot) \leq c \cdot (n^c + \lambda^c + \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)^c).$$

  Note that although $\mathcal{L}_{\mathrm{AR}}$ depends on all of $\mathcal{V}$, its running time is bounded only in terms of the above parameters. i.e., it may not even read all the description of $\mathcal{V}$ in its operation (if it is too long).

- *Value properties: Let $c_{\mathrm{QR}} = c_{\mathrm{QR}}(h') > 0$ be the constant in (121), guaranteed by Theorem 4.36. Given that*

$$|\mathcal{V}| \leq 5\, c_{\mathrm{QR}} \cdot \lambda^{c_{\mathrm{QR}}},$$

  and

$$\forall n \in \mathbb{N}: \quad \mathbb{T}(\mathcal{S}\,;\, n, \cdot, \cdot, \cdot, \cdot, \cdot) \leq c_{\mathrm{QR}}(n^{c_{\mathrm{QR}}} + \lambda^{c_{\mathrm{QR}}}) \quad ; \quad \mathbb{T}(\mathcal{A}\,;\, n, \cdot, \cdot),\, \mathbb{T}(\mathcal{L}\,;\, n, \cdot, \cdot, \cdot, \cdot) \leq 2^{c_{\mathrm{QR}}(n^{c_{\mathrm{QR}}} + \lambda^{c_{\mathrm{QR}}})},$$

  we have that $\mathcal{V}_{\mathrm{AR}}$, the output of $\mathsf{AnswerReduction}_{h,h'}$, satisfies for all $n \geq 2$:

---

[72]Not surprisingly, these assumptions are satisfied by the output of the $\mathsf{QuestionReduction}_{h'}$ algorithm (Theorem 4.36) in the previous section.

1. **_Completeness_**: _If $\mathcal{V}_n$ has a perfect Z-aligned permutation strategy that commutes along edges (ZPC strategy), then so does $(\mathcal{V}_{\mathrm{AR}})_n$._

2. **_Soundness_**: _If $(\mathcal{V}_{\mathrm{AR}})_n$ has quantum value $1 - \varepsilon$, then the value of $\mathfrak{DoubleCover}(\mathcal{V}_n)$ is at least_

$$1 - c \cdot \left((n\lambda)^c \varepsilon^{1/c} + (n\lambda)^{-1/c} \right),$$

where c is again $c_{\mathrm{AR}}(h, h')$ from (122).

3. **_Entanglement bound_**: _For the same constant c, we have_

$$\mathscr{E}((\mathcal{V}_{\mathrm{AR}})_n, 1 - \varepsilon) \geq \mathscr{E}\left(\mathfrak{DoubleCover}(\mathcal{V}_n), 1 - c \cdot \left((n\lambda)^c \varepsilon^{1/c} + (n\lambda)^{-1/c} \right) \right).$$

The main idea underlying Answer Reduction is to use techniques from the field of probabilistically checkable proofs (PCPs). The way this is implemented requires several steps. First, some preprocessing on the verifier needs to be done, and specifically it needs to be _padded and purified_. Then, the task of deciding whether the tuple $(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}})$ passes the checks at a specific edge xy or not is replaced by a succinct SAT instance and a succinct LIN instance, which have a PCP format that is both ZPC-complete and sound.

Let us elaborate more on this last step. After question reduction, the decision procedure in the nth game defined by $\mathcal{V}$ takes exponential time in n, but the sampling mechanism and thus the length of questions are already polynomial in n. So, using standard techniques, in particular the Cook–Levin Theorem 5.32 and Reed–Muller encoding (Definition 5.17), this decision problem can be replaced by a list of low degree polynomial equations that need to be satisfied. Once this is done, there are standard ways of verifying that two provers answer according to a list of low degree polynomials (Section 5.4), called "the low-degree test". Thus, the aforementioned polynomial equations can be checked for a random point. By the Schwartz–Zippel Lemma 5.19, passing such a check with high probability implies the polynomial equations are indeed satisfied, which in turn means that the polynomials encode a tuple of answers that should be accepted by $\mathcal{V}_n$.

The section is structured as follows:

1. Section 5.1 is a Prelude, which contains both basic definitions needed for Answer Reduction — circuits, low-degree encoding and PCPs — as well as sketching the classical MIP = NEXP result.

2. Section 5.2 describes a few transformations that are used as part of Answer Reduction. These include purification, oracularization triangulation and decoupling. Combinatorially, oracularization means applying a barycentric subdivision to the underlying graph of the game — there is also a more compelling dramatized perspective of this transformation, on which we elaborate in Remark 5.44. Triangulation is the standard triangulation of systems of linear equations (see (139), (140) and (141)). Purification is a simple transformation that removes all readable variables from the controlled linear constraints (Definition 5.40). Decoupling is a standard way of "block dividing" a triangulated system of linear equations or a 3CNF formula.

3. Section 5.3 trnaslates the condition "$a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ are accepted by $\mathcal{V}_n$ given xy were asked" to a list of 13 polynomial equations that can be checked probabilistically. This procedure uses various techniques from the field of probabilistically checkable proofs (PCPs), and specifically a decoupled version of the scaled up Cook–Levin transformation (Proposition 5.62).

4. Section 5.4 recalls the quantum low degree test and its soundness properties [JNV$^+$22a]. This test verifies that two provers that use a quantum strategy answer according to a list of low degree polynomials. This is important, as the 13 polynomial equations recovered in Section 5.3 can be checked to be satisfied probabilistically only when the constituting polynomials that are checked have low degree.

5. Section 5.5 provides a complete description of the Answer Reduction transformation, both the combinatorial one and the algorithmic one, yet under some assumptions on the input TNFV. In it, the transformation is showed to be ZPC-complete and sound.

6. Finally, Section 5.6 collects all of the above to prove Theorem 5.1.

5.1 Prelude — Decision Problems, Complexity Classes, Low-Degree Polynomials, Circuits and PCPs

The goal of this prelude is to provide the required definitions and sketch the proof of the influential result MIP = NEXP due to Babai–Fortnow–Lund [BFL91]. Familiarity with these ideas is crucial for better understanding the answer reduction transformation, and thus we encourage any reader not familiar with this result, and the techniques used in it, to read this section.

5.1.1 Decision Problems and Complexity Classes

A *language* L is a subset of bit strings, namely $L \subseteq \{0,1\}^*$. Every language defines a *decision problem*: Given a bit string x, decide whether $x \in L$ or not. A slight generalization of a language is a *promise language*, which consists of two disjoint subsets $L_{yes}, L_{no} \subseteq \{0,1\}^*$, and the decision problem in this case is: Given $x \in L_{yes} \sqcup L_{no}$, decide whether $x \in L_{yes}$ or $x \in L_{no}$. Because promise languages include regular languages, from now on whenever we say language we mean a promise one.

The Halting Problem (HP) is the decision problem induced by the following subset of $\{0,1\}^*$: A bit string x is in HALT if and only if there is a 1-input Turing machine (TM) $\mathcal{M}$ such that x is the encoding of $\mathcal{M}$, namely $x = \overline{\mathcal{M}}$ (see Definition 2.37), and $\mathcal{M}$ halts on the empty input. Famously, Turing [Tur37] proved that this problem is undecidable, namely that there is no TM that takes $x \in \{0,1\}^*$ as input, always halts, and outputs 1 if $x \in$ HALT and 0 if $x \notin$ HALT.

We say that there is a *reduction* from language L^1 to language L^2, if there is an always halting 1-input Turing machine $\mathcal{M}$, such that $x \in L^1_{yes}$ implies $\mathcal{M}(x) \in L^2_{yes}$, and $x \in L^1_{no}$ implies $\mathcal{M}(x) \in L^2_{no}$.[73] Such a reduction is f-time, for a function $f \colon \mathbb{N} \to \mathbb{N}$, if for every instance x the running time of the reduction $\mathcal{M}$ satisfies $\mathbb{T}(\mathcal{M}; x) \leq f(|x|)$. A *class* of problems is a set $\mathscr{C}$ of languages, namely $\mathscr{C} \subseteq \{0,1\}^{\{0,1\}^*}$. A language L is said to be *complete* for $\mathscr{C}$ if $L \in \mathscr{C}$ and there is a reduction from every $L' \in \mathscr{C}$ to L. A language is f-time complete if the reduction is always f-time. Moreover, we say that L is polynomial-time complete for $\mathscr{C}$ if for every $L' \in \mathscr{C}$, there is a polynomial f and a f-time reduction from L' to L.

Definition 5.2 (RE). We say that a (promise) language L is in the class RE if there is a 2-input TM $\mathcal{V}$ such that:

- (Halting condition) $\mathcal{V}$ always halts and outputs a single bit;

- (Completeness) if $x \in L_{yes}$, then there is a $\pi \in \{0,1\}^*$ such that $\mathcal{V}(x, \pi) = 1$;

- (Soundness) if $x \in L_{no}$, then for every $\pi \in \{0,1\}^*$ we have $\mathcal{V}(x, \pi) = 0$.

Remark 5.3. It is straightforward to check that HALT is in RE, and that HALT is complete for RE.

Definition 5.4 (NP and NEXP). Let $f \colon \mathbb{N} \to \mathbb{N}$ be a function. The language L is in the class $\mathsf{NTIME}(f(n))$ (non-deterministic f-time) if there is a 2-input TM $\mathcal{V}$ such that:

- (Time bound) $\mathbb{T}(\mathcal{V}; x, \pi) \leq f(|x|)$, namely $\mathcal{V}$ runs in f-time in its first input;

- (Completeness) if $x \in L_{yes}$, then there is a $\pi \in \{0,1\}^*$ such that $\mathcal{V}(x, \pi) = 1$;

- (Soundness) if $x \in L_{no}$, then for every $\pi \in \{0,1\}^*$ we have $\mathcal{V}(x, \pi) = 0$.

The class NP (non-deterministic polynomial time) is $\bigcup_{C \in \mathbb{N}} \mathsf{NTIME}(Cn^C)$ and the class NEXP (non-deterministic exponential time) is $\bigcup_{C \in \mathbb{N}} \mathsf{NTIME}(2^{Cn^C})$.

Remark 5.5 (Dramatization of NP). There is a resource restricted (in this case, polynomial time) entity, called the *verifier* and which is denoted by $\mathcal{V}$, that wants to decide whether a bit string x is in the language of interest L. It asks an all knowing *prover* $\mathcal{P}$ to provide a written proof that indeed $x \in L$. The prover $\mathcal{P}$ generates (in a single time step for $\mathcal{V}$) such a proof

---

[73]Recall that a k-input TM always defines a partial function from $(\{0,1\}^*)^k$ to $\{0,1\}^*$, and if the TM always halts this is a proper function. Thus, we abuse notation and write $\mathcal{M}(x)$ for the output of $\mathcal{M}$ given the input x.

$\pi \in \{0,1\}^*$, and sends it to $\mathcal{V}$. The verifier then reads the proof (unless it is too long, in which case it reads only part of it), and decides (under its time restrictions) whether to accept (i.e., declare "x is in L") or to reject (i.e., declare "x is not in L"). The language L is in NP if such a verifier will be convinced by **some** proof π given that $x \in L$, and will never be convinced by **any** proof π given that $x \notin L$.

Definition 5.6 (MIP, MIP*, TailoredMIP and TailoredMIP*)**.** Let $f \colon \mathbb{N} \to \mathbb{N}$ be a function. A language L is in the class MIPTIME($f(n), 2, 1$) (multi-prover interactive proofs with an f-time verifier, 2-provers and 1-round) if there is a (tailored) normal form verifier $\mathcal{V}$ (Definition 2.47), such that:

- (Time bound) the TMs $\mathcal{S}, \mathcal{A}$ and $\mathcal{L}$ run in f-time in their first input, namely

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{S}; \overline{n}), \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot), \mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot) \leq f(|\overline{n}|) \approx f(\log n).$$

- (Completeness) if $\overline{n} \in L_{yes}$, then $\mathcal{V}_n$ has a value 1 **classical** strategy (Example 2.20);

- (Soundness) if $\overline{n} \in L_{no}$, then every classical strategy for $\mathcal{V}_n$ has value of at most $1/2$.

The class MIP*TIME($f(n), 2, 1$) is defined almost the same, but with the classical strategies in the completeness and soundness conditions being replaced by **quantum** strategies (Definition 2.18). Furthermore, the class TailoredMIP*TIME($f(n), 2, 1$) is defined the same as MIP*TIME($f(n), 2, 1$), but with the extra condition in the completeness case that the perfect strategy needs to be a ZPC one. Finally,

$$\mathsf{MIPP} = \bigcup_{C \in \mathbb{N}} \mathsf{MIPTIME}(Cn^C, 2, 1) \quad , \quad \mathsf{MIPEXP} = \bigcup_{C \in \mathbb{N}} \mathsf{MIPTIME}(2^{Cn^C}, 2, 1) \, ,$$

and similarly one defines MIP*P, MIP*EXP, TailoredMIP*P and TailoredMIP*EXP. When we write MIP* or TailoredMIP* we mean the polynomial time versions.

Remark 5.7 (Dramatization of MIP*)**.** There is a time bounded entity, the *verifier* denoted by $\mathcal{V}$, that wants to decide whether a bit string x is in the language of interest L. It devises a game $\mathfrak{G}_x$ (Definition 2.16), and describes it to **two** provers, A and B. It then plays one round of this game against the two provers — as was described in Remark 2.23, where the verifier is the referee and the provers are the players. If the provers win the round, then $\mathcal{V}$ accepts (i.e., declares "x is in L"), and if they lose the round, then $\mathcal{V}$ rejects (i.e., declares "x is not in L").

The language L is in MIP* (resp. TailoredMIP*), if given that $x \in L$ the provers can win $\mathfrak{G}_x$ with probability 1 using a quantum strategy (resp. a ZPC-strategy), and given that $x \notin L$, the players lose with probability at least $1/2$, regardless of the quantum strategy they chose.

Remark 5.8. Now it should be clear why Theorem 2.31 is called TailoredMIP* = RE. By choosing the appropriate encoding of games $\mathfrak{G}_{\mathcal{M}}$ using normal form verifiers (Definition 2.47), the theorem states that there is a normal form verifier $\mathcal{V}$, such that

$$\mathbb{T}(\mathcal{S}; \overline{n}), \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot), \mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot) = \mathrm{poly}(|\overline{n}|) = \mathrm{polylog}(n) \, ,$$

and if $\overline{n} = \overline{\mathcal{M}}$ for a TM $\mathcal{M}$, then: $\mathcal{M}$ halting means $\mathcal{V}_n$ has a perfect ZPC strategy; $\mathcal{M}$ not halting means $\mathrm{val}^*(\mathcal{V}_n) \leq 1/2$. This specific normal form verifier $\mathcal{V}$ can be extracted from the proof in Section 2.6: Given $\overline{\mathcal{M}}$, calculate $\lambda = \lambda(\mathcal{M})$ (Lemma 2.61), and then play the game $\mathcal{V}_C^{\mathcal{M}, \lambda}$ for the universal constant C guaranteed by Theorems 2.53 and 4.34. This exactly shows that HALT is in TailoredMIP*, and as HALT is complete for RE, this proves RE $\subseteq$ TailoredMIP*. The reverse inclusion was described in the beginning of Section 2.6.

Remark 5.9. Let L be a language. Note that if one finds a λ-bounded tailored normal form verifier $\mathcal{V}$ such that $\mathcal{V}_n$ has a value 1 strategy if $\overline{n} \in L$, and otherwise $\mathcal{V}_n$ has value bounded from above by $1/2$, then L is only in MIP*EXP and not MIP*. This is because $\mathcal{V}$ runs in time $n^\lambda = 2^{\lambda \log n}$, which is exponential in the input length $|\overline{n}| \approx \log n$.

5.1.2 The Cook–Levin theorem

Recall that this prelude focuses on proving MIP = NEXP (Definitions 5.4 and 5.6). The containment MIP $\subseteq$ NEXP is quite straightforward, and we leave it to the reader. For the other direction, we need the scaled up version of the celebrated Cook–Levin theorem [Coo71, Lev73]. In this section we describe the content, and sketch the proof, of the **standard** Cook–Levin theorem. Later, in Section 5.1.4, we describe the scaled up version of this theorem and the adjustments needed to prove it (see Theorem 5.32). To that end, we first observe that there is a natural complete decision problem for NP (and respectively NEXP):

Definition 5.10 (Time Restricted Halting). The decision problem UnaryTimeHalt (respectively, BinaryTimeHalt) is the following: Given a pair consisting of (the encoding of) a single[74] input TM $\mathcal{M}$ and a positive integer T in unary (respectively binary, see Definition 2.35), decide whether there exists an input $\pi \in \{0,1\}^*$ such that $\mathcal{M}(\pi)$ halts and outputs 1 in less than T time steps.

Claim 5.11 (Time restricted halting is complete for non-deterministic time). *The decision problem* UnaryTimeHalt *is polynomial time complete for* NP, *and similarly* BinaryTimeHalt *is polynomial time complete for* NEXP.

Proof. We leave it for the reader to check that, indeed UnaryTimeHalt is in NP and BinaryTimeHalt is in NEXP. We also omit the reduction from every NEXP language to BinaryTimeHalt, as it is virtually identical to the reduction from every NP language to UnaryTimeHalt (up to the encoding of the integer), which we now present.

By Definition 5.4, a language is in NP if there is a constant C and a 2-input TM $\mathcal{V}$ that runs in time $\mathbb{T}(\mathcal{V}; \mathbf{x}, \pi) \leq C|\mathbf{x}|^C$ such that $\mathbf{x} \in L_{yes}$ implies the existence of a $\pi \in \{0,1\}^*$ for which $\mathcal{V}(\mathbf{x}, \pi) = 1$, and $\mathbf{x} \in L_{no}$ implies that for every $\pi \in \{0,1\}^*$ we have $\mathcal{V}(\mathbf{x}, \pi) = 0$. So, for every $\mathbf{x} \in L_{yes} \cup L_{no}$ we can define the single input TM $\mathcal{M}_{\mathbf{x}} = \mathcal{V}(\mathbf{x}, \cdot)$ and an integer (in unary) $T_{\mathbf{x}} = 1^{*C|\mathbf{x}|^C} = \underbrace{1...1}_{C|\mathbf{x}|^C - \text{times}}$. Hence, $\mathbf{x} \in L_{yes}$ exactly implies that $(\mathcal{M}_{\mathbf{x}}, T_{\mathbf{x}})$ is in UnaryTimeHalt, and $\mathbf{x} \in L_{no}$ implies $(\mathcal{M}_{\mathbf{x}}, T_{\mathbf{x}})$ is not in UnaryTimeHalt. As, given $\mathcal{V}$, calculating $(\mathcal{M}_{\mathbf{x}}, T_{\mathbf{x}})$ takes poly$(|\mathbf{x}|)$-time, the proof is finished. $\square$

Though it is nice to have some complete language to the complexity class of interest, time restricted halting is not a very useful one. So, it is natural to seek some other language in NP (respectively NEXP) to which UnaryTimeHalt (respectively BinaryTimeHalt) can be reduced to (in polynomial time).

Definition 5.12 (CNF). A *literal* is either a formal variable X or its negation $\neg \mathsf{X}$. For $\varepsilon \in \mathbb{F}_2$, we use the notation

$$\mathsf{X}^\varepsilon = \begin{cases} \mathsf{X} & \varepsilon = 0 \,, \\ \neg \mathsf{X} & \varepsilon = 1 \,. \end{cases}$$

A *disjunction* is an OR of smaller formulas, namely $\bigvee_{i=1}^m \varphi_i$, and a *conjunction* is an AND of smaller formulas, namely $\bigwedge_{i=1}^m \varphi_i$. A CNF formula is a conjunction of disjunctions of literals, namely if $\{\mathsf{X}_1, ..., \mathsf{X}_n\}$ is the set of formal variables, then there is an integer m, integers $k_1, ..., k_m$, and functions $i_j \colon [k_j] \to [n]$ and $\varepsilon_j \colon [k_j] \to \mathbb{F}_2$ for every $1 \leq j \leq m$, such that the formula is of the form

$$\varphi(\mathsf{X}_1, ..., \mathsf{X}_n) = \bigwedge_{j=1}^m \bigvee_{t=1}^{k_j} \mathsf{X}_{i_j(t)}^{\varepsilon_j(t)} \,.$$

It is called a k-CNF, if all the k_j's in the above formula are equal to the same integer k. A formula $\varphi(\mathsf{X}_1, ..., \mathsf{X}_n)$ is *satisfiable* if there is an *assignment* $\psi \colon \{\mathsf{X}_1, ..., \mathsf{X}_n\} \to \{\text{True}, \text{False}\}$ such that $\varphi(\psi(\mathsf{X}_1), ..., \psi(\mathsf{X}_n)) = \text{True}$.

A CNF formula can be encoded in many ways. E.g., one can provide the list of integers $n, m, k_1, ..., k_m$ and then the evaluation table of the functions i_j and ε_j. A k-CNF formula has even nicer encodings, e.g., by providing a matrix of size

---

[74]The k-input instance version is similar, and we actually use later the 3-input version to implement answer reduction (see Remark 5.61).

$m \times k$ with entries being pairs of a bit ε and an integer between 1 and n. In any case, after fixing such an encoding scheme for 3-CNF formulas, the language 3SAT is the following: A bit string x is in 3SAT if and only if it encodes a satisfiable 3-CNF formula. We describe Succinct-3SAT later in this section.

Theorem 5.13 (Cook–Levin [Coo71, Lev73]. See also [Kar72]). *The language* 3SAT *is polynomial time* NP-*complete.*

Proof sketch. By Claim 5.11, it is enough to show that one can reduce UnaryTimeHalt to 3SAT in polynomial time. Namely, given a single input TM $\mathcal{M}$ and an integer in unary T, translate them to a 3-CNF formula such that this formula is satisfiable if and only if there is an input that will make $\mathcal{M}$ output 1 in time at most T.

Recall how a TM operates: It has k infinite tapes (3 in the case of a single input TM — one input tape, one memory tape, and one output tape), whose cells are parametrized by $\mathbb{Z}$, and each cell contains either a bit or is empty (which we think of as containing the special symbol $\sqcup$). Each tape has a head, positioned initially at cell number 0. It has a finite list Q of internal states, two of them are the initial state $q_{initial}$ and the halting state q_{halt}; the TM is always initialized to be in $q_{initial}$, and if it arrives at q_{halt} it stops its operation. Finally, there is an instruction table, that tells the machine given the reads from its heads and its current non-halt state, which new values to write in the current position of the heads, which way should each head move (either not move, one step up or one step down), and what should be its new internal state; namely, the instruction table is a mapping from $\{0, 1, \sqcup\}^k \times (Q \setminus \{q_{halt}\})$ to $\{0, 1, \sqcup\}^k \times \{-1, 0, 1\}^k \times Q$.

Since we care about the operation of the TM $\mathcal{M}$ for only T steps, the only cells of the tapes it may visit are in the interval $-T$ to $+T$. So, we can have the following finitely many variables that "remember" everything about the operation of $\mathcal{M}$:

- For each index i between $-T$ and T, each index j between 1 and k, and each time t between 0 and T, there should be a variable that contains the content of the ith cell in the jth tape at time step t of the operation of $\mathcal{M}$. As the content of a cell may be either $0, 1$ or $\sqcup$, and in the end these variables should be part of a boolean formula, we need two variables to encode this information; the combination of values of the two boolean variables will be interpreted as $\{0, 1, \sqcup\}$ according to the encoding map from Definition 2.34.

- In addition, for the same range for i, j, t, there should be a variable whose boolean value answers the question "is the jth head in position i at time t?".

- Finally, for each t in the above range, there should be a variable that indicates the state of $\mathcal{M}$ at time t. Again, as there are $|Q|$ states, we need to choose some encoding of them as length $\lceil \log |Q| \rceil$ bit strings, and then there are $\lceil \log |Q| \rceil$ many boolean variables whose combinations of values encode the state of $\mathcal{M}$ at each time step.

Then, we need to describe the clauses of the 3-CNF formula that uses the above variables. For example, there will be clauses that check that, initially, at time $t = 0$, the machine was in state $q_{initial}$, and that all the heads were in position 0 (and not in any other position, as there is a single head at each tape), and that all but the input tape cells are empty and so on. Then, there will be clauses that check that the variables of time t were well deduced from those of the previous time $t - 1$ according to the instructions table. For example, the head moved at most 1 step from its previous position, and it moved correctly; the new value at each position is indeed what the instructions table says it should be; the new state of the machine is what it should be, and so on. Finally, the variables associated to the output tape at time T need to encode the output 1, and similarly the variables associated to the state of the machine should be the encoding of q_{halt}.

Note that the only "free variables" in the formula are those associated with the content of the input tape(s) at time 0; the content of all the other variables (in a satisfying assignment) is either fixed or can be deduced from the content of the free variables. Hence, a satisfying assignment is essentially an appropriate choice for the free variables, such that indeed the machine halted in less than T steps and its output is 1, which is exactly what we sought after. □

5.1.3 Low-degree polynomials and robust tests for them

Polynomials play a major role in the construction of probabilistically checkable proofs, both as the form of encoding of the proof and as a tool to verify the proof's validity. Although we have not motivated the "why" yet, let us provide some definitions and facts regarding polynomials of low degree.

Definition 5.14 (Polynomials and their degrees). A polynomial with n variables $\vec{X} = (X_1, ..., X_n)$ and coefficients in the field $\mathbb{F}$ is a formal sum

$$f = \sum_{\vec{e} \in (\mathbb{Z}_{\geq 0})^n} c_{\vec{e}} \cdot X_1^{e_1} \cdot ... \cdot X_n^{e_n} ,$$

where each $c_{\vec{e}} \in \mathbb{F}$, and all but finitely many $c_{\vec{e}}$'s are 0. The *total degree* of f is

$$\max_{\vec{e}:\, c_{\vec{e}} \neq 0} (e_1 + ... + e_n) ,$$

while its *individual degree* is

$$\max_{\vec{e}:\, c_{\vec{e}} \neq 0} \max_{1 \leq i \leq n} (e_i) .$$

For a given $i \in [m]$, the X_i-degree of f is

$$\max_{\vec{e}:\, c_{\vec{e}} \neq 0} e_i ;$$

f is said to be *indifferent* to the ith input if the X_i-degree of it is 0. We often emphasize the variable set of the polynomial by denoting $f(\vec{X})$ or $f(X_1, ..., X_n)$ instead of just f.

Remark 5.15. Note that if $f \in \mathbb{F}[X_1, ..., X_m]$ has total degree at most d, then it has individual degree at most d. On the other hand, if f has individual degree at most d, then its total degree is at most md.

It is natural to associate a function with each polynomial via assignments. Namely, if $f : \mathbb{F}[X_1, ..., X_m]$ is a polynomial, it induces a function $\Phi_{\mathbb{F}}(f) : \mathbb{F}^m \to \mathbb{F}$ that takes as input $(x_1, ..., x_m) \in \mathbb{F}^m$ and outputs

$$\Phi_{\mathbb{F}}(f)(x_1, ..., x_m) = \sum_{\vec{e}} c_{\vec{e}} x_1^{e_1} \cdot ... \cdot x_m^{e_m} \in \mathbb{F} , \tag{123}$$

i.e., uses the assignment $X_i \mapsto x_i$. As this transformation is so entrenched in mathematics and computer science, we usually think of f itself as a function and use the same notation for it and for $\Phi_{\mathbb{F}}(f)$. In this paper we mostly do the same, but for the following discussion, we distinguish between the two.

Both $\mathbb{F}[X_1, ..., X_m]$ and $\mathbb{F}^{\mathbb{F}^m}$ are vector spaces over $\mathbb{F}$, and $\Phi_{\mathbb{F}}$ is a linear map between them. In case $\mathbb{F}$ is an infinite field, $\Phi_{\mathbb{F}}$ is an injection (and not a surjection), and in case $\mathbb{F}$ is a finite field, it is a surjection (and not an injection). Let us elaborate more on the finite field case. Let $q = p^t$ be a prime power, and $\mathbb{F} = \mathbb{F}_q$ be the field with q elements, on which we shall focus. Let $(\mathbb{F}_q)_{\leq d}[X_1, ..., X_m]$ be the collection of individual degree at most d polynomials with m variables $X_1, ..., X_m$ and coefficients in $\mathbb{F}_q$ (Definition 5.14). A standard basis for these polynomials is the set of monomials $\mathcal{M}_d = \{X_1^{\alpha_1} \cdot ... \cdot X_m^{\alpha_m} \mid 0 \leq \alpha_i \leq d\}$. On the other hand, the functions $\mathbb{F}_q^{\mathbb{F}_q^m}$ also have a natural basis of indicators $\mathscr{I}$, namely for every $\vec{x} \in \mathbb{F}_q^m$ the indicator $\mathbf{1}_{\vec{x}} : \mathbb{F}_q^m \to \mathbb{F}_q$ defined by

$$\mathbf{1}_{\vec{x}}(\vec{y}) = \begin{cases} 1 & \vec{y} = \vec{x} , \\ 0 & \vec{y} \neq \vec{x} . \end{cases}$$

Every such indicator can be written as (the $\Phi_{\mathbb{F}_q}$-image of) an individual degree at most $q - 1$ polynomial using the (multivariate) Lagrange polynomial

$$\mathbf{1}_{\vec{x}}(X_1, ..., X_m) = (-1)^m \cdot \prod_{i=1}^{m} \prod_{x_i \neq a \in \mathbb{F}_q} (X_i - a) .$$

When restricted to individual degree at most $q - 1$ polynomials, the function $\Phi_{\mathbb{F}_q}$ is a bijection — it is a basis change on the polynomials, moving from the basis $\mathcal{M}$ to the basis $\mathscr{I}$. This basis change is an instance of a Fourier transform (and there is a very efficient algorithm that calculates it, called the *fast Fourier transform* [CT65, Gau86]).

Definition 5.16 (Subcubes in $\mathbb{F}_q^m$). Let $q = p^t$ be a prime power, and let $A \subseteq \mathbb{F}_q$ be a subset. We call the set $A^m \subseteq \mathbb{F}_q^m$ a *subcube*. If $A = \mathbb{F}_p \subseteq \mathbb{F}_q$, then we often call $\mathbb{F}_p^m$ *the subcube*, without referring to a specific A.

Definition 5.17. Let m, t be positive integers, p a prime number, and $q = p^t$. Given a function $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ we denote by $\mathrm{Res}(f) \colon \mathbb{F}_p^m \to \mathbb{F}_q$ the restriction of f to the subcube $\mathbb{F}_p^m \subseteq \mathbb{F}_q^m$, namely $f|_{\mathbb{F}_p^m}$. Given $g \colon \mathbb{F}_p^m \to \mathbb{F}_q$, let the induction of g, $\mathrm{Ind}(g) \colon \mathbb{F}_q^m \to \mathbb{F}_q$, be the individual degree at most $p - 1$ interpolation of g. Namely, $\mathrm{Ind}(g)$ is the unique m-variate individual degree at most $p - 1$ polynomial with coefficients in $\mathbb{F}_q$ that agrees with g.[75]

Remark 5.18. First, note that Ind is a $\mathbb{F}_q$-linear map. In addition, when $g \colon \mathbb{F}_p^m \to \mathbb{F}_p$, namely it outputs only elements in the base field $\mathbb{F}_p$ and not its extension $\mathbb{F}_q$, Ind is the composition of the following

$$\mathbb{F}_p^{\mathbb{F}_p^m} \xrightarrow{\Phi_{\mathbb{F}_p}^{-1}} (\mathbb{F}_p)_{\leq p-1}[\mathsf{X}_1, ..., \mathsf{X}_m] \subseteq (\mathbb{F}_q)_{\leq p-1}[\mathsf{X}_1, ..., \mathsf{X}_m] \xrightarrow{\Phi_{\mathbb{F}_q}} \mathbb{F}_q^{\mathbb{F}_q^m} \ .$$

Furthermore, it can be derived from this perspective that the coefficients of $\mathrm{Ind}(g)$ (as a polynomial, namely $\Phi_{\mathbb{F}_q}^{-1}(\mathrm{Ind}(g))$) are in $\mathbb{F}_p$.

Moreover, note that $\mathrm{Res} \circ \mathrm{Ind} = \mathrm{Id}$, and that $\mathrm{Ind} \circ \mathrm{Res}$ is the identity on m-variable individual degree $p - 1$ polynomials over $\mathbb{F}_q$. Given $f \colon \mathbb{F}_p^m \to \mathbb{F}_p$, $\mathrm{Ind}(f)$ is usually called the (individual degree $p - 1$) *Reed–Muller encoding* of f. This is an error correcting code (as was defined in Section 3.7.2) over $\mathbb{F}_q$ that has good distance (yet not that good of a rate).

Though it is standard (and proving it is not a hard exercise), we include the Schwartz–Zippel Lemma, which proves that the individual degree at most d Reed–Muller codes have good distance, as long as the individual degree d and number of variables m are sufficiently small compared to q:

Lemma 5.19 (Schwartz–Zippel [Sch80, Zip79]). *Let f be a non-zero **total** degree at most d polynomial over m variables with coefficients in $\mathbb{F}_q$. Then, the probability a uniformly random $u \in \mathbb{F}_q$ is a zero of f is bounded from above by $\frac{d}{q}$.*

The goal of low degree tests is to verify that a given function is (the $\Phi_{\mathbb{F}_q}$-image of) a low degree polynomial (in our context, an individual low degree polynomial, but the total degree case is also very useful and usually has better soundness parameters). The basic idea is the following: If a function $f \colon \mathbb{F}_q^n \to \mathbb{F}_q$ is an individual degree at most d polynomial, then restricting it to an axis parallel line of $\mathbb{F}_q^n$ (namely, fixing all the coordinates of the function except one of them) will result in a degree (at most) d univariate polynomial. It turns out that this property works in the other direction as well, and in a somewhat robust manner. Namely, given a function $f \colon \mathbb{F}_q^n \to \mathbb{F}_q$, if its restriction to all of its axis parallel lines is of degree at most d, then f is a polynomial of individual degree at most d. Let us define all of this formally.

Definition 5.20 (Lines). A line $\mathscr{L}$ in $\mathbb{F}_q^m$ is a 1-dimensional affine subspace. Namely, there are $u \in \mathbb{F}_q^m$ and $\vec{0} \neq v \in \mathbb{F}_q^m$ that induce a parametrization affine map $\alpha \mapsto u + \alpha v$ from $\mathbb{F}_q$ to $\mathbb{F}_q^m$, whose image is $\mathscr{L}$; this means $\mathscr{L} = \{u + \alpha v \mid \alpha \in \mathbb{F}_q\}$. Note that there are $q(q-1)$ many pairs u, v that give rise to the same line $\mathscr{L}$, and thus $q(q-1)$ many parametrizations of the same line. We denote by $\mathscr{L}(u, v)$ the *parametrized* line $\{u + \alpha v \mid \alpha \in \mathbb{F}_q\}$, and say that it is in direction v. An *i-axis parallel line* is one whose direction is e_i (or a scalar multiple thereof), and an axis parallel line is an i-axis parallel line for some $i \in [m]$.[76]

We record the following well-known fact, sometimes referred to as a *local characterization* of low individual degree m-variate polynomials [RS96].

Fact 5.21 (Characterizations of low degree polynomials). *The restriction of $f \in \mathbb{F}_q[\mathsf{X}_1, ..., \mathsf{X}_m]$ to the parametrized line $\mathscr{L}(u, v)$ is a polynomial $f|_{\mathscr{L}(u,v)} \in \mathbb{F}_q[\alpha]$ which is derived from f by the assignment $\mathsf{X}_i \mapsto u_i + \alpha v_i$.*

1. *f is of total degree at most d if and only if $f|_{\mathscr{L}(u,v)}$ has degree at most d for every line $\mathscr{L}(u, v)$.*

2. *f is of individual degree at most d if and only if $f|_{\mathscr{L}(u,e_i)}$ has degree at most d for every **axis parallel** line $\mathscr{L}(u, e_i)$.*

3. *Fix $i \in [m]$. Then, f has X_i-degree at most d if and only if $f|_{\mathscr{L}(u,e_i)}$ for every i-axis parallel line $\mathscr{L}(u, e_i)$.*

---

[75]Both restriction and induction depend on m, p and q. Yet, these parameters should be understood from context and are not included in the notation.

[76]We later discuss a (somewhat) canonical way of representing each line in $\mathbb{F}_q^m$. See Definition 5.75.

Definition 5.22 (The low individual degree test). Let d, t and m be positive integers, and $q = 2^t$. Let f and $\mathsf{AL}f$ (acronym for "axis parallel lines of f") be two functions. The input to f is a single $u \in \mathbb{F}_q^m$, and it outputs an element of $\mathbb{F}_q$. The input to $\mathsf{AL}f$ is a pair, consisting of $\hat{u} \in \mathbb{F}_q^{m-1}$ and $i \in [m]$, and it outputs a tuple $(c_0, c_1, ..., c_d)$ of $d+1$ elements from $\mathbb{F}_q$. Given $u \in \mathbb{F}_q^m$, let $\hat{u}^i$ be the $(m-1)$-tuple that results from removing the ith entry of u. The individual degree d test on f and $\mathsf{AL}f$ runs as follows: Sample a pair (u, i) uniformly at random, where u is a point in $\mathbb{F}_q^m$ and i is an axis direction, namely $i \in [m]$. Evaluate $f(u)$ and $\mathsf{AL}f(\hat{u}^i, i) = (c_0, ..., c_d)$. Accept if

$$f(u) = \sum_{j=0}^{d} c_j (u_i)^j, \tag{124}$$

and reject otherwise.

Note that by Fact 5.21, individual degree d polyonomials and their restrictions to axis parallel lines pass the above test with certainty. The following is a quantitative reverse statement, which is a "robust" version of Fact 5.21:

Theorem 5.23 (Classical soundness of the low individual degree test. Babai–Fortnow–Lund [BFL91][77]). *Let m, d, t, q, f and $\mathsf{AL}f$ be as in Definition 5.22, and let $\varepsilon \geq 0$. Assume the probability that $f, \mathsf{AL}f$ pass the individual degree d test is at least $1 - \varepsilon$. Namely*

$$\mathop{\mathbb{P}}_{u \in \mathbb{F}_q^m, i \in [m]} \left[f(u) \neq \sum_{j=0}^{d} c_j(u_i)^j \right] \leq \varepsilon ,$$

where $c_0, ..., c_d \in \mathbb{F}_q$ are the outputs of $\mathsf{AL}f(\hat{u}^i, i)$. Then, there exists an individual degree at most d polynomial $F \colon \mathbb{F}_q^n \to \mathbb{F}_q$ and a universal constant $C > 0$ such that

$$\mathop{\mathbb{P}}_{u \in \mathbb{F}_q^m} \left[f(u) \neq F(u) \right] \leq Cm^C \left(\varepsilon^{1/C} + \left(\frac{d}{q} \right)^{1/C} \right) .$$

Since we care about the time complexity of operations used in our protocols, and Turing machines manipulate bit strings, whenever we deal with a finite field we need to be able to do the arithmetic operations on it efficiently. The following fact guarantees this is possible (in the relevant case for us).

Fact 5.24 (See Section 3.3 in [JNV+21]). *For every odd positive integer t, there is a $\mathrm{poly}(t)$-time algorithm that chooses a basis of $\mathbb{F}_q$ over $\mathbb{F}_2$, where $q = 2^t$, such that the arithmetic operations (products, inverses, sums) and taking traces all take $\mathrm{poly}(t)$-time when the elements of $\mathbb{F}_q$ are represented according to this basis (namely, as elements of $\mathbb{F}_2^t$).*

One last "still to be motivated" definition is needed at this point:

Definition 5.25 (Zero on a subcube and Assignments). Let $\mathbb{F}$ be a finite field, and $A \subseteq \mathbb{F}$ a subset. A function $f \colon \mathbb{F}^m \to \mathbb{F}$ is said to be *zero on the subcube A^m*, if for every $u \in A^m$ the function evaluates to zero, namely $f(u) = 0$.

In case $\mathbb{F} = \mathbb{F}_q$ where $q = 2^t$, and $A = \mathbb{F}_2$, we often use the term f is zero on the subcube, without extra information. Under the same assumptions, a function $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is said to be an *assignment* if $f(\mathbb{F}_2^m) \subset \mathbb{F}_2$.

Claim 5.26 (Assignment condition). *Let $q = 2^t$. A function $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is an assignment, if and only if $f \cdot (f + 1)$ is zero on the subcube.*

Proof. This is immediate from the fact that the only zeros in $\mathbb{F}_q$ of the polynomial $\mathsf{X}(\mathsf{X} + 1)$ are the elements of $\mathbb{F}_2$. $\qquad\square$

---

[77]See also [PS94] and the introduction of [JNV+20].

Claim 5.27 (Combinatorial Nullstellensatz). *Let $q = 2^t$. The polynomial $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is zero on the subcube if and only if there are polynomials $c_i \colon \mathbb{F}_q^m \to \mathbb{F}_q$, often called* the helper polynomials, *such that*

$$f(\mathsf{X}_1, ..., \mathsf{X}_m) = \sum_{i=1}^m c_i(\mathsf{X}_1, ..., \mathsf{X}_m) \cdot (\mathsf{X}_i + 1)\mathsf{X}_i.$$

If $\vec{\mathsf{X}}$ is a tuple of variables indexed by a set I, and $i \in I$, then we denote

$$\mathrm{zero}_i(\vec{\mathsf{X}}) = (\mathsf{X}_i + 1)\mathsf{X}_i . \tag{125}$$

In addition, both the coefficients and the evaluation table of the helper polynomials c_i are linear in the coefficients (or evaluation table) of f, and the individual degree of the helper polynomials is smaller or equal to that of f.

Proof. This is a simple division of polynomials argument [JNV$^+$21, Proposition 10.21], which is a special case of *Combinatorial Nullstellensatz* [Alo99, Theorem 1.1].

Recall that $q = 2^t$. Fixing a univariate polynomial $f \in \mathbb{F}_q[\mathsf{X}_i]$, every multivariate polynomial $g \in \mathbb{F}_q[\mathsf{X}_1, ... \mathsf{X}_m]$ can be written in a unique way as $g = mf + r$, where the X_i-individual degree of r is strictly smaller than the degree of f. The maps $\mathrm{Div}_f(g)$ — which outputs the quotient polynomial m — and $\mathrm{Mod}_f(g)$ — which outputs the remainder polynomial r — are both $\mathbb{F}_q$-linear, and the individual degree of $\mathrm{Div}_f(g)$ is at most that of g. In particular, for every polynomial $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$, if we let

$$\forall i \in [m] : \quad c_i(\mathsf{X}_1, ..., \mathsf{X}_m) = \mathrm{Div}_{\mathsf{X}_i(\mathsf{X}_i+1)} \circ \mathrm{Mod}_{\mathsf{X}_{i-1}(\mathsf{X}_{i-1}+1)} \circ ... \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(f)$$

and

$$c_0(\mathsf{X}_1, ..., \mathsf{X}_m) = \mathrm{Mod}_{\mathsf{X}_m(\mathsf{X}_m+1)} \circ ... \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(f) ,$$

then (i) the coefficients of each c_i are linear combinations of the coefficients of f, (ii) for $i > 0$ the individual degree of each c_i is at most that of f, (iii) the total degree of $c_0(\mathsf{X}_1, ..., \mathsf{X}_m)$ is at most 1, and (iv) we have

$$\begin{aligned}
f(\mathsf{X}_1, ..., \mathsf{X}_m) &= c_0(\mathsf{X}_1, ..., \mathsf{X}_m) + \sum_{i=1}^m \mathsf{X}_i(\mathsf{X}_i + 1) \cdot c_i(\mathsf{X}_1, ..., \mathsf{X}_m) \\
&= c_0(\mathsf{X}_1, ..., \mathsf{X}_m) + \sum_{i=1}^m \mathrm{zero}_i(\mathsf{X}_1, ..., \mathsf{X}_m) \cdot c_i(\mathsf{X}_1, ..., \mathsf{X}_m) .
\end{aligned} \tag{126}$$

Note that for every $\vec{x} \in \mathbb{F}_2^m$, $\mathrm{zero}_i(\vec{x}) = x_i(x_i + 1) = 0$, as $x_i \in \mathbb{F}_2$. Hence,

$$\forall \vec{x} \in \mathbb{F}_2^m : \quad f(\vec{x}) = c_0(\vec{x}) .$$

So, if f is zero on the subcube, we deduce that $c_0(\vec{x}) = 0$ for every $\vec{x} \in \mathbb{F}_2^n$. As its total degree is at most 1, this implies c_0 is the zero polynomial, and this in this case

$$f(\mathsf{X}_1, ..., \mathsf{X}_m) = \sum_{i=1}^m \mathsf{X}_i(\mathsf{X}_i + 1) \cdot c_i(\mathsf{X}_1, ..., \mathsf{X}_m) . \tag{127}$$

as required. □

5.1.4 Circuits and Succinct-3SAT

The Turing machine is a *uniform* computational model, because a given Turing machine can in principle accept inputs of any given length. In contrast, circuits are a *non-uniform* model: a given circuit has a fixed number of input wires, which determine a unique input length that the circuit accepts. For this reason, in complexity one usually considers families of circuits $(\mathcal{C}_n)_{n \geq 1}$ indexed by a growing input length n; the model is called non-uniform because without any further restrictions, each circuit in the family can be quite different from any other (e.g. we do not necessarily require that $n \mapsto \mathcal{C}_n$ is an efficiently computable mapping).

Definition 5.28. A (binary, Boolean) *circuit* $\mathcal{C}$ is a finite, vertex labeled, directed and acyclic graph,[78] whose label set is

$$\{\neg, \oplus, \wedge, \text{Input}, \text{Output}, \text{Copy}, \text{True}, \text{False}\}. \tag{128}$$

In addition, the label of a vertex determines its in-degree (i.e., number of edges oriented into it) and out-degree (i.e., number of edges oriented out of it) according to the table below. See Figure 15 for visualization.

Label	in-degree	out-degree
Input	0	1
True	0	1
False	0	1
Output	1	0
Copy	1	2
$\neg$	1	1
$\wedge$	2	1
$\oplus$	2	1

The vertices of a circuit are often called *gates* and its edges are often called *wires*. Such a circuit is called *linear* if it has no vertices labeled by $\wedge$, i.e., no AND gates are used in it.

For later discussions, note that a circuit $\mathcal{C}$ can be encoded as bit string $\overline{n} \sqcup c_1 \sqcup ... \sqcup c_n$, where $\overline{n}$ is interpreted as the binary encoding of an integer n, which represents the number of gates in $\mathcal{C}$, and each c_i is (the encoding of) a tuple $(\text{Type}, \text{inwire } 1, \text{inwire } 2, \text{outwire } 1, \text{outwire } 2)$, where Type is one of the possible logic gates from (128), the two in-wires are integers between 1 and n that indicate the origins of the two wires that are fed into the gate (if there are less then 2 wires feeding into the gate, then the extra ones are ignored), and the two out-wires are integers between 1 and n that indicate the endpoints of the two wires stemming out of the gate (again, if there are less than 2 wires that stem out of the gate, the extra ones are ignored). This provides an encoding of $\mathcal{C}$ of size $O(n \log n)$, where n is the number of gates (vertices) in the circuit.

Let I (resp. O) be the set of vertices in $\mathcal{C}$ labeled by Input (resp. Output). Then $\mathcal{C}$ encodes a function $P_{\mathcal{C}} \colon \mathbb{F}_2^I \to \mathbb{F}_2^O$: Given $\iota \colon I \to \mathbb{F}_2$, write in each vertex $x \in I$ the value $\iota(x)$. Moreover, write in each True vertex 1 and in each False vertex 0. Then, repeat the following — for every vertex that contains a value, write down this value on all its outgoing edges; if there is a vertex all of whose in-going edges have values written on them, act as follows:

- if the vertex is labeled by Output, then it has one in-going edge; write in it the value appearing on this single edge;

- if the vertex is labeled by Copy, then it has one in-going edge; write in it the value appearing on this single edge;

- if the vertex is labeled by $\neg$, then it has one in-going edge; write in it the value appearing on this single edge plus 1 (in $\mathbb{F}_2$, namely, flip the bit);

- if the vertex is labeled by $\wedge$ then it has two in-going edges; write in it the product of the values on these two edges;

- if the vertex is labeled by $\oplus$ then it has two in-going edges; write in it the sum (in $\mathbb{F}_2$) of the values on these two edges;

To summarize, if a vertex has a single in-going edge e or two in-going edges e_1, e_2 then we assign it a value according to the following table:

Label	value at the vertex
Output	$\text{value}(e)$
Copy	$\text{value}(e)$
$\neg$	$\text{value}(e) + 1$
$\wedge$	$\text{value}(e_1) \cdot \text{value}(e_2)$
$\oplus$	$\text{value}(e_1) + \text{value}(e_2)$

---

[78] A directed graph is acyclic if there is no directed path that starts and ends at the same vertex.

If we write in each input vertex $x \in I$ a formal variable X_x, then each output vertex $y \in O$ will contain some polynomial (with $\mathbb{F}_2$-coefficients) $P_{\mathcal{C},y}$ in these variables — or equivalently, some Boolean formula in them. Note that a linear circuit induces total degree one polynomials, namely $P_{\mathcal{C}} \colon \mathbb{F}_2^I \to \mathbb{F}_2^O$ is an affine map in this case.

There is another way of associating a polynomial $T_{\mathcal{C},y}$ with each output vertex $y \in O$ of the circuit $\mathcal{C}$, which is called the Tseitin polynomial (or Tseitin formula): Associate a formal variable Y_e to each edge e in $\mathcal{C}$. For each vertex z with a directed path to y[79], denote the in-going edges of z by e or e_1, e_2 and its out-going edges by f or f_1, f_2. Then we define a polynomial t_z according to the following table.

Label of z	
Input	$t_z = 1$
True	$t_z = \mathsf{Y}_f$
False	$t_z = \mathsf{Y}_f + 1$
Output	$t_z = \mathsf{Y}_e$
Copy	$t_z = (\mathsf{Y}_e + \mathsf{Y}_{f_1} + 1)(\mathsf{Y}_e + \mathsf{Y}_{f_2} + 1)$
$\neg$	$t_z = \mathsf{Y}_e + \mathsf{Y}_f$
$\wedge$	$t_z = \mathsf{Y}_{e_1}\mathsf{Y}_{e_2} + \mathsf{Y}_f + 1$
$\oplus$	$t_z = \mathsf{Y}_{e_1} + \mathsf{Y}_{e_2} + \mathsf{Y}_f + 1$

Finally, let $T_{\mathcal{C},y} = \prod_z t_z$, where the product runs over all z with a directed path to y.

Remark 5.29. Note that the individual degree of the Tseitin polynomial is at most 3, namely there is no monomial and a variable Y_e whose exponent in this monomial is larger than 3. This is because when $e = xy$, the variable Y_e appears only in t_x and t_y, and its exponent in t_x is at most 1 while its exponent in t_y is at most 2.

As long as we consider functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2$, this is not a useful property, as any such function can be written as a polynomial of individual degree at most 1. But, we are going to view these polynomials over some finite field extension of $\mathbb{F}_2$, namely as functions from $\mathbb{F}_q^n$ to $\mathbb{F}_q$ for some $q = 2^t$, in which case this property of having low individual degree will become very handy.

Claim 5.30. *Let $\mathcal{C}$ be a circuit (Definition 5.28), y an output vertex in $\mathcal{C}$, $P_{\mathcal{C},y}$ the polynomial associated with y (over formal variables $\{\mathsf{X}_x\}_{x \in I}$), and $T_{\mathcal{C},y}$ the Tseitin polynomial associated with y (over formal variables $\{\mathsf{Y}_e\}_e$, where e is running over all edges in $\mathcal{C}$). Then:*

- *(Completeness) There is an assignment to the Y_e variables as polynomials in the X_x variables such that $T_{\mathcal{C},y}(\{\mathsf{Y}_e\}_e) = P_{\mathcal{C},y}(\{\mathsf{X}_x\}_x)$.*

- *(Soundness) If $P_{\mathcal{C},y}$ induces the constant 0 function, then so does $T_{\mathcal{C},y}$.*

Proof. For the completeness requirement, assign to the $\{\mathsf{Y}_e\}$ variables the following values inductively: For every e whose initial vertex x is labeled by Input, let $\mathsf{Y}_e = \mathsf{X}_x$. For e whose initial vertex is False, let $\mathsf{Y}_e = 0$, and for e whose initial vertex is True let $\mathsf{Y}_e = 1$. Then, if e is an edge whose initial vertex is x, and all of the edges whose terminal vertex is x were already assigned values, then assign to it — the input edge value if x is labeled Copy; the input edge value plus 1 if x is labeled $\neg$; the product of the input edges' values if x is labeled $\wedge$; the sum of the input edges' values if x is labeled $\oplus$. It is straightforward to check that indeed, under this assignment, $t_z(\mathsf{Y}_e) = 1$ for every non Output vertex z, and $t_y(\mathsf{Y}_e) = P_{\mathcal{C},y}(\mathsf{X}_x)$. Thus

$$T_{\mathcal{C},y}(\mathsf{Y}_e) = \prod t_z(\mathsf{Y}_e) = t_y(\mathsf{Y}_e) = P_{\mathcal{C},y}(\mathsf{X}_x).$$

For the soundness requirement, note that for $T_{\mathcal{C},y}(\mathsf{Y}_e)$ to be 1, all $t_z(\mathsf{Y}_e)$ need to be 1. But, if all the t_z's for which z is not labeled by Output evaluate to 1, then the Y_e's were assigned the values as in the complete case, which means $t_y(\mathsf{Y}_e) = P_{\mathcal{C},y}(\mathsf{X}_x)$, where $\mathsf{X}_x = \mathsf{Y}_{xz}$ for every $x \in I$ and xz its outgoing edge. But, by assumption, the output of $P_{\mathcal{C},y}$ is always 0, which implies that $T_{\mathcal{C},y}$ must evaluate to 0 as well. $\qquad\square$

---

[79]This property is commonly phrased as "y is reachable from z".

Figure 15: An example of a circuit $\mathcal{C}$ with three input gates, a single output gate, and 15 wires. If we denote by X_i the formal variable associated with Input_i, then the polynomial the circuit induces at the single output vertex is

$$P_{\mathcal{C}}(X_1, X_2, X_3) = 1 + X_1 + X_2 + X_3 + X_1X_2 + X_1X_2X_3 \, .$$

On the other hand, if we let Y_i be the formal variable associated with the edge e_i, then the Tseitin polynomial of this circuit is

$$\begin{aligned}
T_{\mathcal{C}}(Y_1, ..., Y_{15}) = &(Y_1 + Y_5 + 1)(Y_1 + Y_6 + 1)(Y_4 + Y_7 + 1)(Y_4 + Y_8 + 1) \\
&(Y_5 + Y_9)(Y_2Y_6 + Y_{10} + 1)(Y_3 + Y_7 + Y_{11} + 1)(Y_8 + Y_{12}) \\
&(Y_9 + Y_{11} + Y_{13} + 1)(Y_{10}Y_{12} + Y_{14} + 1)(Y_{13} + Y_{14} + Y_{15} + 1)Y_{15} \, .
\end{aligned}$$

Although the Tseitin polynomial $T_{\mathcal{C}}$ is "more complicated", it is guaranteed to have individual degree at most 3, while the circuit polynomial $P_{\mathcal{C}}$ may have arbitrarily large individual degree.

Definition 5.31 (Succinct encodings of 3CNF formulas). A circuit $\mathcal{C}$ with $3n + 3$ input gates and a single output is said to succinctly encode a 3CNF formula $\varphi_{\mathcal{C}}$ on 2^n variables $\{X_u\}_{u \in \mathbb{F}_2^n}$ if the following holds: $\varphi_{\mathcal{C}}$ is the conjunction of all formulas $X_{u_1}^{\varepsilon_1} \vee X_{u_2}^{\varepsilon_2} \vee X_{u_3}^{\varepsilon_3}$ for which $P_{\mathcal{C}}(u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3) = 1$.

The language Succinct-3SAT is the one containing all (encodings, as described in Definition 5.28, of) circuits $\mathcal{C}$ such that $\varphi_{\mathcal{C}}$ is satisfiable.

Theorem 5.32 (Scaled up Cook–Levin. Compare to Theorem 5.13. See also Section 10.2 in [JNV^{+}21]). *The language Succinct-*3SAT *is* NEXP*-complete.*

Proof ideas. Recall again that BinaryTimeHalt is NEXP-complete due to Claim 5.11, so it is enough to reduce it to Succinct-3SAT. Namely, given a TM $\mathcal{M}$ and integer T **in binary**, translate it in polynomial time to a circuit $\mathcal{C}$, such that if $\mathcal{M}$ has an input $\pi \in \{0, 1\}^*$ for which it halts and outputs 1 in T steps, then the 3CNF formula $\varphi_{\mathcal{C}}$ is satisfiable, and otherwise $\varphi_{\mathcal{C}}$ is unsatisfiable.

The idea is the same as the proof of Theorem 5.13. Namely, having formal variables that collect the tape contents, the head position, and the machine's internal state at each time step. Then, to add restrictions on the variables at time 0 so that they represent $\mathcal{M}$ in time 0 (with some input), and restrictions that check that the next time step variables were calculated correctly from the previous time according to the instructions table of $\mathcal{M}$. The important thing is that these restrictions and variables are so well structured, that one can encode them succinctly using a circuit of size $\mathrm{poly}(|\mathcal{M}|, \log T)$. $\square$

Observation 5.33. Let $\mathcal{C}$ be a circuit which succinctly encodes a formula $\varphi_{\mathcal{C}}$ on 2^n variables. An assignment to 2^n variables can be thought of as a function $\psi \colon \mathbb{F}_2^n \to \mathbb{F}_2$ by letting $\psi(u) = 1$ if X_u was assigned True and $\psi(u) = 0$ if it was assigned False. Now, $\mathsf{X}_{u_1}^{\varepsilon_1} \vee \mathsf{X}_{u_2}^{\varepsilon_2} \vee \mathsf{X}_{u_3}^{\varepsilon_3} = \text{True}$ if and only if $(\psi(u_1) + \varepsilon_1 + 1)(\psi(u_2) + \varepsilon_2 + 1)(\psi(u_1) + \varepsilon_3 + 1) = 0$. Therefore, $\varphi_{\mathcal{C}}$ is satisfiable if and only if there is a function $\psi \colon \mathbb{F}_2^n \to \mathbb{F}_2$ such that for all

$$(u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2 \times \mathbb{F}_2 \times \mathbb{F}_2 \,,$$

we have

$$P_{\mathcal{C}}(u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3)(\psi(u_1) + \varepsilon_1 + 1)(\psi(u_2) + \varepsilon_2 + 1)(\psi(u_3) + \varepsilon_3 + 1) = 0 \,.$$

Furthermore, by Claim 5.30, if s is the number of non-input wires (edges that do not stem from an Input-labeled vertex) in $\mathcal{C}$, then ψ satisfies the above condition if and only if for every

$$u = (u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3, z) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2 \times \mathbb{F}_2 \times \mathbb{F}_2 \times \mathbb{F}_2^s \,,$$

we have

$$T_{\mathcal{C}}(u)(\psi(u_1) + \varepsilon_1 + 1)(\psi(u_2) + \varepsilon_2 + 1)(\psi(u_3) + \varepsilon_3 + 1) = 0 \,. \tag{129}$$

Actually, by embedding this setup into a larger field of characteristic 2 via induction (Definition 5.17), we are able to get the following.

Proposition 5.34. *Let* $q = 2^t$ *for a positive integer* t. *Let* $\mathcal{C}$ *be a circuit that succinctly encodes a* 3CNF *formula* $\varphi_{\mathcal{C}}$ *(Definition 5.31), in particular, it has* $3n + 3$ *input gates, a single output gate, and we denote by* s *the number of non-input wires in it. Then:*

- *(Completeness) If* $\varphi_{\mathcal{C}}$ *is satisfiable, then there is a sequence of* $4n + 4 + s$ *individual degree at most* 6 *polynomials*

$$g_\psi \colon \mathbb{F}_q^n \to \mathbb{F}_q \,,$$
$$\forall 1 \le i \le 3n + 3 + s : \quad \alpha_i \colon \mathbb{F}_q^{3n+3+s} \to \mathbb{F}_q \,,$$
$$\forall 1 \le i \le n : \quad \beta_i \colon \mathbb{F}_q^n \to \mathbb{F}_q \,,$$

such that for every $u = (u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3, z) \in \mathbb{F}_q^{3n+3+s}$ *we have*

$$T_{\mathcal{C}}(u) \prod_{j=1}^{3} (g_\psi(u_j) + \varepsilon_j + 1) = \sum_{i=1}^{3n+3+s} \alpha_i(u)\mathrm{zero}_i(u) \,, \tag{130}$$

and for every $u_0 \in \mathbb{F}_q^n$ *we have*

$$g_\psi(u_0)(g_\psi(u_0) + 1) = \sum_{i=1}^{n} \beta_i(u_0)\mathrm{zero}_i(u_0), \tag{131}$$

where $\mathrm{zero}_i(\vec{\mathsf{X}}) = \mathsf{X}_i(\mathsf{X}_i + 1)$ *(as was defined in* (125)*).*

- *(Soundness) If there is a sequence of $4n + 4 + s$ individual degree at most 6 polynomials*

$$g_\psi \colon \mathbb{F}_q^n \to \mathbb{F}_q \, ,$$

$$\forall 1 \le i \le 3n + 3 + s \colon \quad \alpha_i \colon \mathbb{F}_q^{3n+3+s} \to \mathbb{F}_q \, ,$$

$$\forall 1 \le i \le n \colon \quad \beta_i \colon \mathbb{F}_q^n \to \mathbb{F}_q \, ,$$

such that

$$\mathop{\mathbb{P}}_{\substack{u \in \mathbb{F}_q^{3n+3+s} \\ u = (u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3, z)}} \left[T_{\mathcal{C}}(u) \prod_{j=1}^{3} (g_\psi(u_j) + \varepsilon_j + 1) = \sum_{i=1}^{3n+3+s} \alpha_i(u) \mathrm{zero}_i(u) \right] > \frac{21(3n + 3 + s)}{q} \, , \tag{132}$$

and

$$\mathop{\mathbb{P}}_{u_0 \in \mathbb{F}_q^n} \left[g_\psi(u_0)(g_\psi(u_0) + 1) = \sum_{i=1}^{n} \beta_i(u_0) \mathrm{zero}_i(u_0) \right] > \frac{21n}{q} \, , \tag{133}$$

then $\varphi_{\mathcal{C}}$ is satisfiable.

Proof. If $\varphi_{\mathcal{C}}$ is satisfiable, then by Observation 5.33, there is a function $\psi \colon \mathbb{F}_2^n \to \mathbb{F}_2$ that satisfies (129). Let $g_\psi = \mathrm{Ind}(\psi) \colon \mathbb{F}_q^n \to \mathbb{F}_q$, which has individual degree at most 1, and let $\Psi \colon \mathbb{F}_q^{3n+3+s} \to \mathbb{F}_q$ be

$$\Psi(u) = T_{\mathcal{C}}(u) \prod_{j=1}^{3} (g_\psi(u_j) + \varepsilon_j + 1),$$

which has individual degree at most 6 — as $T_{\mathcal{C}}$ has individual degree at most 3 (Remark 5.29). Since ψ satisfies (129), Ψ is zero on the subcube $\mathbb{F}_2^{3n+3+s}$ (Definition 5.25), and by the Combinatorial Nullstellensatz (Claim 5.27), there are $3n + 3 + s$ individual degree at most 6 helper polynomials $\alpha_i \colon \mathbb{F}_q^{3n+3+s} \to \mathbb{F}_q$ such that

$$\forall u = (u_1, u_2, u_3, \varepsilon_1, \varepsilon_2, \varepsilon_3, z) \in \mathbb{F}_q^{3n+3+s} \colon \quad T_{\mathcal{C}}(u) \prod_{j=1}^{3} (g_\psi(u_j) + \varepsilon_j + 1) = \sum_{i=1}^{3n+3+s} \alpha_i(u) \mathrm{zero}_i(u) \, .$$

As $\mathrm{Im}(\psi) \subseteq \mathbb{F}_2$, and $g_\psi = \mathrm{Ind}(\psi)$, g_ψ is an assignment (Definition 5.25). Therefore, by Claim 5.26, the polynomial $g_\psi(g_\psi + 1)$ is zero on the subcube $\mathbb{F}_2^n$ (and has individual degree at most 2). Thus, again by the Combinatorial Nullstellensatz, there are n individual degree at most 2 polynomials $\beta_i \colon \mathbb{F}_q^n \to \mathbb{F}_q$ such that

$$\forall u_0 \in \mathbb{F}_q^n \colon \quad g_\psi(u_0)(g_\psi(u_0) + 1) = \sum_{i=1}^{n} \beta_i(u_0) \mathrm{zero}_i(u_0) \, ,$$

which finishes the proof of completeness.

On the other hand, assume there are g_ψ, α_i and β_i of individual degree at most 6 that satisfy (132) and (133). By the assumptions on the individual degree of $g_\psi, \alpha_i, \beta_i$ and the fact that $T_{\mathcal{C}}$ has individual degree at most 3, the polynomials

$$\heartsuit(u) = T_{\mathcal{C}}(u) \prod_{j=1}^{3} (g_\psi(u_j) + \varepsilon_j + 1) - \sum_{i=1}^{3n+3+s} \alpha_i(u) \mathrm{zero}_i(u)$$

and

$$\clubsuit(u_0) = g_\psi(u_0)(g_\psi(u_0) + 1) - \sum_{i=1}^{n} \beta_i(u_0) \mathrm{zero}_i(u_0)$$

144

have individual degree at most 21. Hence, $\heartsuit$ has total degree at most $21(3n + 3 + s)$ and $\clubsuit$ has total degree at most $21n$. Equation (132) says that a uniformly random vector in $\mathbb{F}_q^{3n+3+s}$ is a zero of $\heartsuit$ with probability greater than $\frac{21(3n+3+s)}{q}$, which combined with the Schwartz–Zippel Lemma 5.19 implies that $\heartsuit$ is the zero function. The same argument, using (133), implies $\clubsuit$ is the zero function. The fact $\clubsuit$ is the zero function implies that g_ψ is an assignment, and we recover a potential boolean assignment $\psi = \text{Res}(g_\psi) \colon \mathbb{F}_2^n \to \mathbb{F}_2$ to $\varphi_{\mathcal{C}}$. The fact $\heartsuit$ is the zero function implies that the ψ recovered from g_ψ satisfies (129), and by Observation 5.33, $\varphi_{\mathcal{C}}$ is satisfied by ψ. In particular $\varphi_{\mathcal{C}}$ is satisfiable in this case. $\qquad\square$

5.1.5 Probabilistically checkable proofs

Recall that a (promise) language L is in NP (resp. NEXP), if for every $\mathbf{x} \in L_{yes}$ there is a polynomial sized (resp. exponential sized) proof π, such that the polynomial time (resp. exponential time) verifier $\mathcal{V}$ will be convinced by π that $\mathbf{x} \in L_{yes}$, and for every $\mathbf{x} \in L_{no}$ no proof π would convince $\mathcal{V}$ that $\mathbf{x} \in L_{yes}$. The goal of probabilistically checkable proofs (PCPs) is to enable the verifier to read only a **small yet random** part of the proof, and still be able to distinguish with high probability between the cases where $\mathbf{x} \in L_{yes}$ and where $\mathbf{x} \in L_{no}$.

A *black-box function* (also known as an *oracle*) $\pi \colon \{0,1\}^* \to \{0,1\}^*$ is a function that a TM can interact with as follows: The TM can, as part of its operation, send to π an input $\mathbf{x}$, and π outputs (in a single time step) the value $\pi(\mathbf{x})$ — such an interaction is called a *query* to the black-box function.

Definition 5.35 (PCP). Let $f, r, qu \colon \mathbb{N} \to \mathbb{N}$ be functions. A language L is in $\text{PCP}(f(n), r(n), qu(n))$ if there is a 1-input probabilistic TM $\mathcal{V}$ such that

- (Time bound) $\mathbb{T}(\mathcal{V}, \overline{n}) \le f(|\overline{n}|)$, namely $\mathcal{V}$ is f-time;

- (Randomness bound) $\mathcal{V}(\overline{n})$ uses at most $r(|\overline{n}|)$ many random bits — these bits, together with $\overline{n}$, will determine what positions of the black-box function $\mathcal{V}$ will query;

- (Completeness) if $\overline{n} \in L_{yes}$, then there exists a black-box function π that $\mathcal{V}$ queries at most $qu(|\overline{n}|)$-many times, and always decides to accept.

- (Soundness) if $\overline{n} \in L_{no}$, then for every black-box function π that $\mathcal{V}$ queries at most $qu(|\overline{n}|)$-many times, the probability that $\mathcal{V}$ accepts is bounded from above by $1/2$.

We denote by PCP the union of the classes $\text{PCP}(f(n), r(n), qu(n))$ overall all polynomials f, r and qu.

Remark 5.36 (Dramatization of PCP). Again, a polynomial time verifier $\mathcal{V}$ wants to decide whether $\mathbf{x} \in L_{yes}$. It sends $\mathbf{x}$ to a prover $\mathcal{P}$. The prover then generates a black-box function π that $\mathcal{V}$ can interact with. $\mathcal{V}$ reads some random bits, and according to them queries π at several locations. According to the outputs of π, $\mathcal{V}$ needs to decide whether to accept or reject.

The black-box function should be thought of as a proof to the claim $\mathbf{x} \in L_{yes}$. If r is a polynomial, the length of this proof is at most exponential in $|\mathbf{x}|$. But, if qu is polynomial, $\mathcal{V}$ reads only a logarithmic part of the proof. This indeed means that $\mathcal{V}$ does not have enough time to be convinced with certainty that the proof is correct. The point is that the verifier can ask the prover to format the proof in such a way that even this logarithmically sized view of the proof will enable it to reject it with constant probability $1/2$ in case the claim is wrong (namely, when $\mathbf{x} \in L_{no}$).

Remark 5.37. For a nice historical survey of this field, and specifically of PCPs, see [GO05].

We are ready to describe the PCP protocol for the language Succinct-3SAT, which is the main step towards showing that $\text{MIP} = \text{NEXP}$.

Observation 5.38 (Succinct-3SAT is in PCP). Let $\mathcal{C}$ be the instance received as input, namely it is (the encoding, as described in Definition 5.28, of) a circuit that succinctly encodes a 3CNF formula $\varphi_{\mathcal{C}}$ (Definition 5.31). In particular, it has $3n + 3$ input gates, a single output gate, and let s be the number of non-input wires of $\mathcal{C}$.

Then, the verifier $\mathcal{V}$ in the PCP protocol, which gets as input both $\mathcal{C}$ and a black box function π, acts as follows.

1. First, $\mathcal{V}$ uses (the encoding of) $\mathcal{C}$ to recover the integers n and s — this can be done in time linear in the encoding length of $\mathcal{C}$. It then chooses an odd positive integer t, according to a rule that we describe later, lets $q = 2^t$, and fixes a basis of $\mathbb{F}_q$ over $\mathbb{F}_2$ (a la Fact 5.24). Thus, the notions of an $\mathbb{F}_q$-input and an $\mathbb{F}_q$-output are well defined, as every element of $\mathbb{F}_q$ has now a fixed encoding as an element of $\mathbb{F}_2^t$.

2. Then, $\mathcal{V}$ expects π to be the evaluation table of functions $g_\psi \colon \mathbb{F}_q^n \to \mathbb{F}_q$, $\alpha_i \colon \mathbb{F}_q^{3n+3+s} \to \mathbb{F}_q$ and $\beta_i \colon \mathbb{F}_q^n \to \mathbb{F}_q$, as well as functions $\mathsf{AL}g_\psi \colon \mathbb{F}_q^{n-1} \times [n] \to \mathbb{F}_q^7$, $\mathsf{AL}\alpha_i \colon \mathbb{F}_q^{3n+2+s} \times [3n+3+s] \to \mathbb{F}_q^7$ and $\mathsf{AL}\beta_i \colon \mathbb{F}_q^{n-1} \times [n] \to \mathbb{F}_q^7$.[80]

   The range of the $\mathsf{AL}\square$ functions is chosen to be 7-dimensional so that their output encodes a degree 6 univariate polynomial over $\mathbb{F}_q$. All in all, $\mathcal{V}$ expects $g_\psi, \alpha_i, \beta_i$ to be individual degree at most 6 polynomials as needed for Proposition 5.34, and $\mathsf{AL}g_\psi, \mathsf{AL}\alpha_i, \mathsf{AL}\beta_i$ be their respective restrictions to axis parallel lines (which must be univariate polyonimals of degree 6 according to Fact 5.21).

3. The verifier $\mathcal{V}$ runs r many — where the procedure for choosing r will be described later — independent rounds of the (classical) individual degree at most 6 test (Definition 5.22) on each of the $4n + 4 + s$ pairs

$$(g_\psi, \mathsf{AL}g_\psi), (\alpha_i, \mathsf{AL}\alpha_i), (\beta_i, \mathsf{AL}\beta_i) .$$

   If **any** of these rounds has rejected, then $\mathcal{V}$ rejects.

4. If all of the low-degree test rounds have accepted, then $\mathcal{V}$ samples two additional points $u_0 \in \mathbb{F}_q^n$ and $u \in \mathbb{F}_q^{3n+3+s}$, each of which uniformly at random, and asks π to send the values

$$g_\psi(u_0), g_\psi(u_1), g_\psi(u_2), g_\psi(u_3), \alpha_i(u), \beta_i(u_0) . \tag{134}$$

   It then evaluates $T_\mathcal{C}(u)$ on its own using $\mathcal{C}$. Finally, it verifies that (130) and (131) are satisfied. If so, then it accepts, and otherwise it rejects.

As Proposition 5.34 shows, if $\varphi_\mathcal{C}$ is satisfiable, then a tuple of individual degree at most 6 polynomials $g_\psi, \alpha_i, \beta_i$ that always pass (130) and (131) exists. If we define $\mathsf{AL}g_\psi, \mathsf{AL}\alpha_i, \mathsf{AL}\beta_i$ according to their restrictions to axis parallel lines, then as the original polynomials were of low degree, according to Fact 5.21, the pairs $(g_\psi, \mathsf{AL}g_\psi), (\alpha_i, \mathsf{AL}\alpha_i), (\beta_i, \mathsf{AL}\beta_i)$ pass the individual degree at most 6 test with certainty. Thus, if the prover chooses π such that it consists of these specific functions, then it passes the above protocol with certainty. Namely, the protocol is complete.

Assume π consisting of functions $g_\psi, \mathsf{AL}g_\psi, \alpha_i, \mathsf{AL}\alpha_i, \beta_i, \mathsf{AL}\beta_i$ passes the above protocol with probability strictly larger than $1/2$. We are going to show that, under an appropriate choice of t and r, this implies $\varphi_\mathcal{C}$ is satisfiable, proving the soundness of the protocol. To pass the entire protocol with probability of at least $1/2$, functions in the proof π need to pass the r rounds of individual degree at most 6 test with probability of at least $1/2$, which implies each pair of $(g_\psi, \mathsf{AL}g_\psi), (\alpha_i, \mathsf{AL}\alpha_i), (\beta_i, \mathsf{AL}\beta_i)$ passes the single round of the low-degree test with probability of at least $(1/2)^{1/r}$, and as r is a positive integer we have the inequalities $(1/2)^{1/r} \geq (1/e)^{1/r} = e^{-1/r} \geq 1 - \frac{1}{r}$, where the last one is Bernoulli's inequality. Using the classical soundness of the low degree test (Theorem 5.23) on each pair $(\square, \mathsf{AL}\square)$ as above, there are individual degree at most 6 polynomials $\widetilde{g}_\psi, \widetilde{\alpha}_i, \widetilde{\beta}_i$ such that

$$\max \left\{ \mathbb{P}[g_\psi(u) \neq \widetilde{g}_\psi(u)] \, , \, \mathbb{P}[\alpha_i(u) \neq \widetilde{\alpha}_i(u)] \, , \, \mathbb{P}[\beta_i(u) \neq \widetilde{\beta}_i(u)] \right\} \leq C(3n+3+s)^C \left(\frac{1}{r^{1/C}} + \frac{6}{q^{1/C}} \right) , \tag{135}$$

where $C \geq 0$ is a universal constant independent of everything.

---

[80]In general, a single black box function can encode any sequence of black box functions. E.g., in our context, $\mathcal{V}$ can verify that π is structured that way by sending to it tuples of the form $(\text{Name}, \text{Input})$ where Name is one of $g_\psi, \mathsf{AL}g_\psi, \alpha_i, \mathsf{AL}\alpha_i, \beta_i, \mathsf{AL}\beta_i$, and Input an appropriate input to the function, and seeing that indeed the outputs are from $\mathbb{F}_q$ or $\mathbb{F}_q^7$ respectively.

Coming back to π, for it to pass the protocol with probability of at least $1/2$, it needs to pass the last check with this probability, namely

$$\mathbb{P}\left[T_{\mathcal{C}}(u)\prod_{j=1}^{3}(g_\psi(u_j)+\varepsilon_j+1)=\sum\alpha_i(u)\text{zero}_i(u)\right],\ \mathbb{P}\left[g_\psi(u_0)(g_\psi(u_0)+1)=\sum\beta_i(u_0)\text{zero}_i(u_0)\right]>\frac{1}{2}.$$

These are the exact expressions as in (132) and (133), which guarantee that $\varphi_{\mathcal{C}}$ is satisfiable according to Proposition 5.34. Alas, we do not know that these functions are individual degree at most 6 polynomials, which is needed for the soundness condition of the proposition to hold. But using a union bound and (135), we can deduce that

$$\mathbb{P}\left[T_{\mathcal{C}}(u)\prod_{j=1}^{3}(\widetilde{g}_\psi(u_j)+\varepsilon_j+1)=\sum\widetilde{\alpha}_i(u)\text{zero}_i(u)\right]\geq \mathbb{P}\left[T_{\mathcal{C}}(u)\prod_{j=1}^{3}(g_\psi(u_j)+\varepsilon_j+1)=\sum\alpha_i(u)\text{zero}_i(u)\right]$$

$$-\sum_{j=1}^{3}\mathbb{P}[\widetilde{g}_\psi(u_j)\neq g_\psi(u_j)]-\sum_{i=1}^{3n+3+s}\mathbb{P}[\widetilde{\alpha}_i(u)\neq\alpha_i(u)]$$

$$>\frac{1}{2}-(3n+6+s)\cdot C(3n+3+s)^C\left(\frac{1}{r^{1/C}}+\frac{6}{q^{1/C}}\right),$$

and similarly

$$\mathbb{P}\left[\widetilde{g}_\psi(u_0)(\widetilde{g}_\psi(u_0)+1)=\sum\widetilde{\beta}_i(u_0)\text{zero}_i(u_0)\right]>\frac{1}{2}-(2+n)C(3n+3+s)^C\left(\frac{1}{r^{1/C}}+\frac{6}{q^{1/C}}\right).$$

Hence, if we choose r and t such that

$$\frac{1}{2}-(3n+6+s)\cdot C(3n+3+s)^C\left(\frac{1}{r^{1/C}}+\frac{6}{q^{1/C}}\right)\geq\frac{21(3n+3+s)}{q},$$

and

$$\frac{1}{2}-(2+n)C(3n+3+s)^C\left(\frac{1}{r^{1/C}}+\frac{6}{q^{1/C}}\right)\geq\frac{21n}{q}$$

then $\varphi_{\mathcal{C}}$ is indeed satisfiable due to the soundness condition of Proposition 5.34. By choosing, for example, $r=(12C)^C(3n+3+s)^{3C^2}$ and the smallest odd t for which $q=2^t\geq(72C)^C(3n+3+s)^{3C^2}$, the above is satisfied and soundness of the protocol is proved.

We leave for the reader to check that indeed the number of random bits used in this protocol, the number of queries to the proof π, and the running time of it are all polynomial in n and s, which in turn means they are polynomial in the input length (since it bounds both of these numbers from above), as needed.

5.1.6 What is missing for MIP = NEXP?

The above protocol assumed that the prover first fixed a function π, and only then the verifier queried it. In MIP (see Remark 5.7), the provers see the questions before they commit to a certain answer. The way to overcome this is for one prover to provide all the needed values in the above protocol, and for the second prover to play a cross-checking role. Namely, the second prover gets just one of the functions and evaluation points, and its answers are checked to be consistent with the first prover. This already demonstrates that one prover should be able to provide "the whole proof", which in terms of the underlying games will require oracularization — namely, for one player to get both questions, and for the other player to get one of the original questions. See Section 5.2.2 for more on that.

The other key component which is not clear in the two provers scenario is the parallel repetition part, namely step 3 in Observation 5.38. In that step, the verifier executes r "independent rounds" of a certain test. In our sketch for the soundness

analysis we used that such repeated checking increases the likelihood of finding an error exponentially. However, in the setting of a two-prover interactive proof, all questions to each prover are sent simultaneously. By sending the prover multiple "independent" questions at once, the verifier potentially allows them to answer each individual question in a way that depends on the entire tuple, thus putting in question the exponential error improvement assumed above. It turns out that this is an intricate issue, which cannot be waved away by arguing using "without loss of generality..." type of arguments. In the case of classical two-prover interactive proofs, this problem was resolved by the celebrated parallel repetition theorem of Raz [Raz95]. There is no general parallel repetition theorem for quantum strategies, but there is a "good enough" version [BVY17] that assumes some simple transformation on the given game, called *anchoring*, was applied before the repetition — see Section 6 for more on that.

5.2 Prerequisites to Answer Reduction: Purification, Oracularization, Triangulation and Decoupling

Before we can apply PCP techniques to reduce the answer lengths of the tailored normal form verifier in mind, we need to define four transformations: *purification*, *oracularization*, *triangulation* and *Decoupling*. The first will be applied to the verifier before answer reduction, and the rest are incorporated in the answer reduction transformation itself.

A game is said to be purely unreadable if the controlled linear constraints function $L_{\mathtt{xy}}$ outputs only constraints on the unreadable variables. One could have defined a tailored non-local game to satisfy this property without hindering the expressiveness of the model; although this would have induced some complications to the presentation and description of games, hence we kept the looser definition. To purify, all one needs to note is that the values for the readable variables are known before choosing the controlled system of equations, so one can assign to the readable variables their already known values, which makes them non-variables, i.e. part of the constants in the equations.

The idea behind oracularization is simple — instead of sampling a pair of questions $\mathtt{x}, \mathtt{y}$ and sending a single question to each player, the oracularized version samples the same pair of questions, but sends one of the players both $\mathtt{x}$ and $\mathtt{y}$, and the other either $\mathtt{x}$ or $\mathtt{y}$. On the level of the underlying graph of the game, this boils down to a barycentric sub-division of the graph. Though this seems to be a naive transformation, and indeed in the classical setup it is, in the quantum setup the completeness of this transformation is dependent on the capability of one player to always measure the needed values along every edge, which is exactly the *commuting along edges* condition from the definition of a ZPC strategy.

Regarding triangulation, again, the transformation on the level of linear systems of equations is straightforward: Every equation which involves k variables is transformed into $k+1$ equations that involve only three variables, by inductively defining a new variable at each step alongside the constraint that the sum of two of the variables currently appearing in the equation should equal the new variable. This is a standard trick, used even in the non-commutative context, e.g. when showing that every group has a presentation with relations of length at most 3 (cf. [Żuk03]). Though this is a straightforward transformation, similar to PCPs, it requires one of the parties to know what the system of equations is.[81] This makes it natural to apply triangulation in tandem with oracularization.

Finally, decoupling is a method of transforming in a complete and sound way a triangulated system of linear equations into a system of linear equations whose variables come from 5 blocks, and each equation in the new system contains at most one variable from each block. The same kind of transformation can be applied for 3CNF formulas, and is essentially baked into the version of the scaled up Cook–Levin transformation that we use in Proposition 5.62.

5.2.1 Purification

Definition 5.39. A system of linear equations $\mathscr{A}\vec{S} = \vec{b}$ over a field $\mathbb{F}$, where

$$S = S_{\mathtt{x}}^{\mathfrak{R}} \cup S_{\mathtt{x}}^{\mathfrak{L}} \cup S_{\mathtt{y}}^{\mathfrak{R}} \cup S_{\mathtt{y}}^{\mathfrak{L}},$$

(as is the case in the controlled linear constraint systems of tailored games), is said to be *purely unreadable*, or *purely linear*, if the columns of $\mathscr{A}$ associated with $S_{\mathtt{x}}^{\mathfrak{R}} \cup S_{\mathtt{y}}^{\mathfrak{R}}$ are all zero. This is the same as saying that no constraint is applied on the

---

[81]There are ways to avoid this assumption under some reasonable bounds on the degree in the underlying graph of the game, but the transformation is somewhat more complicated in this case.

readable variables.

The idea in purification is to assign to the readable variables in the controlled linear constraints of a tailored non-local game their already assigned values, and thereby change them from variables to constants, which means their coefficients can be assumed to be zero.

Definition 5.40 (Combinatorial purification). Let $\mathfrak{G}$ be a tailored game. Define the purification $\mathfrak{G}' = \mathfrak{Pure}(\mathfrak{G})$ of $\mathfrak{G}$ as follows: It has the same underlying graph and distribution along edges as $\mathfrak{G}$. In addition, it has the same length functions and the same formal variable sets. Recall that the controlled linear constraint function is $L_{xy} \colon \mathbb{F}_2^{S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}}} \to \mathbb{F}_2^{S_{xy} \cup \{J\}}$. If $\gamma^{\mathfrak{R}} \colon S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}} \to \mathbb{F}_2$ is the readable variables assignment, then for every constraint $c \colon S_{xy} \cup \{J\} \to \mathbb{F}_2$ in $L_{xy}(\gamma^{\mathfrak{R}})$, $L_{xy}'(\gamma^{\mathfrak{R}})$ will contain the constraint $c' \colon S_{xy} \cup \{J\} \to \mathbb{F}_2$ defined by:

$$
\begin{aligned}
\forall X \in S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}} : \quad & c'(X) = 0, \\
\forall X \in S_x^{\mathfrak{L}} \cup S_y^{\mathfrak{L}} : \quad & c'(X) = c(X), \\
& c'(J) = c(J) + \sum_{X \in S_x^{\mathfrak{R}} \cup S_y^{\mathfrak{R}}} c(X) \gamma^{\mathfrak{R}}(X).
\end{aligned}
\tag{136}
$$

Fact 5.41 (Completeness and soundness of purification). *As the underlying graph and length functions of $\mathfrak{G}$ and $\mathfrak{Pure}(\mathfrak{G})$ are the same, there is a one to one correspondence between the quantum strategies for them. This correspondence is value preserving.*

Claim 5.42 (Algorithmic purification). *There is a polynomial time TM* Purify *that takes as input a tailored (typed or non-typed) h-level normal form verifier $\mathcal{V}$, and outputs a tailored (typed or non-typed) h-level normal form verifier $\mathcal{V}'$, such that:*

- Combinatorial purification: *If $\mathcal{V}_n$ is well defined, then $\mathcal{V}_n'$ is well defined and satisfies $\mathcal{V}_n' = \mathfrak{Pure}(\mathcal{V}_n)$.*

- Running times and description lengths: *The sampler and answer length remain the same (and so their running times and description lengths are the same). Moreover,*

$$
\mathbb{T}(\mathcal{L}'; n, \cdot, \cdot, \cdot, \cdot) = O\big(\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) + \mathbb{T}(\mathcal{A}; n, \cdot, \cdot)\big),
$$

and the description length of $\mathcal{L}'$ is linear in that of $\mathcal{L}$.

Proof. We only need to describe the linear constraints processor. Given $(n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ (the typed version is similar), $\mathcal{L}'$ first runs $\mathcal{L}(n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ to obtain $(c^1, ..., c^k)$, and calculates $\ell_{\cdot}$ using $|\mathrm{dec}(\mathcal{A}(n, \cdot, \cdot))|$. Then, for every i, it replaces c^i with c'^i defined as in (136), which is possible as the positions in c^i associated to $S_{\cdot}$ can be deduced from the values $\ell_{\cdot}$ previously calculated. $\qquad\square$

5.2.2 Oracularization

Definition 5.43 (Combinatorial Oracularization). Let $\mathfrak{G}$ be a tailored non local game. The *oracularization* of $\mathfrak{G}$, $\mathfrak{G}' = \mathfrak{Oracle}(\mathfrak{G})$, is defined as follows. If $G = (V, E)$ was the underlying graph of $\mathfrak{G}$, then the underlying graph of the oracularization is the barycentric subdivision of G, namely $G' = (V', E')$, where $V' = V \sqcup E$, and for $x \in V$ and $e \in E$, $(x, e) \in E'$ if and only if x was one of the endpoints of e. The vertices of G' coming from E are called *oracle player* questions, while those that come from V are called *isolated player* questions. Regarding the length functions $\ell^{\mathfrak{R}}, \ell^{\mathfrak{L}}$, they remain the same on V, and extended to E as follows — for $e = xy$, $\ell^{\cdot}(e) = \ell^{\cdot}(x) + \ell^{\cdot}(y)$. For the distribution over questions, if $(x, e) \in E'$, then $\mu'(x, e) = \mu(e)/2$ — or in words, a pair of questions is sampled as before, both of which are sent to an oracle player, and then one of the other two is sent uniformly to the isolated player. For the formal sets of variables, we leave those on isolated vertices as they were before, and if $e = xy$ is an edge with $S_x = \{X^{\kappa, j} \mid \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, j \in [\ell^{\kappa}(x)]\}$ and

$S_{\mathsf{y}} = \{\mathsf{Y}^{\kappa,j} \mid \kappa \in \{\mathfrak{R},\mathfrak{L}\}, j \in [\ell^{\kappa}(\mathsf{y})]\}$, then $S_e = S_{\mathsf{x}\in e} \sqcup S_{\mathsf{y}\in e} = \{\mathsf{OX}_e^{\cdot,\cdot}, \mathsf{OY}_e^{\cdot,\cdot}\}$. Finally, the controlled linear constraints function L' acts as follows: Assume $(\mathsf{x}, e = \mathsf{xy})$ was sampled, and that $a^{\mathfrak{R}}$ was the assignment to $S_{\mathsf{x}}^{\mathfrak{R}}$ while $a_e^{\mathfrak{R}}, b_e^{\mathfrak{R}}$ was the assignment to $S_e^{\mathfrak{R}}$. Then, L' first outputs the following $\ell(\mathsf{x})$ equations:

$$\mathsf{OX}_e^{\kappa,j} = \mathsf{X}^{\kappa,j} . \tag{137}$$

In addition, it outputs the same linear equations as $L_{\mathsf{xy}}(a_e^{\mathfrak{R}}, b_e^{\mathfrak{R}})$, but on the S_e variables instead of the $S_{\mathsf{x}} \cup S_{\mathsf{y}}$. Namely, if $\sum \alpha_{\kappa,j} \mathsf{X}^{\kappa,j} + \sum \beta_{\kappa,j} \mathsf{Y}^{\kappa,j} = b$ was an equation output by L, then L' will output the equation

$$\sum_{\kappa,j} \alpha_{\kappa,j} \mathsf{OX}_e^{\kappa,j} + \sum_{\kappa,j} \beta_{\kappa,j} \mathsf{OY}_e^{\kappa,j} = b . \tag{138}$$

Remark 5.44. The oracularized game has a simple description: Sample a pair of questions as before, send both of them to an oracle player — and expect it to reply with the appropriate answer to both questions – and send only one of them to the isolated player. The answer of the oracle player should be accepted by the original game, while the answer of the isolated player should be consistent with the appropriate part of the answer of the oracle player.

Remark 5.45 (The pure part of an oracularized pure game)**.** Note that by oracularizing a purely unreadable game, you get a game which is not pure, but the equations (138) are purely unreadable. This will be sufficient for answer reduction to work.

Claim 5.46 (Completeness and soundness of the oracularized game)**.** *Let $\mathfrak{G}$ be a tailored non-local game. Then,*

- *(Completeness): If $\mathfrak{G}$ has a perfect ZPC strategy, then so does $\mathfrak{Oracle}(\mathfrak{G})$.*

- *(Soundness): If $\mathfrak{Oracle}(\mathfrak{G})$ has a value $1 - \varepsilon$ strategy, then $\mathfrak{G}$ has a value $1 - 12\varepsilon$ strategy of the same dimension, and $\mathscr{E}(\mathfrak{Oracle}(\mathfrak{G}), 1 - \varepsilon) \geq \mathscr{E}(\mathfrak{G}, 1 - 12\varepsilon)$.*

Proof. For completeness, assume $\mathfrak{G}$ has a perfect ZPC strategy $\mathscr{S} = \{\mathcal{U}\}$. Then, we can extend $\mathscr{S}$ to the variables S_e at oracle player vertices e in the straightforward manner $\mathcal{U}(\mathsf{OX}_e^{\kappa,j}) = \mathcal{U}(\mathsf{X}^{\kappa,j})$. As $\mathscr{S}$ is commuting along edges, the observables at the oracle player vertices are commuting, which means this extension is a well-defined quantum strategy for $\mathfrak{Oracle}(\mathfrak{G})$ (Definition 2.18) — this is a crucial point, and is the only reason the "commuting along edges" condition is always included in the completeness argument. In addition, once the observables at oracle vertices are commuting and are consistent with the isolated players' observables, the extended $\mathscr{S}$ is commuting along edges of $\mathfrak{Oracle}(\mathfrak{G})$, Z-aligned and induced by a permutation strategy. It is left to be convinced that the extended $\mathscr{S}$ has value 1, but this is also immediate as (137) is satisfied because the observables at $e = \mathsf{xy}$ are consistent with those of x and y, and (138) are satisfied because the original $\mathscr{S}$ was perfect for $\mathfrak{G}$.

For soundness, assume $\mathscr{S} = \{\mathcal{U}\}$ is a value $1 - \varepsilon$ strategy for $\mathfrak{Oracle}(\mathfrak{G})$. The idea is to show that the restriction of $\mathcal{U}$ to the isolated vertices induces a strategy for $\mathfrak{G}$ with value of at least $1 - 12\varepsilon$. To that end, for $e = \mathsf{xy}$, let $\varepsilon_{\mathsf{x},e}$ be the probability $\mathcal{U}$ fails the checks of the edge (x, e) in $\mathfrak{Oracle}(\mathfrak{G})$, and $\varepsilon_{\mathsf{y},e}$ is defined similarly. Then

$$\varepsilon = \mathop{\mathbb{E}}_{(\mathsf{x},e)\sim\mu'}[\varepsilon_{\mathsf{x},e}] = \mathop{\mathbb{E}}_{e\sim\mu}\left[\frac{\varepsilon_{\mathsf{x},e} + \varepsilon_{\mathsf{y},e}}{2}\right] .$$

By Claims 3.36 and 3.22, the fact that $\mathcal{U}$ passes the linear checks (137) with probability $1 - \varepsilon_{\mathsf{x},e}$ implies that

$$\forall \alpha \in \mathbb{F}_2^{S_{\mathsf{x}}} : \; \left\| \prod_{\mathsf{X}\in S_{\mathsf{x}}} \mathcal{U}(\mathsf{X})^{\alpha(\mathsf{X})} - \prod_{\mathsf{OX}_e \in S_{\mathsf{x}\in e}} \mathcal{U}(\mathsf{OX}_e)^{\alpha(\mathsf{X})} \right\|_{hs}^2 \leq 6\varepsilon_{\mathsf{x},e} ,$$

and similarly products of $\mathcal{U}(\mathsf{Y})$'s are close to products of $\mathcal{U}(\mathsf{OY}_e)$'s. In addition, as $\mathcal{U}$ passes the linear check (138) with probability of at least $1 - \min(\varepsilon_{\mathsf{x},e}, \varepsilon_{\mathsf{y},e}) \geq 1 - \frac{\varepsilon_{\mathsf{x},e} + \varepsilon_{\mathsf{y},e}}{2}$, we can deduce by Claim 3.36 that

$$\left\| (-\mathrm{Id})^b \prod \mathcal{U}(\mathsf{OX}_e^{\kappa,j})^{\alpha_{\kappa,j}} - \prod \mathcal{U}(\mathsf{OY}_e^{\kappa,j})^{\beta_{\kappa,j}} \right\|_{hs}^2 \leq 2\varepsilon_{\mathsf{x},e} + 2\varepsilon_{\mathsf{y},e}.$$

Combining the above and using the triangle inequality,

$$\left\|(-\mathrm{Id})^b \prod \mathcal{U}(\mathsf{X}^{\kappa,j})^{\alpha_{\kappa,j}} - \prod \mathcal{U}(\mathsf{Y}^{\kappa,j})^{\beta_{\kappa,j}}\right\|_{hs}^2 \leq 3 \underbrace{\left\|(-\mathrm{Id})^b \prod \mathcal{U}(\mathsf{X}^{\kappa,j})^{\alpha_{\kappa,j}} - (-\mathrm{Id})^b \prod \mathcal{U}(\mathsf{XO}_e^{\kappa,j})^{\alpha_{\kappa,j}}\right\|_{hs}^2}_{\leq 6\varepsilon_{\mathsf{x},e}}$$

$$+ 3 \underbrace{\left\|(-\mathrm{Id})^b \prod \mathcal{U}(\mathsf{OX}_e^{\kappa,j})^{\alpha_{\kappa,j}} - \prod \mathcal{U}(\mathsf{OY}_e^{\kappa,j})^{\beta_{\kappa,j}}\right\|_{hs}^2}_{\leq 2\varepsilon_{\mathsf{x},e}+2\varepsilon_{\mathsf{y},e}}$$

$$+ 3 \underbrace{\left\|\prod \mathcal{U}(\mathsf{OY}_e^{\kappa,j})^{\beta_{\kappa,j}} - \prod \mathcal{U}(\mathsf{Y}^{\kappa,j})^{\beta_{\kappa,j}}\right\|_{hs}^2}_{\leq 6\varepsilon_{\mathsf{y},e}}$$

$$\leq 24(\varepsilon_{\mathsf{x},e} + \varepsilon_{\mathsf{y},e}),$$

which translates to $\mathcal{U}$ passing $\mathfrak{G}$ with probability at least $1 - 6(\varepsilon_{\mathsf{x},e} + \varepsilon_{\mathsf{y},e})$ when e is sampled. Therefore, the value of the restriction of $\mathcal{U}$ to the isolated vertices passes $\mathfrak{G}$ with probability of at least $1 - 12\varepsilon$, as claimed, and the entanglement lower bound is immediate from that. □

Remark 5.47 (The sampling procedure of the oracularized game)**.** We do not know how to induce the sampling procedure described in Definition 5.43 using CLMs, even when $\mathfrak{G}$ has a sampling procedure induced by CLMs. But, If the sampling procedure of $\mathfrak{G}$ was induced by h-level CLMs $\mathfrak{s}^A, \mathfrak{s}^B$, then there is an h-level typed sampling scheme with type graph $\mathtt{A} - \mathtt{Oracle} - \mathtt{B}$ that induces the aforementioned distribution for $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathfrak{G}))$, the oracularized **double cover** of the game (Definition 3.52). This is done by using the same dimension as before, letting $\mathfrak{s}^{\mathtt{A}} = \mathfrak{s}^A$, $\mathfrak{s}^{\mathtt{B}} = \mathfrak{s}^B$, and $\mathfrak{s}^{\mathtt{Oracle}} = \mathfrak{s}^A \times \mathfrak{s}^B$, by which we mean, given input z, $\mathfrak{s}^{\mathtt{Oracle}}$ outputs $(\mathfrak{s}^A(z), \mathfrak{s}^B(z))$. As is common throughout this paper, given that $\mathfrak{G}$ has enough consistency checks (or, given that it was already bipartite), this move to the double cover does not hinder the desired conclusions.

Though the above sampling scheme works, we later let $\mathfrak{s}^{\mathtt{Oracle}} = \mathrm{Id}$ instead. This means that between every two isolated vertices $(\mathtt{A}, \mathtt{x})$ and $(\mathtt{B}, \mathtt{y})$, instead of having a single vertex $(\mathtt{Oracle}, \mathtt{xy})$, there is a vertex $(\mathtt{Oracle}, z)$ for every z for which $\mathfrak{s}^A(z) = \mathtt{x}$ and $\mathfrak{s}^B(z) = \mathtt{y}$. Although this allows the strategies in the oracularized (double cover) game more leniency, which presumably may elevate the value of the game compared to the case of a genuine barycentric subdivision, this turns out not to be the case and the above soundness argument works (essentially) the same.

5.2.3 Triangulation

Definition 5.48 (Triangulated system)**.** A system of linear equations $\mathscr{A}\vec{S} = \vec{b}$ over a field $\mathbb{F}$ is said to be *triangulated* if every row of $\mathscr{A}$ has at most 3 non-zero entries.

Definition 5.49 (Triangulating a system of linear equations)**.** Triangulating a system of linear equations (or a system of word equations over a group) is a standard procedure. The idea is to replace an equation

$$a_0\mathsf{X}_0 + a_1\mathsf{X}_1 + a_2\mathsf{X}_2 + \dots + a_n\mathsf{X}_n = b\,,$$

on $n + 1$ variables, by $n + 2$ triangulated equations

$$a_0\mathsf{X}_0 = \mathsf{Y}_0\,, \tag{139}$$

$$\forall 1 \leq i \leq n: \quad \mathsf{Y}_{i-1} + a_i\mathsf{X}_i = \mathsf{Y}_i\,, \tag{140}$$

$$\mathsf{Y}_n = b\,. \tag{141}$$

on $2n + 2$ variables. On the level of the matrix representation of the system, this is the same as replacing the single row

$$\begin{pmatrix} a_0 & a_1 & \dots & a_n & | & b \end{pmatrix}$$

with the system

$$
\left(
\begin{array}{ccccccccccc|c}
a_0 & 0 & 0 & \dots & 0 & -1 & 0 & 0 & \dots & 0 & 0 & 0 \\
0 & a_1 & 0 & \dots & 0 & 1 & -1 & 0 & \dots & 0 & 0 & 0 \\
0 & 0 & a_2 & \dots & 0 & 0 & 1 & -1 & \dots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \dots & a_n & 0 & 0 & 0 & \dots & 1 & -1 & 0 \\
0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & \dots & 0 & 1 & b
\end{array}
\right).
$$

When there is more than one equation in the system (which is usually the case), one adds $n+1$ new variables to each equation; namely, if there were R equations in the system and $n+1$ variables, the triangulated system has $R(n+2)$ equations over $(R+1)(n+1)$ variables.

More generally, given a system of linear equations with matrix representation $\mathscr{A} \cdot \vec{S} = \vec{b}$ and a non-negative integer Δ, we define $\mathrm{Triangle}(\mathscr{A}, \vec{b}, \Delta) := (\mathscr{A}_\triangle \mid \vec{b}_\triangle)$ to be the matrix representation $\mathscr{A}_\triangle \cdot \overrightarrow{S \sqcup S_\triangle} = \vec{b}_\triangle$ of the triangulated system with $|S_\triangle| = \Delta$ more variables. Namely, if $\mathscr{A}$ is of size $R \times (n+1)$, and $\Delta \geq R(n+1)$, then the system $\mathscr{A}_\triangle \cdot \overrightarrow{S \sqcup S_\triangle} = \vec{b}_\triangle$ is the triangulated system (with extra $\Delta - R(n+1)$ variables that do not appear in any equation). Otherwise, it does not contain the last $R(n+1) - \Delta$ equations in the triangulated system (as there were not enough variables to fully triangulate).

Remark 5.50 (Properties of the triangulated system)**.**

1. The triangulation procedure increases both the number of variables and the number of equations. It is also complete and sound in the following sense: There is a one to one correspondence between solutions to the original system of equations and the triangulated system. Namely, given a system $\mathscr{A} \cdot \vec{S} = \vec{b}$ with R many equations, if we let $\Delta = R(n+1)$, then the solutions to $\mathrm{Triangle}(\mathscr{A}, \vec{b}, \Delta)$ correspond perfectly to those of the original system. More generally, whenever $\Delta \geq R(n+1)$ there is still a perfect correspondence by assigning the value 0 to all variables that do not participate in any equation. Given an assignment $f \colon S \to \mathbb{F}$ to the original variables of the system, let us denote by $f_\triangle \colon S_\triangle \to \mathbb{F}$ the aforementioned unique extension to the triangulation variables $S_\triangle$ — note that the values of $f_\triangle$ are affine combinations of the values of f.

2. Triangulation is efficient. Namely, given a system $\mathscr{A}\vec{x} = \vec{b}$ with $\mathscr{A}$ of size $R \times (n+1)$, and an integer $\Delta \geq 0$, the system $\mathrm{Triangle}(\mathscr{A}, \vec{b}, \Delta)$ is of size $R(n+2) \times (n+1+\Delta)$ and takes $O(R(n+2)(n+1+\Delta))$-time to calculate it.

5.2.4 Decoupling

It is much easier to implement a PCP protocol as a non-local game if every polynomial in the PCP protocol is measured at a single point instead of several points; this should be contrasted with the example in the Prelude Section 5.1, where g_ψ is measured at 4 potentially different points u_0, u_1, u_2, u_3, as described in (134). To that end, we define a more restrictive format of systems of linear equations and k-SAT instances, so that the PCPs we construct measure every polynomial at a single point.

Definition 5.51 (Decoupled systems of equations and decoupled CNFs)**.** A system of linear equations (over a field $\mathbb{F}$) is said to be *k-decoupled* if there are k (disjoint) sets of formal generators $S_1, ..., S_k$ — each of which is called *a block of generators* — such that each equation in the system contains at most one variable from each block; namely, the equations are of the form

$$a_1 \mathsf{X}_1 + ... + a_k \mathsf{X}_k = b,$$

where $a_1, ..., a_k, b \in \mathbb{F}$ and $\mathsf{X}_i \in S_i$ for $1 \leq i \leq k$. Such an equation is uniquely defined by the tuple $(a_1, \mathsf{X}_1, ..., a_k, \mathsf{X}_k, b) \in \mathbb{F} \times S_1 \times ... \times \mathbb{F} \times S_k \times \mathbb{F}$, and thus a k-decoupled system of equations can be encoded as an indicator of a subset of $S_1 \times ... \times S_k \times \mathbb{F}^{k+1}$ (this encoding is finite when S_i and $\mathbb{F}$ are finite).

Let $S_1, ..., S_k$ be, again, disjoint sets of formal generators. A Boolean formula is said to be a k-decoupled CNF (over the blocks $S_1, ..., S_k$) if each disjunctive clause in the conjunction contains exactly one variable from each block; namely, the formula is a conjunction of clauses of the form

$$\mathsf{X}_1^{\varepsilon_1} \vee ... \vee \mathsf{X}_k^{\varepsilon_l},$$

where $\varepsilon_i \in \mathbb{F}_2$ and $\mathsf{X}_i \in S_i$ for every $1 \leq i \leq k$. Assume there are natural numbers $\{n_i\}_{i=1}^k$ such that $S_i = \{\mathsf{X}_{i,u}\}_{u \in \mathbb{F}_2^{n_i}}$ for every $i \in [k]$. Then, a circuit $\mathcal{C}$ (Definition 5.28) with $k + \sum_{i=1}^k n_i$ input gates and a single output encodes a k-decoupled CNF $\varphi_{\mathcal{C}}$ by including the clause

$$\mathsf{X}_{1,u_1}^{\varepsilon_1} \vee ... \vee \mathsf{X}_{k,u_k}^{\varepsilon_k}$$

in the conjunction whenever $P_{\mathcal{C}}(u_1, ..., u_k, \varepsilon_1, ..., \varepsilon_k) = 1$, where $u_i \in \mathbb{F}_2^{n_i}$ and $\varepsilon_i \in \mathbb{F}_2$ for every $i \in [k]$.

The following fact is immediate from the above definition.

Fact 5.52 (Translating satisfiability conditions of decoupled systems and CNFs into polynomial equations. Cf. Observation 5.33)**.** *Let $O \colon S_1 \times S_2 \times ... \times S_k \times \mathbb{F}^{k+1} \to \{0_{\mathbb{F}}, 1_{\mathbb{F}}\} \subseteq \mathbb{F}$ be the encoding of a k-decoupled system of linear equations with blocks $S_1, ..., S_k$ over the field $\mathbb{F}$ (Definition 5.51); namely, the equation*

$$a_1 \mathsf{X}_1 + ... + a_k \mathsf{X}_k = b,$$

where each X_i is in S_i, appears in the system induced by O if and only if $O(\mathsf{X}_1, ..., \mathsf{X}_k, a_1, ..., a_k, b) = 1_{\mathbb{F}}$. Then, the assignments $f_i \colon S_i \to \mathbb{F}$ for $i \in [k]$ satisfy the decoupled system of equations induced by O if and only if

$$\forall (\mathsf{X}_1, ..., \mathsf{X}_k, a_1, ..., a_k, b) \in S_1 \times ... \times S_k \times \mathbb{F}^{k+1} : \quad O(\mathsf{X}_1, ..., \mathsf{X}_k, a_1, ..., a_k, b)(a_1 f_1(\mathsf{X}_1) + ... + a_k f_k(\mathsf{X}_k) - b) = 0 . \quad (142)$$

Similarly, let $\mathcal{C}$ be a circuit with $k + \sum_{i=1}^k n_i$ many input gates, where k and $n_1, ..., n_k$ are positive integers, which encodes a k-decoupled CNF $\varphi_{\mathcal{C}}$ on blocks $S_i = \{\mathsf{X}_{i,u}\}_{\mathbb{F}_2^{n_i}}$ as in Definition 5.51. Then, the assignments $w_i \colon S_i \to \mathbb{F}_2$ for $i \in [k]$ satisfy φ_C if and only if

$$\forall (u_1, ..., u_k, \varepsilon_1, ..., \varepsilon_k) \in \mathbb{F}_2^{n_1} \times ... \times \mathbb{F}_2^{n_k} \times \mathbb{F}_2^k : \quad P_{\mathcal{C}}(u_1, ..., u_k, \varepsilon_1, ..., \varepsilon_k) \cdot \prod_{i=1}^k (w_i(\mathsf{X}_{i,u_i}) + \varepsilon_i + 1) = 0 , \quad (143)$$

where $P_{\mathcal{C}}$ is the function induced by $\mathcal{C}$ (Definition 5.28). Equivalently, if $\mathcal{C}$ has s many non-input wires, then the aforementioned w_i's are a satisfying assignment to $\varphi_{\mathcal{C}}$ if and only if

$$\forall (u_1, ..., u_k, \varepsilon_1, ..., \varepsilon_k, z) \in \mathbb{F}_2^{n_1} \times ... \times \mathbb{F}_2^{n_k} \times \mathbb{F}_2^k \times \mathbb{F}_2^s : \quad T_{\mathcal{C}}(u_1, ..., u_k, \varepsilon_1, ..., \varepsilon_k, z) \cdot \prod_{i=1}^k (w_i(\mathsf{X}_{i,u_i}) + \varepsilon_i + 1) = 0 , \quad (144)$$

where $T_{\mathcal{C}}$ is the Tseitin polynomial associated with $\mathcal{C}$ (Definition 5.28).

Remark 5.53. A triangulated system (Definition 5.48) of m linear equations $\mathscr{A}\vec{S} = \vec{b}$ over $\mathbb{F}$ can be 3-decoupled in a straightforward manner: Let $\vec{S}_1, \vec{S}_2, \vec{S}_3$ be three disjoint copies of $\vec{S}$ — namely, if $\vec{S} = (\mathsf{X}_j)_{j=1}^n$, then $\vec{S}_i = (\mathsf{X}_j^i)_{j=1}^n$. First, regardless of what $\mathscr{A}$ was, add the $2n$ decoupled linear equations

$$\forall 1 \leq j \leq n : \quad \mathsf{X}_j^1 = \mathsf{X}_j^2 = \mathsf{X}_j^3 .$$

Then, for every $1 \leq k \leq m$, if the kth equation of $\mathscr{A}\vec{S} = \vec{b}$ is $a_1 \mathsf{X}_{j_1} + a_2 \mathsf{X}_{j_2} + a_3 \mathsf{X}_{j_3} = b_k$ (where $j_1 < j_2 < j_3$),[82] then add the decoupled equation

$$a_1 \mathsf{X}_{j_1}^1 + a_2 \mathsf{X}_{j_2}^2 + a_3 \mathsf{X}_{j_3}^3 = b_k .$$

All in all, the new decoupled system has $2n + m$ equations over $3n$ variables, and it is straightforward to relate the set of solutions of the two systems of equations.

---

[82]Here we are using the arbitrary ordering $\vec{S}$ on S which is used to write down the system $\mathscr{A}\vec{S} = \vec{b}$.

For our purposes, we need some extra conditions on the new system to be able to answer reduce. We want to decouple only the non consistency linear constraints at the oracle player vertices, under the assumption that they are triangulated and pure. In addition, we still want to be able to check the consistency in an easy manner. To that end, we define a 5-decoupling instead of a 3-decoupling, where the first two blocks should be the "original variables" that will be compared to the isolated player's answers, and three "new blocks" that play a similar role to the above naive decoupling variables — namely, they are an aggregate of all original variables together with the variables added in the triangulation phase.

Definition 5.54 (Combinatorial 5-decoupling of a triangulated linear system of equations)**.** Let S be the disjoint union of 3 sets of formal variables

$$S_A \, , \, S_B \, , \, S_\triangle \, ,$$

of respective sizes $\ell_A, \ell_B, \ell_\triangle$, and let $\ell = \ell_A + \ell_B + \ell_\triangle$. Let $\mathscr{A}\vec{S} = \vec{b}$ be a triangulated system of m equations over S — note that here we assumed some ordering on S was fixed, which is used later. The combinatorial 5-decoupling $\mathfrak{DeCouple}(S_A, S_B, S_\triangle, (\mathscr{A} \mid \vec{b}))$ of $\mathscr{A}\vec{S} = \vec{b}$ has the following 5 blocks of variables

Block number	1	2	3	4	5
Set of variables	S_A	S_B	S_1	S_2	S_3
Size	ℓ_A	ℓ_B	ℓ	ℓ	ℓ

where each S_i is a copy of S, namely it is composed of a disjoint union of sets $S_{A,i}, S_{B,i}, S_{\triangle,i}$ each of which is a respective copy of $S_A, S_B, S_\triangle$. For later use, if $A, B, C \in S$, then the corresponding formal variables in S_i are denoted A^i, B^i, C^i.

For the equations, we have the following. Regardless of what $\mathscr{A}$ or $\vec{b}$ are, it has $3\ell_A + 3\ell_B + 2\ell_\triangle = (\ell_A + \ell_B + 2\ell)$-many decoupled equations

$$
\begin{aligned}
\forall X \in S_A : \quad & X = X^1 = X^2 = X^3 \, , \\
\forall Y \in S_B : \quad & Y = Y^1 = Y^2 = Y^3 \, , \\
\forall Z \in S_\triangle : \quad & Z^1 = Z^2 = Z^3 \, .
\end{aligned}
\tag{145}
$$

In addition, as $\mathscr{A}\vec{S} = \vec{b}$ is triangulated, an equation in it is of the form $a_1 A + a_2 B + a_3 C = b$, where $A, B, C \in S$ with $A < B < C$ according to the ordering $\vec{S}$, and $a_1, a_2, a_3, b \in \mathbb{F}$. For every such equation in $\mathscr{A}\vec{S} = \vec{b}$, add the decoupled equation

$$a_1 A^1 + a_2 B^2 + a_3 C^3 = b \tag{146}$$

to $\mathfrak{DeCouple}(S_A, S_B, S_\triangle, (\mathscr{A} \mid \vec{b}))$, where $A^1 \in S_1, B^2 \in S_2$ and $C^3 \in S_3$ are the respective copies of A, B, C. All in all, we defined a 5-decoupled system of $m + 2\ell + \ell_A + \ell_B$ linear equations over $3\ell + \ell_A + \ell_B$ variables.

The solutions to the original system $(\mathscr{A} \mid \vec{b})$ and $\mathfrak{DeCouple}(S_A, S_B, S_\triangle, (\mathscr{A} \mid \vec{b}))$ are in perfect correspondence: This correspondence is achieved by associating with any assignment $f : S_A \sqcup S_B \sqcup S_\triangle \to \mathbb{F}$ the unique 5-tuple $(f|_{S_A}, f|_{S_B}, f, f, f)$. Indeed, from (145), every satisfying assignment for the decoupled system is of this form, and it is straightforward to check that the image of a satisfying assignment to the original triangulated system is satisfying the decoupled system.

Remark 5.55 (Algorithmic decoupling)**.** Let $\mathscr{A}\vec{S} = \vec{b}$ be the matrix representation of a **triangulated** linear system of equations over $\mathbb{F}_2$, where $S = S_A \sqcup S_B \sqcup S_\triangle$. Then, the encoding (as in Definition 5.51) of its 5-decoupling

$$\mathfrak{DeCouple}(S_A, S_B, S_\triangle, (\mathscr{A} \mid \vec{b}))$$

(Definition 5.54) as an indicator map on

$$S_A \times S_B \times S \times S \times S \times \mathbb{F}_2^6$$

can be calculated efficiently, namely in time polynomial in the input length. We often denote by O the resulting encoding, and may abuse notation and write $O = \mathfrak{DeCouple}(S_A, S_B, S_\triangle, (\mathscr{A} \mid \vec{b}))$.

Corollary 5.56 (Combining triangulation and decoupling)**.** *Let $S = S_A \sqcup S_B$, and $\mathscr{A} \cdot \vec{S} = \vec{b}$ a system of linear equations over $\mathbb{F}$ with R many equations. Let $\Delta \geq R(|S| + 1)$ be a positive integer, and $S_\triangle$ a set of Δ-many formal variables (disjoint from S). Then, there is an **affine** mapping* Extend[83] *from assignments $f \colon S \to \mathbb{F}$ to 5-tuples $(f_1, ..., f_5)$ which are assignments of*

$$\mathfrak{DeCouple}(S_A, S_B, S_\triangle, \mathsf{Triangle}(\mathscr{A}, \vec{b}, \Delta)) \, ,$$

such that:

- Completeness*: Satisfying assignments are mapped to satisfying assignments.*

- Soundness*: Non-satisfying assignments are sent to non-satisfying assignments. In addition, by ignoring the variables in $S_\triangle$ that do not appear in any equation of $\mathsf{Triangle}(\mathscr{A}, \vec{b}, \Delta)$, every satisfying assignment to the decoupled system is the* Extend*-image of a satisfying assignment to the original system.*

Proof. Given $f \colon S_A \sqcup S_B \to \mathbb{F}$, the map Extend first defines the map $f_\triangle \colon S_\triangle \to \mathbb{F}_2$ from clause 1. of Remark 5.50. Hence, it retrieves a map $f' \colon S_A \sqcup S_B \sqcup S_\triangle \to \mathbb{F}$ by letting

$$f'(\mathsf{X}) = \begin{cases} f(\mathsf{X}) & \mathsf{X} \in S_A \sqcup S_B \, , \\ f_\triangle(\mathsf{X}) & \mathsf{X} \in S_\triangle \, . \end{cases}$$

Then, Extend outputs $(f|_{S_A}, f|_{S_B}, f', f', f')$. By combining Remark 5.50 and the observation at the end of Definition 5.54, we deduce the corollary. □

5.3 Translating the verifier's checks into polynomial equations

As described in the Prelude Section 5.1, PCP techniques are fit to decide whether a formula succinctly described by a circuit (Definition 5.1.4) is satisfiable. So, after applying some prerequisite transformations — namely padding and purification — the next step towards answer reduction is to translate some of the checks in the game to succinct SAT (and LIN) instances. The succinct SAT instances described here are slightly different from those in the Prelude 5.1, and are adapted from [JNV[+]21, Sections 10.2 and 10.3]. The plan is as follows:

1. First, we replace the check from (10), which verifies that a tuple $a^\mathfrak{R}, a^\mathfrak{L}, b^\mathfrak{R}, b^\mathfrak{L}$ is accepted by the game $\mathcal{V}_n$ given xy were asked, by two checks that verify the same thing. The first check verifies that a bit string O encodes the (triangulated and decoupled) purely unreadable part of the linear system $L_{xy}(a^\mathfrak{R}, b^\mathfrak{R})$. The second check verifies that (an appropriate extension of) $a^\mathfrak{L}, b^\mathfrak{L}$ solve the system O. This is done in Section 5.3.1, and the equivalence to $\mathcal{V}_n$ accepting this quadruple is stated in Claim 5.59.

2. Then, the check that O encodes the purely unreadable part of $L_{xy}(a^\mathfrak{R}, b^\mathfrak{R})$ is shown to be equivalent, using a version of the Cook–Levin transformation, to the satisfiability of a 6-decoupled CNF formula succinctly encoded by some circuit $\mathcal{C}$. This is done in Section 5.3.2, and the main take away from this section is Corollary 5.65.

3. At this point, the condition "$a^\mathfrak{R}, a^\mathfrak{L}, b^\mathfrak{R}, b^\mathfrak{L}$ are accepted by the game $\mathcal{V}_n$ given xy were asked" was replaced by the satisfiability of a certain succinctly encoded formula $\varphi_\mathcal{C}$ and an appropriate succinctly encoded system of linear equations $(\mathscr{A}_O \mid \vec{b}_O)$. At this point, PCP techniques allow to replace these two satisfiability conditions by 13 polynomial equations (see (170), (171), (172) and (173)), whose satsifiability can be checked *probabilistically* by reading only a logarithmic portion of the polynomias' values. This is done in Section 5.3.3.

---

[83]The mapping Extend depends on the decomposition of S to S_A, S_B, the system $(\mathscr{A} \mid \vec{b})$, and the chosen parameter Δ, but we omit this dependence from the notation.

Let us elaborate on why naively applying the scaled up Cook–Levin theorem on (10) does not work in our case (as opposed to [JNV$^+$21]). The problem is that the resulting PCP does not behave well with regards to permutation assignments to unreadable variables. More on that: Given two commuting unitary involutions, namely matrices $A, B \in U(n)$ such that $A^2 = B^2 = [A, B] = \mathrm{Id}$, there is a well defined notion of their $\wedge$ (AND operation) — As they are mutually diagonalizable with respect to some orthonormal basis, and on the diagonal there are only ± 1's, we can define $A \wedge B$ to be the diagonal matrix (with respect to the same orthonormal basis) whose ii entry is -1 if and only if the ii entries of A and B are both -1.[84] A problem arises when the two matrices are permutation matrices — in this case, though their $\wedge$ is well defined, it is not necessarily a permutation matrix (for example, the matrices in (27)), which is problematic when the evaluation table of the PCP should be generated by measuring a ZPC-strategy. Thus, constructing the PCP requires us to be careful with the exact operations applied to unreadable variables, so that in the complete case the proof can be induced by a ZPC strategy.

Name	Role	See
$\mathcal{L}^*$	The triangulated output indicator	Definition 5.58
Λ	A TM controlling the expected answer length, i.e. $$\lvert a^{\mathfrak{L}} \rvert = \lvert a^{\mathfrak{R}} \rvert = \lvert b^{\mathfrak{L}} \rvert = \lvert b^{\mathfrak{R}} \rvert = 2^{\Lambda(n)}$$	Definition 5.58
Δ	A TM controlling the padding required for triangulation	Definition 5.58
$\diamondsuit$	Number of bits required to specify a variable post-triangulation, i.e. $$\diamondsuit(n) = \lceil \log(2^{\Lambda(n)+1} + \Delta(n)) \rceil$$	Definition 5.58
T	A TM that bounds the running time of $\mathcal{L}^*$ with the first 6 inputs fixed to $\mathcal{V}, \Lambda, \Delta, n, \mathrm{x}, \mathrm{y}$	Definition 5.60
M	Size of blocks in the circuit representing $\mathcal{L}^*$	Definition 5.60
s	Number of non-input wires in the circuit representing $\mathcal{L}^*$	Definition 5.60
Q	A TM that bounds the dimension of the CLM underlying $\mathcal{S}$	Corollary 5.65
D	A bound on the description lengths of $\mathcal{V}, \Lambda, \Delta, T$ and Q	Proposition 5.62
h	The tailored normal form verifiers are h-level	Definition 4.33
m	Dimension of the PCPs, defined as $$m = \lvert S \rvert = 4\Lambda(n) + 3\diamondsuit(n) + 3M(n) + s(n) + 12$$	Definition 5.66
$\heartsuit$	Number of polynomials in a PCP, defined as $$\heartsuit(n) = 12\Lambda(n) + 12\diamondsuit(n) + 6M(n) + s(n) + 35$$	Definition 5.68
q, t	The size of the field $\mathbb{F}_q = \mathbb{F}_{2^t}$ which is used in the PCP	Definition 5.68

Table 2: Summary of some relevant parameters used in the rest of Section 5.

5.3.1 The triangulated output indicator $\mathcal{L}^*$

Recall that our goal is to translate the decision problem "given a tailored normal form verifier $\mathcal{V}$, are $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ accepted in the game $\mathcal{V}_n$ assuming questions x, y were asked?" to a collection of polynomial equations on which PCP techniques can be applied. To that end, we first define a TM $\mathcal{L}^*$ called *the triangulated output indicator* (Definition 5.58) which, under some padding and purification assumptions (Definition 5.57), checks that a bit string O is the encoding (as in Definition 5.51) of the 5-decoupling (Definition 5.54) of the triangulated system (Definition 5.48) of controlled linear constraints in the tailored game $\mathcal{V}_n$. The reason for the name "output indicator" is that, when $\mathcal{L}^*$ halts, it outputs either 0 or 1, and if it outputted 1, then the aforementioned input O is the expected output of the operation of the TM.

Definition 5.57 (A padded, purified TNFV). Let $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ be a h-level tailored normal form verifier, and Λ a single input TM that always halts. We say that $\mathcal{V}$ is 2^{Λ}-padded if $\lvert \mathrm{dec}(\mathcal{A}(n, \mathrm{x}, \kappa)) \rvert = 2^{\Lambda(n)}$ regardless of x and κ. We say that $\mathcal{V}$ is purified if the controlled linear constraints in the game $\mathcal{V}_n$ (whenever it is well defined) are purely unreadable (Definition 5.39).

---
[84]Similar to before, we interpret $-1 = (-1)^1$ as True, and $1 = (-1)^0$ as False.

Definition 5.58 (Triangulated output indicator of a linear constraint processor). The *triangulated output indicator* $\mathcal{L}^*$ is a 9-input TM that takes as input: an h-level tailored normal form verifier $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$; two 1-input TMs Λ, Δ, which induce (partial) functions $\Lambda, \Delta \colon \mathbb{N} \to \mathbb{N}$; an integer n (in binary); five bit strings $x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}}$ and O.

Let us first explain what $\mathcal{L}^*$ expects the given inputs to satisfy: $\mathcal{V}$ was already said to be an h-level normal form verifier. The inputs $n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}}$ are expected to be, as usual, an index of a game, a pair of questions in this game and a pair of readable answers to these questions, all with respect to $\mathcal{V}_n$. The TM Λ is supposed to be the padding parameter in $\mathcal{V}$; namely, $\mathcal{L}^*$ expects the answer length calculator $\mathcal{A}$ to always imply $2^{\Lambda(n)}$-long (readable and linear) answers. The TM Δ controls the number of padding variables used in the triangulation of an intermediate system induced by $\mathcal{L}(n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ — namely, it is the parameter Δ as in Definition 5.49 for some system, or alternatively the size of the formal generating set $S_{\triangle}$ as in Definition 5.54. Finally, O is expected to be the encoding as an indicator (Definition 5.51 and Remark 5.55) of some 5-decoupled (Definition 5.54) triangulated linear system induced by $\mathcal{L}(n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$.

We now describe the operation of the TM $\mathcal{L}^*$, namely its high-level description (Remark 2.33). To make it easier to follow, we add remarks in each clause of the operation, as well as the running time bounds of the specific step.

1. <u>The readable answers are of the appropriate length</u>: $\mathcal{L}^*$ calculates $\Lambda(n)$ (by running Λ on input n), and checks that

$$|a^{\mathfrak{R}}|, |b^{\mathfrak{R}}| = 2^{\Lambda(n)} \; ;$$

   if not, it outputs 0.

   This ensures that the answers are of the length expected by a verifier that is 2^{Λ}-padded (Definition 5.57). This step takes $\mathrm{poly}(\mathbb{T}(\Lambda; n), 2^{\Lambda(n)})$ time.[85]

2. <u>Calculate the system of controlled linear constraints</u>: In this step, $\mathcal{L}^*$ defines a system of linear equations $(\mathscr{A} \mid \vec{b})$ over $2^{\Lambda(n)+2}$ many variables as follows. First, it calculates $\mathcal{L}(n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}})$, and then decodes it (Definition 2.34). If the result was well structured, namely of the form $c^1 \sqcup \ldots \sqcup c^k$ where each c^i is a bit string of length $2^{\Lambda(n)+2} + 1$, then it lets $(\mathscr{A} \mid \vec{b})$ be the system whose rows are c^i (the first $2^{\Lambda(n)+2}$ bits of each c^i belong to $\mathscr{A}$ and the last bit is the ith value in $\vec{b}$) — note that in this case this system has k-many equations. Otherwise, it lets $(\mathscr{A} \mid \vec{b})$ be the system with $2^{\Lambda(n)+2}$-many variables, and a single equation $0 = 1$ (i.e., $\mathscr{A}$ is the zero matrix with a single row and $2^{\Lambda(n)+2}$-many columns, and $\vec{b}$ is the scalar 1).

   The above choice ensures that $(\mathscr{A} \mid \vec{b})$ agrees with the controlled linear constraints $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ in the game $\mathcal{V}_n$ — note that when the output of $\mathcal{L}$ is **not** well formatted, the canonical decider will surely reject, which is the same as assuming $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ is the never accepting system $0 = 1$ — this is how we defined $\mathcal{V}_n$ in Definition 2.48.[86] Hence, there is a natural association between the variables of this system and $S = S_x^{\mathfrak{R}} \sqcup S_x^{\mathfrak{L}} \sqcup S_y^{\mathfrak{R}} \sqcup S_y^{\mathfrak{L}}$, where $S_x^{\mathfrak{R}}, S_x^{\mathfrak{L}}, S_y^{\mathfrak{R}}, S_y^{\mathfrak{L}}$ are the formal variables at the vertices x and y of $\mathcal{V}_n$. This step takes $\mathrm{poly}(\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot), 2^{\Lambda(n)})$ time.

3. <u>The system of linear equations is purely unreadable</u>: As $\mathcal{L}^*$ recovered a linear system $(\mathscr{A} \mid \vec{b})$ with variables $S_x^{\mathfrak{R}} \sqcup S_x^{\mathfrak{L}} \sqcup S_y^{\mathfrak{R}} \sqcup S_y^{\mathfrak{L}}$, it can check whether this system is purely unreadable (Definition 5.39) — namely, that the columns associated to the variables from $S_x^{\mathfrak{R}}$ and $S_y^{\mathfrak{R}}$ are all zeros — and otherwise output 0.

   This step takes $O(k \cdot 2^{\Lambda(n)})$ time, and as $k \leq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)$, the running time of this step is bounded by

$$\mathrm{poly}(\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot), 2^{\Lambda(n)}) \; .$$

4. <u>Extracting the pure system</u>: $\mathcal{L}^*$ removes the columns of $(\mathscr{A} \mid \vec{b})$ associated with $S_x^{\mathfrak{R}}$ and $S_y^{\mathfrak{R}}$, which were checked in the previous step to be zero, resulting in a new system $(\mathscr{A}^{\mathfrak{L}} \mid \vec{b})$ on variables $S_x^{\mathfrak{L}} \sqcup S_y^{\mathfrak{L}}$.

---

[85]The TM Λ with input n may not halt; in this case $\mathcal{L}^*$ too will not halt, but this is consistent with our notation as $\mathbb{T}(\Lambda; n) = \infty$.

[86]There, we used the formulation "L outputs $\{J\}$", but this is exactly the subset representation of the unsolvable system $0 = 1$.

As the original system ought to be pure, we do not lose any information by removing the readable columns — it still encodes the same linear conditions on the unreadable variables that need to be satisfied in $\mathcal{V}_n$. This step again takes time $O(k \cdot 2^{\Lambda(n)})$, which is $\text{poly}(\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot), 2^{\Lambda(n)})$.

5. <u>There are enough triangulation variables:</u> $\mathcal{L}^*$ calculates $\Delta(n)$ (by calling Δ on input n), and checks that $\Delta(n) \geq k \cdot 2^{\Lambda(n)+1}$; otherwise it outputs 0.

   Note that k is the number of rows and $2^{\Lambda(n)+1}$ is the number of columns in $\mathscr{A}^{\mathcal{L}}$. So, this check verifies that there are enough "extra" variables in $S_\triangle$ to completely triangulate the system. This check takes $\text{poly}(\mathbb{T}(\Delta; n), k, 2^{\Lambda(n)}) \leq \text{poly}(\mathbb{T}(\Delta; n), \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot), 2^{\Lambda(n)})$ time.

6. <u>Calculate the triangulated system:</u> $\mathcal{L}^*$ calculates the triangulated system (Definition 5.49)

$$\text{Triangle}(\mathscr{A}^{\mathcal{L}}, \vec{b}, \Delta(n)) = (\mathscr{A}_\triangle \mid \vec{b}_\triangle) .$$

   We interpret this system as having variables in $S_{\mathrm{x}}^{\mathcal{L}} \sqcup S_{\mathrm{y}}^{\mathcal{L}} \sqcup S_\triangle$, where $S_{\mathrm{x}}^{\mathcal{L}}$ and $S_{\mathrm{y}}^{\mathcal{L}}$ were the original variables, and $S_\triangle$ is a new set of $\Delta(n)$-many variables.

   Note that, as Remark 5.50 states, this computation takes at most $\text{poly}(k, 2^{\Lambda(n)}, \Delta(n))$ time.

7. <u>The decoupling of the triangulated system:</u> As $\mathcal{L}^*$ recovered the triangulated system $(\mathscr{A}_\triangle \mid \vec{b}_\triangle)$ with variables in $S_{\mathrm{x}}^{\mathcal{L}} \sqcup S_{\mathrm{y}}^{\mathcal{L}} \sqcup S_\triangle$, it can apply 5-decoupling $\mathfrak{DeCouple}(S_{\mathrm{x}}^{\mathcal{L}}, S_{\mathrm{y}}^{\mathcal{L}}, S_\triangle, (\mathscr{A}_\triangle \mid \vec{b}_\triangle))$ (Definition 5.54) to it, resulting in a 5-decoupled system of equations over blocks $S_{\mathrm{x}}^{\mathcal{L}}, S_{\mathrm{y}}^{\mathcal{L}}, S_1, S_2, S_3$, where $S_1, S_2, S_3 \cong S_{\mathrm{x}}^{\mathcal{L}} \sqcup S_{\mathrm{y}}^{\mathcal{L}} \sqcup S_\triangle$ (that is, each S_i is a copy of $S_{\mathrm{x}}^{\mathcal{L}} \sqcup S_{\mathrm{y}}^{\mathcal{L}} \sqcup S_\triangle$). As described in Definition 5.51, such 5-decoupled system of equations can be encoded as an indicator map on $S_{\mathrm{x}}^{\mathcal{L}} \times S_{\mathrm{y}}^{\mathcal{L}} \times S_1 \times S_2 \times S_3 \times \mathbb{F}_2^6$. By recalling that $|S_{\mathrm{x}}^{\mathcal{L}}| = |S_{\mathrm{y}}^{\mathcal{L}}| = 2^{\Lambda(n)}$ and letting

$$\Diamond(n) = \lceil \log |S_i| \rceil = \lceil \log(2^{\Lambda(n)+1} + \Delta(n)) \rceil , \tag{147}$$

   the set $S_{\mathrm{x}}^{\mathcal{L}} \times S_{\mathrm{y}}^{\mathcal{L}} \times S_1 \times S_2 \times S_3 \times \mathbb{F}_2^6$ naturally embeds into $\mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6}$. Hence, the decoupled system is encoded as a map $U \colon \mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6} \to \mathbb{F}_2$, which is a bit string of length $2^{2\Lambda(n)+3\Diamond(n)+6}$.

   In the notation of Remark 5.55, $U = \mathfrak{DeCouple}(S_{\mathrm{x}}^{\mathcal{L}}, S_{\mathrm{y}}^{\mathcal{L}}, S_\triangle, (\mathscr{A}_\triangle \mid \vec{b}_\triangle))$, and calculating it takes $\text{poly}(k, 2^{\Lambda(n)}, \Delta(n))$-time.

8. <u>The input O matches the expected calculation:</u> Finally, $\mathcal{L}^*$ outputs 1 **only if** $O = U$.

   This ensures that the input O matches the encoding of the decoupled, triangulated, purely unreadable controlled linear constraints of $\mathcal{V}_n$ given $\mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}$. This takes at most $|\mathcal{U}| = 2^{2\Lambda(n)+3\Diamond(n)+6}$ steps, which is $\text{poly}(2^{\Lambda(n)}, \Delta(n))$.

Claim 5.59 (Properties of $\mathcal{L}^*$). *Let*
  – Λ *be a single input TM that always halts;*
  – $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *a purified 2^Λ-padded h-level TNFV (Definition 5.57) such that $\mathcal{V}_n$ is well defined for every n (Definition 4.33);*
  – Δ *an always halting single input TM that satisfies $\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ and induces $\Diamond(n)$ as in (147);*
  – *n a positive integer;*
  – *x, y two bit strings of length $r(n) = \mathcal{S}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot)$.*
Then:

1. *For every $a^{\mathfrak{R}}, b^{\mathfrak{R}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$, there **exists** a **unique** $O \colon \mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6} \to \mathbb{F}_2$ such that*

$$\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}, O) = 1 . \tag{148}$$

In addition, O is the encoding of the 5-decoupled system of linear equations

$$\mathfrak{DeCouple}(S_x^{\mathfrak{L}}, S_y^{\mathfrak{L}}, S_{\triangle}, \mathsf{Triangle}(\mathscr{A}^{\mathfrak{L}}, \vec{b}, \Delta(n))) \, , \tag{149}$$

where $(\mathscr{A}^{\mathfrak{L}} \mid \vec{b})$ is the purely unreadable part of the system of controlled linear constraints $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ from $\mathcal{V}_n$.

2. *The quadruple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ is accepted in the game $\mathcal{V}_n$ given x, y were asked if and only if the 5-tuple $\mathsf{Extend}(a^{\mathfrak{L}}, b^{\mathfrak{L}})$ (Corollary 5.56) satisfies the 5-decoupled system of linear equations defined by (the unique) O which satisfies (148).*

In addition the running time of the triangulated output indicator $\mathcal{L}^*$ satisfies

$$\mathbb{T}(\mathcal{L}^*; \mathcal{V}, \Lambda, \Delta, n, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot), 2^{\Lambda(n)}, \Delta(n)) \, . \tag{150}$$

Proof. Let us start by proving the first clause. By construction, $\mathcal{L}^*$ accepting (i.e., outputs 1) implies O is the encoding of the 5-decoupling of the triangulation of the purely unreadable part of $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ — note that this uses the assumption $\mathcal{V}$ is 2^{Λ}-padded, as otherwise the recovered $(\mathscr{A} \mid \vec{b})$ in step 2. of the operation of $\mathcal{L}^*$ is not $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$. In the other direction, note that as $a^{\mathfrak{R}}, b^{\mathfrak{R}}$ are of length $2^{\Lambda(n)}$ (which makes step 1. in the operation of $\mathcal{L}^*$ not reject, i.e., not output 0), $\mathcal{V}$ is purified (which makes step 3. not reject), and $\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ (which makes step 5. not reject), the triangulated output indicator $\mathcal{L}^*$ will run step 7. and recover $U \colon \mathbb{F}_2^{2\Lambda(n)+3\diamondsuit(n)+6} \to \mathbb{F}_2$. Hence, by choosing $O = U$, the triangulated output indicator $\mathcal{L}^*$ will output 1 on the chosen input.

Let us prove the second clause. The tuple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ is accepted in the tailored game $\mathcal{V}_n$ given x, y were asked if and only if they satisfy the system $L_{xy}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$. As this system is purely unreadable, the quadruple satisfies it if and only if the unreadable parts $a^{\mathfrak{L}}, b^{\mathfrak{L}}$ satisfy the purely unreadable part of the system. Therefore, by Corollary 5.56, this happens if and only if the 5-tuple $\mathsf{Extend}(a^{\mathfrak{L}}, b^{\mathfrak{L}})$ satisfy the system (149) encoded by O.

For the running time, one can follow the running time bounds calculated along the description. $\qquad\square$

5.3.2 Decoupled Cook–Levin: Generating a succinct description of the triangulated output indicator of a linear constraint processor

This section is dedicated to applying a version of the scaled up Cook–Levin transformation (Theorem 5.32) to the triangulated output indicator $\mathcal{L}^*$ described in Definition 5.58, which results in a circuit $\mathcal{C}$ which encodes a decoupled succinct-6SAT instance that describes the operation of $\mathcal{L}^*$, on which PCP techniques can be applied.

Let us elaborate. Recall the definition of a Boolean circuit $\mathcal{C}$ in Definition 5.28, and of the function $P_{\mathcal{C}} \colon \mathbb{F}_2^I \to \mathbb{F}_2^O$ encoded by $\mathcal{C}$. Recall also the notion of a circuit succinctly encoding a formula from Definition 5.31, and more importantly the decoupled version from Definition 5.51. The next definition describes what it means for a circuit $\mathcal{C}$ to succinctly describe the operation of the triangulated output indicator $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, x, y, \cdot, \cdot, \cdot)$, where all the labeled inputs are considered fixed parameters (hardwired to the operation of $\mathcal{L}^*$) and the dotted-inputs are considered variables of the formula (and hence inputs to the circuit).

The last three inputs of $\mathcal{L}^*$ are $a^{\mathfrak{R}}, b^{\mathfrak{R}}$ and O, which have length $2^{\Lambda(n)}, 2^{\Lambda(n)}$ and $2^{2\Lambda(n)+3\diamondsuit(n)+6}$ respectively. The circuit $\mathcal{C}$ will have a block of input gates associated with each of these inputs,[87] of size $\Lambda(n), \Lambda(n)$ and $2\Lambda(n) + 3\diamondsuit(n) + 6$ respectively. Each block of input gates can be used to address a single $\mathbb{F}_2$-value of the corresponding function. In addition we include three blocks of $M(n)$ input gates each, where $M \colon \mathbb{N} \to \mathbb{N}$ is some function to be fixed later, and 6 blocks of 1 input gate. The three blocks of $M(n)$ gates receive variables that are supposed to specify intermediate, internal values used in the computation of $\mathcal{L}^*$ — e.g., the values of the variables described in Theorem 5.13.[88] The resulting circuit $\mathcal{C}$ is a succinct encoding of a 6-decoupled CNF $\varphi_{\mathcal{C}}$ that encodes the claim that $\mathcal{L}^*$ accepts a triple $(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O)$, in the sense that

---

[87] The notion of a block of variables was described in Definition 5.51.

[88] These last three blocks of variables in $\varphi_{\mathcal{C}}$ play an analogous role to the three copies of the original variables inserted in the combinatorial decoupling from Definition 5.54.

this is the case if and only if $(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O)$ can be completed to a "proof" $(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O, w, w', w'')$ that satisfies the formula $\varphi_{\mathcal{C}}$. What we gain from this is that running $\mathcal{L}^*$ can take exponential time in principle, while the succinct representation circuit has polynomial size, can be calculated in polynomial time, and can be verified to be satisfiable in polynomial time using PCP techniques (Section 5.1.5).

Definition 5.60 (Succinct description of $\mathcal{L}^*$). Let $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ be a h-level TNFV, Λ, Δ 1-input TMs that induce (partial) functions $\mathbb{N} \to \mathbb{N}$, n a natural number, and x, y bit strings. In addition, let $T, M, s \colon \mathbb{N} \to \mathbb{N}$ be functions.

A circuit $\mathcal{C}$ is said to *succinctly describe* $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, x, y, \cdot, \cdot, \cdot)$ with parameters $(T(n), M(n), s(n))$ if:

1. It has $s(n)$ many non-input wires, and $4\Lambda(n) + 3\Diamond(n) + 3M(n) + 12$ many input gates (and thus input wires) collected as blocks of sizes

$$\Lambda(n), \Lambda(n), 2\Lambda(n) + 3\Diamond(n) + 6, M(n), M(n), M(n), 1, 1, 1, 1, 1, 1,$$

where $\Diamond(n) = \lceil \log(2^{\Lambda(n)+1} + \Delta(n)) \rceil$, as it was in Definition 5.58.

Thus, $\mathcal{C}$ defines a 6-decoupled CNF $\varphi_{\mathcal{C}}$ (Definition 5.51) on 6 blocks of variables parametrized by

$$\mathbb{F}_2^{\Lambda(n)}, \mathbb{F}_2^{\Lambda(n)}, \mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6}, \mathbb{F}_2^{M(n)}, \mathbb{F}_2^{M(n)}, \mathbb{F}_2^{M(n)}.$$

2. Fix $a^{\mathfrak{R}}, b^{\mathfrak{R}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ and $O \colon \mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6} \to \mathbb{F}_2$. Then, there are $w, w', w'' \colon \mathbb{F}_2^{M(n)} \to \mathbb{F}_2$ such that

$$\varphi_{\mathcal{C}}(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O, w, w', w'') = 1 \tag{151}$$

if and only if the triangulated output indicator $\mathcal{L}^*$ (Definition 5.58) outputs 1 on input $(\mathcal{V}, \Lambda, \Delta, n, x, y, a^{\mathfrak{R}}, b^{\mathfrak{R}}, O)$ in **time at most** $T(n)$.

Remark 5.61. Unsurprisingly, the circuit in Definition 5.60 is very similar to the one produced by the scaled up Cook–Levin (Theorem 5.32) to resolve BinaryTimeHalt (Definition 5.10) for the instance $(\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, x, y, \cdot, \cdot, \cdot), T(n))$ (with some bounds on the various parameters of the resulting circuit). Indeed, the next proposition is just a careful application of the scaled up Cook–Levin theorem to this specific instance of BinaryTimeHalt, but which results, as needed in Definition 5.60, with the encoding of a 6-decoupled CNF instead of a 3CNF.

Proposition 5.62 (Algorithmic generation of a succinct description for the triangulated output indicator $\mathcal{L}^*$). *There is a TM* SuccinctTOI *("succinct triangulated output indicator") that takes as input*

$$(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, x, y),$$

where $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *is (the encoding of) an h-level TNFV, Λ, Δ, T and Q (the encodings of) always halting 1-input TMs, D and n positive integers and x, y bit-strings, and outputs a tuple $(M, s, \mathcal{C})$ consisting of two (binary) integers — which we call the* block size M *and the* number of non-input wires s *for reasons to be understood later — and (the encoding of) a circuit $\mathcal{C}$, such that:*

(1) *Properties of the block size and number of non-input wires: The integers M and s are independent of the inputs $\mathcal{V}, \Lambda,$ $\Delta, x, y,$ and we have*

$$M, s = \mathrm{poly}(\log(T(n)), Q(n), D). \tag{152}$$

(2) *Runtime bound: We have*

$$\mathbb{T}(\mathsf{SuccinctTOI}; \mathcal{V}, \Lambda, \Delta, D, T, Q, n, x, y) = \mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), n, \log(T(n)), Q(n), D). \tag{153}$$

In particular, the runtime is independent of $\mathcal{V}, x$ and y (which means it may not even read them completely).

(3) Properties of the resulting circuit: If

$$|\mathtt{x}|, |\mathtt{y}| \le Q(n) \quad , \quad 2\Lambda(n) + 3\Diamond(n) + 6 \le \log(T(n)) \quad \text{and} \quad |\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \le D \,, \qquad (154)$$

where $\Diamond(n) = \lceil \log(2^{\Lambda(n)+1} + \Delta(n)) \rceil$ as before, then the output circuit $\mathcal{C}$ succinctly describes $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)$ with parameters $(T(n), M, s)$ (Definition 5.60). In particular, $\mathcal{C}$ has s many non-input wires, and $4\Lambda(n) + 3\Diamond(n) + 3M + 12$ many input gates collected as blocks of sizes

$$\Lambda(n), \Lambda(n), 2\Lambda(n) + 3\Diamond(n) + 6, M, M, M, 1, 1, 1, 1, 1, 1 \,.$$

Remark 5.63. Throughout this section, we keep the inputs Λ, Δ, D, T and Q fixed, in which case the M and s calculated by SuccinctTOI depend only on n, and we use the notation $M(n)$ and $s(n)$ for them.

Proof sketch of Proposition 5.62. Let us start with a short discussion on how $\mathcal{C}$ is constructed, so that the conditions from (154), as well as the fact M and s can be taken to be of size (152), will become clearer to the reader. As mentioned in the proof of Theorem 5.13, the first step in the Cook–Levin transformation is to describe the variables on which $\varphi_{\mathcal{C}}$ will be defined. These variables control the contents of the tapes at each time step up to $T(n)$, as well as the head position and the internal state of the TM. The number of these variables is polynomial in the number of time steps — $T(n)$ in our case — and the description length of the appropriate TM — $|\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)|$ in our case. As shown in (155), this number is bounded by $\mathrm{poly}(T(n), Q(n), D)$. There are more variables that need to be added to the formula, but it turns out that the number of them is also polynomial in $T(n)$ and $|\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)|$. All in all, the formula needs $\mathrm{poly}(T(n), Q(n), D)$-many variables, and hence $\mathrm{polylog}(T(n), Q(n), D)$-sized blocks of input gates suffice. As $\mathrm{polylog}(T(n), Q(n), D)$ is smaller than $\mathrm{poly}(\log(T(n)), Q(n), D)$, we can indeed choose M as above and have enough flexibility to encode the required number of variables. Similarly, the number of non-input wires in $\mathcal{C}$ can be taken to be $\mathrm{poly}(\log(T(n)), Q(n), D)$ as well. The fact that the assignments $a^{\mathfrak{R}}, b^{\mathfrak{R}}, O$ for the variables induced by the first three blocks of $\mathcal{C}$ come from inputs that make $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)$ halt and output 1 in at most $T(n)$ steps, can be enforced in a straightforward manner: There are variables of the formula $\varphi_{\mathcal{C}}$ that control the values of the input tapes at time 0, and we can just force equality of these variables with those controlling the values of $a^{\mathfrak{R}}, b^{\mathfrak{R}}$ and O in a succinct way. The high-level description (Remark 2.33) of the algorithm SuccinctTOI is thus as follows:

- First, it runs Λ, Δ, T and Q on input n to recover the values $\Lambda(n), \Delta(n), T(n)$ and $Q(n)$. This takes time at most

$$\mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n)) \,.$$

- The algorithm then calculates M and s. We do not describe this calculation in detail, but these are just fixed polynomials in $\log(T(n)), Q(n)$ and D (as needed for (152) to hold), which are large enough to play the role of the variables block size and number of non-input wires in the output of a decoupled version of the scaled up Cook–Levin Theorem 5.32, as sketched above. As this is just the calculation of fixed polynomials, it takes at most polynomial time in the **bit length** of the appropriate numbers to calculate them, namely

$$\mathrm{polylog}(\log(T(n)), Q(n), D) \,.$$

  Together with the previous step, this already shows that calculating M and s takes no more than the time bound from (153).

- After that, SuccinctTOI checks whether the conditions from (154) are satisfied, namely

$$|\mathtt{x}|, |\mathtt{y}| \le Q(n) \quad , \quad 2\Lambda(n) + 3\Diamond(n) + 6 \le \log(T(n)) \quad \text{and} \quad |\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \le D \,.$$

  This takes at most $\mathrm{poly}(\log(T(n)), Q(n), D)$ time. If these conditions are not satisfied, then there are no guarantees on the output circuit $\mathcal{C}$, and SuccinctTOI can just output some fixed constant sized circuit (e.g., the empty circuit).

- Otherwise, $\mathcal{C}$ needs to succinctly describe $\mathcal{L}^*$ with the inputs $\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}$ fixed, and with parameters $(T(n), M, s)$. To that end, it first needs to calculate the description of the 3-input TM $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)$ — by Fact 2.32 and using $|\mathcal{L}^*| = O(1)$, we have

$$|\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)| = \mathrm{poly}(|\mathcal{V}|, |\Lambda|, |\Delta|, |n|, |\mathtt{x}|, |\mathtt{y}|) = \mathrm{poly}(D, \log(n), Q(n)), \qquad (155)$$

and calculating this description takes $\mathrm{poly}(D, \log(n), Q(n))$-time as well.

As mentioned in Remark 5.61, the question of whether $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)$ will output 1 in at most $T(n)$ time steps is a 3-input version of the decision problem BinaryTimeHalt (Definition 5.10). So, it is natural to apply on it some variant of the Cook–Levin theorem, as this theorem describes a transformation from pairs of a TM and time bound (in binary) to a circuit $\mathcal{C}$, such that satisfiability of the formula $\varphi_{\mathcal{C}}$ is associated with the TM indeed outputting 1 in the respective number of time steps. As opposed to the scaled up Cook–Levin theorem that we described in Theorem 5.32, where the output circuit $\mathcal{C}$ succinctly encodes a 3SAT instance $\varphi_{\mathcal{C}}$ which is not decoupled, here we expect the resulting $\mathcal{C}$ to encode a 6-decoupled formula with some extra properties, that we describe now:

First of all, the first three blocks of inputs in the circuit $\mathcal{C}$ (of sizes $\Lambda(n), \Lambda(n)$ and $2\Lambda(n) + 3\Diamond(n) + 6$) induce three blocks of variables in the decoupled formula $\varphi_{\mathcal{C}}$ (of sizes $2^{\Lambda(n)}, 2^{\Lambda(n)}$ and $2^{2\Lambda(n)+3\Diamond(n)+6}$), and we expect the assignments to these blocks of variables to "remember" the appropriate inputs to the TM that make it halt and output 1. Namely, if $a^{\mathfrak{R}}, b^{\mathfrak{R}}, O$ are the assignments to these blocks of generators of $\varphi_{\mathcal{C}}$, then completing them to a satisfying assignment for the formula $\varphi_{\mathcal{C}}$ needs to be possible only if $\mathcal{L}^*(\mathcal{V}, \Lambda, \Delta, n, \mathtt{x}, \mathtt{y}, \cdot, \cdot, \cdot)$ halts in $T(n)$ time steps, where the three dots are replaced by $a^{\mathfrak{R}}, b^{\mathfrak{R}}, O$ — this is exactly the condition phrased in Item (3) of Definition 5.60. In addition, we need the number of input gates in $\mathcal{C}$ and the number of non-input wires in it to be of specific sizes, $4\Lambda(n) + 3\Diamond(n) + 3M + 12$ and s respectively. Both conditions can be achieved by, for example, adapting the proofs from [JNV$^+$21, Sections 10.2 and 10.3].

$\qquad$ $\square$

Remark 5.64. We decided not to include a full proof of the version of the Cook–Levin Theorem 5.32 required for Proposition 5.62. This is mainly because this transformation is fairly standard, and we tried to provide enough information for the reader to be able to reconstruct this for themselves. Moreover, the authors of [JNV$^+$21] did include a version, and anyone who seeks to prove Proposition 5.62 can adapt their version to imply the above.

Recall that in Claim 5.59 we described an equivalent condition for the game $\mathcal{V}_n$ accepting the answers $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ given the questions $\mathtt{x}, \mathtt{y}$ were asked. One of the conditions (148) was for $\mathcal{L}^*$ to halt and output 1 on a certain tuple of inputs. The whole goal of Proposition 5.62 was to replace a check such as (148) by the satisfiability of a formula succinctly represented by a circuit. Hence, the following is obtained:

Corollary 5.65 (Condition for $\mathcal{V}_n$ accepting). *Let*
 - *Λ be a single input TM that always halts;*
 - *$\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ a purified 2^{Λ}-padded h-level TNFV (Definition 5.57) such that $\mathcal{V}_n$ is well defined for every n (Definition 4.33);*
 - *Δ an always halting 1-input TM that satisfies $\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ and induces $\Diamond(n)$ as in (147);*
 - *Q an always halting 1-input TM satisfying $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \leq Q(n)$;*
 - *T an always halting 1-input TM satisfying*

$$T(n) \geq c \cdot \left(\mathbb{T}(\Lambda; n)^c + \mathbb{T}(\Delta; n)^c + 2^{c \cdot \Lambda(n)} + \Delta(n)^c + \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)^c \right), \qquad (156)$$

 where $c \geq 6$ is the positive integer implied by the poly *notation in (150);*
 - *D a positive integer (in binary) satisfying $|\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \leq D$;*
 - *n a positive integer;*
 - *$\mathtt{x}, \mathtt{y}$ two bit strings of length $r(n) = \mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$.*

– $(M(n), s(n), \mathcal{C}) = \mathsf{SuccinctTOI}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, \mathrm{x}, \mathrm{y})$, *where* $\mathsf{SuccinctTOI}$ *was defined in Proposition 5.62.*

Then:

(1) *For every* $a^{\mathfrak{R}}, b^{\mathfrak{R}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$, *there are functions*

$$O \colon \mathbb{F}_2^{2\Lambda(n)+3\Diamond(n)+6} \to \mathbb{F}_2 \quad \text{and} \quad w, w', w'' \colon \mathbb{F}_2^{M(n)} \to \mathbb{F}_2 \,,$$

such that

$$\varphi_{\mathcal{C}}(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O, w, w', w'') = 1 \,. \tag{157}$$

In addition, the O above is the same as in Claim 5.59 which encodes the 5-decoupled system in (149). Let us denote by $\mathsf{Prove}_{\mathcal{C}}$ the mapping that takes $a^{\mathfrak{R}}, b^{\mathfrak{R}}$ as inputs and outputs a 6-tuple $(a^{\mathfrak{R}}, b^{\mathfrak{R}}, O, w, w', w'')$ which satisfies $\varphi_{\mathcal{C}}$ (i.e., (157)).

(2) Completeness*: If the quadruple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ is accepted in the game $\mathcal{V}_n$ given x, y, then the 6-tuple $\mathsf{Prove}_{\mathcal{C}}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$ satisfies $\varphi_{\mathcal{C}}$ and the 5-tuple $\mathsf{Extend}(a^{\mathfrak{L}}, b^{\mathfrak{L}})$ (Corollary 5.56) satisfies the 5-decoupled system of linear equations defined by O, the third entry of the tuple $\mathsf{Prove}_{\mathcal{C}}(a^{\mathfrak{R}}, b^{\mathfrak{R}})$.*

Soundness*: On the other hand, if the 6-tuple $(f_1^{\mathfrak{R}}, f_2^{\mathfrak{R}}, f_0, f_3^{\mathfrak{R}}, f_4^{\mathfrak{R}}, f_5^{\mathfrak{R}})$ satisfies $\varphi_{\mathcal{C}}$, and $(f_1^{\mathfrak{L}}, f_2^{\mathfrak{L}}, f_3^{\mathfrak{L}}, f_4^{\mathfrak{L}}, f_5^{\mathfrak{L}})$ satisfies the 5-decoupled linear system induced by f_0, then the quadruple $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}}$ is accepted in the game $\mathcal{V}_n$ given x, y were asked.*

Proof. This is immediate from Claim 5.59, Definition 5.60 and Proposition 5.62. $\qquad\square$

5.3.3 Converting the succinct descriptions into PCPs

In this section we translate the condition from Item (2) of Corollary 5.65, which is equivalent to the tuple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ passing the nth game induced by a verifier $\mathcal{V}$ (given that it was padded and purified and satisfies certain bounds on its running time and output lengths) when asked x, y, to somewhat longer conditions that can be checked probabilistically. Namely, we construct an appropriate PCP.

Compare this situation to the one we had in Observation 5.33. There, we had a single assignment $\psi \colon \mathbb{F}_2^n \to \mathbb{F}_2$ and we wanted to verify that it satisfies the 3CNF formula $\varphi_{\mathcal{C}}$ induced by some circuit $\mathcal{C}$. To that end, we expected to have a single low degree polynomial g_{ψ} over $\mathbb{F}_q$ (some field extension of $\mathbb{F}_2$) that plays the role of ψ, and a swath of helper polynomials of two types — α's that verified that indeed the formula is satisfied, and β's that verify that g_{ψ} is an assignment (Definition 5.25). Here, we have two conditions to be checked instead of one, and we have 11 assignments instead of one — a 6-tuple $(f_1^{\mathfrak{R}}, f_2^{\mathfrak{R}}, f_0, f_3^{\mathfrak{R}}, f_4^{\mathfrak{R}}, f_5^{\mathfrak{R}})$ needs to satisfy $\varphi_{\mathcal{C}}$, and a 5-tuple $(f_1^{\mathfrak{L}}, f_2^{\mathfrak{L}}, f_3^{\mathfrak{L}}, f_4^{\mathfrak{L}}, f_5^{\mathfrak{L}})$ needs to satisfy the linear system induced by f_0. Still, we can have an 11-tuple of polynomials $g_\cdot$ over some field extension $\mathbb{F}_q$ that play the role of the $f_\cdot$, and helper polynomials of, again, two types — one type checks that indeed the formula and system of linear equations are satisfied by the tuple, and the other type checks that the $g_\cdot$'s are assignments.

As the PCP consists of polynomials, let us recall and describe some notations. The notation $\mathbb{F}_q[S]$ is the space of polynomials with variables from S and coefficients in $\mathbb{F}_q$. Furthermore, the *evaluation table* of $f \in \mathbb{F}_q[S]$ is the $\Phi_{\mathbb{F}_q}$-image of it (123), namely the function that takes an $\mathbb{F}_q$-assignment to the variables in S and evaluates the polynomial accordingly. In our PCP, the different polynomials are expected to have various, not necessarily disjoint, variable sets S.

Definition 5.66 (Blocks of variables associated to the circuit $\mathcal{C}$). Let $\Lambda, \Diamond, s$ and M be positive integers. In addition, let $\mathcal{C}$ be a circuit with s many non-input wires, as well as $4\Lambda + 3\Diamond + 3M + 12$ input gates collected in blocks of size[89]

$$2\Lambda + 3\Diamond + 6, \Lambda, \Lambda, M, M, M, 1, 1, 1, 1, 1, 1 \,. \tag{158}$$

---

[89]This choice should be compared to Definition 5.60. Note that we put the third block of the original circuit as the first block here. This is done so that the notations will be easier to follow.

Hence, the Tseitin polynomial T_C (Definition 5.28) has a total of

$$m = 4\Lambda + 3\diamondsuit + 3M + 12 + s \tag{159}$$

many variables, which we can collect in blocks of sizes

$$2\Lambda + 3\diamondsuit + 6, \Lambda, \Lambda, M, M, M, 1, 1, 1, 1, 1, 1, s \ . \tag{160}$$

We further decompose the first block, that consists of $2\Lambda + 3\diamondsuit + 6$ variables, into 11 blocks of sizes

$$\Lambda, \Lambda, \diamondsuit, \diamondsuit, \diamondsuit, 1, 1, 1, 1, 1, 1 \ . \tag{161}$$

In total, we have 23 blocks of variables. We now give to each block of variables a name. The block of size $2\Lambda + 3\diamondsuit + 6$ that we decomposed as in (161) will be denoted as S_0, and it is the disjoint union of the 11 blocks of variables[90]

$$S_1^{\mathfrak{L}}, S_2^{\mathfrak{L}}, S_3^{\mathfrak{L}}, S_4^{\mathfrak{L}}, S_5^{\mathfrak{L}}, S_6^{\mathfrak{L}}, S_7^{\mathfrak{L}}, S_8^{\mathfrak{L}}, S_9^{\mathfrak{L}}, S_{10}^{\mathfrak{L}}, S_{11}^{\mathfrak{L}} \ .$$

The other 12 blocks from (160) will be denoted by

$$S_1^{\mathfrak{R}}, S_2^{\mathfrak{R}}, S_3^{\mathfrak{R}}, S_4^{\mathfrak{R}}, S_5^{\mathfrak{R}}, S_6^{\mathfrak{R}}, S_7^{\mathfrak{R}}, S_8^{\mathfrak{R}}, S_9^{\mathfrak{R}}, S_{10}^{\mathfrak{R}}, S_{11}^{\mathfrak{R}}, S_{12}^{\mathfrak{R}} \ .$$

The disjoint union of all these blocks will be denoted by S. All in all we have

$$
\begin{aligned}
|S_1^{\mathfrak{R}}| = |S_2^{\mathfrak{R}}| = |S_1^{\mathfrak{L}}| = |S_2^{\mathfrak{L}}| &= \Lambda \ , \\
|S_3^{\mathfrak{L}}| = |S_4^{\mathfrak{L}}| = |S_5^{\mathfrak{L}}| &= \diamondsuit \ , \\
|S_3^{\mathfrak{R}}| = |S_4^{\mathfrak{R}}| = |S_5^{\mathfrak{R}}| &= M \ , \\
|S_{12}^{\mathfrak{R}}| &= s \ , \\
|S_6^{\mathfrak{L}}| = |S_7^{\mathfrak{L}}| = |S_8^{\mathfrak{L}}| = |S_9^{\mathfrak{L}}| = |S_{10}^{\mathfrak{L}}| = |S_{11}^{\mathfrak{L}}| = |S_6^{\mathfrak{R}}| = |S_7^{\mathfrak{R}}| = |S_8^{\mathfrak{R}}| = |S_9^{\mathfrak{R}}| = |S_{10}^{\mathfrak{R}}| = |S_{11}^{\mathfrak{R}}| &= 1 \ , \\
S_0 = \bigcup_{i=1}^{11} S_i^{\mathfrak{L}} \quad , \quad S = S_0 \cup \left(\bigcup_{i=1}^{12} S_i^{\mathfrak{R}} \right) \ .
\end{aligned}
\tag{162}
$$

Remark 5.67 (A sampled point for the PCP). The blocks of variables from Definition 5.66 are expected to be variables of polynomials. As polynomials in $\mathbb{F}_q[S]$ can be thought of as functions $\mathbb{F}_q^S \to \mathbb{F}_q$ (through the map $\Phi_{\mathbb{F}_q}$ (123)), we will often sample points in $\mathbb{F}_q^S$ and evaluate the polynomials at these points. We use the notation

$$p = (\underbrace{p_1^{\mathfrak{L}}, ..., p_{11}^{\mathfrak{L}}}_{p_0}, p_1^{\mathfrak{R}}, ..., p_{12}^{\mathfrak{R}}) \in \mathbb{F}_q^S \ . \tag{163}$$

Namely, p is a function from S to $\mathbb{F}_q$, and p_i^κ is the restriction of p to the block S_i^κ.

Definition 5.68 (Degree-d PCP over $\mathbb{F}_q$). Let $d, t, \Lambda, \diamondsuit, s$ and M be positive integers, and let $q = 2^t$. Let C be a circuit with s many non-input wires and $4\Lambda + 3\diamondsuit + 3M + 12$ many input gates collected in blocks as in (158). Recall the variable blocks from Definition 5.66. A (individual) *degree-d probabilistically checkable proof over* $\mathbb{F}_q$ with parameters $(\Lambda, \diamondsuit, M, s, C)$, denoted by Π and referred to as just "PCP" from now onwards, is the evaluation table of $\heartsuit = 12\Lambda + 12\diamondsuit + 6M + s + 35$ many individual degree at most d polynomials over $\mathbb{F}_q$ formatted as follows:

$$\forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, \ i \in [5] \ , X \in S_i^\kappa : \quad g_i^\kappa \ , \ \beta_{i,X}^\kappa \in \mathbb{F}_q[S_i^\kappa] \ , \tag{164}$$

$$\forall X \in S_0 : \quad g_0 \ , \ \beta_{0,X} \ , \ \alpha_X^{\mathfrak{L}} \in \mathbb{F}_q[S_0] \ , \tag{165}$$

$$\forall X \in S : \quad \alpha_X^{\mathfrak{R}} \in \mathbb{F}_q[S] \ . \tag{166}$$

---

[90]For now it should not be clear why we use the linear part notation for these blocks.

The eleven g polynomials are called the *assignments* in Π, while the α and β polynomials are called the *helpers* in Π. The linear part $\Pi^{\mathfrak{L}}$ of Π consists of the ($\heartsuit^{\mathfrak{L}} = 4\Lambda + 6\diamondsuit + 11$ many) polynomials $g_i^{\mathfrak{L}}, \alpha_{\mathsf{X}}^{\mathfrak{L}}, \beta_{i,\mathsf{X}}^{\mathfrak{L}}$, and the readable part $\Pi^{\mathfrak{R}}$ consists of the rest ($\heartsuit^{\mathfrak{R}} = 8\Lambda + 6\diamondsuit + 6M + s + 24$ many) of the polynomials, namely $g_0, \beta_{0,\mathsf{X}}, g_i^{\mathfrak{R}}, \alpha_{\mathsf{X}}^{\mathfrak{R}}, \beta_{i,\mathsf{X}}^{\mathfrak{R}}$.

As every S_i^{κ} is contained in S, every $\mathbb{F}_q[S_i^{\kappa}]$ is contained in $\mathbb{F}_q[S]$,[91] and there is a well defined notion of evaluating each of the polynomials consisting of Π — (164), (165) and (166) — at $p\colon S \to \mathbb{F}_q$ (cf. Remark 5.67). *Evaluating* Π *at* p is reading the p-evaluation of all the polynomials consisting of Π, and we denote this tuple of $\heartsuit$-many values in $\mathbb{F}_q$ by $\Pi(p)$. Namely, evaluating the PCP Π induces a function

$$\Pi\colon \mathbb{F}_q^S \to \mathbb{F}_q^{\heartsuit}\ .$$

Observation 5.69. Let $d, t, q, \Lambda, \diamondsuit, s, M$ and $\mathcal{C}$ be as in Definitions 5.66 and 5.68. Such a circuit $\mathcal{C}$ induces a 6-decoupled CNF $\varphi_{\mathcal{C}}$ (Definition 5.51), whose six blocks of variables are parametrized by $\mathbb{F}_2^{S_0}, \mathbb{F}_2^{S_1^{\mathfrak{R}}}, ..., \mathbb{F}_2^{S_5^{\mathfrak{R}}}$. By the structure assumed on S_0, a function $f_0\colon \mathbb{F}_2^{S_0} \to \mathbb{F}_2$ encodes (according to Definition 5.51) a 5-decoupled system $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ of linear equations, with the five blocks of variables being parametrized by $\mathbb{F}_2^{S_1^{\mathfrak{L}}}, ..., \mathbb{F}_2^{S_5^{\mathfrak{L}}}$. By Fact 5.52, given an 11-tuple

$$f_0\colon \mathbb{F}_2^{S_0} \to \mathbb{F}_2 \quad , \quad \forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, \ i \in [5]\colon \ f_i^{\kappa}\colon \mathbb{F}_2^{S_i^{\kappa}} \to \mathbb{F}_2\ ,$$

the tuple $(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}})$ satisfies the formula $\varphi_{\mathcal{C}}$ and the tuple $(f_1^{\mathfrak{L}}, ..., f_5^{\mathfrak{L}})$ satisfies the linear system $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ if and only if, for every

$$u = (\underbrace{u_1^{\mathfrak{L}}, ..., u_{11}^{\mathfrak{L}}}_{u_0}, u_1^{\mathfrak{R}}, ..., u_{12}^{\mathfrak{R}}) \in \mathbb{F}_2^S \tag{167}$$

(cf. Remark 5.67) the following two equations are satisfied:

$$T_{\mathcal{C}}(u)(f_0(u_0) + u_6^{\mathfrak{R}} + 1)\prod_{i=1}^{5}\left(f_i^{\mathfrak{R}}(u_i^{\mathfrak{R}}) + u_{i+6}^{\mathfrak{R}} + 1\right) = 0\ , \tag{168}$$

and

$$f_0(u_0)\left(u_{11}^{\mathfrak{L}} + \sum_{i=1}^{5} u_{i+5}^{\mathfrak{L}} f_i^{\mathfrak{L}}(u_i^{\mathfrak{L}})\right) = 0\ . \tag{169}$$

The goal of a degree d PCP Π over $\mathbb{F}_q$ (Definition 5.68) is to prove that the above two equations are indeed satisfied for every point in the subcube $\mathbb{F}_2^S$, and thus that $\varphi_{\mathcal{C}}$ and $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ are satisfied by restricting (Definition 5.17) the g polynomials in the PCP Π to the subcube. Namely, we expect the g polynomials to be assignments, so that their restrictions to the subcube can play the role of the f polynomials. To that end, the β polynomials in Π are expected to satisfy for every $p \in \mathbb{F}_q^S$ as in (163) — note that this time the point is over $\mathbb{F}_q$ and not $\mathbb{F}_2$ as u was in (167) — that

$$\forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, \ i \in [5]\colon \ g_i^{\kappa}(p_i^{\kappa})(g_i^{\kappa}(p_i^{\kappa}) + 1) = \sum_{\mathsf{X} \in S_i^{\kappa}} \mathrm{zero}_{\mathsf{X}}(p_i^{\kappa}) \cdot \beta_{i,\mathsf{X}}^{\kappa}(p_i^{\kappa}) \tag{170}$$

and

$$g_0(p_0)(g_0(p_0) + 1) = \sum_{\mathsf{X} \in S_0} \mathrm{zero}_{\mathsf{X}}(p_0) \cdot \beta_{0,\mathsf{X}}(p_0)\ , \tag{171}$$

where $\mathrm{zero}_{\mathsf{X}}(p) = p(\mathsf{X})(p(\mathsf{X}) + 1)$, as was defined in (125). In addition, the α polynomials verify that the restrictions of the g polynomials satisfy (168) and (169). Namely, for every $p \in \mathbb{F}_q^S$ as in (163),

$$T_{\mathcal{C}}(p)(g_0(p_0) + p_6^{\mathfrak{R}} + 1)\prod_{i=1}^{5}\left(g_i^{\mathfrak{R}}(p_i^{\mathfrak{R}}) + p_{i+6}^{\mathfrak{R}} + 1\right) = \sum_{\mathsf{X} \in S} \mathrm{zero}_{\mathsf{X}}(p) \cdot \alpha_{\mathsf{X}}^{\mathfrak{R}}(p) \tag{172}$$

---

[91]When embedding all the polynomials of Π in $\mathbb{F}_q[S]$, they become indifferent (Definition 5.14) to the added variable-indexes.

and

$$g_0(p_0)(p_{11}^{\mathfrak{L}} + \sum_{i=1}^{5} p_{i+5}^{\mathfrak{L}} g_i^{\mathfrak{L}}(p_i^{\mathfrak{L}})) = \sum_{\mathsf{X} \in S_0} \mathrm{zero}_{\mathsf{X}}(p_0) \cdot \alpha_{\mathsf{X}}^{\mathfrak{L}}(p_0) \,. \tag{173}$$

The next proposition exactly relates the existence of such PCPs to the satisfiability of $\varphi_{\mathcal{C}}$ and $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$, similar to the role of Proposition 5.34 in the Prelude.

Definition 5.70 (Inducing to a PCP). Let $t, q, \Lambda, \diamondsuit, s, M, \mathcal{C}$ and $S_.$ be as in Definitions 5.66, 5.68 and Observation 5.69. The map $\mathsf{Induce}_{\mathcal{C}}$ takes an 11-tuple

$$f_0 \colon \mathbb{F}_2^{S_0} \to \mathbb{F}_2 \quad , \quad \forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\} \,, \, i \in [5] \colon \; f_i^{\kappa} \colon \mathbb{F}_2^{S_i^{\kappa}} \to \mathbb{F}_2 \,,$$

and outputs the following degree 9 PCP Π over $\mathbb{F}_q$ with parameters $(\Lambda, \diamondsuit, M, s, \mathcal{C})$. First, it lets

$$g_0 = \mathrm{Ind}(f_0) \quad , \quad \forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, i \in [5] \colon \; g_i^{\kappa} = \mathrm{Ind}(f_i^{\kappa}) \,, \tag{174}$$

with $\mathrm{Ind}(\cdot)$ being the induction from Definition 5.17. Then, it defines the polynomials

$$\Psi^{\mathfrak{R}}(p) = T_{\mathcal{C}}(p)(g_0(p_0) + p_6^{\mathfrak{R}} + 1) \prod_{i=1}^{5} \left(g_i^{\mathfrak{R}}(p_i^{\mathfrak{R}}) + p_{i+6}^{\mathfrak{R}} + 1 \right) \,,$$

$$\Psi^{\mathfrak{L}}(p_0) = g_0(p_0)(p_{11}^{\mathfrak{L}} + \sum_{i=1}^{5} p_{i+5}^{\mathfrak{L}} g_i^{\mathfrak{L}}(p_i^{\mathfrak{L}})) \,, \tag{175}$$

$$\forall \kappa \in \{\mathfrak{L}, \mathfrak{R}\} \,, \, i \in [5] \colon \; \Psi_{\kappa,i}^{\mathfrak{A}}(p_i^{\kappa}) = g_i^{\kappa}(p_i^{\kappa})(g_i^{\kappa}(p_i^{\kappa}) + 1) \,,$$

$$\Psi_0^{\mathfrak{A}}(p_0) = g_0(p_0)(g_0(p_0) + 1) \,,$$

where $T_{\mathcal{C}}$ is the Tseitin polynomial associated with $\mathcal{C}$ (Definition 5.28) and $p \in \mathbb{F}_q^S$ is as in (163). As the 11 g polynomials have individual degree at most 1, and $T_{\mathcal{C}}$ has individual degree at most 3 (Remark 5.29), all of the above Ψ polynomials are of individual degree at most 9. By fixing an order on each block $S_.$ from Definition 5.66, and using the notation $\mathsf{X}_1 < \mathsf{X}_2 < \mathsf{X}_3 < \ldots$ for the list of variables in the block, we can define

$$\forall \mathsf{X}_i \in S \colon \; \alpha_{\mathsf{X}_i}^{\mathfrak{R}} = \mathrm{Div}_{\mathsf{X}_i(\mathsf{X}_i+1)} \circ \mathrm{Mod}_{\mathsf{X}_{i-1}(\mathsf{X}_{i-1}+1)} \circ \ldots \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(\Psi^{\mathfrak{R}}) \,,$$

$$\forall \mathsf{X}_i \in S_0 \colon \; \alpha_{\mathsf{X}_i}^{\mathfrak{L}} = \mathrm{Div}_{\mathsf{X}_i(\mathsf{X}_i+1)} \circ \mathrm{Mod}_{\mathsf{X}_{i-1}(\mathsf{X}_{i-1}+1)} \circ \ldots \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(\Psi^{\mathfrak{R}}) \,,$$

$$\forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\} \,, \, j \in [5] \,, \mathsf{X}_i \in S_j^{\kappa} \colon \; \beta_{j,\mathsf{X}_i}^{\kappa} = \mathrm{Div}_{\mathsf{X}_i(\mathsf{X}_i+1)} \circ \mathrm{Mod}_{\mathsf{X}_{i-1}(\mathsf{X}_{i-1}+1)} \circ \ldots \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(\Psi_{\kappa,j}^{\mathfrak{A}}) \,, \tag{176}$$

$$\forall \mathsf{X}_i \in S_0 \colon \; \beta_{0,\mathsf{X}_i} = \mathrm{Div}_{\mathsf{X}_i(\mathsf{X}_i+1)} \circ \mathrm{Mod}_{\mathsf{X}_{i-1}(\mathsf{X}_{i-1}+1)} \circ \ldots \circ \mathrm{Mod}_{\mathsf{X}_1(\mathsf{X}_1+1)}(\Psi_0^{\mathfrak{A}}) \,,$$

where Div and Mod were defined in the proof of Claim 5.27. As Div and Mod can only decrease the individual degree of the polynomial on which they are applied, the resulting α and β polynomial have individual degree at most 9. All in all, $\mathsf{Induce}_{\mathcal{C}}$ recovered a degree 9 PCP over $\mathbb{F}_q$.

Proposition 5.71 (A probabilistically checkable proof for the combined succinct 6-decoupled SAT and 5-decoupled linear system). *Let $t, q, \Lambda, \diamondsuit, s, M, \mathcal{C}$ and $S_.$ be as in Definitions 5.66, 5.68, 5.70 and Observation 5.69. Then:*

- Completeness*: Given an* 11*-tuple*

$$f_0 \colon \mathbb{F}_2^{S_0} \to \mathbb{F}_2 \quad , \quad \forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\} \,, \, i \in [5] \colon \; f_i^{\kappa} \colon \mathbb{F}_2^{S_i^{\kappa}} \to \mathbb{F}_2 \,,$$

the degree 9 *PCP* $\Pi = \mathsf{Induce}_{\mathcal{C}}(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}, f_1^{\mathfrak{L}}, ..., f_5^{\mathfrak{L}})$ *over* $\mathbb{F}_q$ *with parameters* $(\Lambda, \diamondsuit, M, s, \mathcal{C})$ *from Definition 5.70 satisfies that*

- $g_\circ^\circ = \mathrm{Ind}(f_\circ^\circ)$;
- *the equations* (170), (171) *are satisfied by the polynomials in* Π *for every* $p \in \mathbb{F}_q^S$;
- *the readable part* $\Pi^{\mathfrak{R}}$ *of* Π *depends only on* $f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}$;
- *for fixed* $f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}$, *the map* $\mathsf{Induce}_{\mathcal{C}}(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}, \cdot, \cdot, \cdot, \cdot, \cdot)$ *from* $\bigoplus_{i=1}^{5}(\mathbb{F}_2)^{\mathbb{F}_2^{S_i^{\mathfrak{L}}}}$ *to* $(\mathbb{F}_q^{\heartsuit})^{\mathbb{F}_q^S}$, *thought of as a vector space over* $\mathbb{F}_2$, *is* $\mathbb{F}_2$-*linear*;
- *if* $\varphi_{\mathcal{C}}(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}) = 1$, *then* (172) *is satisfied by* Π *for every* $p \in \mathbb{F}_q^S$;
- *if* $(f_1^{\mathfrak{L}}, ..., f_5^{\mathfrak{L}})$ *satisfy the* 5-*decoupled system of linear equations* $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ *induced by* f_0, *then* (173) *is satisfied by* Π *for every* $p \in \mathbb{F}_q^S$.

- Soundness*: Let* $d \geq 3$. *If* Π *is a degree-d PCP over* $\mathbb{F}_q$ *with parameters* $(\Lambda, \Diamond, M, s, \mathcal{C})$ *that passes each of the checks* (170), (171), (172) *and* (173) *with probability strictly larger than* $\frac{7dm}{q}$ *(with* $m = |S|$ *from* (159)*) over the choice of a uniformly random* $p \colon S \to \mathbb{F}_q$, *then it passes them with probability* 1. *This in turn means that by taking* $f_\circ^\circ = \mathrm{Res}(g_\circ^\circ)$, *the resulting* 11-*tuple satisfies both* $\varphi_{\mathcal{C}}$ *and* $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$.

Proof. **Completeness**:

Let us begin by proving the first 4 claimed properties of $\Pi = \mathsf{Induce}_{\mathcal{C}}(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}, f_1^{\mathfrak{L}}, ..., f_5^{\mathfrak{L}})$, as they are non-conditional. The fact $g_\circ^\circ = \mathrm{Ind}(f_\circ^\circ)$ is by construction, see (174). As the g polynomials are inductions of functions from $\mathbb{F}_2^\circ \to \mathbb{F}_2$, they are assignments, and thus the polynomials $\Psi_{\circ,\circ}^{\mathfrak{A}}$ from (175) are zero on the subcube (Definition 5.25). Hence, by the proof of the Combinatorial Nullstellensatz (Claim 5.27), equations (170), (171) are satisfied by the polynomials in Π for every $p \in \mathbb{F}_q^S$. Recall that the readable part $\Pi^{\mathfrak{R}}$ of Π consists of the polynomials $g_0, g_i^{\mathfrak{R}}, \alpha_{\mathsf{X}}^{\mathfrak{R}}, \beta_{0,\mathsf{X}}$ and $\beta_{i,\mathsf{X}}^{\mathfrak{R}}$. The fact the readable g polynomials depend only on $f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}$ is immediate. For the other polynomials, note that $\Psi^{\mathfrak{R}}, \Psi_0^{\mathfrak{A}}$ and $\Psi_{\kappa,i}^{\mathfrak{A}}$ from (175) depend only on $f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}$, and thus taking Div and Mod with respect to fixed polynomials still depends only on them. Finally, let us address the required properties of the linear part $\Pi^{\mathfrak{L}}$ of Π, which needs to depend $\mathbb{F}_2$-linearly on $f_i^{\mathfrak{L}}$ given $f_0, f_i^{\mathfrak{R}}$ were fixed. First, Ind is a $\mathbb{F}_q$-linear map (Remark 5.18), which guarantees that the values of the $g_i^{\mathfrak{L}}$ polynomials depend linearly on those of $f_i^{\mathfrak{L}}$. Now, as f_0 is fixed, g_0 is fixed as well and the values of $\Psi^{\mathfrak{L}}$ are by construction linear combinations of the values of $g_i^{\mathfrak{L}}$. As Div and Mod are $\mathbb{F}_q$-linear functions, the values of the $\alpha_{\mathsf{X}}^{\mathfrak{L}}$ polynomials depend linearly on $f_i^{\mathfrak{L}}$. The map $\mathsf{Y} \mapsto \mathsf{Y}^2 + \mathsf{Y}$ is $\mathbb{F}_2$-linear over $\mathbb{F}_q$, which means that the values of $\Psi_{\mathfrak{L},i}^{\mathfrak{A}}$ depend $\mathbb{F}_2$-linearly on the values of $g_i^{\mathfrak{L}}$. Using again the fact that Div and Mod are $\mathbb{F}_q$-linear proves that the $\beta_{i,\mathsf{X}}^{\mathfrak{L}}$ polynomials depend $\mathbb{F}_2$-linearly on the values of $f_i^{\mathfrak{L}}$. All in all, for a fixed $f_0, f_i^{\mathfrak{R}}$, the readable part $\Pi^{\mathfrak{R}}$ is fixed and the linear part $\Pi^{\mathfrak{L}}$ is an $\mathbb{F}_2$-linear combination of the values of $f_i^{\mathfrak{L}}$.

For the last two claimed properties of Π, as described in Observation 5.69, the fact that $\varphi_{\mathcal{C}}$ or $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ are satisfied by the 11-tuple of $f_\circ^\circ$'s, implies that (168) or (169) are (respectively) satisfied for every $u \in \mathbb{F}_2^S$ (as in (167)). Hence, $\Psi^{\mathfrak{R}}$ or $\Psi^{\mathfrak{L}}$ are zero on their respective subcubes, and by the Combinatorial Nullstellensatz (Claim 5.27), our choice of helper polynomials in Π make (172) or (173) perfectly satisfied, as claimed.

Soundness:

Let us analyze the 13 polynomials

$$\clubsuit^{\mathfrak{R}}(p) = T_{\mathcal{C}}(p)(g_0(p_0) + p_6^{\mathfrak{R}} + 1) \prod_{i=1}^{5} \left(g_i^{\mathfrak{R}}(p_i^{\mathfrak{R}}) + p_{i+6}^{\mathfrak{R}} + 1 \right) - \sum_{\mathsf{X} \in S} \mathsf{zero}_{\mathsf{X}}(p) \cdot \alpha_{\mathsf{X}}^{\mathfrak{R}}(p) \,,$$

$$\clubsuit^{\mathfrak{L}}(p_0) = g_0(p_0)(p_{11}^{\mathfrak{L}} + \sum_{i=1}^{5} p_{i+5}^{\mathfrak{L}} g_i^{\mathfrak{L}}(p_i^{\mathfrak{L}})) - \sum_{\mathsf{X} \in S_0} \mathsf{zero}_{\mathsf{X}}(p_0) \cdot \alpha_{\mathsf{X}}^{\mathfrak{L}}(p_0) \,,$$

$$\clubsuit_{\circ,\circ}^{\mathfrak{A}}(p_\circ^\circ) = g_\circ^\circ(p_\circ^\circ)(g_\circ^\circ(p_\circ^\circ) + 1) - \sum_{\mathsf{X} \in S_\circ^\circ} \mathsf{zero}_{\mathsf{X}}(p_\circ^\circ) \cdot \beta_{\circ,\mathsf{X}}^\circ(p_\circ^\circ) \,.$$

As Π is of (individual) degree d, $\mathsf{zero}_{\mathsf{X}}$ is of individual degree $2 \leq d$, and $T_{\mathcal{C}}$ is of individual degree $3 \leq d$, all of the $\clubsuit$ polynomials are of individual degree at most $7d$, and are thus of total degree at most $7md$ (as the number of variables

in each of them is at most $m = |S|$). By the assumptions on Π passing each of the checks (170), (171), (172) and (173) with probability strictly larger than $\frac{7dm}{q}$, the proportion of roots of each $\clubsuit$ polynomial is strictly larger than $\frac{7dm}{q}$. By the Schwartz–Zippel Lemma 5.19, this implies all the $\clubsuit$ polynomials are identically zero. From the fact that $\clubsuit_{\circ,\circ}^{\mathfrak{A}}$ is identically zero, we deduce that $\Psi_{\circ,\circ}^{\mathfrak{A}}$ is zero on the subcube, and thus $g_{\circ}^{\circ}$ is an assignment (Definition 5.25). Hence, $f_{\circ}^{\circ} = \mathrm{Res}(g_{\circ}^{\circ})$ outputs only values in $\mathbb{F}_2$. As $\clubsuit^{\mathfrak{R}}$ and $\clubsuit^{\mathfrak{L}}$ are identically zero, we deduce that (168) and (169) are satisfied, and thus the tuple of f polynomials indeed satisfy $\varphi_{\mathcal{C}}$ and $(\mathscr{A}_{f_0} \mid \vec{b}_{f_0})$ (by Observation 5.69). This finishes the proof. $\qquad\square$

Corollary 5.72 (The functional viewpoint for $\mathcal{V}_n$ accepting)**.** *Let*
- *Λ be a single input TM that always halts;*
- *$\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ a purified 2^{Λ}-padded h-level TNFV (Definition 5.57) such that $\mathcal{V}_n$ is well defined for every n (Definition 4.33);*
- *Δ an always halting 1-input TM that satisfies $\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ and induces $\diamondsuit(n)$ as in (147);*
- *Q an always halting 1-input TM satisfying $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot) \leq Q(n)$;*
- *T an always halting 1-input TM satisfying*

$$T(n) \geq c \cdot \left(\mathbb{T}(\Lambda; n)^c + \mathbb{T}(\Delta; n)^c + 2^{c \cdot \Lambda(n)} + \Delta(n)^c + \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)^c\right), \qquad (177)$$

  where $c \geq 6$ is the positive integer implied by the poly *notation in (150);*
- *D a positive integer (in binary) satisfying $|\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \leq D$;*
- *n and t positive integers, and $q = 2^t$;*
- *x, y two bit strings of length $r(n) = \mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$;*
- *$(M(n), s(n), \mathcal{C}) = \mathsf{SuccinctTOI}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, \mathrm{x}, \mathrm{y})$, where $\mathsf{SuccinctTOI}$ was defined in Proposition 5.62.*

Then:

(1) Completeness*: For every quadruple $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$, there is a degree-9 PCP Π over $\mathbb{F}_q$ (Definition 5.68) which satisfies the equations (170), (171) and (172) for every $p \in \mathbb{F}_q^{\mathcal{S}}$,[92] and such that*

$$\forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\} : \quad \mathrm{Ind}(a^{\kappa}) = g_1^{\kappa} \quad , \quad \mathrm{Ind}(b^{\kappa}) = g_2^{\kappa} ,$$

where $g_{\cdot}^{\cdot}$ are the appropriate polynomials in Π. Furthermore, the readable part $\Pi^{\mathfrak{R}}$ of the PCP depends only on $a^{\mathfrak{R}}, b^{\mathfrak{R}}$, while the linear part $\Pi^{\mathfrak{L}}$ of the PCP depends on $a^{\mathfrak{L}}, b^{\mathfrak{L}}$ in an $\mathbb{F}_2$-affine manner (which may depend on $a^{\mathfrak{R}}, b^{\mathfrak{R}}$); namely, by choosing a basis of $\mathbb{F}_q$ over $\mathbb{F}_2$, the bit representation of $\Pi^{\mathfrak{L}}$ is an $\mathbb{F}_2$-affine combination of the bit representation of $a^{\mathfrak{L}}, b^{\mathfrak{L}}$. In addition, if $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ are accepted in the game $\mathcal{V}_n$ given x, y were asked, then (173) is also satisfied by Π for every $p \in \mathbb{F}_q^{\mathcal{S}}$.

For later use, we denote by $\mathsf{PCP}_{\mathrm{xy}}$ the mapping that takes $a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}$ as inputs and outputs this promised degree 9 PCP Π.

(2) Soundness*: If there is a degree 9 PCP Π over $\mathbb{F}_q$ that passes each of the checks (170), (171), (172) and (173) with probability strictly larger than $\frac{63m}{q}$ (with $m = |S|$ as in (159)), then the quadruple $\mathrm{Res}(g_1^{\mathfrak{R}}), \mathrm{Res}(g_1^{\mathfrak{L}}), \mathrm{Res}(g_2^{\mathfrak{R}}), \mathrm{Res}(g_2^{\mathfrak{L}})$ passes the game $\mathcal{V}_n$ given x, y were asked.*

Proof. This is a combination of Corollary 5.65 and Proposition 5.71. Let us just spell out explicitly how $\mathsf{PCP}_{\mathrm{xy}}$ operates. First, it lets $\mathsf{Prove}_{\mathcal{C}}(a^{\mathfrak{R}}, b^{\mathfrak{R}}) = (f_1^{\mathfrak{R}}, f_2^{\mathfrak{R}}, f_0, f_3^{\mathfrak{R}}, f_4^{\mathfrak{R}}, f_5^{\mathfrak{R}})$. It then calculates $\Delta(n)$ and $\mathcal{L}(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$ to retrieve $L_{\mathrm{xy}}(a^{\mathfrak{R}}, b^{\mathfrak{R}}) = (\mathscr{A} \mid \vec{b})$ and thus its unreadable part $(\mathscr{A}^{\mathfrak{L}} \mid \vec{b})$. Then, the function Extend from Corollary 5.56 is well defined with respect to $(\mathscr{A}^{\mathfrak{L}} \mid \vec{b})$ and $\Delta(n)$, and we let $\mathsf{Extend}(a^{\mathfrak{L}}, b^{\mathfrak{L}}) = (f_1^{\mathfrak{L}}, f_2^{\mathfrak{L}}, f_3^{\mathfrak{L}}, f_4^{\mathfrak{L}}, f_5^{\mathfrak{L}})$. Finally $\mathsf{PCP}_{\mathrm{xy}}$ outputs

$$\Pi = \mathsf{Induce}_{\mathcal{C}}(f_0, f_1^{\mathfrak{R}}, ..., f_5^{\mathfrak{R}}, f_1^{\mathfrak{L}}, ..., f_5^{\mathfrak{L}}) . \qquad\square$$

---

[92]Note, and this is crucial, that (173) may not be satisfied in this case

5.4   The low individual degree test for MIP^* protocols

A crucial, and technically involved, part of the proof of $\mathsf{MIP} = \mathsf{NEXP}$ (described in the Prelude 5.1) is to verify that the functions involved in the proof are indeed low individual degree polynomials (Theorem 5.23). The low individual degree test for MIP^* protocols, or just low degree test from now onwards, is, as its name suggests, a non-local game analogue of the test described in Theorem 5.23. Namely, it is designed to verify that a player in a MIP^* protocol returns an answer that can be interpreted as the simultaneous evaluation of a tuple of individual degree-d polynomials $(g_1, \ldots, g_k)$ at a point $z \in \mathbb{F}_q^m$. The degree bound d, the number of polynomials k, the field size $q = 2^t$ and the number of variables m are all parameters of this game. The soundness proof for this test is involved, and is the main theorem of [JNV$^+$22b].

Preliminaries   Recall the notions of polynomial degrees (Definition 5.14), lines in $\mathbb{F}_q^m$ (Definition 5.20) and the characterization of low degree polynomials via restrictions to lines (Fact 5.21). There is no canonical way of choosing "orthogonal projections" and their "complements" in a finite vector space. So, we need to agree how to choose them in a consistent manner. The following definition provides a way of choosing such maps, which will be canonical for us.

Definition 5.73 (Canonical linear maps). Let $\mathscr{B} = \{v_1, \ldots, v_k\} \subseteq \mathbb{F}_q^m$ be a set of linearly independent vectors. Complete them to a basis of $\mathbb{F}_q^m$ as follows — at each step, add to the set the standard basis vector e_i with the largest possible i so that the new sequence is still linearly independent.[93]   So, we now have a basis $v_1, \ldots, v_k, e_{i_1}, \ldots, e_{i_{m-k}}$. The canonical projection $\mathfrak{null}_{\mathscr{B}}$ with kernel basis $\mathscr{B}$ is the map that takes a vector v, writes it as a linear combination $\sum \alpha_i v_i + \sum \beta_j e_{i_j}$ (in the unique way), and returns $\sum \beta_j e_{i_j}$.[94] When $\mathscr{B}$ is a single vector v, we denote $\mathfrak{null}_v$ instead of $\mathfrak{null}_{\{v\}}$.

Fact 5.74. *Let $u \in \mathbb{F}_q^m$ and $\vec{0} \neq v \in \mathbb{F}_q^m$. Then, there is a point $u_0 \in \mathscr{L}(u, v)$ such that for every $u' \in \mathscr{L}(u, v)$ we have $\mathfrak{null}_v(u') = u_0$.*

Definition 5.75 (Canonical representation of a line). Let $\mathscr{L} \subseteq \mathbb{F}_q^m$ be a line in direction $v \neq \vec{0}$. Then, the canonical representation of $\mathscr{L}$ is $\mathscr{L}(u_0, v_0)$, where $u_0 = \mathfrak{null}_v(\mathscr{L})$ and v_0 is a non-zero scalar multiple of v which is biggest in lexicographic order.[95]

The low individual degree game   We begin by defining the game $\mathfrak{LowDegree}(d, q, m, 1)$, which is designed to check that the provers hold a single global $f\colon \mathbb{F}_q^m \to \mathbb{F}_q$ which is induced by a polynomial of individual degree at most d, and answer according to its restriction to various points and lines. The general case of $\mathfrak{LowDegree}(d, q, m, k)$, which checks that k functions $f_1, \ldots, f_k\colon \mathbb{F}_q^m \to \mathbb{F}_q$ are of individual degree at most d, will be explained afterwards.

The idea of the game is similar to the classical case (Theorem 5.23), namely to use the truth tables of restrictions of f to lines and points, checking that they are consistent with one another, and that the lines satisfy condition 2 in Fact 5.21. In the quantum case, we need to add some diagonal line checks, where the diagonal lines are sampled according to a somewhat peculiar distribution. These checks force commutation between all of the observables in the game. This should be seen as a quirk of the proof in [JNV$^+$22b], and we do not have much insight to it except that it allows the inductive step therein to work.

The vertices in the underlying graph are of three types: `Point`, `ALine` (axis parallel line), and `DLine` (diagonal line). The `Point` vertices are parametrized by $\mathbb{F}_q^m$, namely $\{\mathtt{Point}^u \mid u \in \mathbb{F}_q^m\}$. The `ALine` vertices are parametrized by axis parallel lines, namely $\{\mathtt{ALine}^{\mathscr{L}} \mid \mathscr{L} \text{ is an axis parallel line} \subseteq \mathbb{F}_q^m\}$. Finally, the `DLine` vertices are parametrized by diagonal lines, namely $\{\mathtt{DLine}^{\mathscr{L}} \mid \mathscr{L} \text{ is any line} \subseteq \mathbb{F}_q^m\}$. We now specify the generators associated to each vertex, which also determines the length functions of the game, and we provide some notations that will clarify both this game as well as the Answer Reduced game (Section 5.5). The game $\mathfrak{LowDegree}(d, q, m, 1)$ is an LCS (recall Example 2.29), which means it can

---

[93]This agrees with the canonical complement of a set defined in [JNV$^+$21, Definition 3.6].

[94]This agrees with [JNV$^+$21, Definition 3.10].

[95]The reason to choose the biggest and not smallest element in the lexicographic order is that the number 1 in $\mathbb{F}_q$ is always maximal in lexicographic order when considering the basis from Fact 5.24 — this results in the vector $v_0 = (0, \ldots, 0, 1, \ldots)$, which is a somewhat natural choice.

Question Content	Formal variables (unreadable)	Interpretation of answer
$\mathtt{Point}^u$, $u \in \mathbb{F}_q^m$	$S_{\mathtt{Point}^u}^{\mathfrak{L}} = \{\mathtt{Point}^{u,j,i} \mid j \in [k]$, $i \in [t]\}$	$f_1(u), ..., f_k(u)$, where each $f_j(u) \in \mathbb{F}_q$
$\mathtt{ALine}^{\mathscr{L}}$, $\mathscr{L}$ is an axis-parallel line in $\mathbb{F}_q^m$	$S_{\mathtt{ALine}^{\mathscr{L}}}^{\mathfrak{L}} = \{\mathtt{ALine}^{\mathscr{L},j,s,i} \mid j \in [k], 0 \le s \le d, i \in [t]\}$	$(\mathsf{AL}f_1(\mathscr{L}), ... \mathsf{AL}f_k(\mathscr{L}))$, where $\mathsf{AL}f_j(\mathscr{L}) = (a_{j,0}, ..., a_{j,d}) \in \mathbb{F}_q^{d+1}$ encodes a degree d univariate polynomial $\sum a_{j,s} \mathsf{X}^s$.
$\mathtt{DLine}^{\mathscr{L}}$, $\mathscr{L}$ is a line in $\mathbb{F}_q^m$	$S_{\mathtt{DLine}^{\mathscr{L}}}^{\mathfrak{L}} = \{\mathtt{DLine}^{\mathscr{L},j,s,i} \mid j \in [k]$, $0 \le s \le md$, $i \in [t]\}$	$(\mathsf{DL}f_1(\mathscr{L}), ... \mathsf{DL}f_k(\mathscr{L}))$, where $\mathsf{DL}f_j(\mathscr{L}) = (a_{j,0}, ..., a_{j,md}) \in \mathbb{F}_q^{md+1}$ encodes a degree md univariate polynomial $\sum a_{j,s} \mathsf{X}^s$.

1. (**Axis-parallel line-versus-point test**) If $\mathtt{ALine}^{\mathscr{L}} - \mathtt{Point}^u$ was sampled, then $u \in \mathscr{L} = \mathscr{L}(u_0, e_i)$ and thus $u = u_0 + \alpha e_i$. For $s \in \{0, ..., d\}$ let $a_{j,s} = \mathsf{AL}f_j(\mathscr{L})_s$. Check that for every $j \in [k]$, $f_j(u) = \sum_{s=0}^{d} a_{j,s} \alpha^s$.

2. (**Diagonal line-versus-point test**) If $\mathtt{DLine}^{\mathscr{L}} - \mathtt{Point}^u$ was sampled, then $u \in \mathscr{L} = \mathscr{L}(u_0, v_0)$ and thus $u = u_0 + \alpha v_0$. For $s \in \{0, ..., d\}$ let $a_{j,s} = \mathsf{DL}f_j(\mathscr{L})_s$. Check that for every $j \in [k]$, $f_j(u) = \sum_{s=0}^{md} a_{j,s} \alpha^s$.

Figure 16: Description of the low degree game $\mathfrak{LowDegree}(d, q, m, k)$.

be tailored by making all variables unreadable. Recall that $q = 2^t$, and using the basis from Fact 5.24, an element of $\mathbb{F}_q$ is encoded as a length-t bit string.

- For $\mathtt{Point}^u$, we set $S_{\mathtt{Point}^u}^{\mathfrak{L}} = \{\mathtt{Point}^{u,i} : i \in [t]\}$. Here, $\mathtt{Point}^{u,i}$ represents the ith bit of the $\mathbb{F}_q$-value assigned to the point u by the supposed low individual degree polynomial f which controls the answers of the players. Hence, if γ is the assignment to the variables, we denote $f(u) = \gamma(\mathtt{Point}^{u,i})_{i=1}^t \in \mathbb{F}_q$.

- For $\mathtt{ALine}^{\mathscr{L}}$, we set $S_{\mathtt{ALine}^{\mathscr{L}}}^{\mathfrak{L}} = \{\mathtt{ALine}^{\mathscr{L},j,i} : 0 \le j \le d, i \in [t]\}$. As the restriction of an individual degree-d polynomial to an axis parallel line is a univariate polynomial of degree at most d (Fact 5.21), it can be written as $a_0 + a_1\alpha + a_2\alpha^2 + ... + a_d\alpha^d$ with the a_i being $\mathbb{F}_q$-coefficients. The value of the variable $\mathtt{ALine}^{\mathscr{L},j,i}$ is interpreted as the ith bit of the encoding of a_j in the supposed restriction of the global low degree f to the line $\mathscr{L}$.[96] Hence, if γ is the assignment to the variables, we denote $\mathsf{AL}f(\mathscr{L}) = (a_0, ..., a_d) = (\gamma(\mathtt{ALine}^{\mathscr{L},j,i})_{i=1}^t)_{j=0}^d \in \mathbb{F}_q^{d+1}$.

- For $\mathtt{DLine}^{\mathscr{L}}$, we set $S_{\mathtt{DLine}^{\mathscr{L}}}^{\mathfrak{L}} = \{\mathtt{DLine}^{\mathscr{L},j,i} : 0 \le j \le md, i \in [t]\}$. As an individual degree-d polynomial is a total degree at most md polynomial, and the restriction of such a polynomial to a line is a univariate polynomial of degree at most md (Fact 5.21), it can be written as $a_0 + a_1\alpha + a_2\alpha^2 + ... + a_{md}\alpha^{md}$ with the a_i being $\mathbb{F}_q$-coefficients. The value of the variable $\mathtt{DLine}^{\mathscr{L},j,i}$ is interpreted as the ith bit of the encoding of a_j in the supposed restriction of the global low degree f to the line $\mathscr{L}$. Hence, if γ is the assignment to the variables, we denote $\mathsf{DL}f(\mathscr{L}) = (a_0, ..., a_{md}) = (\gamma(\mathtt{ALine}^{\mathscr{L},j,i})_{i=1}^t)_{j=0}^{md} \in \mathbb{F}_q^{md+1}$.

The underlying graph of $\mathfrak{LowDegree}(d, q, m, 1)$ is induced by the incidence relation between points and lines. Namely, $\mathtt{Point}^u$ is connected to $\mathtt{ALine}^{\mathscr{L}}$ (respectively $\mathtt{DLine}^{\mathscr{L}}$) if and only if $u \in \mathscr{L}$. For the sampling scheme of edges, let us

---

[96]Here it is important that we fixed a canonical representation to each line, as the restriction of a polynomial to a line depends on its representation (see Fact 5.21).

provide a typed 3-level CLM (Definition 4.38) that describes it exactly. The type set consists of three types `Point`, `ALine` (axis parallel line), and `DLine` (diagonal line), and the type graph contains all loops as well as the edges `Point` − `ALine` and `Point` − `DLine`. The dimension of the space the CLMs act on is $(2m+1)\log q = (2m+1)t$, and by using the basis guaranteed by Fact 5.24, we can interpret each element from this space unambiguously as a triple in $\mathbb{F}_q^m \times \mathbb{F}_q \times \mathbb{F}_q^m$.

- The CLM $\mathfrak{s}^{\texttt{Point}}$ is 1-level, and is defined by

$$\forall u, v \in \mathbb{F}_q^m, \, s \in \mathbb{F}_q : \quad \mathfrak{s}^{\texttt{Point}}(u, s, v) = (u, 0, 0) \, . \tag{178}$$

  Namely, the vertex $(\texttt{Point}, u, 0, 0)$ corresponds to the vertex $\texttt{Point}^u$ introduced above.

- The CLM $\mathfrak{s}^{\texttt{ALine}}$ is 2-level, and is defined by:

$$\forall u, v \in \mathbb{F}_q^m, \, s \in \mathbb{F}_q : \quad \mathfrak{s}^{\texttt{ALine}}(u, s, v) = \left(\mathrm{null}_{e_{\chi(s)}}(u), s, 0 \right) , \tag{179}$$

  where $\mathrm{null}.$ is the canonical map with kernel $\cdot$ (Definition 5.73), $e.$ is the appropriate standard basis element of $\mathbb{F}_q^m$, and $\chi(s)$ is one more than the residue of the devision of s by m, namely

$$\chi(s) = 1 + (s \pmod m) \in [m] \, , \tag{180}$$

  where $s \in \mathbb{F}_q$ is associated with the integer with the same binary representation (again, according to the fixed basis of $\mathbb{F}_q$ over $\mathbb{F}_2$ chosen in Fact 5.24). The resulting pair maps naturally to a canonical representation (Definition 5.75) of an axis parallel line $\mathscr{L} = \mathscr{L}(u_0, e_i)$, where $i = \chi(s)$. Namely, the vertex $(\texttt{ALine}, u_0, s, 0)$ corresponds to (a copy of) the vertex $\texttt{ALine}^{\mathscr{L}}$ defined above.

- The CLM $\mathfrak{s}^{\texttt{DLine}}$ is 3-level, and is defined by:

$$\forall u, v \in \mathbb{F}_q^m, \, s \in \mathbb{F}_q : \quad \mathfrak{s}^{\texttt{DLine}}(u, s, v) = \left(\mathrm{null}_{\pi_{\chi(s)-1}(v)}(u), s, \pi_{\chi(s)-1}(v) \right) , \tag{181}$$

  where $\pi_i \colon \mathbb{F}_q^m \to \mathbb{F}_q^m$ is the linear map that zeroes out the first i coordinates of the input. It easily seen to be 3-level CLM, as the first register space is the copy of $\mathbb{F}_q$ on which $\mathfrak{s}^{\texttt{DLine}}$ acts with the identity. Then, the second register space is the last copy of $\mathbb{F}_q^m$, on which the linear map $\pi_{\chi(s)-1}$ is applied (and indeed, it depends only on the image of the previous linear map). And finally, the third register space is the first copy of $\mathbb{F}_q^m$, on which $\mathrm{null}_{\pi_{\chi(s)-q}(v)}$ is applied, which is dependent on the result of the previous linear map. By ignoring s, we get a representation of a diagonal line $\mathscr{L} = \mathscr{L}\left(\mathrm{null}_{\pi_{\chi(s)-1}(v)}(u), \pi_{\chi(s)-1}(v) \right)$ — note that the incidence point is canonical, while the direction may not be. So, each such $\left(\texttt{DLine}, \mathrm{null}_{\pi_{\chi(s)-1}(v)}(u), s, \pi_{\chi(s)-1}(v) \right)$ is (a copy of) the vertex $\texttt{DLine}^{\mathscr{L}}$ introduced above.[97]

Finally we specify the decision procedure. Recall the canonical representation of lines from Definition 5.75.

- If $\texttt{Point}^u - \texttt{ALine}^{\mathscr{L}}$ is sampled, then $u \in \mathscr{L} = \mathscr{L}(u_0, e_i)$, and in particular $u = u_0 + \alpha e_i$ for some $\alpha \in \mathbb{F}_q$. Let γ be the answer of the players, and denote as before $\mathsf{AL}f(\mathscr{L})_j = a_j = \gamma(\texttt{ALine}^{\mathscr{L},j,i})_{i=1}^t$ and $f(u) = (\texttt{Point}^{u,i})_{i=1}^t$, which are elements of $\mathbb{F}_q$. Then the decision procedure accepts if and only if $\sum_{j=0}^d a_j \alpha^j = f(u)$. This can easily be written as t linear equations over $\mathbb{F}_2$.

- If $\texttt{Point}^u - \texttt{DLine}^{\mathscr{L}}$ is sampled, then $u \in \mathscr{L} = \mathscr{L}(u_0, v_0)$, and in particular $u = u_0 + \alpha v_0$ for some $\alpha \in \mathbb{F}_q$. Let γ be the answer of the players. The decision here is almost identical to the previous one — we denote as before $\mathsf{DL}f(\mathscr{L})_j = a_j = \gamma(\texttt{DLine}^{\mathscr{L},j,i})_{i=1}^t$ and $f(u) = (\texttt{Point}^{u,i})_{i=1}^t$ as elements of $\mathbb{F}_q$, and accept if and only if $\sum_{j=0}^{md} a_j \alpha^j = f(u)$. This can again be written as t linear equations over $\mathbb{F}_2$.

---

[97]It can already be noticed that the probability of sampling diagonal lines is far from being uniform over them. This is a technical thing needed for the induction in the soundness proof in [JNV+22b] to work out.

For the general case of $k > 1$, $\mathfrak{LowDegree}(d,q,m,k)$ uses the same question distribution, but now the sets of generators are k times larger. For example, for the vertex $\texttt{Point}^u$, we have $S^{\mathfrak{Q}}_{\texttt{Point}^u} = \{\texttt{Point}^{u,j,i} : i \in [t], j \in [k]\}$ — in this case, $\texttt{Point}^{u,j,i}$ is supposed to be the ith bit of the evaluation of a global function f_j, that is supposed to be of low degree, evaluated at u. In this case, given an assignment γ, we denote by $(f_1(u), ..., f_k(u))$ the answer $(\gamma(\texttt{Point}^{u,j,i})^t_{i=1})^k_{j=1}$ (and similarly we denote $(\mathsf{AL}f_1(\mathscr{L}), ..., \mathsf{AL}f_k(\mathscr{L}))$ and $(\mathsf{DL}f_1(\mathscr{L}), ..., \mathsf{DL}f_k(\mathscr{L}))$ for the other types). The check performed is the same check as for the case $k = 1$, executed independently k times, once for each group of generators associated with the same $j \in [k]$.[98]

The following is based on [JNV$^+$22b]. We state the theorem for the case where the base code is the Reed–Solomon code with degree d — i.e., all univariate polynomials of degree at most d over $\mathbb{F}_q$ — as this is the only case we use. The only fact about this code that is used in the theorem statement is that it has distance $1 - d/q$, by the Schwartz–Zippel Lemma 5.19.

Theorem 5.76 (Soundness of the low-degree game. See the main theorem in [JNV$^+$22b] and Theorem 4.43 in [NW19])**.** *There exists a universal positive integer constant*

$$c = c_{\text{LD}} , \tag{182}$$

and a function

$$\delta_{\text{LD}}(m,d,k,\varepsilon,q^{-1}) = c \cdot (m^c + d^c + k^c) \cdot (\varepsilon^{1/c} + q^{-1/c} + 2^{-md/c}) \tag{183}$$

such that the following holds. Let $\mathscr{S} = \{\mathcal{P}\}$ be a strategy that is accepted in $\mathfrak{LowDegree}(d,q,m,k)$ with probability $1 - \varepsilon$. Then there exists a PVM $\{\mathcal{G}_{f_1,...,f_k}\}$, acting on the same Hilbert space as $\mathcal{P}$, with outcomes in k-tuples $f_1, ..., f_k \colon \mathbb{F}^m_q \to \mathbb{F}_q$ of polynomials of individual degree at most d, such that

$$\mathcal{G}_{[\text{eval}_u(\cdot)]} \approx_\delta \mathcal{P}^{\texttt{Point}^u} , \tag{184}$$

where $\delta = \delta_{\text{LD}}(m,d,k,\varepsilon,q^{-1})$ and eval_u is the "evaluate at u" function, namely $\text{eval}_u(f_1, ..., f_k) = (f_1(u), ..., f_k(u))$. In addition, by letting $\text{eval}_{\mathscr{L}}$ be the function that restricts an individual degree at most d polynomial to the line $\mathscr{L}$ and represents it in coefficient representation, we have that

$$\mathcal{G}_{[\text{eval}_{\mathscr{L}}(\cdot)]} \approx_\delta \mathcal{P}^{\texttt{ALine}^{\mathscr{L}}} \quad \text{and} \quad \mathcal{G}_{[\text{eval}_{\mathscr{L}}(\cdot)]} \approx_\delta \mathcal{P}^{\texttt{DLine}^{\mathscr{L}}} . \tag{185}$$

Proof. We first apply [JNV$^+$22b, Theorem 4.1] to the degree-d Reed–Solomon code over $\mathbb{F}_q$. The relative distance of this code is at least $(1 - d/q)$. This gives the statement of the theorem for $k = 1$. The extension to general k can be done via a standard reduction, following the same steps as the derivation of Theorem 4.43 from Theorem 4.40 in [NW19]. $\qquad\square$

Remark 5.77. Let us recall the meaning of the notations in (184) and (185). Recall the data processing notation (Definition 3.32). Then (184) is equivalent to

$$\sum_{a_1,...,a_k \in \mathbb{F}_q} \mathbb{E}_{u \sim \mathbb{F}^m_q} \left[\left\| \mathcal{P}^{\texttt{Point}^u}_{a_1,...,a_k} - \sum_{\substack{f_1,...,f_k \\ f_i(u)=a_i}} \mathcal{G}_{f_1,...,f_k} \right\|^2_{hs} \right] \leq \delta ,$$

while (185) is equivalent to

$$\sum_{c_{i,j} \in \mathbb{F}_q} \mathbb{E}_{\substack{\mathscr{L}=\mathscr{L}(u,e_i) \\ i \in [m], u \in \mathbb{F}^m_q}} \left[\left\| \mathcal{P}^{\texttt{ALine}^{\mathscr{L}}}_{(c_{1,0},...,c_{1,d}),...,(c_{k,0},...,c_{k,d})} - \sum_{\substack{f_1,...,f_k \\ f_i|_{\mathscr{L}}=(c_{i,0},...,c_{i,d})}} \mathcal{G}_{f_1,...,f_k} \right\|^2_{hs} \right] \leq \delta$$

and

$$\sum_{c_{i,j} \in \mathbb{F}_q} \mathbb{E}_{\substack{\mathscr{L}=\mathscr{L}(u,v) \\ u,v \in \mathbb{F}^m_q}} \left[\left\| \mathcal{P}^{\texttt{DLine}^{\mathscr{L}}}_{(c_{1,0},...,c_{1,md}),...,(c_{k,0},...,c_{k,md})} - \sum_{\substack{f_1,...,f_k \\ f_i|_{\mathscr{L}}=(c_{i,0},...,c_{i,md})}} \mathcal{G}_{f_1,...,f_k} \right\|^2_{hs} \right] \leq \delta .$$

---

[98]Note that question types are *not* mixed according the different copies of the test, e.g. a point or line is sampled simultaneously for all copies, not a mixture of points and lines.

Fact 5.78 (Algorithmic Low Degree test). *There is a (4-input version of a) 3-level tailored normal form verifier* $\mathcal{V}^{\mathrm{LD}} = (\mathcal{S}^{\mathrm{LD}}, \mathcal{A}^{\mathrm{LD}}, \mathcal{L}^{\mathrm{LD}}, \mathcal{D})$ *with the following properties:*[99]

1. *Combinatorial Low Degree test: For every* $d, t, m, k \in \mathbb{N}$, $\mathcal{V}^{\mathrm{LD}}_{d,t,m,k} = \mathfrak{LowDegree}(d, 2^t, m, k)$.

2. *Running time and description length: The runtimes of* $\mathcal{S}^{\mathrm{LD}}, \mathcal{A}^{\mathrm{LD}}, \mathcal{L}^{\mathrm{LD}}$ *are all bounded by* $\mathrm{poly}(d, t, m, k)$. *In addition, their description length is constant (up to appending the inputs* d, t, m, k, *which contributes length* $O(\log(d \cdot t \cdot m \cdot k))$).

Proof Sketch. Regardless of the rest, $\mathcal{S}^{\mathrm{LD}}, \mathcal{A}^{\mathrm{LD}}$ and $\mathcal{L}^{\mathrm{LD}}$ run the algorithm of Fact 5.24 with respect to t, and recover a fixed basis of $\mathbb{F}_q$ over $\mathbb{F}_2$, so that bit strings of length t can be interpreted and manipulated as elements of the field $\mathbb{F}_q$ in time $\mathrm{poly}(t)$.

The sampler $\mathcal{S}^{\mathrm{LD}}$ follows the CLMs defined in equations (178), (179) and (181). Note that all the calculations are in $\mathbb{F}_q^{2m+1} \cong \mathbb{F}_2^{t(2m+1)}$, which takes time $\mathrm{poly}(t, m)$.

The answer length calculator $\mathcal{A}^{\mathrm{LD}}$ outputs a string of 1's of length kt in case the type of question is `Point`; $kt(d + 1)$ in case the type of question is `ALine`; $kt(md + 1)$ in case the type of question is `DLine`. All in all, this takes at most $kt(md + 1) = \mathrm{poly}(d, t, m, k)$-time.

The linear constraints processor $\mathcal{L}^{\mathrm{LD}}$, in case the sampled edge is $\mathtt{Point}^u - \mathtt{ALine}^{\mathscr{L}}$, calculates the canonical representation of $\mathscr{L} = \mathscr{L}(u_0, v_0)$ — this takes $\mathrm{poly}(t, m)$-time. Then, it interprets u as $u_0 + \alpha v_0$ — which takes again $\mathrm{poly}(t, m)$-time. Only according to that, it can write kt many equations which amount to verifying that each bit of $f_i(u)$ is the appropriate bit of $\sum \mathsf{AL} f_i(\mathscr{L})_j \cdot \alpha^j$ — note that the constants are coming from the powers of α and the variables are the bits of $\mathsf{AL} f_i(\mathscr{L})_j$. All in all, this requires $\mathrm{poly}(d, t, m, k)$-time. The case of $\mathtt{Point}^u - \mathtt{DLine}^{\mathscr{L}}$ is similar and its runtime is also bounded by $\mathrm{poly}(d, t, m, k)$. $\qquad\square$

5.5 Combinatorial and Algorithmic Answer Reduction

As opposed to the question reduction (Section 4) and parallel repetition (Section 6) transformations, which have non-complexity theoretic combinatorial descriptions, even the combinatorial transformation of answer reduction is tied to complexity theoretic aspects — as should already be clear from the previous subsections. Regardless, before describing the answer reduction transformation on the level of normal form verifiers, we describe it on a combinatorial level, with the hopes it clarifies its operation as well as its completeness and soundness properties.

Let us describe the idea briefly. Given a previously 2^Λ-padded and purified verifier $\mathcal{V}$, with certain bounds on the running times of its sampler, answer length calculator and linear constraint processor, and fixing an index $n \in \mathbb{N}$, we aim to reduce the length of answers in $\mathcal{V}_n$ exponentially, as well as reducing the time it takes to decide whether to accept or reject them. At first, we choose a field size $q = 2^t$ where t is odd. A question in the answer reduced game $\mathfrak{AnsRed}(\mathcal{V}_n)$ would be a pair of questions, where the first is from the oracularization (Section 5.2.2) of the double cover (Definition 3.52) of $\mathcal{V}_n$, namely $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ (Remark 5.47 clarifies this point), and the other from the low degree test $\mathfrak{LowDeg}(9, q, m, \cdot)$, where m is the same as in the definition of a PCP (Definition 5.68). The isolated player is assumed to answer with the evaluation at a point or restriction to a line of the multilinear encoding (i.e., individual degree at most 1 Reed–Muller encoding) of its pair of answers. The oracle player is assumed to answer with the evaluation at a point or restriction to a line of a PCP as in Definition 5.68. Then, the part of the PCP that is supposed to be consistent with the isolated player is checked to be so, and the PCP itself is checked to satisfy (170), (171), (172) and (173). Moreover, if in the edge sampled in $\mathfrak{AnsRed}(\mathcal{V}_n)$, which is a pair of pairs, both pairs agree on the left coordinate, namely the question sampled from $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ is the same in both pairs, then the game is just an instance of $\mathfrak{LowDeg}(9, q, m, \cdot)$. Namely, in such a case, the restriction of the polynomials to lines are checked to be consistent with their evaluations at a point.

---

[99]Though we did not define this 4-input version, we hope it is clear from context what do we mean by that. Instead of having a sequence of games that are generated uniformly using a single input n, we have a sequence of games that are generated uniformly using 4 inputs d, t, m, k. We spell out explicitly the dependencies of each TM in the normal form verifier on each input, and so do not need the more intricate notion of being λ-bounded and so on.

A minor problem arises with this idea, as the low individual degree test guarantees that the polynomials are low-degree yet all have the same number of variables m, while the PCP needs to be with polynomials that depend only on subsets of the m variables, namely, they should be indifferent to certain inputs (Definition 5.14). To that end, we add a "constant on certain axis parallel lines" condition that ensures that, indeed, the polynomials that should depend only on a subset of the variables are such.

Definition 5.79 (Combinatorial Answer Reduction). We defined the game $\mathfrak{G} = \mathfrak{Ans}\mathfrak{Red}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, t)$ given the following provided data: Let

- Λ be a single input TM that always halts;
- $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ a purified 2^{Λ}-padded h-level TNFV (Definition 5.57) such that $\mathcal{V}_n$ is well defined for every n (Definition 4.33);
- Δ an always halting 1-input TM that satisfies $\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ and induces $\Diamond(n)$ as in (147);
- Q an always halting 1-input TM satisfying $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \leq Q(n)$;
- T an always halting 1-input TM satisfying

$$T(n) \geq c \cdot \left(\mathbb{T}(\Lambda; n)^c + \mathbb{T}(\Delta; n)^c + 2^{c \cdot \Lambda(n)} + \Delta(n)^c + \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)^c \right), \tag{186}$$

  where $c \geq 6$ is the positive integer implied by the poly notation in (150);
- D a positive integer (in binary) satisfying $|\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \leq D$;
- n and t positive integers, and $q = 2^t$;
- $(M(n), s(n), \cdot) = \mathsf{SuccinctTOI}(\cdot, \cdot, \cdot, D, T, Q, n, \cdot, \cdot)$, where $\mathsf{SuccinctTOI}$ was defined in Proposition 5.62.

Underlying graph and sampling scheme of $\mathfrak{G}$:

The answer reduced game has a typed $\max(h, 3)$-level CL sampling scheme (Definition 4.38). First, the type graph is the tensor product of the graph $\mathtt{A} - \mathtt{Oracle} - \mathtt{B}$ (including self loops) and the graph $\mathtt{ALine} - \mathtt{Point} - \mathtt{DLine}$ (including self loops) — see Figure 17.

For the CLMs associated with each type:

- Denote by $\mathfrak{s}^{\mathtt{A}}$ the h-level CLM induced by $\mathcal{S}(n, \cdot, A, \cdot, \cdot, \cdot)$, and similarly $\mathfrak{s}^{\mathtt{B}}$ for the one induced by $\mathcal{S}(n, \cdot, B, \cdot, \cdot, \cdot)$.

- Furthermore, let r be the dimension of $\mathfrak{s}^{\mathtt{A}}$ and $\mathfrak{s}^{\mathtt{B}}$, namely the output of $\mathcal{S}(n, \mathsf{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and let $\mathfrak{s}^{\mathtt{Oracle}} : \mathbb{F}_2^r \to \mathbb{F}_2^r$ be the identity map (which is a linear map, and hence a 1-level CLM).

- Finally, recall the CLMs $\mathfrak{s}^{\mathtt{Point}}, \mathfrak{s}^{\mathtt{ALine}}$ and $\mathfrak{s}^{\mathtt{DLine}}$ defined in (178), (179) and (181) respectively, which act on $\mathbb{F}_2^{t(2m+1)}$, where $m = |S| = 4\Lambda(n) + 3\Diamond(n) + 3M(n) + s(n) + 12$ is the number of variables in a PCP (Definition 5.68).

Then, the CLMs of $\mathfrak{G}$ act on the space $\mathbb{F}_2^r \times \mathbb{F}_2^{t(2m+1)}$, and for every

$$(\mathtt{Player}, \mathtt{Space}) \in \{\mathtt{A}, \mathtt{B}, \mathtt{Oracle}\} \times \{\mathtt{Point}, \mathtt{ALine}, \mathtt{DLine}\} \quad \text{and} \quad (z, (u, s, v)) \in \mathbb{F}_2^r \times \mathbb{F}_2^{t(2m+1)},$$

we have

$$\mathfrak{s}^{(\mathtt{Player}, \mathtt{Space})}(z, (u, s, v)) = (\mathfrak{s}^{\mathtt{Player}}(z), \mathfrak{s}^{\mathtt{Space}}(u, s, v)).$$

By endowing $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ with the appropriate typed CL sampling scheme (see Remark 5.47), the above is just the direct sum ([JNV$^+$21, Lemma 4.8]) of it and the CL sampling scheme of $\mathfrak{LowDegree}(9, q, m, \cdot)$ — namely, a pair of questions is sampled in each game independently, and the resulting edge is the pair of pairs. Note that in particular, this typed CL sampling scheme has level $\max(h, 3)$, as claimed.

As the vertices of $\mathfrak{LowDegree}(9, q, m, \cdot)$ are either $\mathtt{Point}^p$ for a point $p \in \mathbb{F}_q^m$, $\mathtt{ALine}^{\mathscr{L}}$ for an axis parallel line $\mathscr{L}$ in $\mathbb{F}_q^m$, or $\mathtt{DLine}^{\mathscr{L}}$ for any line $\mathscr{L}$ in $\mathbb{F}_q^m$, we denote the vertices in $\mathfrak{G}$ as $((\mathtt{Player}, \mathtt{w}), \mathtt{Space}^{\rho})$, where $\mathtt{Player} \in \{\mathtt{A}, \mathtt{B}, \mathtt{Oracle}\}$, $\mathtt{w} \in \mathbb{F}_2^r$, $\mathtt{Space} \in \{\mathtt{Point}, \mathtt{ALine}, \mathtt{DLine}\}$ and ρ is either a point or a line in $\mathbb{F}_q^m$. When $\mathtt{Player}$ is either A or B, we call it an

Figure 17: The type graph of $\mathfrak{AnsRed}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, t)$. Though not drawn, all self loops are also edges in this type graph. We added (five) colours that can be compared to the checks in the game. When the edge is either pink or green, Item (1), which is a low-degree test, is checked — when pink, it is an instance of $\mathfrak{LowDegree}(9, q, m, 2)$, and when green it is an instance of $\mathfrak{LowDegree}(9, q, m, \heartsuit(n))$ (with $\heartsuit$ being recalled in (187)). When an edge is orange, Item (2) is checked, which is a consistency check between the (evaluation of the) g polynomials provided by the isolated player versus the respective g polynomials in the (evaluation of the) PCP proof Π provided by the oracle player. When an edge incident to a purple vertex is sampled, Item (3) is checked, which is an indifference of the relevant polynomials in the direction the sampled axis parallel line is. Finally, when an edge incident to the (single) blue vertex is sampled, Item (4) is checked, which is interpreting the answers at the $(\texttt{Oracle}, \texttt{Point})$-typed vertex as an evaluation of a PCP at a point (Definition 5.68), and verifying that these values satisfy (170), (171), (172) and (173) with respect to the circuit $\mathcal{C}$.

isolated player question, and when $\texttt{Player}$ is $\texttt{Oracle}$, we call it an oracle player question (compare to the naming convention in the oracularized game $\mathfrak{Oracle}(\cdot)$ in Section 5.2.2).

Answer lengths and structure of answers:

A full description appears in Table 3. Some guidance to parse the table: Whenever the second coordinate of the question is $\texttt{Point}^p$, the answer consists of a sequence of values in $\mathbb{F}_q$ (as length t bit strings), that are supposed to be the evaluation of polynomials of degree at most 9 in m variables at the point $p \in \mathbb{F}_q^m$ — in case the first coordinate of the question is $(\texttt{A}, \texttt{x})$ or $(\texttt{B}, \texttt{y})$, evaluations of just two polynomials, one readable and one linear, and in case the first coordinate is $(\texttt{Oracle}, z)$, the supposed evaluation of a PCP Π (Definition 5.68) at a point $p \in \mathbb{F}_q^m$, where $\Pi^{\mathfrak{R}}(p)$ is the readable part of the answer and $\Pi^{\mathfrak{L}}(p)$ the linear part. Similarly, when $\texttt{ALine}^{\mathscr{L}}$ (respectively $\texttt{DLine}^{\mathscr{L}}$) is the second coordinate of the question, the answer

consists of a sequence of 10-tuples (respectively $(9m + 1)$-tuples) of values in $\mathbb{F}_q$ that encode the restrictions of "the same" polynomials as before to the respective line $\mathscr{L}$. As the answer when Oracle is asked are restrictions of a PCP to a certain subspace (either a point or a line), this is a list of

$$\heartsuit(n) = 12\Lambda(n) + 12\Diamond(n) + 6M(n) + s(n) + 35 \tag{187}$$

many values (in $\mathbb{F}_q$ when evaluating at a point, and tuples in $\mathbb{F}_q$ when evaluating at a line).

Controlled linear constraints function:

We can now describe the game checks. They can be collected into four (not mutually exclusive) checks, namely if a condition holds, then it is checked, and for certain pairs of questions more than one condition holds. Assume the sampled edge was $((\texttt{Player}, \texttt{w}), \texttt{Space}_1^{\rho_1}) - ((\texttt{Player}', \texttt{w}'), \texttt{Space}_2^{\rho_2})$, where $\texttt{Player}, \texttt{Player}' \in \{\texttt{A}, \texttt{B}, \texttt{Oracle}\}$, $\texttt{w}, \texttt{w}' \in \mathbb{F}_2^r$, $\texttt{Space}_i \in \{\texttt{Point}, \texttt{ALine}, \texttt{DLine}\}$ and ρ_i is either a point $p \in \mathbb{F}_q^m$ or a line $\mathscr{L} \subseteq \mathbb{F}_q^m$.

(1) <u>Low degree test</u>: In case the pair of questions agree on their first coordinate, namely $\texttt{Player} = \texttt{Player}'$ (and thus $\texttt{w} = \texttt{w}'$, as this is a CL sampling scheme), the second coordinates pair $\texttt{Space}_1^{\rho_1} - \texttt{Space}_2^{\rho_2}$ is an edge in the game $\mathfrak{LowDeg}(9, q, m, \cdot)$. Hence, we run the checks from $\mathfrak{LowDeg}(9, q, m, k)$ on the respective answers of the players, where $k = 2$ in case $\texttt{Player} = \texttt{A}$ or $\texttt{B}$, and $k = \heartsuit(n)$ in case $\texttt{Player} = \texttt{Oracle}$. In more words, assuming $\texttt{Space}_1 = \texttt{Point}$, $\rho_1 = p$, $\texttt{Space}_2 = \texttt{ALine}$ (respectively $\texttt{DLine}$) and $\rho_2 = \mathscr{L}$, and letting f be one of the presumed polynomials, the answer of the first player is treated as $f(p) \in \mathbb{F}_q$ (which also agrees with our notations in Table 3) and the answer of the second player is treated as $\mathsf{AL}f(\mathscr{L}) = (a_0, ..., a_9) \in \mathbb{F}_q^{10}$ (respectively $\mathsf{DL}f(\mathscr{L}) = (a_0, ..., a_{9m}) \in \mathbb{F}_q^{9m+1}$). As $p \in \mathscr{L}$, and letting $\mathscr{L} = \mathscr{L}(p_0, v_0)$ be its canonical representation (Definition 5.75), there is a scalar $\alpha \in \mathbb{F}_q$ such that $p_0 + \alpha v_0 = p$. Then, check that $f(p) = \sum a_i \alpha^i$. This can be implemented as t many $\mathbb{F}_2$-linear checks, and thus the tailored structure of the game is preserved.

(2) <u>Consistency check induced by the oracularized game</u>: In case the pair of questions agree on their second coordinate, namely $\texttt{Space}_1 = \texttt{Space}_2$ (and thus $\rho := \rho_1 = \rho_2$, as this is a CL sampling scheme), the first coordinates pair $(\texttt{Player}, \texttt{w}) - (\texttt{Player}', \texttt{w}')$ is a valid pair of questions (see Remark 5.47) in the double cover of the oracularized game

$$\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n)) \,.$$

In such a case, we check consistency between the $g_{\texttt{Player}, \texttt{w}}^\kappa$-polynomials provided by the isolated player and the appropriate part of the PCP Π provided by the oracle player. Namely, if $(\texttt{Player}, \texttt{w}) = (\texttt{A}, \texttt{x})$ (respectively $(\texttt{B}, \texttt{y})$) and $(\texttt{Player}', \texttt{w}') = (\texttt{Oracle}, z)$, then the first player responds with $\square g_{\texttt{A}, \texttt{x}}^{\mathfrak{R}}(\rho), \square g_{\texttt{A}, \texttt{x}}^{\mathfrak{L}}(\rho)$ (respectively $\square g_{\texttt{B}, \texttt{y}}^{\mathfrak{R}}(\rho), \square g_{\texttt{B}, \texttt{y}}^{\mathfrak{L}}(\rho)$), where $\square$ is either blank, AL or DL, and part of the second player's response is $\square g_1^{\mathfrak{R}}(\rho), \square g_1^{\mathfrak{L}}(\rho)$ (respectively $\square g_2^{\mathfrak{R}}(\rho), \square g_2^{\mathfrak{L}}(\rho)$). Thus, it checks that

$$\square g_{\texttt{A}, \texttt{x}}^{\mathfrak{R}}(\rho) = \square g_1^{\mathfrak{R}}(\rho) \quad \text{and} \quad \square g_{\texttt{A}, \texttt{x}}^{\mathfrak{L}}(\rho) = \square g_1^{\mathfrak{L}}(\rho)$$

(respectively,

$$\square g_{\texttt{B}, \texttt{y}}^{\mathfrak{R}}(\rho) = \square g_2^{\mathfrak{R}}(\rho) \quad \text{and} \quad \square g_{\texttt{B}, \texttt{y}}^{\mathfrak{L}}(\rho) = \square g_2^{\mathfrak{L}}(\rho) \,.$$

These can be realized as $2t$-many $\mathbb{F}_2$-linear checks and thus this check preserves the tailored structure of the game.

(3) <u>Indifference check</u>: Recall the notion of indifference of a polynomial to a certain index (Definition 5.14), and specifically which polynomials of the PCP Π (Definition 5.68) are indifferent to which indexes (namely, which variables from S do not appear in the polynomial). Assume $\texttt{Space}_1 = \texttt{ALine}$ (the case of $\texttt{Space}_2 = \texttt{ALine}$ is essentially the same, so we do not spell it out) $\rho_1 = \mathscr{L}(p, e_{\mathsf{X}})$ where $\mathsf{X} \in S$ — here we use the bijection between $[m]$ and S induced by (163) — and let f be one of the polynomials provided by the player ($g_{\circ,\circ}^\circ$ in case of an isolated player, and one of $g_{\circ,\circ}^\circ, \alpha_{\circ,\circ}^\circ, \beta_{\circ,\circ}^\circ$ in case of an oracle player). If f should be indifferent to X, as X is not in the formal generating set defining f (compare to Definition 5.66), then f should be constant along the axis parallel line ρ_1. Hence, if $\mathsf{AL}f(\rho_1) = (a_0, ..., a_9)$, then we

		$\mathtt{Point}^p$	$\mathtt{ALine}^{\mathscr{L}}$	$\mathtt{DLine}^{\mathscr{L}}$
(A,x)	Readable	$g_{\mathsf{A},\mathsf{x}}^{\mathfrak{R}}(p)$	$\mathsf{AL}g_{\mathsf{A},\mathsf{x}}^{\mathfrak{R}}(\mathscr{L})$	$\mathsf{DL}g_{\mathsf{A},\mathsf{x}}^{\mathfrak{R}}(\mathscr{L})$
	Linear	$g_{\mathsf{A},\mathsf{x}}^{\mathfrak{L}}(p)$	$\mathsf{AL}g_{\mathsf{A},\mathsf{x}}^{\mathfrak{L}}(\mathscr{L})$	$\mathsf{DL}g_{\mathsf{A},\mathsf{x}}^{\mathfrak{L}}(\mathscr{L})$
(B,y)	Readable	$g_{\mathsf{B},\mathsf{y}}^{\mathfrak{R}}(p)$	$\mathsf{AL}g_{\mathsf{B},\mathsf{y}}^{\mathfrak{R}}(\mathscr{L})$	$\mathsf{DL}g_{\mathsf{B},\mathsf{y}}^{\mathfrak{R}}(\mathscr{L})$
	Linear	$g_{\mathsf{B},\mathsf{y}}^{\mathfrak{L}}(p)$	$\mathsf{AL}g_{\mathsf{B},\mathsf{y}}^{\mathfrak{L}}(\mathscr{L})$	$\mathsf{DL}g_{\mathsf{B},\mathsf{y}}^{\mathfrak{L}}(\mathscr{L})$
(Oracle,z)	Readable	$\Pi_{0,z}^{\mathfrak{R}}(p)$	$\mathsf{AL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$	$\mathsf{DL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$
	Linear	$\Pi_{0,z}^{\mathfrak{L}}(p)$	$\mathsf{AL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$	$\mathsf{DL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$

Table 3: In the above table, $g_{\circ,\circ}^{\circ}(p)$ is an element of $\mathbb{F}_q$ (namely a bit string of length t), $\mathsf{AL}g_{\circ,\circ}^{\circ}(\mathscr{L})$ is a tuple of 10 elements in $\mathbb{F}_q$ (which encode a degree 9 univariate polynomial), and $\mathsf{DL}g_{\circ,\circ}^{\circ}(\mathscr{L})$ is a tuple of $9m+1$ elements in $\mathbb{F}_q$ (which encode a degree $9m$ univariate polynomial). Recall the notion of a degree 9 PCP Π over $\mathbb{F}_q$ (Definition 5.68), and specifically the number $\heartsuit^{\mathfrak{R}}(n) = 8\Lambda(n) + 6\Diamond(n) + 6M(n) + s(n) + 24$ of polynomials in the readable part $\Pi^{\mathfrak{R}}$ of Π and the number $\heartsuit^{\mathfrak{L}}(n) = 4\Lambda(n) + 6\Diamond(n) + 11$ of polynomials in the linear part $\Pi^{\mathfrak{L}}$ of Π. Then, the answer $\Pi_{0,z}^{\mathfrak{R}}(p)$ consists of $\heartsuit^{\mathfrak{R}}(n)$-many values in $\mathbb{F}_q$, which we denote according to the names of the polynomials in the readable part of a PCP; namely, $\Pi_{0,z}^{\mathfrak{R}}(p)$ consists of the $\mathbb{F}_q$-values

$$\forall i \in [5] \, , \, \mathsf{X} \in S_i^{\kappa} : \ g_i^{\mathfrak{R}}(p) \, , \, \beta_{i,\mathsf{X}}^{\mathfrak{R}}(p)$$
$$\forall \mathsf{X} \in S : \ \alpha_{\mathsf{X}}^{\mathfrak{R}}(p) \, ,$$
$$\forall \mathsf{X} \in S_0 : \ g_0(p) \, , \, \beta_{0,\mathsf{X}}(p) \, .$$

Similarly, the answer $\mathsf{AL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$ (respectively $\mathsf{DL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$) consists of $\heartsuit^{\mathfrak{R}}(n)$-many 10-tuples (respectively $9m + 1$-tuples) of values in $\mathbb{F}_q$, which we denote according to the names of the polynomials in the readable part of a PCP as well; namely, $\mathsf{AL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$ (respectively $\mathsf{DL}\Pi_{0,z}^{\mathfrak{R}}(\mathscr{L})$) consists of the 10-tuples (respectively $9m + 1$-tuples) of $\mathbb{F}_q$-values

$$\forall i \in [5] \, , \, \mathsf{X} \in S_i^{\kappa} : \ \mathsf{AL}g_i^{\mathfrak{R}}(\mathscr{L}) \, , \, \mathsf{AL}\beta_{i,\mathsf{X}}^{\mathfrak{R}}(\mathscr{L}) \, , \quad (\text{resp. } \mathsf{DL}g_i^{\mathfrak{R}}(\mathscr{L}) \, , \, \mathsf{DL}\beta_{i,\mathsf{X}}^{\mathfrak{R}}(\mathscr{L})) \, ,$$
$$\forall \mathsf{X} \in S : \ \mathsf{AL}\alpha_{\mathsf{X}}^{\mathfrak{R}}(\mathscr{L}) \, , \quad (\text{resp. } \mathsf{DL}\alpha_{\mathsf{X}}^{\mathfrak{R}}(\mathscr{L})) \, ,$$
$$\forall \mathsf{X} \in S_0 : \ \mathsf{AL}g_0(\mathscr{L}) \, , \, \mathsf{AL}\beta_{0,\mathsf{X}}(\mathscr{L}) \, , \quad (\text{resp. } \mathsf{DL}g_0(\mathscr{L}) \, , \, \mathsf{DL}\beta_{0,\mathsf{X}}(\mathscr{L})) \, .$$

For the linear part, the answer $\Pi_{0,z}^{\mathfrak{L}}(p)$ consists of $\heartsuit^{\mathfrak{L}}(n)$-many values in $\mathbb{F}_q$, which we denote according to the names of the polynomials in the linear part of a PCP; namely, $\Pi_{0,z}^{\mathfrak{L}}(p)$ consists of the $\mathbb{F}_q$-values

$$\forall i \in [5] \, , \, \mathsf{X} \in S_i^{\kappa} : \ g_i^{\mathfrak{L}}(p) \, , \, \beta_{i,\mathsf{X}}^{\mathfrak{L}}(p)$$
$$\forall \mathsf{X} \in S_0 : \ \alpha_{\mathsf{X}}^{\mathfrak{L}}(p) \, .$$

Similarly, the answer $\mathsf{AL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$ (respectively $\mathsf{DL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$) consists of $\heartsuit^{\mathfrak{L}}(n)$-many 10-tuples (respectively $9m + 1$-tuples) of values in $\mathbb{F}_q$, which we denote according to the names of the polynomials in the linear part of a PCP as well; namely, $\mathsf{AL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$ (respectively $\mathsf{DL}\Pi_{0,z}^{\mathfrak{L}}(\mathscr{L})$) consists of the 10-tuples (respectively $9m + 1$-tuples) of $\mathbb{F}_q$-values

$$\forall i \in [5] \, , \, \mathsf{X} \in S_i^{\kappa} : \ \mathsf{AL}g_i^{\mathfrak{L}}(\mathscr{L}) \, , \, \mathsf{AL}\beta_{i,\mathsf{X}}^{\mathfrak{L}}(\mathscr{L}) \, , \quad (\text{resp. } \mathsf{DL}g_i^{\mathfrak{L}}(\mathscr{L}) \, , \, \mathsf{DL}\beta_{i,\mathsf{X}}^{\mathfrak{L}}(\mathscr{L})) \, ,$$
$$\forall \mathsf{X} \in S : \ \mathsf{AL}\alpha_{\mathsf{X}}^{\mathfrak{L}}(\mathscr{L}) \, , \quad (\text{resp. } \mathsf{DL}\alpha_{\mathsf{X}}^{\mathfrak{L}}(\mathscr{L})) \, .$$

check that for every $j \geq 1$, $a_j = 0$. This is done for all f's in the answer that are indifferent to the specific variable X. Also, equating to 0 is clearly a linear check, and thus tailored (as a sanity check, note that these are $9t$-many $\mathbb{F}_2$-linear equations).

(4) <u>Proof check</u>: In case one of the questions is $((\texttt{Oracle}, z), \texttt{Point}^p)$, where $p \in \mathbb{F}_q^S$ is as in (163), calculate the following:

- $\texttt{x} = \mathfrak{s}^A(z)$ and $\texttt{y} = \mathfrak{s}^B(z)$;
- $\mathcal{C}$ which is the third output of $\mathsf{SuccinctTOI}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, \texttt{x}, \texttt{y})$;
- $T_{\mathcal{C}}(p) \in \mathbb{F}_q$, where $T_{\mathcal{C}}$ is the Tseitin polynomial (Definition 5.28) associated with $\mathcal{C}$.

Treating the answer of the player who receives this question as the evaluation of a PCP Π at the point p, we can check that the equations (170), (171), (172) and (173) are satisfied at this specific point. Namely, the following 13 equations over $\mathbb{F}_q$ are checked:

$$\forall \kappa \in \{\mathfrak{R}, \mathfrak{L}\}, \ i \in [5] : \ g_i^\kappa(p)(g_i^\kappa(p) + 1) = \sum_{\mathsf{X} \in S_i^\kappa} \mathrm{zero}_\mathsf{X}(p) \cdot \beta_{i,\mathsf{X}}^\kappa(p), \tag{188}$$

$$g_0(p)(g_0(p) + 1) = \sum_{\mathsf{X} \in S_0} \mathrm{zero}_\mathsf{X}(p) \cdot \beta_{0,\mathsf{X}}(p), \tag{189}$$

$$T_{\mathcal{C}}(p)(g_0(p) + p_6^{\mathfrak{R}} + 1) \prod_{i=1}^{5} \left(g_i^{\mathfrak{R}}(p) + p_{i+6}^{\mathfrak{R}} + 1 \right) = \sum_{\mathsf{X} \in S} \mathrm{zero}_\mathsf{X}(p) \cdot \alpha_\mathsf{X}^{\mathfrak{R}}(p), \tag{190}$$

$$g_0(p)(p_{11}^{\mathfrak{L}} + \sum_{i=1}^{5} p_{i+5}^{\mathfrak{L}} g_i^{\mathfrak{L}}(p)) = \sum_{\mathsf{X} \in S_0} \mathrm{zero}_\mathsf{X}(p) \cdot \alpha_\mathsf{X}^{\mathfrak{L}}(p). \tag{191}$$

Let us briefly describe why these equations are tailored, as this is not obvious. The right hand side in all of the equations is already a linear combination of the values provided by the oracle player. For the left hand side of (188) and (189), note that the map $\mathsf{X} \mapsto \mathsf{X}^2 + \mathsf{X}$ from $\mathbb{F}_q$ to itself is $\mathbb{F}_2$-linear, and thus these are also just linear combinations of the t-bits representing each $\mathbb{F}_q$ value over $\mathbb{F}_2$. Hence, (188) and (189) are fixed linear checks that are independent of the value of the readable answers, and in particular are tailored. The left hand side of (190) is some function of p and the readable parts $g_0(p), g_1^{\mathfrak{R}}(p), ..., g_5^{\mathfrak{R}}(p)$ of the answer, which makes it a constant once the readable part of the answer is given — therefore this check is tailored as well. Finally, the left hand side of (191) is a linear combination of the linear parts $g_1^{\mathfrak{L}}(p), ..., g_5^{\mathfrak{L}}(p)$, with the exact coefficients depending on the value $g_0(p)$, which is readable — this shows that, indeed, this check is properly tailored.

Proposition 5.80 (Completeness and Soundness of the Answer Reduced game). *Recall all the data provided in Definition 5.79 and the assumptions on it, and recall the definition of the answer reduced game $\mathfrak{G} = \mathfrak{AnsRed}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, t)$. Then:*

1. Completeness: *If the game $\mathcal{V}_n$ has a perfect ZPC-strategy, then so does $\mathfrak{G}$.*

2. Soundness: *If $\mathfrak{G}$ has a strategy with value $1 - \varepsilon$, then $\mathfrak{DoubleCover}(\mathcal{V}_n)$ has a strategy with value of at least*

$$1 - \frac{10^6 \cdot m}{1 - \frac{72m}{q}} \cdot \delta^{1/8}$$

of the same dimension, where $\delta = \delta_{\mathrm{LD}}(m, 9, \heartsuit(n), 2\varepsilon, q^{-1})$ and δ_{LD} is as defined in (183).

3. Entanglement lower bound: *Furthermore,*

$$\mathscr{E}(\mathfrak{G}, 1 - \varepsilon) \geq \mathscr{E}\left(\mathfrak{DoubleCover}(\mathcal{V}_n), 1 - \frac{10^6 \cdot m}{1 - \frac{72m}{q}} \cdot \delta^{1/8} \right).$$

Proof. **Completeness**:

Assume $\mathcal{V}_n$ has a perfect ZPC-strategy. By the completeness of the double cover transformation (Claim 3.54) and the completeness of the oraculatrization transformation (Claim 5.46), there is a perfect[100] ZPC-strategy $\mathcal{U}$ for

$$\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n)) \, .$$

As discussed in Remark 5.47, the vertices in this game can be parametrized by $(\texttt{Player}, \texttt{w})$, where $\texttt{Player} \in \{\texttt{A}, \texttt{B}, \texttt{Oracle}\}$ and $\texttt{w} \in \mathbb{F}_2^r$ with $r = \mathcal{S}(n, \texttt{Dimension}, \cdot, \cdot, \cdot, \cdot)$. Since $\mathcal{V}$ is 2^Λ-padded, the size of the generating sets at the isolated player vertices is $2^{\Lambda(n)}$ and at the oracle player vertices is $2 \cdot 2^{\Lambda(n)}$; namely, the answers at an A-player vertex consist of two functions $a^{\mathfrak{R}}, a^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$, the answers at a B-player vertex consist of two functions $b^{\mathfrak{R}}, b^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$, and the answers at an Oracle-player vertex consist of four functions $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$.

We will now construct the PVMs of the perfect ZPC-strategy for $\mathfrak{G}$ at the isolated player vertices. This is done by first inducing and then evaluating the outcomes of the PVM $\mathcal{U}$ associates with the vertices $(\texttt{A}, \texttt{x})$ and $(\texttt{B}, \texttt{y})$, namely data-processing; as both induction and evaluation are linear maps, this preserves the Z-alignment and permutation properties. Let us elaborate more. Recall the discussion from Remark 5.18: Every map $f : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ can be thought of as a multilinear polynomial in $\mathbb{F}_2[S_f]$, where S_f is a formal set of $\Lambda(n)$-many variables. As $\mathbb{F}_2[S_f]$ embeds naturally into $\mathbb{F}_q[S_f]$, we associated with every function f a polynomial in $\mathbb{F}_q[S_f]$. As the variables of all of our polynomials are contained in the set S described in Definition 5.66, $\mathbb{F}_q[S_f]$ naturally embeds into $\mathbb{F}_q[S]$. As every polynomial in $\mathbb{F}_q[S]$ has an evaluation table (its $\Phi_{\mathbb{F}_q}$-image (123)), it induces a function $\mathbb{F}_q^m \to \mathbb{F}_q$, where as usual $m = |S|$. Let us denote by $\mathrm{Ind}(f) : \mathbb{F}_q^m \to \mathbb{F}_q$ this final function — although this is a **variation** on the $\mathrm{Ind}(\cdot)$ map from Definition 5.17 and Remark 5.18, we use the same notation, and this map is linear as well. Note that the exact way S_f is embedded in S plays a role in the induction function. For example, a^κ is associated with S_1^κ from Definition 5.66, while b^κ is associated with S_2^κ. Recall also the evaluation at a point map $\mathrm{eval}_p(\cdot)$ and the evaluation at a line map $\mathrm{eval}_{\mathscr{L}}(\cdot)$ described in Theorem 5.76, which are also $\mathbb{F}_2$-linear. Define for every $\texttt{x}, \texttt{y} \in \mathbb{F}_2^r$, $\texttt{Space} \in \{\texttt{Point}, \texttt{ALine}, \texttt{DLine}\}$ and ρ an appropriate point or line, the PVM

$$\mathcal{W}^{(\texttt{A},\texttt{x}), \texttt{Space}^\rho} = \mathcal{U}^{(\texttt{A},\texttt{x})}_{[\mathrm{eval}_\rho \circ \mathrm{Ind}(\cdot)]} \quad , \quad \mathcal{W}^{(\texttt{B},\texttt{y}), \texttt{Space}^\rho} = \mathcal{U}^{(\texttt{B},\texttt{y})}_{[\mathrm{eval}_\rho \circ \mathrm{Ind}(\cdot)]} \; ; \tag{192}$$

namely,

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_q : \quad \mathcal{W}^{(\texttt{A},\texttt{x}), \texttt{Point}^p}_{\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}} = \sum_{\substack{a^{\mathfrak{R}}, a^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{Ind}(a^{\mathfrak{R}})(p) = \alpha^{\mathfrak{R}} , \, \mathrm{Ind}(a^{\mathfrak{L}})(p) = \alpha^{\mathfrak{L}}}} \mathcal{U}^{(\texttt{A},\texttt{x})}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}$$

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_q^{10} : \quad \mathcal{W}^{(\texttt{A},\texttt{x}), \texttt{ALine}^{\mathscr{L}}}_{\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}} = \sum_{\substack{a^{\mathfrak{R}}, a^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{Ind}(a^{\mathfrak{R}})(\mathscr{L}) = \alpha^{\mathfrak{R}} , \, \mathrm{Ind}(a^{\mathfrak{L}})(\mathscr{L}) = \alpha^{\mathfrak{L}}}} \mathcal{U}^{(\texttt{A},\texttt{x})}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}$$

$$\forall \alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}} \in \mathbb{F}_q^{9m+1} : \quad \mathcal{W}^{(\texttt{A},\texttt{x}), \texttt{DLine}^{\mathscr{L}}}_{\alpha^{\mathfrak{R}}, \alpha^{\mathfrak{L}}} = \sum_{\substack{a^{\mathfrak{R}}, a^{\mathfrak{L}} : \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{Ind}(a^{\mathfrak{R}})(\mathscr{L}) = \alpha^{\mathfrak{R}} , \, \mathrm{Ind}(a^{\mathfrak{L}})(\mathscr{L}) = \alpha^{\mathfrak{L}}}} \mathcal{U}^{(\texttt{A},\texttt{x})}_{a^{\mathfrak{R}}, a^{\mathfrak{L}}}$$

and similarly for the $(\texttt{B}, \texttt{y})$ case. In words, these measurements are the following: First, measure according to $\mathcal{U}^{(\texttt{A},\texttt{x})}$ and retrieve $a^{\mathfrak{R}}, a^{\mathfrak{L}}$. Then, induce both function to get $\mathrm{Ind}(a^{\mathfrak{R}}), \mathrm{Ind}(a^{\mathfrak{L}})$. Finally, evaluate these functions on the respective point p or line $\mathscr{L}$ to get a value in $\mathbb{F}_q$ or the coefficients of a low-degree polynomial. As both Ind and eval_ρ are linear maps, by Corollary 3.39 the resulting PVMs are readably Z-aligned and consist of signed permutations.

We now aim to define the PVMs of the perfect ZPC-strategy for $\mathfrak{G}$ at the oracle player vertices. The process is similar to the isolated player vertices, but we need to retrieve more polynomials. Recall the mapping $\mathsf{PCP}_{\texttt{xy}}$ from Corollary 5.72

---

[100]Note that here we are using the commuting along edges condition, essentially, for the first and last time in the proof of compression.

that takes as input a quadruple $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ (which is the format of measurement outcomes of the PVM $\mathcal{U}^{(\texttt{Oracle},z)}$), and outputs a degree 9 PCP Π over $\mathbb{F}_q$ with parameters $(\Lambda(n), \Diamond(n), M(n), s(n), \mathcal{C})$ (Definition 5.68), such that:

1. The values of readable part $\Pi^{\mathfrak{R}}$, i.e., the polynomials $g_0, \beta_{0,\mathsf{X}}, g_i^{\mathfrak{R}}, \alpha_{\mathsf{X}}^{\mathfrak{R}}, \beta_{\mathsf{X}}^{\mathfrak{R}}$, depends only on $f_1^{\mathfrak{R}}$ and $f_2^{\mathfrak{R}}$, and the rest of the polynomials in Π depend in addition on the values of $f_1^{\mathfrak{L}}$ and $f_2^{\mathfrak{L}}$, but in an affine manner; namely, the values of the additional polynomials are affine combinations of the values of $f_1^{\mathfrak{L}}$ and $f_2^{\mathfrak{L}}$, with the coefficients depending only on $f_1^{\mathfrak{R}}$ and $f_2^{\mathfrak{R}}$.

2. In addition, if $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$ are accepted in $\mathcal{V}_n$ given $\mathrm{x} = \mathfrak{s}^{\mathsf{A}}(z), \mathrm{y} = \mathfrak{s}^{\mathsf{B}}(z)$ were asked, then the PCP Π passes (170), (171), (172) and (173) for every $p \in \mathbb{F}_q^S$.

For ease of notation, given a seed z, instead of denoting $\mathrm{PCP}_{\mathfrak{s}^{\mathsf{A}}(z)\mathfrak{s}^{\mathsf{B}}(z)}$ we will use PCP_z to mean the same function. Then, recall the notion of evaluating a PCP Π at a point (Definition 5.68), and denote this function by $\mathrm{eval}_p(\Pi) = \Pi(p)$. Similarly, one can evaluate a PCP Π at a line $\mathscr{L}$, by providing the coefficient representation of the restriction of each polynomial in Π to this line. We denote this function by $\mathrm{eval}_{\mathscr{L}}(\Pi) = \Pi(\mathscr{L})$. So, let

$$\mathcal{W}^{(\texttt{Oracle},z),\texttt{Space}^\rho} = \mathcal{U}^{(\texttt{Oracle},z)}_{[\mathrm{eval}_\rho \circ \mathrm{PCP}_z(\cdot)]} \; ; \tag{193}$$

namely,

$$\forall \alpha \in \mathbb{F}_q^{\heartsuit(n)} \colon \quad \mathcal{W}_\alpha^{(\texttt{Oracle},z),\texttt{Point}^p} = \sum_{\substack{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{PCP}_z(f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}})(p) = \alpha}} \mathcal{U}^{(\texttt{Oracle},z)}_{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}}}$$

$$\forall \alpha \in (\mathbb{F}_q^{10})^{\heartsuit(n)} \colon \quad \mathcal{W}_\alpha^{(\texttt{Oracle},z),\texttt{ALine}^{\mathscr{L}}} = \sum_{\substack{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{PCP}_z(f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}})(\mathscr{L}) = \alpha}} \mathcal{U}^{(\texttt{Oracle},z)}_{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}}}$$

$$\forall \alpha \in (\mathbb{F}_q^{9m+1})^{\heartsuit(n)} \colon \quad \mathcal{W}_\alpha^{(\texttt{Oracle},z),\texttt{DLine}^{\mathscr{L}}} = \sum_{\substack{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \\ \mathrm{PCP}_z(f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}})(\mathscr{L}) = \alpha}} \mathcal{U}^{(\texttt{Oracle},z)}_{f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}}} \; .$$

Though the notation is confusing, we can describe these measurements simply with words: First, measure according to $\mathcal{U}^{(\texttt{Oracle},z)}$ to retrieve a quadruple $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2$. Then, use PCP_z with input $f_1^{\mathfrak{R}}, f_1^{\mathfrak{L}}, f_2^{\mathfrak{R}}, f_2^{\mathfrak{L}}$ to get a degree 9 PCP Π. Finally, evaluate the resulting PCP Π at ρ to get $\heartsuit(n)$ many values — in $\mathbb{F}_q$ when ρ is a point, in $\mathbb{F}_q^{10}$ when ρ is an axis parallel line, and in $\mathbb{F}_q^{9m+1}$ when ρ is a diagonal line.

Let us convince ourselves that the resulting strategy $\mathcal{W}$ is a perfect ZPC strategy for $\mathfrak{G}$. As eval_ρ and Ind are linear functions, and PCP_z is $\mathbb{F}_2$-affine for fixed readable values, by Corollary 3.41, data processing along $\mathrm{eval}_\rho \circ \mathrm{Ind}$ and along $\mathrm{eval}_\rho \circ \mathrm{PCP}_z$ preserves Z-alignment and being a signed permutation PVM. Since $\mathcal{U}^{(\texttt{Player},\mathtt{w})}$ are Z-aligned permutation PVMs, the same is true for $\mathcal{W}^{(\texttt{Player},\mathtt{w}),\texttt{Space}^\rho}$. Also, data processing (Definition 3.32) does not change the commuting along edges condition; namely, if $\mathcal{P}, \mathcal{Q}$ are commmuting PVMs, then $\mathcal{P}_{[f(\cdot)=\cdot]}$ commutes with $\mathcal{Q}_{[g(\cdot)=\cdot]}$. Hence, as $\mathcal{U}$ is commuting along edges, $\mathcal{W}$ is commuting along edges as well. All in all, $\mathcal{W}$ is a ZPC strategy. Let us now elaborate on why $\mathcal{W}$ is perfect for $\mathfrak{G}$:

1. As the isolated player measurements are, by definition, the evaluation at lines and points of $\mathrm{Ind}(f)$ for measured functions of the form $f \colon \mathbb{F}_2^m \to \mathbb{F}_2$, they pass the low degree test $\mathfrak{LowDeg}(d, q, m, 2)$ perfectly for every $d \geq 1$ (and in particular for 9). Similarly, the oracle player measurements are the evaluation at lines and points of $\mathrm{PCP}_z(\cdot, \cdot, \cdot, \cdot)$, which are of individual degree at most 9, and will thus pass $\mathfrak{LowDeg}(9, q, m, \heartsuit(n))$. This proves that $\mathcal{W}$ passes check (1) perfectly (the pink and green edges from Figure 17).

180

2. As $\mathcal{U}$ was perfect for $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, given z for which $\mathfrak{s}^{\mathsf{A}}(z) = \mathsf{x}$ and $\mathfrak{s}^{\mathsf{B}}(z) = \mathsf{y}$, the isolated player measurement $\mathcal{U}^{(\mathsf{A},\mathsf{x})}$ (respectively $\mathcal{U}^{(\mathsf{B},\mathsf{y})}$) is perfectly consistent with the first two outputs of the oracle player measurement $\mathcal{U}^{(\mathtt{Oracle},z)}$ (respectively the last two outputs of $\mathcal{U}^{(\mathtt{Oracle},z)}$); namely

$$\forall a^{\mathfrak{R}}, a^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \colon\ \mathcal{U}^{(\mathsf{A},\mathsf{x})}_{a^{\mathfrak{R}},a^{\mathfrak{L}}} = \sum_{b^{\mathfrak{R}},b^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2} \mathcal{U}^{(\mathtt{Oracle},z)}_{a^{\mathfrak{R}},a^{\mathfrak{L}},b^{\mathfrak{R}},b^{\mathfrak{L}}} \ ,$$

$$\forall b^{\mathfrak{R}}, b^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2 \colon\ \mathcal{U}^{(\mathsf{B},\mathsf{y})}_{b^{\mathfrak{R}},b^{\mathfrak{L}}} = \sum_{a^{\mathfrak{R}},a^{\mathfrak{L}} \colon \mathbb{F}_2^{\Lambda(n)} \to \mathbb{F}_2} \mathcal{U}^{(\mathtt{Oracle},z)}_{a^{\mathfrak{R}},a^{\mathfrak{L}},b^{\mathfrak{R}},b^{\mathfrak{L}}} \ .$$

Following the definition of $\mathsf{PCP}_z(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}) = \Pi$ (Corollary 5.72), the polynomial g_1^{κ} in Π is $\mathrm{Ind}(a^{\kappa})$ and g_2^{κ} is $\mathrm{Ind}(b^{\kappa})$, and thus evaluating these polynomials at a point or a line always agree. Recalling the definition of $\mathcal{W}$ from (192) and (193), we deduce that $\mathcal{W}$ passes check (2) perfectly (the orange edges from Figure 17).

3. Recalling the way we redefined $\mathrm{Ind}(\cdot)$ in this completeness proof, $\mathrm{Ind}(a_i^{\kappa}) \colon \mathbb{F}_q^S \to \mathbb{F}_q$ is indifferent to every X from S which is not in S_1^{κ} (and similarly $\mathrm{Ind}(b_i^{\kappa})$ is indifferent to every X not in S_2^{κ}). Also, by the definition of a PCP Π (Definition 5.68), the various polynomials are indifferent to the variables not from their respective block. Hence, the $\mathcal{W}$ strategy passes check (3) perfectly (the purple vertices from Figure 17).

4. Now, as $\mathcal{U}$ was perfect for $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, the measurement outcome of $\mathcal{U}^{(\mathtt{Oracle},z)}$ always passes $\mathcal{V}_n$ given $\mathfrak{s}^{\mathsf{A}}(z)\mathfrak{s}^{\mathsf{B}}(z) = \mathsf{xy}$ were asked. Hence, by the completeness clause of Corollary 5.72, the measurement outcome of $\mathcal{U}^{(\mathtt{Oracle},z)}_{[\mathsf{PCP}_z(\cdot,\cdot,\cdot,\cdot)]}$ is a PCP Π such that for every point $p \in \mathbb{F}_q^S$, its evaluation at a point $\Pi(p)$ always passes (188), (189), (190) and (191). Therefore, $\mathcal{W}$ passes check (4) perfectly (the blue vertex from Figure 17).

Soundness and entanglement lower bound:

The idea is similar to soundness of the question reduction transformation. Namely, we perturb the original almost perfect strategy bit by bit, so that in each step it passes more of the checks of $\mathfrak{G}$ perfectly, while worsening the probability of passing the remaining checks. This is done in the following order: First, perturbing the strategy so as to pass Item (1) perfectly, which guarantees that there is a global PVM at any vertex of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ such that the strategy is close to restrictions of it to the appropriate lines and points. Then, using the Schwartz–Zippel Lemma 5.19, these PVMs can be further purturbed to pass Item (3) perfectly. Then using the consistency checks of Item (2) and the PCP check of Item (4), we deduce that the restrictions of the polynomials measured by the PVMs at isolated player vertices are almost perfect as a strategy for $\mathfrak{DoubleCover}(\mathcal{V}_n)$.

Preliminary analysis:

Recall that the vertices of $\mathfrak{G}$ are a direct product of the vertices of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ and the vertices of $\mathfrak{LowDegree}(9, t, m, \cdot)$, namely a vertex in $\mathfrak{G}$ is of the form

$$(\mathtt{Player}, \mathtt{w}), \mathtt{Space}^{\rho} \ ,$$

where $(\mathtt{Player}, \mathtt{w}) \in \{\mathtt{A}, \mathtt{B}, \mathtt{Oracle}\} \times \mathbb{F}_2^r$, $\mathtt{Space} \in \{\mathtt{Point}, \mathtt{ALine}, \mathtt{DLine}\}$ and ρ is either a point or a line in $\mathbb{F}_q^m$. So, a sampled edge in $\mathfrak{G}$ is of the form

$$e := (\mathtt{Player}, \mathtt{w}), \mathtt{Space}_1^{\rho_1} - (\mathtt{Player}', \mathtt{w}'), \mathtt{Space}_2^{\rho_2} \ . \tag{194}$$

For a vertex $\mathtt{v}$ of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, we say that an edge e of $\mathfrak{G}$ as in (194) is first-$\mathtt{v}$-incident if $(\mathtt{Player}, \mathtt{w}) = \mathtt{v}$, second-$\mathtt{v}$-incident if $(\mathtt{Player}', \mathtt{w}') = \mathtt{v}$, and just $\mathtt{v}$-incident if it is either first-$\mathtt{v}$-incident or second-$\mathtt{v}$-incident (or both). Similarly, such an edge is said to be $\mathtt{v}$-internal if it is both first-$\mathtt{v}$-incident and second-$\mathtt{v}$-incident; namely, e is an edge of the form

$$e := \mathtt{v}, \mathtt{Space}_1^{\rho_1} - \mathtt{v}, \mathtt{Space}_2^{\rho_2} \ . \tag{195}$$

181

Let $\mu(e)$ be the probability the edge e is sampled in $\mathfrak{G}$, and note that for each edge e there is a unique $\mathtt{v}$ for which e is first-$\mathtt{v}$-incident and a unique $\mathtt{v}'$ such that e is second-$\mathtt{v}'$-incident. Now, μ induces a marginal distribution on the vertices of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$ by the following sampling mechanism: sample $e \sim \mu$ as in (194); output $\mathtt{v} = (\mathtt{Player}, \mathtt{w})$ or $\mathtt{v} = (\mathtt{Player}', \mathtt{w}')$ uniformly at random[101] — we denote the probability $\mathtt{v}$ is the output by $\mu(\mathtt{v})$. Furthermore, let $\mu^{\mathtt{v}}$ be the probability e was the sampled edge in the aforementioned sampling mechanism given $\mathtt{v}$ was the output, namely: if e is not $\mathtt{v}$-incident, then $\mu^{\mathtt{v}}(e) = 0$; if e is $\mathtt{v}$-incident but not $\mathtt{v}$-internal, then $\mu^{\mathtt{v}}(e) = \frac{\mu(e)}{2\mu(\mathtt{v})}$; if e is $\mathtt{v}$-internal, then $\mu^{\mathtt{v}}(e) = \frac{\mu(e)}{\mu(\mathtt{v})}$.

Assume $\mathscr{S} = \{\mathcal{P}\}$ is a quantum strategy for the answer reduced game $\mathfrak{G}$ with value $1 - \varepsilon$. For an edge e in $\mathfrak{G}$, let ε^e be the probability $\mathcal{P}$ loses the round of the game given e was sampled. Then,

$$\varepsilon = \mathop{\mathbb{E}}_{e \sim \mu}[\varepsilon^e] .$$

For a vertex $\mathtt{v}$ of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, let $\varepsilon^{\mathtt{v}}$ be the probability that $\mathcal{P}$ loses the round of $\mathfrak{G}$, given an edge was sampled according to $\mu^{\mathtt{v}}$; namely

$$\varepsilon^{\mathtt{v}} = \mathop{\mathbb{E}}_{e \sim \mu^{\mathtt{v}}}[\varepsilon^e] .$$

By our choices, sampling and edge e by first sampling $\mathtt{v} \sim \mu$ and then $e \sim \mu^{\mathtt{v}}$ is the same as just sampling $e \sim \mu$, and thus $\varepsilon = \mathbb{E}_{\mathtt{v} \sim \mu}[\varepsilon^{\mathtt{v}}]$.

First perturbation:

Recall that $\mathscr{S} = \{\mathcal{P}\}$ is a strategy for $\mathfrak{G}$ with value $1 - \varepsilon$. For the next discussion, let us fix a $\mathtt{v} = (\mathtt{Player}, \mathtt{w})$. By inspecting the type graph of $\mathfrak{G}$ (Figure 17), and recalling that the type graph contains all self loops, one can deduce that the probability $e \sim \mu^{\mathtt{v}}$ is $\mathtt{v}$-internal is at least $1/2$. Hence, the probability $\mathcal{P}$ loses versus e, given that the sampled e is $\mathtt{v}$-internal (195), is at most $2\varepsilon^{\mathtt{v}}$. In case a $\mathtt{v}$-internal edge was sampled (which corresponds to a pink or green edge in Figure 17), Item (1) in $\mathfrak{G}$ is checked. Namely, $\mathfrak{G}$ runs $\mathfrak{LowDegree}(9, q, m, k)$, where $k = 2$ in case $\mathtt{Player} = \mathtt{A}$ or $\mathtt{B}$ and $k = \heartsuit(n)$ in case $\mathtt{Player} = \mathtt{Oracle}$. Hence, by Theorem 5.76, there is a PVM $\mathcal{G}^{\mathtt{v}}$ (of the same dimension as $\mathcal{P}$), with outcomes in k-tuples of individual degree at most 9 polynomials from $\mathbb{F}_q^m$ to $\mathbb{F}_q$, that satisfies

$$\mathcal{P}^{\mathtt{v},\mathtt{Space}^{\rho}} \approx_{\delta^{\mathtt{v}}} \mathcal{G}^{\mathtt{v}}_{[\mathrm{eval}_{\rho}(\cdot)]} , \tag{196}$$

where $\delta^{\mathtt{v}} = \delta_{\mathrm{LD}}(m, 9, k, 2\varepsilon^{\mathtt{v}}, q^{-1})$ and δ_{LD} is the function from (183). Let $\mathscr{S}' = \{\mathcal{Q}\}$ be the strategy for $\mathfrak{G}$ that satisfies $\mathcal{Q}^{\mathtt{v},\mathtt{Space}^{\rho}} = \mathcal{G}^{\mathtt{v}}_{[\mathrm{eval}_{\rho}(\cdot)]}$; namely, the way it answers the question $\mathtt{v}, \mathtt{Space}^{\rho}$ is by first measuring a k-tuple $f_1, ..., f_k$ according to $\mathcal{G}^{\mathtt{v}}$, and then evaluating it at the space ρ. By construction, $\mathcal{Q}$ always passes check (1) of $\mathfrak{G}$, and from (196) (see Remark 5.77), and the notion of distance between strategies (Definition 3.26), we can deduce that $\mathscr{S}$ is $\mathbb{E}_{\mathtt{v} \sim \mu}[\delta^{\mathtt{v}}]$-close to $\mathscr{S}'$. Now, as δ_{LD} from (183) is monotonously increasing in all its inputs (in particular, in its third input), and is concave with respect to its fourth input, we can deduce that

$$\mathop{\mathbb{E}}_{\mathtt{v} \sim \mu}[\delta^{\mathtt{v}}] \leq \delta_{\mathrm{LD}}(m, 9, \heartsuit(n), 2 \cdot \underbrace{\mathop{\mathbb{E}}_{\mathtt{v} \sim \mu}[\varepsilon^{\mathtt{v}}]}_{=\varepsilon}, q^{-1}) =: \delta .$$

Therefore, by Claim 3.29, the value of $\mathscr{S}'$ is at least

$$1 - \varepsilon - 10\sqrt{\delta} \geq 1 - 11\sqrt{\delta} .$$

Second perturbation:

---

[101] Note that this agrees the marginal distribution from edges to vertices in the game $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$.

For every vertex $\mathtt{v} = (\mathtt{Player}, \mathtt{w})$ of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, let $\mathcal{G}^{\mathtt{v}}$ be a PVM of dimension N with outcomes in k-tuples of degree at most 9 polynomials, where $k = 2$ in case $\mathtt{Player} = \mathtt{A}$ or $\mathtt{B}$, and $k = \heartsuit(n)$ in case $\mathtt{Player} = \mathtt{Oracle}$. Assume the N-dimensional quantum strategy $\mathscr{S} = \{\mathcal{P}\}$ for $\mathfrak{G}$ defined by $\mathcal{P}^{\mathtt{v}, \mathtt{Space}^\rho} = \mathcal{G}^{\mathtt{v}}_{[\mathrm{eval}_\rho(\cdot)]}$ has value $1 - \varepsilon$.

An edge e that is sampled according to $\mu^{\mathtt{v}}$ has a probability of at least $\frac{3}{10}$ to have the $\mathtt{Space}$ associated with $\mathtt{v}$ be $\mathtt{ALine}$, namely for one of its endpoints to be $\mathtt{v}, \mathtt{ALine}^{\mathscr{L}}$. Hence, the strategy $\mathcal{P}$ loses against an edge with endpoint $\mathtt{v}, \mathtt{ALine}^{\mathscr{L}}$ with probability of at most $\frac{10}{3}\varepsilon^{\mathtt{v}} \leq 4\varepsilon^{\mathtt{v}}$, given that such an edge was sampled. In such a case, $\mathfrak{G}$ checks whether Item (3) is satisfied.

Let f be an individual degree at most 9 polynomial which is not indifferent to the ith variable of S. Then, $f = \sum_{j=0}^{9} \alpha_j \mathsf{X}_i^j$, where X_i is the ith variable of S, each α_j is a polynomial of individual degree at most 9 in the other $m - 1$ variables from S, and at least one of the α_j's for $j \geq 1$ is not identically zero — denote by $1 \leq l \leq 9$ the index such that α_l is not the zero polynomial. For $u \in \mathbb{F}_q^m$, recall from Theorem 5.23 the notation $\hat{u}^i$ for the vector in $\mathbb{F}_q^{m-1}$ recovered by removing from u its ith coefficient. As α_l is an individual degree at most 9 polynomial, it has total degree at most $9(m-1)$. By applying the Schwartz-Zippel Lemma 5.19, we deduce that for at least $1 - \frac{9(m-1)}{q}$ of the points $u \in \mathbb{F}_q^m$, we have $\alpha_l(\hat{u}^i) \neq 0$, and thus the restriction of f to the axis parallel line $\mathscr{L}(u, e_i)$ is not constant.

Recall that the probability the measurement outcome of $\mathcal{G}^{\mathtt{v}}$ is a specific tuple $f_1, ..., f_k$ is $\tau(\mathcal{G}^{\mathtt{v}}_{f_1,...,f_k})$. A tuple $f_1, ..., f_k$ is said to be *bad* if there is an index $s \in [k]$ and a direction $i \in [m]$ such that f_s is not indifferent to the ith variable of S, although it should according to the game checks. By the analysis from the previous paragraph, given a bad tuple $f_1, ..., f_k$, and sampling an axis parallel line $\mathscr{L}$ uniformly at random, there is a probability of at least $1/m$ that $\mathscr{L}$ is in direction i, and a probability of at least $1 - \frac{9(m-1)}{q}$ that $f_s(\mathscr{L})$ is not constant, which makes $\mathfrak{G}$ reject $\mathrm{eval}_{\mathscr{L}}(f_1, ..., f_k)$ due to Item (3). Therefore,

$$\frac{1}{m} \cdot \left(1 - \frac{9(m-1)}{q}\right) \cdot \tau\left(\sum_{f_1,...,f_k \text{ is bad}} \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}\right) \leq 4\varepsilon^{\mathtt{v}} . \tag{197}$$

By choosing $\gamma^{\mathtt{v}} = \frac{4m}{1 - \frac{9(m-1)}{q}} \cdot \varepsilon^{\mathtt{v}}$, one deduces that $\tau\left(\sum_{f_1,...,f_k \text{ is bad}} \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}\right) \leq \gamma^{\mathtt{v}}$. We use the PVM $\mathcal{G}^{\mathtt{v}}$ to define a new PVM $\mathcal{H}^{\mathtt{v}}$ with outcomes in k-tuples of individual degree at most 9 polynomials (as is $\mathcal{G}^{\mathtt{v}}$): for a bad tuple $f_1, ..., f_k$, we let $\mathcal{H}^{\mathtt{v}}_{f_1,...,f_k} = 0$; for a good tuple $f_1, ..., f_k$ such that not all the polynomials are 0, we let $\mathcal{H}^{\mathtt{v}}_{f_1,...,f_k} = \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}$; for the tuple of only zero polynomials, we let $\mathcal{H}^{\mathtt{v}}_{0,...,0} = \mathcal{G}^{\mathtt{v}}_{0,...,0} + \sum_{f_1,...,f_k \text{ is bad}} \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}$. In other words, if we define the map Ξ from k-tuples of individual degree at most 9 polynomials to itself that is the identity on good tuples, but sends all bad tuples to the all zero tuple, then $\mathcal{H}^{\mathtt{v}}$ is the Ξ-evaluated PVM (Defintion 3.32), namely $\mathcal{H}^{\mathtt{v}} = \mathcal{G}^{\mathtt{v}}_{[\Xi(\cdot)]}$. Now,

$$\sum_{f_1,...,f_k} \|\mathcal{H}^{\mathtt{v}}_{f_1,...,f_k} - \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}\|_{hs}^2 = \sum_{f_1,...,f_k \text{ is bad}} \|\mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}\|_{hs}^2 + \left\|\sum_{f_1,...,f_k \text{ is bad}} \mathcal{G}^{\mathtt{v}}_{f_1,...,f_k}\right\|_{hs}^2$$

$$=_{\mathcal{G} \text{ is a PVM}} 2 \cdot \sum_{f_1,...,f_k \text{ is bad}} \tau(\mathcal{G}^{\mathtt{v}}_{f_1,...,f_k})$$

$$\leq 2\gamma^{\mathtt{v}} ,$$

which means

$$\mathcal{H}^{\mathtt{v}} \approx_{2\gamma^{\mathtt{v}}} \mathcal{G}^{\mathtt{v}} . \tag{198}$$

As $\mathcal{G}^{\mathtt{v}}$ and $\mathcal{H}^{\mathtt{v}}$ are PVMs, by Claim 3.35, (198) implies $\mathcal{H}^{\mathtt{v}}_{[\mathrm{eval}_\rho(\cdot)]} \approx_{2\gamma^{\mathtt{v}}} \mathcal{G}^{\mathtt{v}}_{[\mathrm{eval}_\rho(\cdot)]}$. Hence, if we define a strategy $\mathscr{S}' = \{\mathcal{Q}\}$ for $\mathfrak{G}$ by letting $\mathcal{Q}^{\mathtt{v}, \mathtt{Space}^\rho} = \mathcal{H}^{\mathtt{v}}_{[\mathrm{eval}_\rho(\cdot)]}$, then $\mathscr{S}'$ passes checks (1) and (3) perfectly, and by (198) $\mathscr{S}$ and $\mathscr{S}'$ are $2\mathbb{E}_{\mathtt{v}\sim\mu}[\gamma^{\mathtt{v}}]$-close. Letting $\gamma = 2\mathbb{E}_{\mathtt{v}\sim\mu}[\gamma^{\mathtt{v}}] = \frac{8m}{1 - \frac{9(m-1)}{q}} \cdot \varepsilon$, and using Claim 3.29, the value of $\mathscr{S}'$ is at least

$$1 - \varepsilon - 10\sqrt{\gamma} \geq 1 - \frac{81m}{1 - \frac{9(m-1)}{q}} \cdot \sqrt{\varepsilon} .$$

183

Translating the resulting strategy into a high value strategy for $\mathfrak{DoubleCover}(\mathfrak{G})$:

For every vertex $\mathtt{v} = (\mathtt{Player}, \mathtt{w})$ of $\mathfrak{Oracle}(\mathfrak{DoubleCover}(\mathcal{V}_n))$, let $\mathcal{H}^{\mathtt{v}}$ be a PVM of dimension N with outcomes in good k-tuples of degree at most 9 polynomials (on the variable set S), where $k = 2$ in case $\mathtt{Player} = \mathtt{A}$ or $\mathtt{B}$, and $k = \heartsuit(n)$ in case $\mathtt{Player} = \mathtt{Oracle}$. Assume the N-dimensional quantum strategy $\mathscr{S} = \{\mathcal{P}\}$ for $\mathfrak{G}$ defined by $\mathcal{P}^{\mathtt{v}, \mathtt{Space}^\rho} = \mathcal{H}^{\mathtt{v}}_{[\mathrm{eval}_\rho(\cdot)]}$ has value $1 - \varepsilon$. Note that $\mathscr{S}$ passes checks (1) and (3) of $\mathfrak{G}$ perfectly. In particular, at oracle player vertices, the $\heartsuit(n)$-tuple of polynomials measured by $\mathcal{H}^{(\mathtt{Oracle}, z)}$ is a degree 9 PCP over $\mathbb{F}_q$ in the sense of Definition 5.68 — not only these are polynomials from $\mathbb{F}_q^S$ to $\mathbb{F}_q$, the consisting polynomials are indeed indifferent to the relevant directions, as check (3) is always passed.

For a fixed $z \in \mathbb{F}_2^r$, an edge e sampled according to $\mu^{(\mathtt{Oracle}, z)}$ has a probability of at least $\frac{2}{5}$ to have the $\mathtt{Space}$ associated with $(\mathtt{Oracle}, z)$ be $\mathtt{Point}$, namely for one of its endpoints to be $(\mathtt{Oracle}, z), \mathtt{Point}^p$. Hence, the probability $\mathcal{P}$ loses against an edge with endpoint $(\mathtt{Oracle}, z), \mathtt{Point}^p$ is at most $\frac{5}{2}\varepsilon^{(\mathtt{Oracle}, z)} \le 3\varepsilon^{(\mathtt{Oracle}, z)}$ (given such an edge was sampled). In this case, $\mathfrak{G}$ checks Item (4).

A PCP Π is sad to be *bad* (with respect to z) if it does not pass the checks (170), (171), (172) and (173) perfectly for all $p \in \mathbb{F}_q^m$, which is the same as failing with positive probability check (4) given $(\mathtt{Oracle}, z), \mathtt{Point}^p$ is an endpoint of the sampled edge and Π was the outcome of measuring $\mathcal{H}^{(\mathtt{Oracle}, z)}$. By the soundness clause of Corollary 5.72, given that Π is a bad PCP, it will fail versus at least $1 - \frac{63m}{q}$ of the points of $\mathbb{F}_q^m$. Combined with the observation from the previous paragraph, we can deduce that

$$\left(1 - \frac{63m}{q}\right) \cdot \tau \left(\sum_{\Pi \text{ is bad}} \mathcal{H}_\Pi^{(\mathtt{Oracle}, z)}\right) \le 3\varepsilon^{(\mathtt{Oracle}, z)} \; ;$$

namely, when sampling Π according to $\mathcal{H}^{(\mathtt{Oracle}, z)}$, there is a probability of at most $\frac{3}{1 - \frac{63m}{1}} \cdot \varepsilon^{(\mathtt{Oracle}, z)}$ that Π is bad. Therefore, the soundness clause of Corollary 5.72 again, when sampling Π according to $\mathcal{H}^{(\mathtt{Oracle}, z)}$ the restriction of the polynomials $g_1^{\mathfrak{R}}, g_1^{\mathfrak{L}}, g_2^{\mathfrak{R}}, g_2^{\mathfrak{L}}$ in Π to the subcube pass the game $\mathcal{V}_n$ given $\mathfrak{s}^{\mathtt{A}}(z)\mathfrak{s}^{\mathtt{B}}(z)$ were asked.

Let $\mathrm{Restrict}_1$ (respectively $\mathrm{Restrict}_2$ and $\mathrm{Restrict}_{12}$) be the function that take a PCP Π as input, and outoputs only the polynomilals $g_1^{\mathfrak{R}}, g_1^{\mathfrak{L}}$ (respectively $g_2^{\mathfrak{R}}, g_2^{\mathfrak{L}}$ and $g_1^{\mathfrak{R}}, g_1^{\mathfrak{L}}, g_2^{\mathfrak{R}}, g_2^{\mathfrak{L}}$) from it. For a fixed $z \in \mathbb{F}_2^r$, the probability $e \sim \mu^{(\mathtt{Oracle}, z)}$ is of the form

$$(\mathtt{A}, \mathtt{x}), \mathtt{Point}^p - (\mathtt{Oracle}, z), \mathtt{Point}^p \,, \tag{199}$$

where $\mathtt{x} = \mathfrak{s}^{\mathtt{A}}(z)$, is at least $\frac{1}{25}$ (and the same is true by replacing for $(\mathtt{A}, \mathtt{x})$ by $(\mathtt{B}, \mathtt{y})$ for $\mathtt{y} = \mathfrak{s}^{\mathtt{B}}(z)$). In this case, check (2) of $\mathfrak{G}$ is checked. Therefore, the probability $\mathcal{P}$ loses against check (2) given an edge of the form (199) was sampled is at most $25\varepsilon^{(\mathtt{Oracle}, z)}$. Assume we mutually measured $(g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{R}}, g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{L}}), (g_1^{\mathfrak{R}}, g_1^{\mathfrak{L}}) \sim (\mathcal{H}^{(\mathtt{A}, \mathtt{x})}, \mathcal{H}_{[\mathrm{Restrict}_1(\cdot)]}^{(\mathtt{Oracle}, z)})$. Then, as all of these polynomials are of individual degree at most 9, if $g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{R}} \ne g_1^{\mathfrak{R}}$ or $g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{L}} \ne g_1^{\mathfrak{L}}$, then by the Schwartz–Zippel Lemma 5.19, for at least $1 - \frac{9m}{q}$ of the points in $\mathbb{F}_q^m$ we have $g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{R}}(p) \ne g_1^{\mathfrak{R}}(p)$ or $g_{\mathtt{A}, \mathtt{x}}^{\mathfrak{L}}(p) \ne g_1^{\mathfrak{L}}(p)$. Hence, by letting $\zeta^{(\mathtt{Oracle}, z)} = \frac{25}{1 - \frac{9m}{q}} \cdot \varepsilon^{(\mathtt{Oracle}, z)}$, and recalling the relation between distance and inconsistency of PVMs, we deduce that

$$\mathcal{H}^{(\mathtt{A}, \mathtt{x})} \approx_{2\zeta^{(\mathtt{Oracle}, z)}} \mathcal{H}_{[\mathrm{Restrict}_1(\cdot)]}^{(\mathtt{Oracle}, z)} \,.$$

Similarly, we can deduce that

$$\mathcal{H}^{(\mathtt{B}, \mathtt{y})} \approx_{2\zeta^{(\mathtt{Oracle}, z)}} \mathcal{H}_{[\mathrm{Restrict}_2(\cdot)]}^{(\mathtt{Oracle}, z)} \,.$$

By applying Claim 3.18 twice, we can deduce that the distribution on quadruples of degree at most 9 polynomials induced by mutually measuring according to $(\mathcal{H}^{(\mathtt{A}, \mathtt{x})}, \mathcal{H}^{(\mathtt{B}, \mathtt{y})})$ is at most $4\sqrt{2\zeta^{(\mathtt{Oracle}, z)}}$ away in L^1-distance from the distribution induced by mutually measuring according to $(\mathcal{H}_{[\mathrm{Restrict}_1(\cdot)]}^{(\mathtt{Oracle}, z)}, \mathcal{H}_{[\mathrm{Restrict}_2(\cdot)]}^{(\mathtt{Oracle}, z)})$, which in turn is equal to the distribution induced by measuring according to $\mathcal{H}_{[\mathrm{Restrict}_{12}(\cdot)]}^{(\mathtt{Oracle}, z)}$. As measuring according to $\mathcal{H}^{(\mathtt{Oracle}, z)}$ produces a good PCP with probability of

at least $1 - \frac{3}{1-\frac{63m}{1}} \cdot \varepsilon^{(\texttt{Oracle},z)}$, by the soundness clause of Corollary 5.72, the restriction to the subcuce of the measurement outcome of $\mathcal{H}^{(\texttt{Oracle},z)}_{[\text{Restrict}_{12}(\cdot)]}$ passes $\mathcal{V}_n$ given $\mathfrak{s}^{\texttt{A}}(z)\mathfrak{s}^{\texttt{B}}(z)$ was asked with at least the same probability. Hence, the restriction to the subcuce of the mutual measurement according to $(\mathcal{H}^{(\texttt{A},\texttt{x})}, \mathcal{H}^{(\texttt{B},\texttt{y})})$ passes $\mathcal{V}_n$ given $\mathfrak{s}^{\texttt{A}}(z)\mathfrak{s}^{\texttt{B}}(z)$ was asked with probability of at least $1 - \frac{3}{1-\frac{63m}{1}} \cdot \varepsilon^{(\texttt{Oracle},z)} - 4\sqrt{2\zeta^{(\texttt{Oracle},z)}}$.

Let $\mathscr{S}' = \{\mathcal{Q}\}$ be the strategy for $\mathfrak{DoubleCover}(\mathcal{V}_n)$ defined by $\mathcal{Q}^{(\texttt{Player},\texttt{w})} = \mathcal{H}^{(\texttt{Player},\texttt{w})}_{[\text{Res}(\cdot)]}$, where $\text{Res}(\cdot)$ is the restriction to the subcube function from Definition 5.17, not to be confused with $\text{Restrict}_\circ$. According to the previous analysis, the value of $\mathscr{S}'$ versus $\mathfrak{DoubleCover}(\mathcal{V}_n)$ is at least

$$1 - \mathop{\mathbb{E}}_{z\in\mathbb{F}_2^r}\left[\frac{3}{1-\frac{63m}{1}} \cdot \varepsilon^{(\texttt{Oracle},z)} - 4\sqrt{2\zeta^{(\texttt{Oracle},z)}} \right],$$

where the expectation is over a uniformly random $z \in \mathbb{F}_2^r$. By the definition of $\zeta^{(\texttt{Oracle},z)}$, the concavity of $\sqrt{\cdot}$, and the fact $x \le x^2$ for $x \ge 1$, we have that

$$\text{val}(\mathfrak{DoubleCover}(\mathcal{V}_n), \mathscr{S}') \ge 1 - \frac{3}{1-\frac{63m}{q}} \mathop{\mathbb{E}}_{z\in\mathbb{F}_2^r}\left[\varepsilon^{(\texttt{Oracle},z)}\right] - \frac{200}{1-\frac{9m}{q}} \cdot \sqrt{\mathop{\mathbb{E}}_{z\in\mathbb{F}_2^r}\left[\varepsilon^{(\texttt{Oracle},z)}\right]} .$$

By inspecting the type graph of $\mathfrak{G}$, and recalling the marginal distribution μ induces on the vertices $(\texttt{Player},\texttt{w})$, we have $\mu((\texttt{Oracle},z)) = \frac{8}{25} \cdot \frac{1}{2^r}$. Therefore,

$$\mathop{\mathbb{E}}_{z\in\mathbb{F}_2^r}\left[\varepsilon^{(\texttt{Oracle},z)}\right] = \sum_{z\in\mathbb{F}_2^r} \frac{1}{2^r} \cdot \varepsilon^{(\texttt{Oracle},z)}$$

$$\le \frac{25}{8} \sum_{z\in\mathbb{F}_2^r} \mu((\texttt{Oracle},z))\varepsilon^{(\texttt{Oracle},z)}$$

$$\le 4 \cdot \sum_{\texttt{v}} \mu(\texttt{v})\varepsilon^{\texttt{v}}$$

$$= 4\varepsilon .$$

All in all, we found a strategy for $\mathfrak{DoubleCover}(\mathcal{V}_n)$ with value of at least

$$1 - \frac{12\varepsilon}{1-\frac{63m}{q}} - \frac{400\sqrt{\varepsilon}}{1-\frac{9m}{q}} \ge 1 - \frac{412}{1-\frac{63m}{q}} \cdot \sqrt{\varepsilon} .$$

Combining all of the above to deduce soundness:

- Starting with an N-dimensional value $1 - \varepsilon$ strategy for $\mathfrak{G}$, the first perturbation allows us to find an N-dimensional value at least $1 - \varepsilon'$ strategy for $\mathfrak{G}$ that always passes check (1), where $\varepsilon' = 11\sqrt{\delta}$ for $\delta = \delta_{\text{LD}}(m, 9, \heartsuit(n), 2\varepsilon, q^{-1})$.

- By applying the second perturbation on the resulting strategy, we find an N-dimensional strategy for $\mathfrak{G}$ with value of at least $1 - \varepsilon''$ for $\mathfrak{G}$ that always pass both check (1) and (3) perfectly, where $\varepsilon'' = \frac{81m}{1-\frac{9(m-1)}{q}} \cdot \sqrt{\varepsilon'}$.

- Finally, by restricting the recovered strategy to the isolated player vertices, we showed that the resulting N-dimensional strategy for $\mathfrak{DoubleCover}(\mathcal{V}_n)$ has value of at least $1 - \varepsilon'''$, where $\varepsilon''' = \frac{412}{1-\frac{63m}{q}} \cdot \sqrt{\varepsilon''}$.

- Now,

$$\varepsilon''' \le \frac{11 \cdot 81m \cdot 412}{\left(1 - \frac{63m}{q}\right)\left(1 - \frac{9(m-1)}{q}\right)} \cdot \delta^{1/8} \le \frac{10^6 \cdot m}{1 - \frac{72m}{q}} \cdot \delta^{1/8} ,$$

which finishes the proof.

$\square$

Though we tried to emphasize the combinatorial aspects of the answer reduced game $\mathfrak{AnsRed}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, t)$, it is already quite "normal form verify"ish in nature. The following is thus quite straightforward.

Claim 5.81 (Algorithmic (partial) Answer Reduction). *There is a polynomial time TM* PartialAnsRed *that takes as input a h-level TNFV $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$, a positive integer D and five 1-input TMs $\Lambda, \Delta, T, Q, \mathcal{FE}$,[102] and outputs a typed* $\max(h, 3)$-*level TNFV*

$$\mathcal{V}' = (\mathcal{S}', \mathcal{A}', \mathcal{L}', \mathcal{D}) = \mathsf{PartialAnsRed}(\mathcal{V}, \Lambda, \Delta, D, T, Q, \mathcal{FE})$$

with the following properties:

- Combinatorial Answer Reduction: *Given that the inputs satisfy that —*
  - *the input TMs always halt;*
  - *$\mathcal{V}$ is purified and 2^Λ-padded;*
  - *$\Delta(n) \geq \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1}$ for every n;*
  - *$Q(n) \geq \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$ for every n;*
  - *$\forall n \in \mathbb{N}: \quad T(n) \geq c \cdot \left(\mathbb{T}(\Lambda; n)^c + \mathbb{T}(\Delta; n)^c + 2^{c \cdot \Lambda(n)} + \Delta(n)^c + \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)^c \right)$, where $c \geq 6$ is the positive integer implied by the* poly *notation in* (150);
  - *$|\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \leq D$;*

  then $\mathcal{V}'_n = \mathfrak{AnsRed}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, 2\mathcal{FE}(n) + 1)$, where $\mathfrak{AnsRed}$ is from Definition 5.79.

- Running time and description length: *The running time of $\mathcal{A}'$ is*

  $$\mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), \mathbb{T}(\mathcal{FE}; n), \Lambda(n), \log(\Delta(n)), \log(T(n)), Q(n), D, \mathcal{FE}(n)),$$

  while the running times of $\mathcal{S}'$ and $\mathcal{L}'$ depend polynomially on the same parameters and also on $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$. For description lengths, we have that

  $$|\mathcal{S}'| = \mathrm{poly}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|, |\mathcal{S}|),$$
  $$|\mathcal{A}'| = \mathrm{poly}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|),$$
  $$|\mathcal{L}'| = \mathrm{poly}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|, |\mathcal{S}|, |\mathcal{L}|).$$

  *Note that the running times of $\mathcal{A}$ and $\mathcal{L}$ **do not participate** in the above bounds, only the **log** of the upper bound on them, namely $\log(T(n))$ — which is the goal of this transformation.*

Proof. In the operation of all of the TMs in $\mathcal{V}'$ there is a pre-processing phase in which the following values are calculated:

- $t = 2\mathcal{FE}(n) + 1$ — this requires $O(\mathbb{T}(\mathcal{FE}; n))$ time;

- use Fact 5.24 to retrieve a basis for $\mathbb{F}_q$ over $\mathbb{F}_2$, where $q = 2^t$ — requires $\mathrm{poly}(t) = \mathrm{poly}(\mathcal{FE}(n))$ time;

- compute $\Lambda(n), \Delta(n), T(n)$ and $Q(n)$ — requires $\mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n))$ time;

- compute $M(n)$ and $s(n)$ as defined in Proposition 5.62 — this requires

  $$\mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), n, \log(T(n)), Q(n), D) - \text{time},$$

  and also $M(n), s(n)$ are $\mathrm{poly}(\log(T(n)), Q(n), D)$-sized;

---

[102]The role of $\mathcal{FE}$ is to be the "field exponent", namely to calculate the appropriate (log of the) field size $q = 2^t$ for the nth game of the normal form verifier.

186

- compute $\Diamond(n)$ from (147), $m = |S|$ from (159) and $\heartsuit(n)$ from (187) — requires

$$\mathrm{polylog}(\Lambda(n), \log(\Delta(n)), \log(T(n)), Q(n), D) - \text{time} .$$

The Sampler: $\mathcal{S}'$ is the product of $\mathcal{S}$ and $\mathcal{S}^{\mathrm{LD}}$ (from Fact 5.78) with parameters $(9, m, 2^t, \cdot)$, where $\cdot$ does not affect the sampler. Running such a sampler is just the sum of the running times of the component samplers $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$ and

$$\mathbb{T}(\mathcal{S}^{\mathrm{LD}}; 9, m, 2^t, \cdot, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(t, m) = \mathrm{poly}(\Lambda(n), \log(\Delta(n)), \log(T(n)), Q(n), D, \mathcal{FE}(n)) .$$

The Answer length calculator: $\mathcal{A}'$ follows Table 3. This requires outputting (the encoding of) at most $t \cdot (9m + 1) \cdot \heartsuit(n)$-many 1s, which takes less than

$$\mathrm{poly}(t, m, \heartsuit(n)) = \mathrm{poly}(\Lambda(n), \log(\Delta(n)), \log(T(n)), Q(n), D, \mathcal{FE}(n)) \quad \text{time} .$$

The Linear constraints processor: $\mathcal{L}'$ needs to check given its input which of Item (1), Item (2), Item (3) and Item (4) are needed. Implementing Item (1) requires manipulating length $\mathrm{poly}(m)$ vectors of $\mathbb{F}_q$ inputs, and doing arithmetic using them, which in turn takes $\mathrm{poly}(m, \log(q)) = \mathrm{poly}(m, t)$-time. In any case, it needs to output t-many constraints, writing which again takes at most $\mathrm{poly}(m, t)$-time. Implementing Item (2) is just adding a fixed number of consistency equations, where this number does not surpass $2t(9m + 1) = \mathrm{poly}(t, m)$-many equations; this again takes at most $\mathrm{poly}(m, t)$-time. Implementing Item (3) is adding at most $40tm^2$-many equations that force a certain variable to be zero, which again takes at most $\mathrm{poly}(m, t)$-time. Finally, implementing Item (4) is slightly more involved: It requires us to calculate $\mathfrak{s}^{\mathsf{A}}(z)$ and $\mathfrak{s}^{\mathsf{B}}(z)$, which are calls to $\mathcal{S}$ and this takes at most $2\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$-time. It requires us to calculate $\mathcal{C}$ which is the third output of $\mathsf{SuccinctTOI}(\mathcal{V}, \Lambda, \Delta, D, T, Q, n, \mathfrak{s}^{\mathsf{A}}(z), \mathfrak{s}^{\mathsf{B}}(z))$, which takes, according to Proposition 5.62 at most

$$\mathrm{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), n, \log(T(n)), Q(n), D) \quad \text{time} .$$

Finally, evaluating the Tseitin polynomial $T_{\mathcal{C}}$ at p requires us to calculate the value at every non-input gate, which in turn is some arithmetic operation in $\mathbb{F}_q$ which takes $\mathrm{poly}(t)$-time. As there are $s(n)$ non-input gates, this takes $\mathrm{poly}(t, s(n)) = \mathrm{poly}(\mathcal{FE}(n), \log(T(n)), Q(n), D)$-time. Combining all of the above provides the running time bound.

Description lengths: The desecription lengths of $\mathcal{S}', \mathcal{A}', \mathcal{L}'$ depend polynomially on the parameters that play a role in their operation, which is only $\Lambda, \Delta, D, T, Q, \mathcal{FE}$ for $\mathcal{A}'$, all of the above in addition to $\mathcal{S}$ for $\mathcal{S}'$, and all of the above including both $\mathcal{S}$ and $\mathcal{L}$ for $\mathcal{L}'$. $\qquad \square$

Remark 5.82. The reason Claim 5.81 is only **partial** answer reduction, is that it needs to be combined with the padding and purification transformations, as well as choosing the TMs $\mathcal{FE}, \Lambda, \Delta, \mathcal{T}, Q$ depending only on h, h' and λ. This is done in the following proof of Answer Reduction

.

5.6 Proving the main theorem of Answer Reduction: Theorem 5.1

The goal is to describe the TM $\mathsf{AnswerReduction}_{h,h'}$ and prove it possesses the desired properties according to the Theorem. To make it easier for the reader to follow, let us say where each TM that needs to be defined is located: Λ and Q are defined in (200); Δ is defined in (206); T is defined in (207); D is defined in (208); $\mathcal{FE}$ is defined in the paragraph preceding (210).

Throughout the proof, we need to consider many constants which control previous asymptotic notations — behind every $g(n) = \mathrm{poly}(f(n))$ (respectively $g(n) = O(f(n))$) there is a universal constant c such that $g(n) \le cf(n)^c$ (respectively $g(n) \le cf(n)$). We list all of the used constants in Table 4.

Recall that $c_{\mathrm{QR}}(h')$ is the constant guaranteed in Theorem 4.36. Let c_1 be a later to be fixed constant, such that $c_1 \ge c_{\mathrm{QR}}(h')$, which in particular means it depends on h'. Define Λ and Q to be the TMs that take n (in binary) as input and output $c_1(\lambda^{c_1} + n^{c_1})$, namely

$$Q(n) = \Lambda(n) = c_1(\lambda^{c_1} + n^{c_1}) , \tag{200}$$

Constant	Origin
c_{LD}	Theorem 5.76 and (209)
$c_{\text{QR}}(h')$	Theorem 4.36
c_1	(200)
c_2	(201)
c_3	(202)
c_4	Claim 4.49 and (203)
c_5	Claim 5.42 and (204)
c_6	(206)
c_7	(207)
c_8	(208)
c	(211)

Table 4: Constants along the proof of Theorem 5.1.

which have runtime and description length

$$
\begin{aligned}
\mathbb{T}(Q;n) = \mathbb{T}(\Lambda;n) = \text{polylog}_{h'}(\lambda, n) &\le c_2(\log^{c_2}\lambda + \log^{c_2} n)\,, \\
|Q| = |\Lambda| = \text{polylog}_{h'}(\lambda) &\le c_2 \log^{c_2}\lambda\,,
\end{aligned}
\tag{201}
$$

for some positive integer c_2 that depends on c_1 and thus on h'. Let 2^Λ be the TM that outputs $2^{\Lambda(n)}$ many 1s given n as input, which has runtime and description length

$$
\begin{aligned}
\mathbb{T}(2^\Lambda;n) = \text{poly}(\mathbb{T}(\Lambda;n), 2^{\Lambda(n)}) &\le c_3 \cdot 2^{c_3(\lambda^{c_3}+n^{c_3})} \\
|2^\Lambda| = O(|\Lambda|) &\le c_3 \log^{c_3}\lambda\,,
\end{aligned}
\tag{202}
$$

for some positive integer c_3 that depends on c_1, c_2 and thus on h'.

Let $\mathcal{V}^{(1)} = (\mathcal{S}, \mathcal{A}^{(1)}, \mathcal{L}^{(1)}, \mathcal{D}) = \text{Padding}(\mathcal{V}, 2^\Lambda)$ — the sampler stays the same in this transformation — where Padding was defined in Claim 4.49. By that claim,

$$
\begin{aligned}
\mathbb{T}(\mathcal{A}^{(1)};n,\cdot,\cdot) = O(\mathbb{T}(2^\Lambda;n)) &\le c_4 \cdot 2^{c_4(\lambda^{c_4}+n^{c_4})}\,, \\
|\mathcal{A}^{(1)}| = O(|2^\Lambda|) &\le c_4 \log^{c_4}\lambda\,, \\
\mathbb{T}(\mathcal{L}^{(1)};n,\cdot,\cdot,\cdot,\cdot) = \text{poly}(\mathbb{T}(2^\Lambda;n), \mathbb{T}(\mathcal{S};n,\cdot,\cdot,\cdot,\cdot,\cdot), \mathbb{T}(\mathcal{A};n,\cdot,\cdot), \mathbb{T}(\mathcal{L};n,\cdot,\cdot,\cdot,\cdot)) & \\
\le c_4(2^{c_4(\lambda^{c_4}+n^{c_4})} + \mathbb{T}(\mathcal{S};n,\cdot,\cdot,\cdot,\cdot,\cdot)^{c_4} + \mathbb{T}(\mathcal{A};n,\cdot,\cdot)^{c_4} + &\mathbb{T}(\mathcal{L};n,\cdot,\cdot,\cdot,\cdot)^{c_4})\,, \\
|\mathcal{L}^{(1)}| = \text{poly}(|2^\Lambda|, |\mathcal{S}|, |\mathcal{A}|, |\mathcal{L}|) &\le c_4(\log^{c_4}\lambda + |\mathcal{S}|^{c_4} + |\mathcal{A}|^{c_4} + |\mathcal{L}|^{c_4})\,,
\end{aligned}
\tag{203}
$$

for some constant c_4 that depends on the asymptotic bounds from Claim 4.49 as well as c_1, c_2, c_3 (and thus h').

Let $\mathcal{V}^{(2)} = (\mathcal{S}, \mathcal{A}^{(1)}, \mathcal{L}^{(2)}, \mathcal{D}) = \text{Purify}(\mathcal{V}^{(1)})$ — the sampler and answer length calculator stay the same in this transformation — where Purify was defined in Claim 5.42. Then, we have

$$
\begin{aligned}
\mathbb{T}(\mathcal{L}^{(2)};n,\cdot,\cdot,\cdot,\cdot) = \text{poly}(\mathbb{T}(\mathcal{A}^{(1)};n,\cdot,\cdot), \mathbb{T}(\mathcal{L}^{(1)};n,\cdot,\cdot,\cdot,\cdot)) & \\
\le c_5(2^{c_5(\lambda^{c_5}+n^{c_5})} + \mathbb{T}(\mathcal{S};n,\cdot,\cdot,\cdot,\cdot,\cdot)^{c_5} + &\mathbb{T}(\mathcal{A};n,\cdot,\cdot)^{c_5} + \mathbb{T}(\mathcal{L};n,\cdot,\cdot,\cdot,\cdot)^{c_5})\,, \\
|\mathcal{L}^{(2)}| = \text{poly}(|\mathcal{L}^{(1)}|, |\mathcal{A}^{(1)}|) & \\
\le c_5(\log^{c_5}\lambda + |\mathcal{S}|^{c_5} + |\mathcal{A}|^{c_5} + &|\mathcal{L}|^{c_5})\,,
\end{aligned}
\tag{204}
$$

for some c_5 that depends on the asymptotic bounds from Claim 5.42 as well as c_1, c_2, c_3, c_4. Now, $\mathcal{V}^{(2)}$ is 2^Λ-padded and purified.

We are left to define $\Delta, T, \mathcal{FE}$ and D. Let Δ be the TM that calculates

$$\Delta(n) = 2^{2\Lambda(n)+2} - 2^{\Lambda(n)+1}. \tag{205}$$

Its runtime bound and description length satisfy

$$\mathbb{T}(\Delta; n) = \text{poly}(\mathbb{T}(\Lambda; n), \Lambda(n)) \leq c_6(\lambda^{c_6} + n^{c_6}),$$
$$|\Lambda| = O(|\Lambda|) \leq c_6 \log^{c_6} \lambda, \tag{206}$$

for some positive integer c_6 that depends on c_1, c_2. This choice of Δ is made so that $\Diamond(n)$ is a nice expression $2\Lambda(n) + 2$.

As T needs to satisfy (156) given that

$$\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \leq 2^{c_{\text{QR}}(h')(\lambda^{c_{\text{QR}}(h')} + n^{c_{\text{QR}}(h')})},$$

by the previous upper bounds on $\mathbb{T}(\Lambda; n)$, $\mathbb{T}(\Delta; n)$, $\Lambda(n)$ and $\Delta(n)$, we can choose a positive integer c_7, that depends on all the previous constants $c_1, ..., c_6$ as well as the asymptotic notation from (150) — but not on anything else, and in particular not on $\mathcal{L}$ — so that

$$T(n) = c_7 \cdot 2^{c_7(\lambda^{c_7} + n^{c_7})} \tag{207}$$

indeed satisfies (156) under the upper bound assumption on $\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)$.[103] Moreover,

$$\mathbb{T}(T; n) = \text{poly}_{h'}(\lambda, n) \quad \text{and} \quad |T| = O_{h'}(\log \lambda).$$

As D needs to bound $|\Lambda|, |\Lambda|, |T|$ and $|Q|$, and also $|\mathcal{V}|$ but only under the assumption that $|\mathcal{V}| \leq 5c_{\text{QR}}(h')\lambda^{c_{\text{QR}}(h')}$, by the previous calculated bounds there is a large enough positive integer c_8 (that depends on all previous constants and thus on h') so that

$$D = c_8 \lambda^{c_8} \tag{208}$$

indeed upper bounds all of the above.

The choice of $\mathcal{FE}$ requires us to associate more constants with previously used asymptotic notation. Let c_{LD} be the constant from the definition of δ_{LD} (183), namely

$$\delta_{\text{LD}}(m, d, k, \varepsilon, q^{-1}) = c_{\text{LD}}(m^{c_{\text{LD}}} + d^{c_{\text{LD}}} + k^{c_{\text{LD}}})(\varepsilon^{1/c_{\text{LD}}} + q^{-1/c_{\text{LD}}} + 2^{-md/c_{\text{LD}}}). \tag{209}$$

By running SuccinctTOI from Proposition 5.62 on $(\cdot, \cdot, \cdot, D, T, Q, n, \cdot, \cdot)$ — which takes

$$\text{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), n, \log(T(n)), Q(n), D) = \text{poly}_{h'}(\lambda, n)$$

time — we can retrieve $M(n)$ and $s(n)$ which are of size

$$\text{poly}(\log(T(n)), Q(n), D) = \text{poly}_{h'}(\lambda, n),$$

and thus

$$m = |S| = 4\Lambda(n) + 3\Diamond(n) + 3M(n) + s(n) + 12 = \text{poly}_{h'}(\lambda, n).$$

Note that

$$\heartsuit(n) = 12\Lambda(n) + 12\Diamond(n) + 6M(n) + s(n) + 35 \leq 4m,$$

and thus

$$m^{c_{\text{LD}}} + 9^{c_{\text{LD}}} + \heartsuit(n)^{c_{\text{LD}}} \leq (14m)^{c_{\text{LD}}}.$$

---

[103]Note that we do not assume at this point that $\mathcal{L}$ satisfies the runtime upper bound, only that c_7 can be calculated to satisfy (156) **in case** the upper bound holds. This is a crucial point, as the final sampler and answer length calculator depend on T, and are not allowed to depend on $\mathcal{L}$.

We now fix c_1 to be the smallest constant which is larger than $c_{\text{QR}}(h')$ and makes $\Lambda(n)$ big enough so that m satisfies $c_{\text{LD}}(14m)^{c_{\text{LD}}} \cdot 2^{-9m/c_{\text{LD}}} \leq m^{-16}$ for every $n \geq 2$. Now $\mathcal{FE}$ will be the TM that takes n as input, outputs an integer $\frac{t-1}{2}$, where t is the smallest positive odd integer for which $q = 2^t$ satisfies both

$$\frac{72m}{q} \leq \frac{1}{2} \qquad \text{and} \qquad c_{\text{LD}}(14m)^{c_{\text{LD}}} \cdot q^{-1/c_{\text{LD}}} \leq m^{-16} \,, \tag{210}$$

for every $n \geq 2$. Note that $t = \Theta(\log m)$, and as $\mathcal{FE}$ only needs to calculate m and thus $M(n), s(n)$, it takes it at most $\text{poly}_{h'}(\lambda, n)$ time. Also, its description is fixed up to appending the values h' and λ, which means $|\mathcal{FE}| = \text{polylog}_{h'}(\lambda)$. By the above choices and equation (209), we have

$$\delta = \delta_{\text{LD}}(m, 9, \heartsuit(n), 2\varepsilon, q^{-1}) \leq c_{\text{LD}}(14m)^{c_{\text{LD}}} \cdot (2\varepsilon)^{1/c_{\text{LD}}} + 2m^{-16} \,;$$

by the concavity of $(\cdot)^{1/8}$ and the fact it only shrinks positive integers, we have

$$\delta^{1/8} \leq c_{\text{LD}}(14m)^{c_{\text{LD}}} \cdot (2\varepsilon)^{1/8c_{\text{LD}}} + 2m^{-2} \,,$$

and as $\frac{72m}{q} \leq \frac{1}{2}$ we have

$$\frac{10^6 \cdot m}{1 - \frac{72m}{q}} \delta^{1/8} \leq 2 \cdot 10^6 (c_{\text{LD}} \cdot m \cdot (14m)^{c_{\text{LD}}} \cdot (2\varepsilon)^{1/8c_{\text{LD}}} + 2m^{-1}).$$

As we have $m \geq \Lambda(n) \geq \lambda + n \geq \sqrt{\lambda n}$, for every positive integer $c \geq 2$ we have $m^{-1} \leq (\lambda n)^{-1/c}$. All in all, there is a large enough positive integer c (that depends on all previous constants and bounds, and thus on h') such that

$$\frac{10^6 \cdot m}{1 - \frac{72m}{q}} \delta^{1/8} \leq c((\lambda n)^c \varepsilon^{1/c} + (\lambda n)^{-1/c}) \,. \tag{211}$$

We can finally define $\mathcal{V}_{\text{AR}} = (\mathcal{S}_{\text{AR}}, \mathcal{A}_{\text{AR}}, \mathcal{L}_{\text{AR}}, \mathcal{D}) = \text{PartialAnsRed}(\mathcal{V}^{(2)}, \Lambda, \Delta, T, Q, \mathcal{FE})$, where PartialAnsRed is from Claim 5.81. Let us prove that, indeed, $\mathcal{V}_{\text{AR}}$ satisfies the requirements of Theorem 5.1:

Time bounds and description lengths:
By Claim 5.81 and the above bonds, we have

$$|\mathcal{S}_{\text{AR}}| = \text{poly}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|, |\mathcal{S}|) = \text{poly}_{h'}(\log \lambda, |\mathcal{S}|) \,,$$
$$|\mathcal{A}_{\text{AR}}| = \text{poly}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|) = \text{polylog}_{h'}(\lambda) \,,$$
$$|\mathcal{L}_{\text{AR}}| = \text{poly}_{h'}(|\Lambda|, |\Delta|, |Q|, |T|, |\mathcal{FE}|, |\mathcal{S}|, |\mathcal{L}^{(2)}|) = \text{poly}_{h'}(\log \lambda, |\mathcal{V}|) \,.$$

For running times, by Claim 5.81, we have that $\mathbb{T}(\mathcal{A}_{\text{AR}}; n, \cdot, \cdot)$ is bounded by

$$\text{poly}(\mathbb{T}(\Lambda; n), \mathbb{T}(\Delta; n), \mathbb{T}(T; n), \mathbb{T}(Q; n), \mathbb{T}(\mathcal{FE}; n),$$
$$\Lambda(n), \log(\Delta(n)), \log(T(n)), Q(n), D, \mathcal{FE}(n))$$
$$= \text{poly}_{h'}(n, \lambda) \,,$$

while $\mathcal{S}_{\text{AR}}$ and $\mathcal{L}_{\text{AR}}$ running times also depend polynomially on $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$, i.e.,

$$\mathbb{T}(\mathcal{S}_{\text{AR}}; n, \cdot, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{L}_{\text{AR}}; n, \cdot, \cdot, \cdot, \cdot) = \text{poly}_{h'}(n, \lambda, \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)) \,,$$

as is needed for the theorem to hold.

Completeness and Soundness:

Assume

$$|\mathcal{V}| \le 5\, c_{\mathrm{QR}}(h') \cdot \lambda^{c_{\mathrm{QR}}(h')}\,,$$

$$\forall n \in \mathbb{N}:\ \ \mathbb{T}(\mathcal{S}\,;\, n, \cdot, \cdot, \cdot, \cdot, \cdot) \le c_{\mathrm{QR}}(h')(n^{c_{\mathrm{QR}}(h')} + \lambda^{c_{\mathrm{QR}}(h')})$$

$$\forall n \in \mathbb{N}:\ \ \mathbb{T}(\mathcal{A}\,;\, n, \cdot, \cdot)\,,\ \mathbb{T}(\mathcal{L}\,;\, n, \cdot, \cdot, \cdot, \cdot) \le 2^{c_{\mathrm{QR}}(h')(n^{c_{\mathrm{QR}}(h')} + \lambda^{c_{\mathrm{QR}}(h')})}\,.$$

As $c_1 \ge c_{\mathrm{QR}}(h')$, the answer length of $\mathcal{V}_n$ is bounded by $2^{\Lambda(n)}$, which, by Fact 4.48, means $\mathcal{V}_n^{(1)} = \mathfrak{Padding}(\mathcal{V}_n, 2^{\Lambda(n)})$ has the same value as $\mathcal{V}_n$, and in particular has a perfect ZPC strategy if $\mathcal{V}_n$ has one. Now $\mathcal{V}_n^{(2)} = \mathfrak{Pure}(\mathcal{V}_n^{(1)})$, and again this transformation preserves values (Fact 5.41). Let us check that $\mathcal{V}^{(2)}$ satisfies the conditions that ensure

$$(\mathcal{V}_{\mathrm{AR}})_n = \mathfrak{AnsRed}(\mathcal{V}^{(2)}, \Lambda, \Delta, D, T, Q, n, 2\mathcal{FE}(n) + 1).$$

If this is satisfied, then by Proposition 5.80, the completeness is deduced from the completeness of $\mathcal{V}^{(2)}$ which we already settled, and the soundness is due to our choice of t in (210) and the bound from (211). So, we are left to verify the conditions for the completeness and soundness of Proposition 5.80:

– Λ always halts;
– $\mathcal{V}^{(2)}$ is purified and 2^{Λ}-padded;
– Δ always halts; as $c_1 \ge c_{\mathrm{QR}}(h')$, we have $\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \le 2^{\Lambda(n)}$, and thus

$$\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \cdot 2^{\Lambda(n)+1} \le 2^{2\Lambda(n)+1} \le 2^{2\Lambda(n)+2} - 2^{\Lambda(n)+1} = \Delta(n)\,;$$

– Q always halts, and again as $c_1 \ge c_{\mathrm{QR}}(h')$ we have $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \le Q(n)$;
– T is always halting and we chose c_7 exactly for (150) to hold given the appropriate upper bound on $\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)$;
– D a positive integer (in binary), and we chose c_8 so that, given the upper bound on $|\mathcal{V}|$, we will have

$$|\mathcal{V}|, |\Lambda|, |\Delta|, |T|, |Q| \le D\,.$$

6 Parallel repetition

In this section we define a gap amplification transformation on games. This transformation is based on taking repeated products (Definition 3.43) of a game with itself. It is designed so that, if a game has a ZPC strategy with value 1, then this remains the case for the repeated game; while, if the game has value bounded away from 1, then the value of the repeated game goes to zero (exponentially) with the number of repetitions.

The product of two (tailored) games was introduced in Definition 3.43. In Lemma 3.50 we showed that if $\mathfrak{G}_1$ and $\mathfrak{G}_2$ each have a perfect ZPC strategy, then so does $\mathfrak{G}_1 \otimes \mathfrak{G}_2$. One might more generally expect that the value is multiplicative under game product — however, this is *not* the case (see e.g. [MPTW23, Example 4.3]). Fortunately, it is known that under mild assumptions on the structure of the game $\mathfrak{G}$, the value of its k-fold repetition $\mathfrak{G}^{\otimes k}$ does go to zero exponentially fast as k goes to infinity (provided the value of $\mathfrak{G}$ is strictly smaller than 1). Actually, under these mild assumptions, what is known is that if the *non-synchronous* value of $\mathfrak{G}$ is smaller than 1, then the *non-synchronous* value of the repeated game tends to zero exponentially in the number of repetitions (see Section 3.6 for the non-synchronous setup). To be able to translate this fact to our *synchronous* setup (namely, the notions of value and entanglement lower bounds used throughout this paper), we need to use the results of [Vid22] that say, essentially, that if $\mathfrak{G}$ has many self consistency checks, then its synchronous value is not much smaller than its non-synchronous one.

As usual, this transformation needs to be implemented on the level of tailored normal form verifiers. The following is the main theorem proved in this section. Recall the asymptotic notation from Remark 1.2.

Theorem 6.1 (Parallel Repetition. Proved in Section 6.3). *There exists a 2-input Turing machine* ParRep_h, *that takes as input a **typed** h-level TNFV* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *with type set* $\mathcal{T}$, *and a 1-input TM* $\mathcal{K}$ *(which induces, as usual, a partial function* $\mathcal{K} \colon \mathbb{N} \to \mathbb{N}$*), and outputs an* $(h+2)$*-level **untyped** TNFV*

$$\mathsf{ParRep}_h(\mathcal{V}, \mathcal{K}) = \mathcal{V}_{\mathrm{REP}} = (\mathcal{S}_{\mathrm{REP}}, \mathcal{A}_{\mathrm{REP}}, \mathcal{L}_{\mathrm{REP}}, \mathcal{D})$$

with the following properties. There is a positive integer constant

$$c = c_{\mathrm{REP}}(|\mathcal{T}|) \tag{212}$$

depending only on the number of types in $\mathcal{V}$, *such that:*

- *Sampler properties: $\mathcal{S}_{\mathrm{REP}}$ depends only on $\mathcal{K}$ and the original sampler $\mathcal{S}$ (but not on $\mathcal{A}$ or $\mathcal{L}$), and ParRep_h can calculate its description in time $\mathrm{poly}(|\mathcal{K}|, |\mathcal{S}|)$ from them; in particular, $|\mathcal{S}_{\mathrm{REP}}| \leq c \cdot (|\mathcal{K}|^c + |\mathcal{S}|^c)$. In addition, $\mathcal{S}_{\mathrm{REP}}$ runs in time*

$$\mathrm{poly}(n, \mathcal{K}(n), \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot)),$$

  namely

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{S}_{\mathrm{REP}}; n, \cdot, \cdot, \cdot, \cdot) \leq c \cdot (n^c + \mathcal{K}(n)^c + \mathbb{T}(\mathcal{K}; n)^c + \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot)^c),$$

  where c is from (212).

- *Answer length calculator properties: $\mathcal{A}_{\mathrm{REP}}$ depends only on $\mathcal{K}, \mathcal{S}$ and $\mathcal{A}$, and ParRep_h can calculate its description in time $\mathrm{poly}(|\mathcal{K}|, |\mathcal{S}|, |\mathcal{A}|)$; in particular $|\mathcal{A}_{\mathrm{REP}}| \leq c \cdot (|\mathcal{K}|^c + |\mathcal{S}|^c + |\mathcal{A}|^c)$. In addition, $\mathcal{A}_{\mathrm{AR}}$ runs in time*

$$\mathrm{poly}(n, \mathcal{K}(n), \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; n, \cdot, \cdot)),$$

  namely

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{A}_{\mathrm{REP}}; n, \cdot, \cdot) \leq c \cdot (n^c + \mathcal{K}(n)^c + \mathbb{T}(\mathcal{K}; n)^c + \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot)^c + \mathbb{T}(\mathcal{A}; n, \cdot, \cdot)^c).$$

Finally, if for every $\mathrm{x} \in \mathbb{F}_2^{r(n)}$, *where* $r(n) = \mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$, *and* $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, *the output of* $\mathcal{A}(n, \mathrm{x}, \kappa)$ *never decodes (Definition 2.34) to an* $\mathfrak{error}$ *sign, then for every* $\mathrm{y} \in \mathbb{F}_2^{\mathcal{K}(n) \cdot r(n)}$, $\mathcal{A}_{\mathrm{REP}}(n, \mathrm{y}, \kappa)$ *never decodes to an* $\mathfrak{error}$ *sign.*

192

- *Linear constraints process properties: $\mathcal{L}_{\mathrm{REP}}$ depends on all inputs. Also,* ParRep_h *can calculate its description in time* $\overline{\mathrm{poly}(|\mathcal{K}|,|\mathcal{V}|)}$; *in particular,* $|\mathcal{L}_{\mathrm{REP}}| \leq c \cdot (|\mathcal{K}|^c + |\mathcal{V}|^c)$. *In addition,* $\mathcal{L}_{\mathrm{REP}}$ *runs in time*

$$\mathrm{poly}(n, \mathcal{K}(n), \mathbb{T}(\mathcal{K};n), \mathbb{T}(\mathcal{S};n,\cdot,\cdot,\cdot,\cdot,\cdot), \mathbb{T}(\mathcal{A};n,\cdot,\cdot), \mathbb{T}(\mathcal{L};n,\cdot,\cdot,\cdot,\cdot)),$$

  namely

$$\forall n \in \mathbb{N}: \ \mathbb{T}(\mathcal{L}_{\mathrm{REP}};n,\cdot,\cdot,\cdot,\cdot) \leq c \cdot (n^c + \mathcal{K}(n)^c + \mathbb{T}(\mathcal{K};n)^c + \mathbb{T}(\mathcal{S};n,\cdot,\cdot,\cdot,\cdot,\cdot)^c + \mathbb{T}(\mathcal{A};n,\cdot,\cdot)^c + \mathbb{T}(\mathcal{L};n,\cdot,\cdot,\cdot,\cdot)^c).$$

- *Value properties: For $c = c_{\mathrm{REP}}(|\mathcal{T}|)$ as in (212) and $\mathcal{V}_{\mathrm{REP}}$ the output of* ParRep, *we have for all $n \geq 2$:*

  1. ***Completeness**: If $\mathcal{V}_n$ has a value-1 ZPC strategy, then $(\mathcal{V}_{\mathrm{REP}})_n$ has a value-1 ZPC strategy.*

  2. ***Soundness**: For all $\varepsilon > 0$, let*

$$p = p(\varepsilon,n) = \frac{c}{\varepsilon^c} \cdot 2^{-\frac{\varepsilon^c}{c} \cdot \mathcal{K}(n) \cdot \mathbb{T}(\mathcal{A};n,\cdot,\cdot)^{-1}}. \tag{213}$$

     Then, if $(\mathcal{V}_{\mathrm{REP}})_n$ has a strategy with value of at least p, then $\mathcal{V}_n$ has a strategy with value of at least $1 - \varepsilon$.

  3. ***Entanglement bound**: For the same parameters as in the soundness property,*

$$\mathscr{E}((\mathcal{V}_{\mathrm{REP}})_n, p) \geq \mathscr{E}(\mathcal{V}_n, 1 - \varepsilon).$$

The structure of this section is as follows: In Section 6.1 we introduce a simple transformation that can be performed on any game, and is such that the resulting game, which is called *anchored*, will behave well under repetition. In Section 6.2 we introduce the parallel repetition transformation, and analyze its completeness and soundness properties assuming the game was anchored beforehand. Finally, in Section 6.3, we combine the two transformations (together with an in-between detyping) so as to prove the main theorem of this section (Theorem 6.1).

6.1 The anchoring transformation

The anchoring transformation is studied in [BVY17] in the context of nonlocal games with quantum players sharing entanglement, and extended in [JNV+21] to normal form verifiers. Here we consider the anchoring transformation applied to a typed game $\mathfrak{G}$. Informally, anchoring a game consists in adding a single additional type $\mathtt{Anchor}$ such that, whenever a question of type $\mathtt{Anchor}$ is sampled, any answer is accepted.

Definition 6.2 (Combinatorial anchoring). Let $\mathfrak{G}$ be a tailored nonlocal game with an h-level **typed** CL sampling scheme (Definition 4.38), and let $(\mathcal{T}, \mathcal{E})$ be the associated type graph. The anchoring $\mathfrak{Anchor}(\mathfrak{G})$ of $\mathfrak{G}$ is another typed tailored game $\mathfrak{G}_\perp$ with an underlying h-level typed CL sampling scheme; its type set contains an additional element $\mathtt{Anchor}$, i.e. $\mathcal{T}_\perp = \mathcal{T} \sqcup \{\mathtt{Anchor}\}$. The anchored type graph contains the original type graph as the induced subgraph on the vertices $\mathcal{T}$, while attaching $\mathtt{Anchor}$ to every other type; namely

$$\mathcal{E}_\perp = \{tt' \in \mathcal{T}_\perp \times \mathcal{T}_\perp \mid t = \mathtt{Anchor} \text{ or } t' = \mathtt{Anchor} \text{ or } tt' \in \mathcal{E}\}.$$

The CLMs at each type $t \in \mathcal{T}$ are kept as before, and we let $\mathfrak{s}^{\mathtt{Anchor}}$ be the zero function (which is 0-level and thus does not change the level of the sampling scheme). The readable and unreadable answer lengths of the question $(\mathtt{Anchor}, \vec{0})$ are both zero. Finally, the decision function of the anchored game executes $D_{(t,\mathbf{x})(t',\mathbf{y})}$ whenever $t, t' \in \mathcal{T}$, and always accepts whenever t or t' equal $\mathtt{Anchor}$.

Recall that in a game with a typed CL sampling scheme (Definition 4.38) a uniform edge from the type graph is first sampled. Thus, $\mathfrak{G}_\perp$ runs $\mathfrak{G}$ with probability $\frac{|\mathcal{E}|}{|\mathcal{E}|+2|\mathcal{T}|+1}$, and otherwise accepts. This translates to

$$\mathrm{val}^*(\mathfrak{Anchor}(\mathfrak{G})) = \frac{2|\mathcal{T}|+1}{|\mathcal{E}|+2|\mathcal{T}|+1} + \frac{|\mathcal{E}|}{|\mathcal{E}|+2|\mathcal{T}|+1}\mathrm{val}^*(\mathfrak{G}). \tag{214}$$

Actually, **every** strategy $\mathscr{S}$ for $\mathfrak{Anchor}(\mathfrak{G})$ is also a strategy for $\mathfrak{G}$ (as the anchor vertex is of length 0), and the above relation is on that level, namely

$$\operatorname{val}(\mathfrak{Anchor}(\mathfrak{G}),\mathscr{S}) = \frac{2|\mathcal{T}|+1}{|\mathcal{E}|+2|\mathcal{T}|+1} + \frac{|\mathcal{E}|}{|\mathcal{E}|+2|\mathcal{T}|+1}\operatorname{val}(\mathfrak{G},\mathscr{S}) .$$

In particular, we have that $\operatorname{val}^*(\mathfrak{Anchor}(\mathfrak{G})) = 1$ if and only if $\operatorname{val}^*(\mathfrak{G}) = 1$. Moreover, if the latter is achieved with a ZPC strategy then so is the former; as is easily verified by using exactly the same strategy.

Claim 6.3 (Algorithmic anchoring). *There is a polynomial time TM* Anchor *that takes a typed h-level tailored normal form verifier* $\mathcal{V} = (\mathcal{S},\mathcal{A},\mathcal{L},\mathcal{D})$ *as input, and outputs an h-level tailored normal form verifier* $\mathsf{Anchor}(\mathcal{V}) = \mathcal{V}' = (\mathcal{S}',\mathcal{A}',\mathcal{L}',\mathcal{D})$, *such that:*

1. *Combinatorial anchoring: If $\mathcal{V}_n$ is well defined, then $\mathcal{V}'_n$ is well defined and $\mathcal{V}'_n = \mathfrak{Anchor}(\mathcal{V}_n)$.*

2. *Complexity: The time complexities of the TMs in $\mathcal{V}'$ satisfy, for every integer in binary $\overline{n} \in \{0,1\}^*$,*

$$\mathbb{T}(\mathcal{S}';\overline{n},\cdot,\cdot,\cdot,\cdot) = O(\mathbb{T}(\mathcal{S};\overline{n},\cdot,\cdot,\cdot,\cdot)) ,$$
$$\mathbb{T}(\mathcal{A}';\overline{n},\cdot,\cdot) = O(\mathbb{T}(\mathcal{A};\overline{n},\cdot,\cdot)) ,$$
$$\mathbb{T}(\mathcal{L}';\overline{n},\cdot,\cdot,\cdot,\cdot) = O(\mathbb{T}(\mathcal{L};\overline{n},\cdot,\cdot,\cdot,\cdot)) .$$

In addition, the description of $\mathcal{S}'$, $\mathcal{A}'$ and $\mathcal{L}'$ can be computed in time which is linear in the descriptions of $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{L}$. Moreover, the description of $\mathcal{S}'$ only depends on $\mathcal{S}$.

Proof. Let $\mathcal{T}$ be the type set of $\mathcal{V}$. Define the type set $\mathcal{T}_\perp = \mathcal{T} \sqcup \{\texttt{Anchor}\}$. If the type graph of $\mathcal{V}$ is $(\mathcal{T},\mathcal{E})$ then the type graph of $\mathcal{V}'$ is $(\mathcal{T}_\perp,\mathcal{E}_\perp)$ where according to Definition 6.2 we set

$$\mathcal{E}_\perp = \left\{ tt' \in \mathcal{T}_\perp \times \mathcal{T}_\perp \mid t = \text{ANCH or } t' = \text{ANCH or } tt' \in \mathcal{E} \right\} . \tag{215}$$

The Turing machine $\mathcal{S}'$ is defined as follows:

1. On input $(\cdot,\text{Graph},\cdot,\cdot,\cdot,\cdot)$ it executes $\mathcal{S}(\cdot,\text{Graph},\cdot,\cdot,\cdot,\cdot)$ and modifies the output appropriately to match (215) (adding a type $\texttt{Anchor}$ and a corresponding row and column of 1's to the adjacency matrix).

2. On input $(n,\text{Dimension},\cdot,\cdot,\cdot,\cdot)$, it returns $\mathcal{S}(n,\text{Dimension},\cdot,\cdot,\cdot,\cdot)$.

3. On all other inputs
$$(n,\text{Action},\text{Type},j,\mathsf{x},z) ,$$
if Type $\neq \texttt{Anchor}$, then it operates the same as $\mathcal{S}$. Otherwise, it encodes the zero map.

Define the Turing machine $\mathcal{L}'$ that, on input $(n,(t,\mathsf{x}),(t',\mathsf{y}),a^{\mathfrak{R}},b^{\mathfrak{R}})$, if either t or t' is equal to $\texttt{Anchor}$, then it accepts automatically (returns no constraints). Otherwise, it returns the output of $\mathcal{L}(n,(t,\mathsf{x}),(t',\mathsf{y}),a^{\mathfrak{R}},b^{\mathfrak{R}})$. Finally, define the Turing machine $\mathcal{A}'$ that returns the same output as $\mathcal{A}$ when the type is not $\texttt{Anchor}$, and returns 0 when the type is $\texttt{Anchor}$.

All the above TMs clearly run in time which is linear in their original counterparts, and their description is constant up to appending $\mathcal{V}$. Finally, $\mathcal{S}'$ indeed uses only $\mathcal{S}$ in its definition. □

6.2 The parallel repetition transformation

Let $\mathfrak{G}$ be a (untyped) tailored game with underlying graph $G = (V, E)$ and question distribution μ over edges. Recall that in Section 3.5 we define the tensor product of two games (Definition 3.43). Iterating the definition, we obtain the following k-fold tensored game $\mathfrak{G}^{\otimes k}$, which we make explicit for convenience:

1. The underlying graph is the k-fold tensor product of the graph G with itself, denoted $G^{\otimes k}$, whose vertex set is V^k. The vertex $(x_1, ..., x_k)$ is a neighbor of $(y_1, ..., y_k)$ in $G^{\otimes k}$ if and only if x_i is a neighbor of y_i for every $i \in [k]$ in the original graph G.

2. The question distribution is μ^k; namely, k edges $x_i y_i$ from G are sampled independently according to μ, and are combined to a single edge $(x_1, ..., x_k)(y_1, ..., y_k)$ in $G^{\otimes k}$.

3. The length functions satisfy $\ell^{\otimes k, \mathfrak{R}}(x) = \sum_i \ell^{\mathfrak{R}}(x_i)$ and $\ell^{\otimes k, \mathfrak{L}}(x) = \sum_i \ell^{\mathfrak{L}}(x_i)$, and for each $x = (x_1, \ldots, x_k) \in V^k$ the sets of formal generators are $S_x^{\otimes k, \mathfrak{R}} = S_{x_1}^{\mathfrak{R}} \sqcup \cdots \sqcup S_{x_k}^{\mathfrak{R}}$ and $S_x^{\otimes k, \mathfrak{L}} = S_{x_1}^{\mathfrak{L}} \sqcup \cdots \sqcup S_{x_k}^{\mathfrak{L}}$; namely, the answer at $(x_1, ..., x_k)$ is a k-tuple of answers to the respective x_i's. Note that there is a copy of S_{x_i} in S_x, and thus linear constraints over $S_{x_i} \cup S_{y_i}$ can be naturally interpreted as linear constraints over $S_x \cup S_y$.

4. The function $L_{xy}^{\otimes k}$ is obtained by returning the "union" of all $L_{x_i y_i}$. I.e., for $x = (x_1, \ldots, x_k)$ and $y = (y_1, \ldots, y_k)$, an answer to x is formatted as $a^{\mathfrak{R}} = (a_1^{\mathfrak{R}}, ..., a_k^{\mathfrak{R}})$, $a^{\mathfrak{L}} = (a_1^{\mathfrak{L}}, ..., a_k^{\mathfrak{L}})$ and to y as $b^{\mathfrak{R}} = (b_1^{\mathfrak{R}}, ..., b_k^{\mathfrak{R}})$, $b^{\mathfrak{L}} = (b_1^{\mathfrak{L}}, ..., b_k^{\mathfrak{L}})$. So, $L_{x_i y_i}(a_i^{\mathfrak{R}}, b_i^{\mathfrak{R}})$ returns a sequence of linear constraints on variables in $S_{x_i} \cup S_{y_i}$ which we can interpret as linear constraints over $S_x \cup S_y$. Combining all of this we let

$$L_{xy}^{\otimes k}(a^{\mathfrak{R}}, b^{\mathfrak{R}}) = \bigsqcup_{i=1}^{k} L_{x_i y_i}(a_i^{\mathfrak{R}}, b_i^{\mathfrak{R}}) .$$

Note that on the level of the decision predicate, this gives the following relation:

$$D_{xy}^{\otimes k}(a^{\mathfrak{R}}, a^{\mathfrak{L}}, b^{\mathfrak{R}}, b^{\mathfrak{L}}) = \prod_{i=1}^{k} D_{x_i y_i}(a_i^{\mathfrak{R}}, a_i^{\mathfrak{L}}, b_i^{\mathfrak{R}}, b_i^{\mathfrak{L}}) ,$$

which is the standard view of parallel repetition.

The following theorem states the effect that the k-fold tensoring has on the value of a game, at least when it is applied to the detyping of a typed game that is anchored according to Definition 6.2.

Theorem 6.4 (Completeness and soundness of the parallel repeated game)**.** *Let $\mathfrak{G}$ be a tailored nonlocal game with underlying typed h-level CL sampling scheme. Assume that the type graph of $\mathfrak{G}$ has self-loops at each vertex, and let Λ be an upper bound on the answer length in $\mathfrak{G}$. Let $\mathfrak{G}' = \mathfrak{Anchor}(\mathfrak{G})$ (Definition 6.2) and $\widetilde{\mathfrak{G}} = \mathfrak{DeType}(\mathfrak{G}')$ (Definition 4.40). Then, there is a positive integer constant c_0 depending only on $|\mathcal{T}|$, such that for any $k \geq 1$, the game $\widetilde{\mathfrak{G}}^{\otimes k}$ has the following properties.*

(1) Perfect ZPC Completeness*: If $\mathfrak{G}$ has a perfect ZPC strategy, then so does $\widetilde{\mathfrak{G}}^{\otimes k}$.*

(2) Soundness*: For any $\varepsilon > 0$, let*

$$p = p(\varepsilon) = \frac{c_0}{\varepsilon^{c_0}} \cdot 2^{-\frac{\varepsilon^{c_0}}{c_0 \cdot \Lambda} \cdot k} . \tag{216}$$

If $\widetilde{\mathfrak{G}}^{\otimes k}$ has a strategy with value at least p, then $\mathfrak{G}$ has a strategy of the same dimension with value at least $1 - \varepsilon$.

(3) Entanglement*: For all $\varepsilon > 0$ and p as in (216),*

$$\mathscr{E}(\widetilde{\mathfrak{G}}^{\otimes k}, p) \geq \mathscr{E}(\mathfrak{G}, 1 - \varepsilon) .$$

Proof. We first verify perfect completeness. Assume $\mathfrak{G}$ has a perfect ZPC strategy. By the analysis immediately after Definition 6.2, the anchored game $\mathfrak{G}' = \mathfrak{Anchor}(\mathfrak{G})$ has a perfect ZPC strategy as well. Hence, by Corollary 4.42, the detyped game $\widetilde{\mathfrak{G}} = \mathfrak{DeType}(\mathfrak{G}')$ has a perfect ZPC strategy. Finally, by Lemma 3.50, taking the sum (Definition 3.47) of the perfect ZPC strategy for $\widetilde{\mathfrak{G}}$ with itself k times produces a perfect ZPC strategy for $\widetilde{\mathfrak{G}}^{\otimes k}$. This finishes the perfect completeness argument.

Let us prove the soundness and entanglement lower bound. Recall the notions and notations of Section 3.6 on the non-synchronous setup. Let $\delta > 0$. If for the synchronous quantum value (Definition 2.21) we have $\mathrm{val}^*(\widetilde{\mathfrak{G}}^{\otimes k}) \geq \delta$, then the same can be deduced for the non-synchronous value (Definition 3.61): $\mathrm{val}^{\mathrm{non-sync}}(\widetilde{\mathfrak{G}}^{\otimes k}) \geq \delta$, as every synchronous strategy is also a general strategy (Remark 3.62) — note that the same holds for the entanglement lower bound. We now need to apply a deep theorem from the study of general quantum strategies.

Theorem 6.5 (Non-synchronous soundness of parallel repetition of anchored games, Theorem 6.1 of [BVY17]). *There exists a universal positive integer constant c' such that for all two-player anchored games $\mathfrak{G}$ and for all positive integers k, for all $0 < \varepsilon \leq 1$, and for δ satisfying*

$$\delta = \frac{4}{\varepsilon} \cdot 2^{-\frac{\varepsilon^{17}}{c' \cdot \Lambda} \cdot k} \,, \tag{217}$$

we have that every strategy for $\mathfrak{G}^{\otimes k}$ of dimension n with value at least δ induces[104] a strategy of the same dimension n for $\mathfrak{G}$ with value $1 - \varepsilon$. Namely,

$$\mathscr{E}^{\mathrm{non-sync}}(\mathfrak{G}^{\otimes k}, \delta) \geq \mathscr{E}^{\mathrm{non-sync}}(\mathfrak{G}, 1 - \varepsilon) \,.$$

Remark 6.6. Here we do not repeat the precise definition of a game being "anchored," referring to [BVY17] for it. Suffice it to say that any game produced by the transformation $\mathfrak{DeType} \circ \mathfrak{Anchor}(\cdot)$ is anchored.

As we fixed δ in the beginning of the proof, let ε_δ be the value that satisfies equation (217) with respect to it. Then, given a general strategy $\mathscr{S}$ for $\widetilde{\mathfrak{G}}^{\otimes k}$ with value δ, there is a general strategy $\mathscr{S}'$ of the same dimension for $\widetilde{\mathfrak{G}}$ with value of at least $1 - \varepsilon_\delta$. As, combinatorially, detyping (Definition 4.40) is just playing with some constant probability (that depends on the number of types $|\mathcal{T}|$) the double cover of the original game, and the double cover of the game has the same non-synchronous value as the game itself (Remark 3.60), we can deduce that $\mathfrak{G}'$ has a general strategy with value of at least $1 - C\varepsilon_\delta$ and the same dimension (C here depends only on $|\mathcal{T}|$). Furthermore, as anchoring is again just playing the original game with some positive probability (and accepting otherwise), the same general strategy has value of at least $1 - C'C\varepsilon_\delta$ against $\mathfrak{G}$, where again C' depends only on $|\mathcal{T}|$. Now, we assumed that the type graph of $\mathfrak{G}$ contains all self loops, and thus for every vertex x we have

$$\frac{\mu(\mathrm{xx})}{\mu(\mathrm{x})} \geq \frac{1}{2|\mathcal{T}|} \,.$$

Therefore, we can apply Fact 3.63, and retrieve a synchronous strategy for $\mathfrak{G}$ of the same dimension with value of at least $1 - C'' \cdot (4|\mathcal{T}|^2 C'C)^{C''} \cdot \varepsilon_\delta^{1/C''}$, where this time C'' is a universal positive integer constant.

Tracing back these equations, if we want to understand what is the p for which $\mathrm{val}^*(\widetilde{\mathfrak{G}}^{\otimes k}) \geq p$ implies $\mathrm{val}^*(\mathfrak{G}) \geq 1 - \varepsilon$, we first need to choose ε_δ so that

$$\varepsilon \geq C'' \cdot (4|\mathcal{T}|^2 C'C)^{C''} \cdot \varepsilon_\delta^{1/C''} \,. \tag{218}$$

In particular, by letting $c = (C''(4|\mathcal{T}|^2 C'C)^{C''})^{C''}$ and $\varepsilon_\delta = \frac{\varepsilon^c}{c}$, (218) is satisfied. We then plug this into equation (217) and deduce that for

$$p = \frac{4c}{\varepsilon^c} \cdot 2^{-\frac{\varepsilon^{17c}}{c^{17} \cdot c' \cdot \Lambda} \cdot k}$$

the conclusion is satisfied. As $c \geq 4$ and $c' \geq 1$, $c^{17} \cdot c' \geq 17c \geq 4c \geq c$, and thus, if we take $c_0 = c^{17} \cdot c'$, the number

$$\frac{c_0}{\varepsilon^{c_0}} \cdot 2^{-\frac{\varepsilon^{c_0}}{c_0 \cdot \Lambda} \cdot k}$$

---

[104]Finding this induced strategy is not immediate at all, and is one of the main issues in the proof of this theorem.

is always larger than the aforementioned p, and we can conclude. Note that indeed c_0 depends only on $|\mathcal{T}|$ and nothing else. $\qquad \square$

The following claim shows that, given a function $\mathcal{K}\colon \mathbb{N} \to \mathbb{N}$ and a normal form verifier $\mathcal{V}$, one can efficiently calculate a normal form verifier $\mathcal{V}'$ whose n^{th} game is the $\mathcal{K}(n)$-fold tensor power of the nth game of $\mathcal{V}$, namely $\mathcal{V}'_n = \mathcal{V}_n^{\otimes \mathcal{K}(n)}$.

Claim 6.7 (Algorithmic parallel repetition). *There is a 2-input TM PartialParRep that takes as input an h-level untyped tailored normal form verifier $\mathcal{V}$ and a 1-input always halting TM $\mathcal{K}$ (in particular, it induces a map $\mathcal{K}\colon \mathbb{N} \to \mathbb{N}$), and outputs a new h-level normal form verifier*

$$\mathsf{PartialParRep}(\mathcal{V}, \mathcal{K}) = \mathcal{V}' = (\mathcal{S}', \mathcal{A}', \mathcal{L}', \mathcal{D})$$

with the following properties.

1. *Combinatorial parallel repetition: If $\mathcal{V}_n$ is well defined, then $\mathcal{V}'_n$ is well defined and $\mathcal{V}'_n = \mathcal{V}_n^{\otimes \mathcal{K}(n)}$.*

2. *Complexity: The time bounds are*

$$\mathbb{T}(\mathcal{S}'; \overline{n}, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(\mathcal{K}(n), \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot)) \,,$$
$$\mathbb{T}(\mathcal{A}'; \overline{n}, \cdot, \cdot) = \mathrm{poly}(\mathcal{K}(n), \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot)) \,, \qquad (219)$$
$$\mathbb{T}(\mathcal{L}'; \overline{n}, \cdot, \cdot, \cdot, \cdot) = \mathrm{poly}(\mathcal{K}(n), \mathbb{T}(\mathcal{K}; n), \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot), \mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot, \cdot)) \,.$$

In addition, the description lengths are linear in that of $\mathcal{V}$ and $\mathcal{K}$. Finally, the sampler $\mathcal{S}'$ depends only on $\mathcal{K}$ and $\mathcal{S}$, while the answer length calculator $\mathcal{A}'$ depends only on $\mathcal{K}, \mathcal{S}$ and $\mathcal{A}$.

Proof. Let us define the TMs one by one.

The Sampler: First, $\mathcal{S}'$ calculates $k = \mathcal{K}(n)$. If the input is $(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$, $\mathcal{S}'$ calculates $\mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot) = r$ and outputs $r \cdot k$. Otherwise, given an input of the form

$$(\overline{n}, \mathrm{Action}, \mathrm{Player}, j, \mathrm{x}, z) \,,$$

as x and z are length $r \cdot k$ bit-strings, they can be interpreted as $\mathrm{x} = (\mathrm{x}_1, ..., \mathrm{x}_k)$ and $z = (z_1, ..., z_k)$. So, $\mathcal{S}'$ does k calls to $\mathcal{S}$, by evaluating it on $(\overline{n}, \mathrm{Action}, \mathrm{Player}, j, \mathrm{x}_i, z_i)$ for every $i \in [k]$. It then concatenates the result so as to produce the appropriate output in $(\mathbb{F}_2^r)^k$.

This TM has a constant description up to appending $\mathcal{K}$ and $\mathcal{S}$. It runs in time linear in $\mathcal{K}(n)$ times $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$, as required, and depends only on $\mathcal{S}$ and $\mathcal{K}$.

The Answer length calculator: Given (n, x, κ), $\mathcal{A}'$ calls $\mathcal{K}(n)$ and $\mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ to retrieve k and r respectively. It then checks that x is indeed of length $k \cdot r$, and outputs error otherwise. Finally, if x is indeed of the appropriate length, it denotes it $\mathrm{x} = (\mathrm{x}_1, ..., \mathrm{x}_k) \in (\mathbb{F}_2^r)^k$, calculates $\mathcal{A}(n, \mathrm{x}_i, \kappa)$ for all $i \in [k]$, and concatenates the outputs to a single bit string.[105]

This TM has a constant description length up to appending $\mathcal{K}$ and $\mathcal{S}$ and $\mathcal{A}$. It runs in time linear in $\mathcal{K}(n)$, $\mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot)$ and $\mathbb{T}(\mathcal{A}; n, \cdot, \cdot)$, as required, and depends only on $\mathcal{S}, \mathcal{A}$ and $\mathcal{K}$.

The Linear constraints processor: The TM $\mathcal{L}'$ first calls $\mathcal{K}(n)$ and $\mathcal{S}(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$ to retrieve k and r respectively. Now, given input $(n, \mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}})$, it parses $\mathrm{x}, \mathrm{y}, a^{\mathfrak{R}}, b^{\mathfrak{R}}$ as k-tuples of questions and readable answers, respectively. It then calls $\mathcal{A}(n, \mathrm{x}_i, \kappa), \mathcal{A}(n, \mathrm{y}_i, \kappa)$ to retrieve $\ell^\kappa(\mathrm{x}_i)$ and $\ell^\kappa(\mathrm{y}_i)$ for every $i \in [k]$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$. Then, it calls $\mathcal{L}(n, \mathrm{x}_i, \mathrm{y}_i, a_i^{\mathfrak{R}}, b_i^{\mathfrak{R}})$, for all $i \in [k]$. The output bit-strings represent constraints on some subset of the variables used in the parallel repeated game, i.e., the output of $\mathcal{L}(n, \mathrm{x}_i, \mathrm{y}_i, a_i^{\mathfrak{R}}, b_i^{\mathfrak{R}})$ is (the encoding) of a sequence $(c_1^i, ..., c_m^i)$, each of which is of length $\ell^{\mathfrak{R}}(\mathrm{x}_i) + \ell^{\mathfrak{L}}(\mathrm{x}_i) + \ell^{\mathfrak{R}}(\mathrm{y}_i) + \ell^{\mathfrak{L}}(\mathrm{y}_i) + 1$,[106] and we want to interpret it as a sequence of m constraints of length

---

[105] As $\mathcal{A}'$ needs to output the (encoded) answer length in unary, this concatenation is exactly taking the sum of the appropriate lengths.

[106] As usual, if the outputs are not well formatted, $\mathcal{L}'$ should halt and output error.

$1 + \sum_{j=1}^{k} \ell^{\mathfrak{R}}(x_j) + \ell^{\mathfrak{L}}(x_j) + \ell^{\mathfrak{R}}(y_j) + \ell^{\mathfrak{L}}(y_j)$. To do that, we need to pad them with zeros appropriately so that c_j^i would be the same constraint as before, but on the variables at the ith position. After doing it to the output of each call to $\mathcal{L}$, $\mathcal{L}'$ collects them all to a long list of constraints and outputs (the encoding) of it.

This TM has a constant description up to appending $\mathcal{K}, \mathcal{S}, \mathcal{A}$ and $\mathcal{L}$. It runs in time linear in

$$\mathcal{K}(n) \, , \, \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \, , \, \mathbb{T}(\mathcal{A}; n, \cdot, \cdot)$$

and $\mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot)$ as required. $\qquad\qquad\square$

6.3 Proving the main theorem of Parallel Repetition: Theorem 6.1

Theorem 6.8 (Anchored parallel repetition of tailored normal form verifiers)**.** *There exists a function*

$$c_{\text{REP}} \colon \mathbb{N} \to \mathbb{N} \tag{220}$$

and a 2-input polynomial-time Turing machine ParRep_h*, that takes as input a typed h-level tailored normal form verifier* $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ *and a 1-input TM* $\mathcal{K}$ *(which induces, as usual, a partial function* $\mathcal{K} \colon \mathbb{N} \to \mathbb{N}$*), and outputs an* $(h+2)$*-level **untyped** tailored normal form verifier* $\mathsf{ParRep}_h(\mathcal{V}, \mathcal{K}) = \mathcal{V}_{\text{REP}} = (\mathcal{S}_{\text{REP}}, \mathcal{A}_{\text{REP}}, \mathcal{L}_{\text{REP}}, \mathcal{D})$ *such that the following hold.*

1. *Combinatorial parallel repetition: If* $\mathcal{V}_n$ *is well defined, then* $(\mathcal{V}_{\text{REP}})_n$ *is well defined, and*

$$(\mathcal{V}_{\text{REP}})_n = (\mathfrak{DeType}(\mathfrak{Anchor}(\mathcal{V}_n)))^{\otimes \mathcal{K}(n)} \, ,$$

   where $\mathfrak{Anchor}$ *is the anchoring transformation (Definition 6.2),* $\mathfrak{DeType}$ *is the detyping transformation (Definition 4.40), and* $(\cdot)^{\otimes k}$ *is the k-fold tensor power of a game (Definition 3.43 and the beginning of Section 6.2). In particular, for* $c = c_{\text{REP}}(|\mathcal{T}|)$*, where* $|\mathcal{T}|$ *is the number of types in the type graph of* $\mathcal{V}$ *and* c_{REP} *is from (212), we have —*

   - *Completeness: If* $\mathcal{V}_n$ *has a value-1 ZPC strategy, then* $(\mathcal{V}_{\text{REP}})_n$ *has a value-1 ZPC strategy.*
   - *Soundness: For all* $\varepsilon > 0$*, letting* $\Lambda(n) = \mathbb{T}(\mathcal{A}; n, \cdot, \cdot)$*, if*

$$p \geq \frac{c}{\varepsilon^c} \cdot 2^{-\frac{\varepsilon^c}{c\Lambda(n)} \cdot \mathcal{K}(n)} \, ,$$

   then $\mathscr{E}((\mathcal{V}_{\text{REP}})_n, p) \geq \mathscr{E}(\mathcal{V}_n, 1 - \varepsilon)$*.*

2. *Complexity: Letting* $c = c_{\text{REP}}(|\mathcal{T}|)$ *as above, the time complexities of the output verifier* $(\mathcal{V}_{\text{REP}})_n$ *are*

$$\mathbb{T}(\mathcal{S}_{\text{REP}}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \leq c \big(\mathcal{K}(n) \cdot \mathbb{T}(\mathcal{K}; n) \cdot \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \big)^c \, ,$$
$$\mathbb{T}(\mathcal{A}_{\text{REP}}; \overline{n}, \cdot, \cdot) \leq c \big(\mathcal{K}(n) \cdot \mathbb{T}(\mathcal{K}; n) \cdot \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \cdot \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot) \big)^c \, ,$$
$$\mathbb{T}(\mathcal{L}_{\text{REP}}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \leq c \big(\mathcal{K}(n) \cdot \mathbb{T}(\mathcal{K}; n) \cdot \mathbb{T}(\mathcal{S}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \cdot \mathbb{T}(\mathcal{A}; \overline{n}, \cdot, \cdot) \cdot \mathbb{T}(\mathcal{L}; \overline{n}, \cdot, \cdot, \cdot, \cdot) \big)^c \, .$$

   The repeated sampler $\mathcal{S}_{\text{REP}}$ *only depends on* $\mathcal{S}$ *and* $\mathcal{K}$*, the answer length calculator* $\mathcal{A}_{\text{REP}}$ *only depends on* $\mathcal{S}, \mathcal{A}$ *and* $\mathcal{K}$*. Finally, the description lengths of all output TMs are linear in the TMs they depend on.*

Proof. Letting $\mathcal{V}_{\text{REP}} = \mathsf{PartialParRep}(\mathsf{DeTyping}(\mathsf{Anchor}(\mathcal{V})), \mathcal{K})$, and using Claims 6.7, 4.46 and 6.3, we deduce that indeed

$$(\mathcal{V}_{\text{REP}})_n = (\mathfrak{DeType}(\mathfrak{Anchor}(\mathcal{V}_n)))^{\otimes \mathcal{K}(n)} \, ,$$

whenever $\mathcal{V}_n$ is well defined. Following all these claims as well as Theorem 6.4, and taking $c_{\text{REP}}(|\mathcal{T}|)$ large enough to bound all the constants appearing in them (in particular, bound c_0 from Theorem 6.4), we deduce the soundness clause, as well as the time complexities. Completeness is deduced by the same theorem and claims. $\qquad\square$

7 Proving the compression theorem

Let us recall the version of the compression theorem, first phrased in Theorem 4.34, that we ought to prove. Recall also the asymptotic notation from Remark 1.2.

Theorem 7.1 (Compression of tailored h-level normal form verifiers). *For every positive integer h, there exist two positive integers*

$$c = c(h) \quad \text{and} \quad C = C(h)$$

*that depend only on h, and a 2-input Turing machine $\mathsf{Compress}_h$, that takes as input a h-level TNFV $\mathcal{V} = (\mathcal{S}, \mathcal{A}, \mathcal{L}, \mathcal{D})$ and a positive integer λ (in binary), and outputs a 5-**level** TNFV $\mathsf{Compress}_h(\mathcal{V}, \lambda) = \mathcal{V}' = (\mathcal{S}^\lambda, \mathcal{A}^\lambda, \mathcal{L}', \mathcal{D})$, such that:*

- *Sampler properties: The 5-level CL sampler $\mathcal{S}^\lambda$ depends only on λ and h, (but not the specific $\mathcal{V}$), and $\mathsf{Compress}_h$ can calculate its description in time $\mathrm{polylog}_h(\lambda)$;[107] in particular, $|S^\lambda| \leq c \log^c \lambda$. In addition, $\mathcal{S}^\lambda$ runs in $\mathrm{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{S}^\lambda; n) \leq c \cdot (n^c + \lambda^c) .$$

- *Answer length calculator properties: $\mathcal{A}^\lambda$ depends only on λ and h, and $\mathsf{Compress}_h$ can calculate its description in time $\mathrm{polylog}_h(\lambda)$; in particular $|\mathcal{A}^\lambda| \leq c \log^c \lambda$. In addition, $\mathcal{A}^\lambda$ runs in $\mathrm{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{A}^\lambda; n, \cdot, \cdot) \leq c \cdot (n^c + \lambda^c) .$$

  Finally, given that $\mathrm{x} \in \mathbb{F}_2^{r(n)}$, where $r(n) = \mathcal{S}^\lambda(n, \mathrm{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and that $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the output of $\mathcal{A}^\lambda(n, \mathrm{x}, \kappa)$ never decodes (Definition 2.34) to an $\mathfrak{error}$ sign.

- *Linear constraints processor properties: $\mathcal{L}'$ depends on both λ and $\mathcal{V}$, and $\mathsf{Compress}_h$ can calculate its description in time $\mathrm{poly}_h(\log \lambda, |\mathcal{V}|)$; in particular, $|\mathcal{L}'| \leq c \cdot (\log^c \lambda + |\mathcal{V}|^c)$. In addition, $\mathcal{L}'$ runs in $\mathrm{poly}_h(n, \lambda)$-time, namely*

$$\forall n \in \mathbb{N} : \quad \mathbb{T}(\mathcal{L}'; n, \cdot, \cdot, \cdot, \cdot) \leq c \cdot (n^c + \lambda^c) .$$

- *Decider properties: The canonical decider $\mathcal{D}$ (Definition 2.45) is fixed and runs in time which is linear in its input length.*

- *Value properties: If $\mathcal{V}$ is λ-bounded, then $\mathcal{V}'$, the output of $\mathsf{Compress}_h$, satisfies: For all $n \geq C$,*

  1. ***Completeness**: If $\mathcal{V}_{2^n}$ has a perfect Z-aligned permutation strategy that commutes along edges (ZPC strategy), then so does $\mathcal{V}'_n$.*

  2. ***Soundness**: $\mathscr{E}(\mathcal{V}'_n, \frac{1}{2}) \geq \max \left\{ \mathscr{E}(\mathcal{V}_{2^n}, \frac{1}{2}), 2^{2^{\lambda n} - 1} \right\}.$*

Proof. Though our parameters are different, we use the same proof structure as [JNV$^+$21, Theorem 12.1]. To that end, let $\mathcal{K} = \mathcal{K}_{\lambda, h}$ be a TM that takes an integer n in binary as input, and outputs

$$\mathcal{K}(n) = c_0 (n\lambda)^{c_0} \tag{221}$$

in binary, where c_0 is a integer parameter that will be fixed later, and which depends only on h. Note that $\mathbb{T}(\mathcal{K}; n) = \mathrm{polylog}_h(n, \lambda)$ and $|\mathcal{K}| = \mathrm{polylog}_h(\lambda)$, which means there is a positive integer constant $C_0 = C_0(h)$ such that $|\mathcal{K}| \leq C_0 \log^{C_0} \lambda$ and $\mathbb{T}(\mathcal{K}; n) \leq C_0 (\log^{C_0} n + \log^{C_0} \lambda)$. Let us collect other constants that will be used along the proof: $c_1 = c_{\mathrm{QR}}(h)$ defined in (121) from Theorem 4.36; $c_2 = c_{\mathrm{AR}}(3, h)$ defined in (122) from Theorem 5.1; $c_3 = c_{\mathrm{REP}}(9)$ defined in (212) from Theorem 6.1.

Now, define three tailored normal form verifiers (TNFVs):

---

[107]Recall our asymptotic notation from Remark 1.2 to parse $\mathrm{polylog}_h$.

1. Let $\mathsf{QuestionReduction}_h(\mathcal{V},\lambda) = \mathcal{V}^{(1)} = (\mathcal{S}^{(1)}, \mathcal{A}^{(1)}, \mathcal{L}^{(1)}, \mathcal{D})$.

2. Let $\mathsf{AnswerReduction}_{3,h}(\mathcal{V}^{(1)},\lambda) = \mathcal{V}^{(2)} = (\mathcal{S}^{(2)}, \mathcal{A}^{(2)}, \mathcal{L}^{(2)}, \mathcal{D})$.

3. Let $\mathsf{ParRep}_3(\mathcal{V}^{(2)},\mathcal{K}) = \mathcal{V}^{(3)} = (\mathcal{S}^{(3)}, \mathcal{A}^{(3)}, \mathcal{L}^{(3)}, \mathcal{D})$.

4. Finally, choose $\mathcal{S}^\lambda := \mathcal{S}^{(3)}$, $\mathcal{A}^\lambda := \mathcal{A}^{(3)}$ and $\mathcal{L}' := \mathcal{L}^{(3)}$.

Level:
By Theorem 4.36, the output of $\mathsf{QuestionReduction}_h$ is always a 3-level TNFV, and thus $\mathcal{V}^{(1)}$ is such. By Theorem 5.1, given that its input was 3-level, the output of $\mathsf{AnswerReduction}_{3,h}$ is always a $\max(3,3) = 3$-level typed TNFV, and thus $\mathcal{V}^{(2)}$ is such. By Theorem 6.1, ParRep_3 outputs a 5-level TNFV given that the input was a typed 3-level TNFV, and thus $\mathcal{V}^{(3)}$ is a 5-level TNFV, as needed.

Sampler properties:
By Theorem 4.36, the sampler $\mathcal{S}^{(1)}$ depends only on λ and h, can be calculated in time $\mathrm{polylog}_h(\lambda)$, runs in time $c_1(n^{c_1} + \lambda^{c_1})$, and has description length bounded by $c_1 \log^{c_1} \lambda$. By Theorem 5.1, $\mathcal{S}^{(2)}$ depends on $\mathcal{S}^{(1)}, \lambda, 3$ and h, can be calculated in time

$$\mathrm{poly}_{3,h}(\log \lambda, \underbrace{|\mathcal{S}^{(1)}|}_{c_1 \log^{c_1} \lambda}) = \mathrm{polylog}_h(\lambda) \, ,$$

has description length bounded by

$$|S^{(2)}| \le c_2(\log^{c_2} \lambda + |\mathcal{S}^{(1)}|^{c_2}) \le 2c_1^{c_2} \cdot c_2 \log^{c_1 c_2} \lambda \, ,$$

and runs in time at most

$$\mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \le c_2(n^{c_2} + \lambda^{c_2} + (\underbrace{\mathbb{T}(\mathcal{S}^{(1)}; n, \cdots, \cdot, \cdot)}_{c_1(n^{c_1} + \lambda^{c_1})})^{c_2}) \le 2c_1^{c_2} \cdot c_2 (n + \lambda)^{c_1 c_2} \, .$$

By Theorem 6.1,

$$|\mathcal{S}^{(3)}| \le c_3(|\mathcal{K}|^{c_3} + |\mathcal{S}^{(2)}|^{c_3}) \le c_3((C_0 \log^{C_0} \lambda)^{c_3} + (c_2 \log^{c_1 c_2} \lambda)^{c_3}) = \mathrm{polylog}_h(\lambda) \, ,$$

and

$$
\begin{aligned}
\mathbb{T}(\mathcal{S}^{(3)}; n, \cdot, \cdot, \cdot, \cdot, \cdot) &\le c_3(n^{c_3} + \mathcal{K}(n)^{c_3} + \mathbb{T}(\mathcal{K}; n)^{c_3} + \mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot, \cdot)^{c_3}) \\
&\le c_3(n^{c_3} + (c_0 n^{c_0} \lambda^{c_0})^{c_3} + (C_0 \log^{C_0} n + C_0 \log^{C_0} \lambda)^{c_3} + (2c_1^{c_2} \cdot c_2 (n + \lambda)^{c_1 c_2})^{c_3}) \\
&= \mathrm{poly}_h(n, \lambda) \, .
\end{aligned}
$$

This proves the sampler properties of Theorem 7.1.

Answer length calculator properties:
By Theorem 5.1, $\mathcal{A}^{(2)}$ depends only on $3, h$ and λ, can be calculated from them in $\mathrm{polylog}_h(\lambda)$ time, and in particular $|\mathcal{A}^{(2)}| \le c_2 \log^{c_2} \lambda$. In addition,

$$\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot) \le c_2(n^{c_2} + \lambda^{c_2}) \, .$$

Also, whenever $\mathrm{x} \in \mathbb{F}_2^{r_2(n)}$ for $r_2(n) = \mathcal{S}^{(2)}(n, \mathsf{Dimension}, \cdot, \cdot, \cdot, \cdot)$, and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the output of $\mathcal{A}^{(2)}(n, \mathrm{x}, \kappa)$ does not decode to error. By Theorem 6.1, $\mathcal{A}^{(3)}$ depends on $\mathcal{K}, \mathcal{S}^{(2)}$ and $\mathcal{A}^{(2)}$, and can be calculated from their description in

polynomial time. As we showed in the sampler properties $|\mathcal{S}^{(2)}| = \text{polylog}_h(\lambda)$, and by the analysis above $|\mathcal{A}^{(2)}|, |\mathcal{K}| = \text{polylog}_h(\lambda)$. All in all, $|\mathcal{A}^{(3)}| = \text{poly}(|\mathcal{S}^{(2)}|, |\mathcal{A}^{(2)}|, |\mathcal{K}|) = \text{polylog}_h(\lambda)$. Furthermore,

$$
\begin{aligned}
\mathbb{T}(\mathcal{A}^{(3)}; n, \cdot, \cdot) &\leq c_3 \cdot (n^{c_3} + \mathcal{K}(n)^{c_3} + \mathbb{T}(\mathcal{K}; n)^{c_3} + \mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot)^{c_3} + \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)^{c_3}) \\
&\leq c_3 \cdot (n^{c_3} + (c_0 n \lambda)^{c_0 c_3} + (C_0 \log^{C_0} n + C_0 \log^{C_0} \lambda)^{c_3} \\
&\quad + (2 c_1^{c_2} \cdot c_2 (n + \lambda)^{c_1 c_2})^{c_3} + (c_2 (n^{c_2} + \lambda^{c_2}))^{c_3}) \\
&= \text{poly}_h(n, \lambda) \,.
\end{aligned}
$$

Now, as $r_3(n) = \mathcal{S}^{(3)}(n, \text{Dimension}, \cdot, \cdot, \cdot, \cdot) = \mathcal{K}(n) \cdot r_2(n)$, and by the answer length calculator properties guaranteed by Theorem 6.1, we can deduce that whenever $\mathrm{x} \in \mathbb{F}_2^{r_3(n)}$ and $\kappa \in \{\mathfrak{R}, \mathfrak{L}\}$, the decoding of the output of $\mathcal{A}^{(3)}(n, \mathrm{x}, \kappa)$ is not error. This finishes the proof of the answer length calculator properties of Theorem 7.1.

Linear constraints processor properties:

By Theorem 4.36, $\mathcal{L}^{(1)}$ depends on λ, h and $\mathcal{V}$, and can be calculated in $\text{poly}_h(\log \lambda, |\mathcal{V}|)$ time. In particular, $|\mathcal{L}^{(1)}| \leq c_1(\log^{c_1} \lambda + |\mathcal{V}|^{c_1})$. Moreover, $\mathbb{T}(\mathcal{L}^{(1)}; n, \cdot, \cdot, \cdot, \cdot) \leq 2^{c_1(n^{c_1} + \lambda^{c_1})}$. From Theorem 5.1, we can deduce that $\mathcal{L}^{(2)}$ depends on $3, h, \lambda$ and $\mathcal{V}^{(1)}$, and can be calculated from them in time $\text{poly}_h(\log \lambda, |\mathcal{V}^{(1)}|)$. By Theorem 4.36,

$$
|\mathcal{V}^{(1)}| = \underbrace{|\mathcal{S}^{(1)}|}_{\leq c_1 \log^{c_1} \lambda} + \underbrace{|\mathcal{A}^{(1)}|}_{\leq c_1 \log^{c_1} \lambda} + \underbrace{|\mathcal{L}^{(1)}|}_{\leq c_1(\log^{c_1} \lambda + |\mathcal{V}|^{c_1})} + |\mathcal{D}| \leq c_1(3 \log^{c_1} \lambda + |\mathcal{V}|^{c_1}) + \underbrace{|\mathcal{D}|}_{O(1)} = \text{poly}_h(\log \lambda, |\mathcal{V}|) \,, \tag{222}
$$

and thus $|\mathcal{V}^{(2)}| = \text{poly}_h(\log \lambda, |\mathcal{V}|)$. In addition, by Theorem 5.1,

$$
\mathbb{T}(\mathcal{L}^{(2)}; n, \cdot, \cdot, \cdot, \cdot) \leq c_2(n^{c_2} + \lambda^{c_2} + \mathbb{T}(\mathcal{S}^{(1)}; n, \cdot, \cdot, \cdot, \cdot)^{c_2}) \,.
$$

Now, by Theorem 6.1,

$$
\begin{aligned}
\mathbb{T}(\mathcal{L}^{(3)}; n, \cdot, \cdot, \cdot, \cdot) &\leq c_3 \cdot (n^{c_3} + \mathcal{K}(n)^{c_3} + \mathbb{T}(\mathcal{K}; n)^{c_3} + \mathbb{T}(\mathcal{S}^{(2)}; n, \cdot, \cdot, \cdot, \cdot)^{c_3} + \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)^{c_3} + \mathbb{T}(\mathcal{L}^{(2)}; n, \cdot, \cdot, \cdot, \cdot)^{c_3}) \\
&\leq c_3 \cdot (n^{c_3} + (c_0 n \lambda)^{c_0 c_3} + (C_0 \log^{C_0} n + C_0 \log^{C_0} \lambda)^{c_3} + (2 c_1^{c_2} \cdot c_2 (n + \lambda)^{c_1 c_2})^{c_3} \\
&\quad + (c_2 (n^{c_2} + \lambda^{c_2}))^{c_3} + (c_2 (n^{c_2} + \lambda^{c_2}) + (c_1 (n^{c_1} + \lambda^{c_1}))^{c_2})^{c_3}) \\
&= \text{poly}_h(n, \lambda) \,.
\end{aligned}
$$

Also, the description of $\mathcal{L}^{(3)}$ can be calculated in time

$$
\text{poly}(\underbrace{|\mathcal{K}|}_{\text{polylog}_h(\lambda)}, \underbrace{|\mathcal{V}^{(2)}|}_{\text{poly}_h(\log \lambda, |\mathcal{V}|)}) = \text{poly}_h(\log \lambda, |\mathcal{V}|) \,.
$$

This finishes the proof of the linear constraints processor properties of Theorem 7.1.

Value properties:

In this part, we may assume $\mathcal{V}$ is λ-bounded. Namely,

$$
|\mathcal{V}| \leq \lambda \quad , \quad \mathbb{T}(\mathcal{S}; n, \cdot, \cdot, \cdot, \cdot, \cdot), \mathbb{T}(\mathcal{A}; n, \cdot, \cdot), \mathbb{T}(\mathcal{L}; n, \cdot, \cdot, \cdot, \cdot) \leq n^{\lambda} \,.
$$

As $|\mathcal{V}| \leq \lambda$, and assuming c_1 is larger than $|\mathcal{D}|$ regardless of h, we have by (222) that

$$
|\mathcal{V}^{(1)}| \leq c_1(3 \log^{c_1} \lambda + \lambda^{c_1} + 1) \leq 5 c_1 \lambda^{c_1} \,. \tag{223}
$$

By Theorem 4.36,

$$
\mathbb{T}(\mathcal{S}^{(1)}; n, \cdot, \cdot, \cdot, \cdot, \cdot) \leq c_1(n^{c_1} + \lambda^{c_1}) \,, \quad \mathbb{T}(\mathcal{A}^{(1)}; n, \cdot, \cdot), \mathbb{T}(\mathcal{L}^{(1)}; n, \cdot, \cdot, \cdot, \cdot) \leq 2^{c_1(n^{c_1} + \lambda^{c_2})} \,. \tag{224}
$$

Let us assume $\mathcal{V}_{2^n}$ has a perfect ZPC strategy. By Theorem 4.36, given that $\mathcal{V}$ is λ-bounded, $\mathcal{V}_n^{(1)}$ has a perfect ZPC strategy. Now, by (223) and (224), $\mathcal{V}^{(1)}$ satisfies the conditions of the value properties in Theorem 5.1, and thus $\mathcal{V}_n^{(1)}$ having a perfect ZPC strategy implies $\mathcal{V}_n^{(2)}$ has a perfect ZPC strategy. As Theorem 6.1 has no conditions for the value properties, $\mathcal{V}_n^{(2)}$ having a perfect ZPC strategy implies $\mathcal{V}_n^{(3)}$ has a perfect ZPC strategy, proving the perfect completeness condition of Theorem 7.1.

Let $\varepsilon_1 = \left(\frac{1}{2c_1}\right)^{16}$, which means $1 - c_1 \varepsilon_1^{1/16} = 1/2$ and $1 - c_1\varepsilon_1 > 1/2$. Then, as $\mathcal{V}^{(1)}$ is λ-bounded, the entanglement lower bound from the value properties of Theorem 4.36 holds, which means

$$\mathscr{E}(\mathcal{V}_n^{(1)}, 1 - \varepsilon_1) \geq \underbrace{(1 - c_1\varepsilon_1)}_{\geq 1/2} \cdot 2^{2^{\lambda n}} \cdot \mathscr{E}(\mathcal{V}_{2^n}, \underbrace{1 - c_1\varepsilon_1^{1/16}}_{=1/2}) \geq 2^{2^{\lambda n}-1} \cdot \mathscr{E}(\mathcal{V}_{2^n}, 1/2) . \tag{225}$$

As the last step in $\mathsf{QuestionReduction}_h$ (see the proof of Theorem 4.36 in Section 4.6.1) is to apply DeType_1 (Claim 4.46), the game $\mathcal{V}_n^{(1)}$ is $\mathfrak{DeType}(\mathfrak{G})$ (Definition 4.40) for some other game $\mathfrak{G}$. Hence, excluding the anchor vertices in $\mathcal{V}_n^{(1)}$ (whose lengths are anyway 0 and whenever sampled against the game accepts), the underlying graph of $\mathcal{V}_n^{(1)}$ is bipartite. Therefore, by Remark 3.55, the value and entanglement lower bounds of $\mathcal{V}_n^{(1)}$ and $\mathfrak{DoubleCover}(\mathcal{V}_n^{(1)})$ (recall the double cover game from Definition 3.52) are the same. In particular,

$$\mathscr{E}(\mathfrak{DoubleCover}(\mathcal{V}_n^{(1)}), 1 - \varepsilon_1) = \mathscr{E}(\mathcal{V}_n^{(1)}, 1 - \varepsilon_1) .$$

Let $C = ((2c_1)^{16}c_2)^{c_2}$,[108] which implies in particular (as λ is a positive integer) that $c_2(C\lambda)^{-1/c_2} \leq \frac{\varepsilon_1}{2}$, and let $\varepsilon_2(n) = \left(\frac{\varepsilon_1}{2c_2(n\lambda)^{c_2}}\right)^{c_2}$, which implies

$$c_2(n\lambda)^{c_2}(\varepsilon_2(n))^{1/c_2} = \frac{\varepsilon_1}{2} .$$

By (223) and (224), $\mathcal{V}^{(1)}$ satisfies the conditions of the value properties in Theorem 5.1, and thus

$$\mathscr{E}(\mathcal{V}_n^{(2)}, 1 - \varepsilon_2(n)) \geq \mathscr{E}(\mathfrak{DoubleCover}(\mathcal{V}_n^{(1)}), 1 - c_2((n\lambda)^{c_2}(\varepsilon_2(n))^{1/c_2} + (n\lambda)^{-1/c_2})) .$$

Assuming $n \geq C$, we have $c_2(n\lambda)^{-1/c_2} \leq c_2(C\lambda)^{-1/c_2} \leq \frac{\varepsilon_1}{2}$ and combined with our choice of $\varepsilon_2(n)$ we have

$$\mathscr{E}(\mathfrak{DoubleCover}(\mathcal{V}_n^{(1)}), 1 - \underbrace{c_2(n\lambda)^{c_2}(\varepsilon_2(n))^{1/c_2}}_{=\varepsilon_1/2} - \underbrace{c_2(n\lambda)^{-1/c_2}}_{\leq \varepsilon_1/2}) \geq \mathscr{E}(\mathfrak{DoubleCover}(\mathcal{V}_n^{(1)}), 1 - \varepsilon_1) .$$

All in all, for every $n \geq C$, we have

$$\mathscr{E}(\mathcal{V}_n^{(2)}, 1 - \varepsilon_2(n)) \geq \mathscr{E}(\mathcal{V}_n^{(1)}, 1 - \varepsilon_1) . \tag{226}$$

Recall the parameter from (213) in Theorem 6.1:

$$p(\varepsilon_2(n), n) = \frac{c_3}{(\varepsilon_2(n))^{c_3}} \cdot 2^{-\frac{(\varepsilon_2(n))^{c_3}}{c_3} \cdot \mathcal{K}(n) \cdot \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)^{-1}} = 2^{\log \frac{c_3}{(\varepsilon_2(n))^{c_3}} - \frac{(\varepsilon_2(n))^{c_3}}{c_3} \cdot \mathcal{K}(n) \cdot \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)^{-1}} .$$

By the value properties of Theorem 6.1,

$$\mathscr{E}(\mathcal{V}_n^{(3)}, p(\varepsilon_2(n), n)) \geq \mathscr{E}(\mathcal{V}^{(2)}, 1 - \varepsilon_2(n)) . \tag{227}$$

If we choose $\mathcal{K}$ from (221) so that

$$\log \frac{c_3}{(\varepsilon_2(n))^{c_3}} - \frac{(\varepsilon_2(n))^{c_3}}{c_3} \cdot \mathcal{K}(n) \cdot \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)^{-1} \leq -1 , \tag{228}$$

---

[108] Note that as c_1 and c_2 are constants that depend only on h, the same is true for C.

then we have $p(\varepsilon_2(n), n) \le 1/2$ and thus

$$\mathscr{E}(\mathcal{V}_n^{(3)}, 1/2) \ge \mathscr{E}(\mathcal{V}_n^{(3)}, p(\varepsilon_2(n), n)) . \tag{229}$$

Combining (225), (226), (227) and (229), we deduce that for every $n \ge C$ (assuming (228) is satisfied),

$$\mathscr{E}(\mathcal{V}_n^{(3)}, 1/2) \ge 2^{2^{\lambda n}-1} \cdot \mathscr{E}(\mathcal{V}_{2^n}, 1/2) \ge \max\{2^{2^{\lambda n}-1}, \mathscr{E}(\mathcal{V}_{2^n}, 1/2)\} ,$$

finishing the proof of the soundness condition from Theorem 7.1. So, let us finally choose c_0 and thus the TM $\mathcal{K}$ from (221) so that (228) is satisfied. Manipulating (228), we need to have

$$\mathcal{K}(n) \ge \left(1 + \log \frac{c_3}{(\varepsilon_2(n))^{c_3}}\right) \cdot \frac{c_3}{(\varepsilon_2(n))^{c_3}} \cdot \mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot) .$$

Recalling our choices of ε_1 and $\varepsilon_2(n)$, we have

$$\frac{c_3}{(\varepsilon_2(n))^{c_3}} = c_3(2c_2(n\lambda)^{c_2}(2c_1)^{16})^{c_2} \le (4c_1c_2c_3)^{16c_2}(n\lambda)^{c_2^2} .$$

Also, using the fact that for positive integers $1 + \log_2 x \le 2x$, we have that

$$1 + \log \frac{c_3}{(\varepsilon_2(n))^{c_3}} \le \frac{2c_3}{(\varepsilon_2(n))^{c_3}} \le 2(4c_1c_2c_3)^{16c_2}(n\lambda)^{c_2^2} .$$

Let us also recall the upper bound on $\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot)$ previously calculated,

$$\mathbb{T}(\mathcal{A}^{(2)}; n, \cdot, \cdot) \le c_2(n^{c_2} + \lambda^{c_2}) \le 2c_2(n\lambda)^{c_2} .$$

So, if we choose c_0 so that

$$c_0(n\lambda)^{c_0} = \mathcal{K}(n) \ge 4c_2(4c_1c_2c_3)^{32c_2}(n\lambda)^{c_2^2+c_2} ,$$

we are done. As $4c_2(4c_1c_2c_3)^{32c_2} \ge c_2^2 + c_2$, choose $c_0 = 4c_2(4c_1c_2c_3)^{32c_2}$. Since each of c_1, c_2, c_3 depends only on h (in the case of c_3, it is actually a universal constant), c_0 is only a function of h, as needed. This finishes the proof of Theorem 7.1. $\square$

References

[AB09]   Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009. 2.32

[AD22]   Danil Akhtiamov and Alon Dogon. On uniform hilbert schmidt stability of groups. *Proceedings of the American Mathematical Society*, 150(4):1799–1809, 2022. 4.2

[AL07]   David Aldous and Russell Lyons. Processes on Unimodular Random Networks. *Electronic Journal of Probability*, 12(none):1454 – 1508, 2007. (document), 1

[Ald07]   David Aldous. Local weak limits and unimodularity. *Blog post*, 2007. (document)

[Alo99]   Noga Alon. Combinatorial nullstellensatz. *Combinatorics, Probability and Computing*, 8(1-2):7–29, 1999. 5.1.3

[Ara02]   PK Aravind. A simple demonstration of Bell's theorem involving two observers and no probabilities or inequalities. *arXiv preprint quant-ph/0206070*, 2002. 2.4

[BCLV24]   Lewis Bowen, Michael Chapman, Alex Lubotzky, and Thomas Vidick. The Aldous–Lyons Conjecture I: Subgroup Tests. *preprint*, 2024. (document), 1, 1.1, 1, 1, 2.19

[Bel64]   John S Bell. On the einstein podolsky rosen paradox. *Physics Physique Fizika*, 1(3):195, 1964. 3

[BFL91]   László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *computational complexity*, 1(1):3–40, 1991. 1, 5.1, 5.23

[BL06]   Yonatan Bilu and Nathan Linial. Lifts, discrepancy and nearly optimal spectral gap. *Combinatorica*, 26(5):495–519, 2006. 3.5.3

[Bla06]   Bruce Blackadar. *Operator algebras: theory of C*-algebras and von Neumann algebras*, volume 122. Springer Science & Business Media, 2006. 3.1

[Bro06]   Nathanial Patrick Brown. *Invariant Means and Finite Representation Theory of C*-Algebras*, volume 13. American Mathematical Soc., 2006. 1

[BSGH+04] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust pcps of proximity, shorter pcps and applications to coding. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 1–10, 2004. 1.1

[BVY17]   Mohammad Bavarian, Thomas Vidick, and Henry Yuen. Hardness amplification for entangled games via anchoring. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 303–316. ACM, 2017. 1.1, 5.1.6, 6.1, 6.5, 6.6

[CL23]   Michael Chapman and Alex Lubotzky. Stability of homomorphisms, coverings and cocycles I: Equivalence. *preprint*, 2023. 1.1

[CLS17]   Richard Cleve, Li Liu, and William Slofstra. Perfect commuting-operator strategies for linear system games. *Journal of Mathematical Physics*, 58(1), 2017. 2.4

[CM14]   Richard Cleve and Rajat Mittal. Characterization of binary constraint system games. In *International Colloquium on Automata, Languages, and Programming*, pages 320–331. Springer, 2014. 1, 1

[Con76]   Alain Connes. Classification of injective factors cases II_1, II_∞, III_λ, $\lambda \neq 1$. *Annals of Mathematics*, pages 73–115, 1976. (document), 1

[Coo71]   Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, page 151–158, New York, NY, USA, 1971. Association for Computing Machinery. 5.1.2, 5.13

[CT65]   James W Cooley and John W Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965. 5.1.3

[CVY23]   Michael Chapman, Thomas Vidick, and Henry Yuen. Efficiently stable presentations from error-correcting codes. *Preprint*, 2023. 3.12, 3.2.1, 3.21, 3.2.3, 3.7, 3.74, 3.8

[dlS22a]   Mikael de la Salle. Orthogonalization of Positive Operator Valued Measures. *Comptes Rendus. Mathématique*, 360:549–560, 2022. 3.21

[dlS22b]   Mikael de la Salle. Spectral gap and stability for groups and non-local games, 2022. (document), 1.1, 1.1, 3, 3.2.2, 3.20, 3.7, 3.73, 3.74, 3.8, 3.75, 3.78, 4.2

[Gau86]   CF Gauss. Theoria interpolationis methodo nova tractata werke band 3, 265–327. *Göttingen: Königliche Gesellschaft der Wissenschaften*, 1886. 5.1.3

[GH17]   William T. Gowers and Omid Hatami. Inverse and stability theorems for approximate representations of finite groups. *Mat. Sb.*, 208(12):70–106, 2017. 1.1, 3.2, 3.74

[GO05]   Venkatesan Guruswami and Ryan O'Donnell. A history of the PCP Theorem. *https://courses.cs.washington.edu/courses/cse533/05au/pcp-history.pdf*, 2005. 5.37

[Göd31]   Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für mathematik und physik*, 38:173–198, 1931. 2.5.1

[GR09]   Lev Glebsky and Luis Manuel Rivera. Almost solutions of equations in permutations. *Taiwanese J. Math.*, 13(2A):493–500, 2009. 1.1

[HBD+15]   Bas Hensen, Hannes Bernien, Anaïs E Dréau, Andreas Reiserer, Norbert Kalb, Machiel S Blok, Just Ruitenberg, Raymond FL Vermeulen, Raymond N Schouten, Carlos Abellán, et al. Loophole-free bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526(7575):682–686, 2015. 1

[HLW06]   Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. Amer. Math. Soc.*, 43:439–561, 2006. 3.74

[HMU06]   John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., USA, 2006. 2.32, 2.5.1

[HS18]   Don Hadwin and Tatiana Shulman. Stability of group relations under small Hilbert-Schmidt perturbations. *J. Funct. Anal.*, 275(4):761–792, 2018. 1.1

[Ioa]   Adrian Ioana. Stability for product groups and property (τ). 4.2

[JNV+20]   Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. Quantum soundness of the classical low individual degree test. *arXiv preprint arXiv:2009.12982*, 2020. 77

[JNV+21]   Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. MIP*=RE. *Commun. ACM*, 64(11):131–138, 2021. (document), 1, 1, 1, 1.1, 1.1, 1.1, 1.3, 2.4, 2.5, 2.5.1, 2.5.1, 2.5.2, 2.54, 2.6.3, 2.6.3, 3.7, 11, 4.3, 4.12, 4.3, 4.18, 4.25, 57, 4.30, 4.42, 5, 6, 5.24, 5.1.3, 5.32, 5.3, 5.3, 5.3.2, 5.64, 93, 94, 5.79, 6.1, 7

[JNV+22a]   Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. Quantum soundness of testing tensor codes. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 586–597. IEEE, 2022. 3.2, 3.7, 3.12, 3.13, 4

[JNV+22b]   Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. Quantum soundness of testing tensor codes. *Discrete Analysis*, 12 2022. 5.4, 5.4, 97, 5.4, 5.76, 5.4

[Jus72]   Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on information theory*, 18(5):652–656, 1972. 3.7.2

[Kar72]   Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972. 5.13

[KPS18]   Se-Jin Kim, Vern Paulsen, and Christopher Schafhauser. A synchronous game for binary constraint systems. *Journal of Mathematical Physics*, 59(3):032201, 2018. 1, 1, 1, 1.1

[Lev73]   Leonid Anatolevich Levin. Universal sequential search problems. *Problemy peredachi informatsii*, 9(3):115–116, 1973. 5.1.2, 5.13

[LFKN90]   Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. In *Proceedings of 31st Annual Symposium on Foundations of Computer Science*, pages 2–10. IEEE, 1990. 1

[MNY22]   Hamoon Mousavi, Seyed Sajjad Nezhadi, and Henry Yuen. Nonlocal games, compression theorems, and the arithmetical hierarchy. In *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing*, pages 1–11, 2022. 1

[MPTW23]   L Mančinska, VI Paulsen, IG Todorov, and A Winter. Products of synchronous games. *Studia Mathematica*, 272:299–317, 2023. 6

[MSNY24]   Andrew Marks, Seyed Sajjad Nezhadi, and Henry Yuen. The recursive compression method for proving undecidability results. *Manuscript*, 2024. 1.1, 2.6.3

[NV17]   Anand Natarajan and Thomas Vidick. A quantum linearity test for robustly verifying entanglement. In *STOC'17—Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1003–1015. ACM, New York, 2017. 3.7

[NV18a]   Anand Natarajan and Thomas Vidick. Low-degree testing for quantum states, and a quantum entangled games PCP for QMA. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 731–742. IEEE, 2018. 1.1

[NV18b]   Anand Natarajan and Thomas Vidick. Two-player entangled games are NP-hard. In *Proceedings of the 33rd Computational Complexity Conference*, page 20. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. 3, 3.74

[NW19]   Anand Natarajan and John Wright. NEEXP is contained in MIP*. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 510–518. IEEE, 2019. 1.1, 3.12, 3.13, 3.35, 3.7, 5.76, 5.4

[Pau03]   Vern Paulsen. *Completely Bounded Maps and Operator Algebras*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2003. 3.9

[PS94]   Alexander Polishchuk and Daniel A Spielman. Nearly-linear size holographic proofs. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 194–203, 1994. 77

[PS23]   Connor Paddock and William Slofstra. Satisfiability problems and algebras of boolean constraint system games. *arXiv preprint arXiv:2310.07901*, 2023. 1, 1, 1.1

[PSS$^+$16] Vern I Paulsen, Simone Severini, Daniel Stahlke, Ivan G Todorov, and Andreas Winter. Estimating quantum chromatic numbers. *Journal of Functional Analysis*, 270(6):2188–2222, 2016. 1

[Raz95]   Ran Raz. A parallel repetition theorem. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 447–456, 1995. 5.1.6

[Rog87]   Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. MIT Press, Cambridge, MA, USA, 1987. 2.6.3

[RS96]   Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996. 5.1.3

[Sch80]   Jacob T Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM (JACM)*, 27(4):701–717, 1980. 5.19

[Sha90]   Adi Shamir. IP = PSPACE. In *Proceedings of 31st Annual Symposium on Foundations of Computer Science*, pages 11–15. IEEE, 1990. 1

[Sip12]   Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2012. 2.5.1

[Slo19]   William Slofstra. Tsirelson's problem and an embedding theorem for groups arising from non-local games. *Journal of the American Mathematical Society*, 2019. 1

[Tsi06]   Boris S Tsirelson. Bell inequalities and operator algebras, 2006. Problem statement from website of open problems at TU Braunschweig (2006), available at http://web.archive.org/web/20090414083019/http://www.imaph.tu-bs.de/qi/problems 1

[Tur37]   Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society*, 2(1):230–265, 1937. 2.5.1, 30, 5.1.1

[Vid22]   Thomas Vidick. Almost synchronous quantum correlations. *Journal of mathematical physics*, 63(2), 2022. (document), 1.1, 1.1, 3.6, 6

[Zip79]   Richard Zippel. Probabilistic algorithms for sparse polynomials. In *International symposium on symbolic and algebraic manipulation*, pages 216–226. Springer, 1979. 5.19

[Żuk03]   Andrzej Żuk. Property (T) and Kazhdan constants for discrete groups. *Geometric & Functional Analysis GAFA*, 13:643–670, 2003. 5.2