Bandits with Knapsacks

ASHWINKUMAR BADANIDIYURU, Google Research
ROBERT KLEINBERG, Cornell University
ALEKSANDRS SLIVKINS, Microsoft Research

Multi-armed bandit problems are the predominant theoretical model of exploration-exploitation tradeoffs in learning, and they have countless applications ranging from medical trials, to communication networks, to Web search and advertising. In many of these application domains, the learner may be constrained by one or more supply (or budget) limits, in addition to the customary limitation on the time horizon. The literature lacks a general model encompassing these sorts of problems. We introduce such a model, called *bandits with knapsacks*, that combines bandit learning with aspects of stochastic integer programming. In particular, a bandit algorithm needs to solve a stochastic version of the well-known *knapsack problem*, which is concerned with packing items into a limited-size knapsack. A distinctive feature of our problem, in comparison to the existing regret-minimization literature, is that the optimal policy for a given latent distribution may significantly outperform the policy that plays the optimal fixed arm. Consequently, achieving sublinear regret in the bandits-with-knapsacks problem is significantly more challenging than in conventional bandit problems.

We present two algorithms whose reward is close to the information-theoretic optimum: one is based on a novel "balanced exploration" paradigm, while the other is a primal-dual algorithm that uses multiplicative updates. Further, we prove that the regret achieved by both algorithms is optimal up to polylogarithmic factors. We illustrate the generality of the problem by presenting applications in a number of different domains, including electronic commerce, routing, and scheduling. As one example of a concrete application, we consider the problem of dynamic posted pricing with limited supply and obtain the first algorithm whose regret, with respect to the optimal dynamic policy, is sublinear in the supply.

CCS Concepts: • **Theory of computation** → **Online learning algorithms**; **Online learning theory**; **Regret bounds**; *Algorithmic mechanism design*; *Computational pricing and auctions*;

Additional Key Words and Phrases: Multi-armed bandits, dynamic pricing, knapsack constraints

An extended abstract of this paper [14] was published in *IEEE FOCS 2013*. This paper has undergone several rounds of revision since the original "full version" has been published on `arxiv.org` in May 2013. Presentation has been thoroughly revised throughout, from low-level edits to technical intuition and details to high-level restructuring of the paper. Some of the results have been improved: a stronger regret bound in one of the main results (Theorem 4.1), and a more general example of preadjusted discretization for dynamic pricing (Theorem 7.7). Introduction discusses a significant amount of follow-up work, and is up-to-date regarding open questions.

Parts of this research have been done while A. Badanidiyuru was a research intern at Microsoft Research and a graduate student at Cornell University, and while R. Kleinberg was a Consulting Researcher at Microsoft Research.

A. Badanidiyuru was partially supported by NSF grant IIS-0905467. R. Kleinberg was partially supported by NSF grants CCF-0643934, IIS-0905467 and AF-0910940, a Microsoft Research New Faculty Fellowship, and a Google Research Grant.

Authors' addresses: A. Badanidiyuru, Google, 1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA; email: ashwinkumarbv@gmail.com; R. Kleinberg, 317 Gates Hall, Department of Computer Science, Cornell University, Ithaca, NY 14853, USA; email: rdk@cs.cornell.edu; A. Slivkins, Microsoft Research New York City, 641 6th Ave, 7th fl., New York, NY 10011, USA; email: slivkins@microsoft.com.

https://doi.org/10.1145/3164539

ACM Reference format:
Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. 2018. Bandits with Knapsacks. *J. ACM* 65, 3, Article 13 (March 2018), 55 pages.
https://doi.org/10.1145/3164539

1 INTRODUCTION

For more than 50 years, the multi-armed bandit problem (henceforth, *MAB*) has been the predominant theoretical model for sequential decision problems that embody the tension between exploration and exploitation, "the conflict between taking actions which yield immediate reward and taking actions whose benefit (e.g., acquiring information or preparing the ground) will come only later," to quote Whittle's apt summary [57]. Owing to the universal nature of this conflict, it is not surprising that MAB algorithms have found diverse applications ranging from medical trials, to communication networks, to Web search and advertising.

A common feature in many of these application domains is the presence of one or more limited-supply resources that are consumed during the decision process. For example, scientists experimenting with alternative medical treatments may be limited not only by the number of patients participating in the study but also by the cost of materials used in the treatments. A website experimenting with displaying advertisements is constrained not only by the number of users who visit the site but by the advertisers' budgets. A retailer engaging in price experimentation faces inventory limits along with a limited number of consumers. The literature on MAB problems lacks a general model that encompasses these sorts of decision problems with supply limits. Our article contributes such a model, called *bandits with knapsacks* (henceforth, BwK), in which a bandit algorithm needs to solve a stochastic, multi-dimensional version of the well-known *knapsack problem*. We present algorithms whose regret (normalized by the payoff of the optimal policy) converges to zero as the resource budget and the optimal payoff tend to infinity. In fact, we prove that this convergence takes place at the information-theoretically optimal rate.

1.1 Our Model: Bandits with Knapsacks (BwK)

Problem definition. A learner has a fixed set of potential actions, a.k.a. *arms*, denoted by X and called *action space*. (In our main results, X will be finite, but we will also consider extensions with an infinite set of arms, see Sections 8 and 7.) There are d resources being consumed by the learner. Over a sequence of time steps, the learner chooses an arm and observes two things: a *reward* and a *resource consumption vector*. Rewards are scalar-valued, whereas resource consumption vectors are d-dimensional: the ith component represents consumption of resource i. For each resource i, there is a pre-specified *budget* B_i representing the maximum amount that may be consumed, in total. The process stops at the first time τ when the total consumption of some resource exceeds its budget. The objective is to maximize the total reward received before time τ.

We assume that the environment does not change over time. Formally, the observations for a fixed arm x in each time step (i.e., the reward and resource consumption vector) are independent samples from a fixed joint distribution on $[0,1] \times [0,1]^d$, called the *latent distribution* for arm x.

There is a known, finite time horizon T. We model it as one of the resources, one unit of which is deterministically consumed in each decision period, and the budget is T.

Notable examples. The conventional MAB problem, with a finite time horizon T, naturally fits into this framework. A more interesting example is the *dynamic pricing* problem faced by a retailer selling B items to a population of T unit-demand consumers who arrive sequentially. Modeling this as a BwK problem, rounds correspond to consumers, and arms correspond to the possible prices, which may be offered to a consumer. Reward is the revenue from a sale, if any. Resource consumption vectors express the number of items sold and consumers seen, respectively. Thus, if a price p is offered and accepted, the reward is p and the resource consumption is $\left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$. If the offer is declined, then the reward is 0 and the resource consumption is $\left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$.

A "dual" problem of dynamic pricing is *dynamic procurement*, where the algorithm is "dynamically buying" rather than dynamically selling. The reward refers to the number of bought items, and the budget constraint B now applies to the amount spent (which is why the two problems are not merely identical up to sign reversal). If a price p is offered and accepted, then the reward is 1 and the resource consumption is $\left[\begin{smallmatrix}p\\1\end{smallmatrix}\right]$. If the offer is declined, then the reward is 0 and the resource consumption is $\left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$. This problem is also relevant to the domain of crowdsourcing: the items bought then correspond to microtasks ordered on a crowdsourcing platform such as Amazon Mechanical Turk.

Another simple example concerns *dynamic ad allocation* for pay-per-click ads with unknown click probabilities. There is one advertiser with several ads and budget B across all ads, and T users to show the ads to. The ad platform allocates one ad to a new user in each round. Whenever a given ad x is chosen and clicked on, the advertiser pays a known amount π_x. To model this as a BwK problem, arms correspond to ads, rewards are the advertiser's payments, and resource consumption refers to the amount spent by the advertiser and the number of users seen. Thus, if ad x is chosen and clicked, the reward is π_x and the resource consumption is $\left[\begin{smallmatrix}\pi_x\\1\end{smallmatrix}\right]$; otherwise, the reward is 0 and the resource consumption is $\left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$.

All three examples can be easily generalized to multiple resource constraints: respectively, to selling multiple products, procuring different types of goods, and allocating ads from multiple advertisers.

Benchmark and regret. The performance of an algorithm will be measured by its *regret*: the worst case, over all possible tuples of latent distributions, of the difference between OPT and the algorithm's expected total reward. Here, OPT is the expected total reward of the benchmark: an *optimal dynamic policy*, an algorithm that maximizes expected total reward given foreknowledge of the latent distributions.

In a conventional MAB problem, the optimal dynamic policy is to play a fixed arm, namely the one with the highest expected reward. In the BwK problem, the optimal dynamic policy is more complex, as the choice of an arm in a given round depends on the remaining supply of each resource. In fact, we doubt there is a polynomial-time algorithm to compute the optimal dynamic policy given the latent distributions; similar problems in optimal control have long been known to be PSPACE-hard [49].

It is easy to see that the optimal dynamic policy may significantly out-perform the best fixed arm. To take a simple example, consider a problem instance with d resources and d arms such that pulling arm i deterministically produces a reward of 1, consumes one unit of resource i, and does not consume any other resources. We are given an initial endowment of B units of each resource. Any policy that plays a fixed arm i in each round is limited to a total reward of B before running out of its budget of resource i. Whereas an algorithm that alternates arms in a round-robin fashion achieves reward dB: d times larger. Similar, but somewhat more involved examples can be found for application domains of interest; see Appendix A. Interestingly, in all these examples it suffices to consider a *time-invariant mixture* of arms, i.e., a policy that samples in each period from a fixed probability distribution over arms regardless of the remaining resource supplies. In particular, in the simple example above it suffices to consider a uniform distribution.

Alternative definitions. More generally, we could model the budget constraints as a downward-closed polytope $\mathscr{P} \subset \mathbb{R}_+^d$ such that the process stops when the sum of resource consumption vectors is no longer in $\mathscr{P}$. However, our assumption that $\mathscr{P}$ is a box constraint is virtually without loss of generality. If $\mathscr{P}$ is instead specified by a system of inequalities $\{Ax \leq b\}$, then we can redefine the resource consumption vectors to be Ax instead of x and then the budget constraint is the box constraint defined by the vector b. The only potential downside of this transformation is that it increases the dimension of the resource vector space, when the constraint matrix A has more rows than columns. However, one of our algorithms has regret depending only logarithmically on d, so this increase typically has only a mild effect on regret.

Our stopping condition halts the algorithm as soon as any budget is exceeded. Alternatively, we could restrict the algorithm to actions that cannot possibly violate any constraint if chosen in the current round, and stop if there is no such action. This alternative is essentially equivalent to the original version: each budget constraint changes by at most one, which does not affect our regret bounds in any significant way.

1.2 Main Results

We seek regret bounds that are sublinear in OPT, whereas in analyzing MAB algorithms one typically expresses regret bounds as a sublinear function of the time horizon T. This is because a regret guarantee of the form $o(T)$ may be unacceptably weak for the BwK problem, because supply limits prevent the optimal dynamic policy from achieving a reward close to T. An illustrative example is the dynamic pricing problem with supply $B \ll T$: the seller can only sell B items, each at a price of at most 1, so bounding the regret by any number greater than B is worthless. To achieve sublinear regret, the algorithm must be able to explore each arm a significant number of times without exhausting its resource budgets. Accordingly, we parameterize our regret bound by $B = \min_i B_i$, the smallest budget constraint.

Algorithms. We present an algorithm, called PrimalDualBwK, whose regret is sublinear in OPT as both OPT and B tend to infinity. More precisely, denoting the number of arms by m, our algorithm's regret is

$$\widetilde{O}\left(\sqrt{m\,\mathtt{OPT}} + \mathtt{OPT}\sqrt{m/B}\right), \tag{1}$$

where the $\widetilde{O}()$ notation hides logarithmic factors. Note that without resource constraints, i.e., setting $B = T$, we recover regret $\widetilde{O}(\sqrt{m\mathtt{OPT}})$, which is optimal up to log factors [10]. In fact, we prove a slightly stronger regret bound that has an optimal scaling property: if all budget constraints, including the time horizon, are increased by the factor of α, then the regret bound scales as $\sqrt{\alpha}$.[1] The algorithm is computationally efficient, in a strong sense: with machine word size of $\log T$ bits or more, the per-round running time is $O(md)$. Moreover, if each arm j consumes only d_j resources that are known in advance, then the per-round running time is $O(m + d + \sum_j d_j)$.

We also present another algorithm, called BalancedExploration, whose regret bound is the same up to logarithmic factors for $d = O(1)$. The regret bounds for the two algorithms are incomparable: while PrimalDualBwK achieves a better dependence on d, BalancedExploration performs better in some special cases, see Appendix B for a simple example. While PrimalDualBwK is very computationally efficient, the specification of BalancedExploration involves a mathematically well-defined optimization step for which we do not provide a specific implementation, see Remark 4.2 fur further discussion.

[1] The square-root scaling is optimal even for the basic MAB problem, as proved in Auer et al. [10].

Lower bound. We provide a matching lower bound: we prove that the regret bound Equation (1) is optimal up to polylogarithmic factors; moreover, this holds *for any given tuple of parameters*. Specifically, we show that for any given tuple $(m, B, \texttt{OPT})$, any algorithm for $\texttt{BwK}$ must incur regret

$$\Omega\left(\min\left(\texttt{OPT},\ \texttt{OPT}\sqrt{m/B} + \sqrt{m\,\texttt{OPT}}\right)\right), \tag{2}$$

in the worst-case over all instances of $\texttt{BwK}$ with these $(m, B, \texttt{OPT})$. We also show that this dependence on the *smallest* budget constraint is inevitable in the worst case.

Applications and special cases. We derive corollaries for the three examples outlined in Section 1.1:

- We obtain regret $\widetilde{O}(B^{2/3})$ for the basic version of dynamic pricing. This is optimal for each (B, T) pair [12]. Prior work [12, 56] achieved $\widetilde{O}(B^{2/3})$ regret w.r.t. the best fixed price, and $\widetilde{O}(\sqrt{B})$ regret assuming "regularity."[2] The former result is much weaker than ours, see Appendix A for a simple example, and the latter result is incomparable.
- We obtain regret $\widetilde{O}(T/B^{1/4})$ for the basic version of dynamic procurement. Prior work [13] achieves a constant-factor approximation to the optimum with a prohibitively large constant (at least in the tens of thousands), so our result is a big improvement unless $\texttt{OPT} \gg T/B^{1/4}$.
- We obtain regret $\widetilde{O}(\sqrt{B})$ for the basic version of dynamic ad allocation. This is optimal when $B = T$ (i.e., when the budget constraint is void), by the basic $\sqrt{T}$ lower bound for MAB.

Our model admits numerous generalizations of these three examples, as well as applications to several other domains. To emphasize the generality of our contributions, we systematically discuss applications and corollaries in Section 8. Pointers to prior work on special cases of $\texttt{BwK}$ can be found in Section 1.5.

1.3 Challenges and Techniques

Challenges. As with all MAB problems, a central issue in $\texttt{BwK}$ is the tradeoff between exploration and exploitation. A naïve way to resolve this tradeoff is to separate exploration and exploitation: before the algorithm starts, the rounds are partitioned into "exploration rounds" and "exploitation rounds," so that the arms chosen in the former does not depend on the feedback, and the feedback from the latter is discarded.[3] For example, an algorithm may pick an arm uniformly at random for a pre-defined number of rounds, then choose the best arm given the observations so far, and stick to this arm from then on. However, it tends to be much more efficient to combine exploration and exploitation by adapting the exploration schedule to observations. Typically, in such algorithms all but the first few rounds serve both exploration and exploitation. Thus, one immediate challenge is to implement this approach in the context of $\texttt{BwK}$.

The $\texttt{BwK}$ problem is significantly more difficult to solve than conventional MAB problems for the following three reasons. First, to estimate the performance of a given time-invariant policy, one needs to estimate the expected *total* reward of this policy, rather than the per-round expected reward (because the latter does not account for resource constraints). Second, since exploration consumes resources other than time, the negative effect of exploration is not limited to the rounds in which it is performed. Since resource consumption is stochastic, this negative effect is not known in advance, and can only be estimated over time. Finally, and perhaps most

[2] "Regularity" is a standard (but limiting) condition that states that the mapping from prices to expected rewards is concave.

[3] While the intuition behind this definition has been well-known for some time, the precise definition is due to Babaioff et al. [11], Devanur and Kakade [24].

importantly, the optimal dynamic policy can significantly outperform the best fixed arm, as mentioned above. To compete with the optimal dynamic policy, an algorithm needs, essentially, to search over mixtures of arms rather than over arms themselves, which is a much larger search space. In particular, our algorithms improve over the performance of the best fixed arm, whereas algorithms for explore-exploit learning problems typically do not.[4]

Our algorithms. Algorithm `BalancedExploration` explicitly optimizes over mixtures of arms, based on a simple idea: balanced exploration inside confidence bounds. The design principle underlying many confidence-bound based algorithms for stochastic MAB, including the famous `UCB1` algorithm [9] and our algorithm `PrimalDualBwK`, is generally, "Exploit as much as possible, but use confidence bounds that are wide enough to encourage some exploration." The design principle in `BalancedExploration`, in contrast, could be summarized as, "Explore as much as possible, but use confidence bounds that are narrow enough to eliminate obviously suboptimal alternatives." Our algorithm balances exploration across arms, exploring *each arm* as much as possible given the confidence bounds. More specifically, there are designated rounds when the algorithm picks a mixture that approximately maximizes the probability of choosing this arm, among the mixtures that are not obviously suboptimal given the current confidence bounds.

Algorithm `PrimalDualBwK` is a primal-dual algorithm based on the multiplicative weights update method. It maintains a vector of "resource costs" that is adjusted using multiplicative updates. In every period it estimates each arm's expected reward and expected resource consumption, using upper confidence bounds for the former and lower confidence bounds for the latter; then it plays the most "cost-effective" arm, namely the one with the highest ratio of estimated resource consumption to estimated resource cost, using the current cost vector. Although confidence bounds and multiplicative updates are the bread and butter of online learning theory, we consider this way of combining the two techniques to be quite novel. In particular, previous multiplicative-update algorithms in online learning theory—such as the `Exp3` algorithm for MAB [10] or the weighted majority [44] and `Hedge` [31] algorithms for learning from expert advice—applied multiplicative updates to the probabilities of choosing different arms (or experts). Our application of multiplicative updates to the dual variables of the LP relaxation of `BwK` is conceptually quite a different usage of this technique.

Having alternative techniques to solve the same problem is generally useful in a rich problem space such as MAB. Indeed, one often needs to apply techniques beyond the original models for which they were designed, perhaps combining them with techniques that handle other facets of the problem. When pursuing such extensions, some alternatives may be more suitable than others, in particular, because they are more compatible with the other techniques. We already see examples of that in the follow-up work: Agrawal and Devanur [3] and Badanidiyuru et al. [15] use some of the techniques from `BalancedExploration` and `PrimalDualBwK`, respectively; see Section 1.4 for more details.

LP-relaxation. To compare our algorithms to $\mathtt{OPT}$, we compare both to a more tractable benchmark given by time-invariant mixtures of arms. More precisely, we define a linear programming relaxation for the expected total reward achieved by a time-invariant mixture of arms, and prove that the optimal value $\mathtt{OPT}_{\mathtt{LP}}$ achieved by this LP-relaxation is an upper bound for $\mathtt{OPT}$. Therefore it suffices to relate our algorithms to the time-invariant mixture of arms that achieves $\mathtt{OPT}_{\mathtt{LP}}$, and bound their regret with respect to $\mathtt{OPT}_{\mathtt{LP}}$.

Lower bounds. The lower bound Equation (2) is based on a simple example in which all arms have reward 1 and 0-1 consumption of a single resource, and one arm has slightly smaller expected

[4] A few notable exceptions are in References [1, 10, 13, 17]. Of these, Besbes and Zeevi [17] and Badanidiyuru et al. [13] are on special cases of `BwK` and are discussed later.

resource consumption than the rest. To analyze this example, we apply the KL-divergence technique from the MAB lower bound in Auer et al. [10]. Some technical difficulties arise, compared to the derivation in Auer et al. [10], because the arms are different in terms of the expected consumption rather than expected reward, and because we need to match the desired value for OPT.

Discretization. In some applications, such as dynamic pricing and dynamic procurement, the action space X is very large or infinite, so our main algorithmic result is not immediately applicable. However, the action space has some structure that our algorithms can leverage: e.g., a price is just a number in some fixed interval. To handle such applications, we discretize the action space: we apply a BwK algorithm with a restricted, finite action space $S \subset X$, where S is chosen in advance. Immediately, we obtain a bound on regret with respect to the optimal dynamic policy restricted to S. Further, we select S to balance the tradeoff between $|S|$ and the *discretization error*: the decrease in the performance benchmark due to restricting the action space to S. We call this approach *preadjusted discretization*. While it has been used in prior work, the key step of bounding the discretization error is now considerably more difficult, as one needs to take into account resource constraints and argue about mixtures of arms rather than individual arms.

We bound discretization error for subset S, which satisfies certain axioms, and apply this result to handle dynamic pricing with a single product and dynamic procurement with a single budget constraint. While the former application is straightforward, the latter takes some work and uses a nonstandard mesh of prices. Bounding the discretization error for more than one resource constraints (other than time) appears to be much more challenging; we only achieve this for a special case.

1.4 Follow-up Work and Open Questions

Since the BwK problem provides a novel general problem formulation in online learning, it lends itself to a rich set of research questions in a similar way as the stochastic MAB problem did following Lai and Robbins [42] and Auer et al. [9]. Some of these questions were researched in the follow-up work.

Follow-up work. Following the conference publication of this article [14], there have been several developments directly inspired by BwK.

Agrawal and Devanur [3] extend BwK from hard resource constraints and additive rewards to a more general model that allows penalties and diminishing returns. In particular, the time-averaged outcome vector $\bar{v}$ is constrained to lie in an arbitrary given convex set, and the total reward can be an arbitrary concave, Lipschitz-continuous function of $\bar{v}$. They provide several algorithms for this model whose regret scales optimally as a function of the time horizon. Remarkably, these algorithms specialize to *three* new algorithms for BwK, based on different ideas. One of these new BwK algorithms follows the "optimism under uncertainty" approach from Reference [9] (with an additional trick of rescaling the resource constraints). Despite the apparent simplicity, it is shown to satisfy our main regret bound Equation (1).

Badanidiyuru et al. [15] extend BwK to *contextual bandits*: a bandit model where in each round the "context" is revealed (e.g., a user profile), then the algorithm selects an arm, and the resulting outcome (in our case, reward and resource consumption) depends on both the chosen arm and the context. Badanidiyuru et al. [15] merge BwK and *contextual bandits with policy sets* [43], a well-established, very general model for contextual bandits. They achieve regret that scales optimally in terms of the time horizon and the number of policies (respectively, square-root and logarithmic). Akin to BalancedExploration, their algorithm is not computationally efficient.

Both Agrawal and Devanur [3] and Badanidiyuru et al. [15] take advantage of various techniques developed in this article. First, both articles use (a generalization of) linear relaxations from Section 3. In fact, the two claims in Section 3 are directly used in Badanidiyuru et al. [15] to derive the corresponding statements for the contextual version. Second, Badanidiyuru et al. [15] build on the design and analysis of BalancedExploration, and merging them with a technique from prior work on contextual bandits [28]. Third, the analysis of one of the algorithms in Agrawal and Devanur [3] relies on the bound on error terms (Lemma 5.6) from our analysis of PrimalDualBwK. Fourth, the analysis of discretization errors in Badanidiyuru et al. [15] uses a technique from Section 7.

Two recent developments, Agrawal et al. [6] and Agrawal and Devanur [4], concern the contextual version of BwK. Agrawal et al. [6] consider a common generalization of the extended BwK model in Reference [3] and the contextual BwK model in Reference [15]. In particular, for the latter model they achieve the same regret as Badanidiyuru et al. [15], but with a computationally efficient algorithm, resolving the main open question in that article. On a technical level, their work combines ideas from Reference [3] and a recent break-through in contextual bandits [2]. Agrawal and Devanur [4] extend the model in Reference [3] to contextual bandits with a linear dependence on contexts (e.g., see Reference [22]), achieving an algorithm with optimal dependence on the time horizon and the dimensionality of contexts.[5]

Open questions (current status). While the general regret bound in Equation (1) is optimal up to logarithmic factors, better algorithms may be possible for various special cases. To rule out a domain-specific result that improves upon the general regret bound, one would need to prove a lower bound that, unlike the one in Equation (2), is specific to that domain. Currently domain-specific lower bounds are known only for the basic K-armed bandit problem and for dynamic pricing.

For problems with infinite multi-dimensional action spaces, such as dynamic pricing with multiple products and dynamic procurement with multiple budgets, we are limited by the lack of a general approach to upper-bound the discretization error and choose the preadjusted discretization in a principled way. A similar issue arises in the contextual extension of BwK studied in Badanidiyuru et al. [15] and Agrawal et al. [6], even for a single resource constraint. To obtain regret bounds that do not depend on a specific choice of preadjusted discretization, one may need to go beyond preadjusted discretization.

The study of multi-armed bandit problems with large strategy sets has been a very fruitful line of investigation. It seems likely that some of the techniques introduced here could be wedded with the techniques from that literature. In particular, it would be intriguing to try combining our primal-dual algorithm PrimalDualBwK with confidence-ellipsoid algorithms for stochastic linear optimization (e.g., see Dani et al. [23]), or enhancing the BalancedExploration algorithm with the technique of adaptively refined discretization, as in the zooming algorithm of Kleinberg et al. [41].

It is tempting to ask about a version of BwK in which the rewards and resource consumptions are chosen by an adversary. Achieving sublinear regret bounds for this version appears hopeless even for the fixed-arm benchmark. To make progress in the positive direction, one may require a more subtle notion of benchmark and/or restrictions on the power of the adversary.

1.5 Related Work

The study of prior-free algorithms for stochastic MAB problems was initiated by Lai and Robbins [42] and Auer et al. [9]. Subsequent work supplied algorithms for stochastic MAB problems in which the set of arms can be infinite and the payoff function is linear, concave, or Lipschitz-continuous; see a recent survey [20] for more background. Confidence bound techniques have been an integral part of this line of work, and they remain integral to ours.

[5] Agrawal and Devanur [3] prove a similar result for a special case when contexts do not change over time. They also claimed an extension to time-varying contexts, which has subsequently been retracted (see Footnote 1 in Agrawal and Devanur [4]).

As explained earlier, stochastic MAB problems constitute a very special case of bandits with knapsacks, in which there is only one type of resource and it is consumed deterministically at rate 1. Several articles have considered the natural generalization in which there is a single resource (other than time), with deterministic consumption, but different arms consume the resource at different rates. Guha and Munagala [33] gave a constant-factor approximation algorithm for the Bayesian case of this problem, which was later generalized by Gupta et al. [34] to settings in which the arms' reward processes need not be martingales. Tran-Thanh et al. [53, 54] presented prior-free algorithms for this problem; the best such algorithm achieves a regret guarantee qualitatively similar to that of the UCB1 algorithm.

Several recent articles study models that, in hindsight, can be cast as special cases of BwK:

- The two articles [53, 54] mentioned above and Ding et al. [27] consider models with a single resource and unlimited time.
- Dynamic pricing with limited supply has been studied in [12, 16, 17, 56].[6]
- The basic version of dynamic procurement (as per Section 1.1) has been studied in [13, 51].[7] More background on the connection to crowdsourcing can be found in the survey Slivkins and Vaughan [52].
- Dynamic ad allocation (without budget constraints) and various extensions thereof that incorporate user/webpage context have received a considerable attention, starting with [43, 47, 48]. In fact, the connection to pay-per-click advertising has been one of the main drivers for the recent surge of interest in MAB.
- [7, 55] study repeated bidding on a budget, and Cesa-Bianchi et al. [21] study adjusting a repeated auction (albeit without inventory constraints); see Section 8 for more details on these special cases.
- Perhaps the earliest article on resource consumption in MAB is György et al. [35]. They consider a contextual bandit model where the only resource is time, consumed at different rate depending on the context and the chosen arm. The restriction to a single context is a special case of BwK.

Preadjusted discretization has been used in prior work on MAB on metric spaces (e.g., References [36, 37, 41, 45]) and dynamic pricing (e.g., References [12, 16, 18, 39]). However, bounding the discretization error in BwK is much more difficult.

Our BalancedExploration algorithm extends the "active arms elimination" algorithm [29] for the stochastic MAB problem, where one iterates over arms that are not obviously suboptimal given the current confidence bounds. The novelty is that our algorithm chooses over *mixtures* of arms, and the choice is "balanced" across arms. "Policy elimination" algorithm of Dudíik et al. [28] extends "active arms elimination" in a different direction: to contextual bandits. Like BalancedExploration, policy elimination algorithm makes a "balanced" choice among objects that are more complicated than arms, and this choice is not computationally efficient; however, the technical details are very different.

While BwK is primarily an online learning problem, it also has elements of a stochastic packing problem. The literature on prior-free algorithms for stochastic packing has flourished in recent years, starting with prior-free algorithms for the stochastic AdWords problem [25],

[6] The earlier articles [18, 39] focus on the special case of unlimited supply. While we only cited articles that pursue regret-minimizing formulation of dynamic pricing, Bayesian and parametric formulations versions have a rich literature in Operations Research and Economics; see Boer [19] for a literature review.

[7] The regret bound in Reference [51] is against the best-fixed-price benchmark, which may be much smaller than OPT, see Appendix A for a simple example. Benchmarks aside, one cannot directly compare our regret bound and theirs, because they do not derive a worst-case regret bound. Reference [51] is simultaneous work w.r.t. our conference publication.

and continuing with a series of articles extending these results from AdWords to more general stochastic packing integer programs while also achieving stronger performance guarantees [5, 26, 30, 46]. A running theme of these articles (and also of the primal-dual algorithm in this article) is the idea of estimating of an optimal dual vector from samples, then using this dual to guide subsequent primal decisions. Particularly relevant to our work is the algorithm of Devanur et al. [26], in which the dual vector is adjusted using multiplicative updates, as we do in our algorithm. However, unlike the BwK problem, the stochastic packing problems considered in prior work are not learning problems: they are full information problems in which the costs and rewards of decisions in the past and present are fully known. (The only uncertainty is about the future.) As such, designing algorithms for BwK requires a substantial departure from past work on stochastic packing. Our primal-dual algorithm depends upon a hybrid of confidence-bound techniques from online learning and primal-dual techniques from the literature on solving packing LPs; combining them requires entirely new techniques for bounding the magnitude of the error terms that arise in the analysis. Moreover, our BalancedExploration algorithm manages to achieve strong regret guarantees without even computing a dual solution.

2 PRELIMINARIES

BwK: problem formulation. There is a fixed and known, finite set of m *arms* (possible actions), denoted X. There are d resources being consumed. The time proceeds in T rounds, where T is a finite, known time horizon. In each round t, an algorithm picks an arm $x_t \in X$, receives reward $r_t \in [0,1]$, and consumes some amount $c_{t,i} \in [0,1]$ of each resource i. The values r_t and $c_{t,i}$ are revealed to the algorithm after the round. There is a hard constraint $B_i \in \mathbb{R}_+$ on the consumption of each resource i; we call it a *budget* for resource i. The algorithm stops at the earliest time τ when one or more budget constraint is violated; its total reward is equal to the sum of the rewards in all rounds strictly preceding τ. The goal of the algorithm is to maximize the expected total reward.

The vector $(r_t; c_{t,1}, c_{t,2}, \ldots, c_{t,d}) \in [0,1]^{d+1}$ is called the *outcome vector* for round t. We assume *stochastic outcomes*: if an algorithm picks arm x, the outcome vector is chosen independently from some fixed distribution π_x over $[0,1]^{d+1}$. The distributions π_x, $x \in X$ are not known to the algorithm. The tuple $(\pi_x : x \in X)$ comprises all latent information in the problem instance. A particular BwK setting (such as "dynamic pricing with limited supply") is defined by the set of all feasible tuples $(\pi_x : x \in X)$. This set, called the *BwK domain*, is known to the algorithm.

We compare the performance of our algorithms to the expected total reward of the optimal dynamic policy given all the latent information, which we denote by OPT. (Note that OPT depends on the latent information, and therefore is a latent quantity itself.) *Regret* is defined as OPT minus the expected total reward of the algorithm.

W.l.o.g. assumptions. For technical convenience, we make several assumptions that are w.l.o.g.

We express the time horizon as a resource constraint: we model time as a specific resource, say resource 1, such that every arm deterministically consumes B_1/T units of this resource whenever it is picked. W.l.o.g., $B_i \leq T$ for every resource i.

We assume there exists an arm, called the *null arm*, which yields no reward and no consumption of any resource other than time. Equivalently, an algorithm is allowed to spend a unit of time without doing anything. Any algorithm ALG that uses the null arm can be transformed, without loss in expected total reward, to an algorithm ALG′ that does not use the null arm. Indeed, in each round ALG′ runs ALG until it selects a non-null arm x or halts. In the former case, ALG′ selects x and returns the observe feedback to ALG. After ALG halts, ALG′ selects arms arbitrarily.

We say that the budgets are *uniform* if $B_i = B$ for each resource i. Any BwK instance can be reduced to one with uniform budgets by dividing all consumption values for every resource i by

B_i/B, where $B = \min_i B_i$. (That is tantamount to changing the units in which we measure consumption of resource i.) Our technical results are for BwK with uniform budgets. We will assume uniform budgets B from here on.

Useful notation. Let $\mu_x = \mathbb{E}[\pi_x] \in [0,1]^{d+1}$ be the expected outcome vector for each arm x, and denote $\mu = (\mu_x : x \in X)$. We call μ the *latent structure* of a problem instance. The BwK domain induces a set of feasible latent structures, which we denote $\mathcal{M}_{\texttt{feas}}$.

For notational convenience, we will write $\mu_x = (r(x,\mu);\ c_1(x,\mu), \ldots, c_d(x,\mu))$. Also, we will write the expected consumption as a vector $c(x,\mu) = (c_1(x,\mu), \ldots, c_d(x,\mu))$.

If $\mathcal{D}$ is a distribution over arms, then let $r(\mathcal{D},\mu) = \sum_{x\in X} \mathcal{D}(x)\, r(x,\mu)$ and $c(\mathcal{D},\mu) = \sum_{x\in X} \mathcal{D}(x)\, c(x,\mu)$ be, respectively, the expected reward and expected resource consumption in a single round if an arm is sampled from distribution $\mathcal{D}$. Let $\texttt{REW}(\mathcal{D},\mu)$ denote the expected total reward of the time-invariant policy that uses distribution $\mathcal{D}$.

High-probability events. We will use the following expression, which we call the *confidence radius.*

$$\texttt{rad}(\nu, N) = \sqrt{\frac{C_{\texttt{rad}}\, \nu}{N}} + \frac{C_{\texttt{rad}}}{N}. \tag{3}$$

Here, $C_{\texttt{rad}} = \Theta(\log(d\,T|X|))$ is a parameter that we will fix later; we will keep it implicit in the notation. The meaning of Equation (3) and $C_{\texttt{rad}}$ is explained by the following tail inequality from References [12, 41].[8]

Theorem 2.1 ([12, 41]). *Consider some distribution with values in* $[0,1]$ *and expectation* ν. *Let* $\widehat{\nu}$ *be the average of* N *independent samples from this distribution. Then,*

$$\Pr\left[\ |\nu - \widehat{\nu}| \le \texttt{rad}(\widehat{\nu}, N) \le 3\,\texttt{rad}(\nu, N)\ \right] \ge 1 - e^{-\Omega(C_{\texttt{rad}})}, \quad \textit{for each } C_{\texttt{rad}} > 0. \tag{4}$$

More generally, Equation (4) holds if $X_1, \ldots, X_N \in [0,1]$ *are random variables,* $\widehat{\nu} = \frac{1}{N}\sum_{t=1}^{N} X_t$ *is the sample average, and* $\nu = \frac{1}{N}\sum_{t=1}^{N} \mathbb{E}[X_t \mid X_1, \ldots, X_{t-1}]$.

If the expectation ν is a latent quantity, then Equation (4) allows us to estimate ν by a high-confidence interval,

$$\nu \in [\widehat{\nu} - \texttt{rad}(\widehat{\nu}, N),\ \widehat{\nu} + \texttt{rad}(\widehat{\nu}, N)], \tag{5}$$

whose endpoints are observable (known to the algorithm). This estimate is on par with the one provided by Azuma-Hoeffding inequality (up to constant factors), but is much sharper for small ν.[9]

It is sometimes useful to argue about *any* ν that lies in the high-confidence interval Equation (5), not just the latent $\nu = \mathbb{E}[\widehat{\nu}]$. We use the following claim, which is implicit in Kleinberg et al. [41].

Claim 2.2 ([41]). *For any* $\nu, \widehat{\nu} \in [0,1]$, *Equation (5) implies that* $\texttt{rad}(\widehat{\nu}, N) \le 3\,\texttt{rad}(\nu, N)$.

3 LP RELAXATION FOR POLICY VALUE

OPT—the expected reward of the optimal dynamic policy given foreknowledge of the distribution of outcome vectors—is typically difficult to characterize exactly. In fact, even for a time-invariant policy, it is difficult to give an exact expression for the expected reward due to the dependence of the reward on the random stopping time when the resource budget is exhausted. To approximate these quantities, we consider the fractional relaxation of BwK in which the number of rounds in

[8] Specifically, this follows from Lemma 4.9 in the full version of Kleinberg et al. [41], and Theorem 4.8 and Theorem 4.10 in the full version of Babaioff et al. [12] (both full versions can be found on arxiv.org).

[9] Essentially, Azuma-Hoeffding inequality states that $|\nu - \widehat{\nu}| \le O(\sqrt{C_{\texttt{rad}}/N})$, whereas by Theorem 2.1 for small ν it holds with high probability that $\texttt{rad}(\widehat{\nu}, N) \sim C_{\texttt{rad}}/N$.

which a given arm is selected (and also the total number of rounds) can be fractional, and the reward and resource consumption per unit time are deterministically equal to the corresponding expected values in the original instance of BwK.

The following linear program constitutes our fractional relaxation of the optimal dynamic policy:

$$\begin{array}{llcll} \max & \sum_{x\in X} \xi_x\, r(x,\mu) & & & \text{in } \xi_x \in \mathbb{R}, \text{ for each } x \in X \\ \text{s.t.} & \sum_{x\in X} \xi_x\, c_i(x,\mu) & \leq & B & \text{for each resource } i \\ & \xi_x & \geq & 0 & \text{for each arm } x. \end{array} \tag{LP-primal}$$

The variables ξ_x represent the fractional relaxation for the number of rounds in which a given arm x is selected. This is a bounded LP (because $\sum_x \xi_x\, r(x,\mu) \leq \sum_x \xi_x \leq T$). The optimal value of this LP is denoted by $\mathtt{OPT}_{\mathtt{LP}}$. We will also use the dual LP, shown below.

$$\begin{array}{llcll} \min & B\sum_i \eta_i & & & \text{in } \eta_i \in \mathbb{R}, \text{ for each resource } i \\ \text{s.t.} & \sum_i \eta_i\, c_i(x,\mu) & \geq & r(x,\mu) & \text{for each arm } x \in X \\ & \eta_i & \geq & 0 & \text{for each resource } i. \end{array} \tag{LP-dual}$$

The dual variables η_i can be interpreted as a unit cost for the corresponding resource i.

Lemma 3.1. $\mathtt{OPT}_{\mathtt{LP}}$ *is an upper bound on the value of the optimal dynamic policy:* $\mathtt{OPT}_{\mathtt{LP}} \geq \mathtt{OPT}$.

One way to prove this lemma is to define ξ_x to be the expected number of times arm x is played by the optimal dynamic policy, and argue that the vector $(\xi_x, x \in X)$ is primal-feasible and that $\sum_x \xi_x\, r(x,\mu)$ is the expected reward of the optimal dynamic policy. We instead present a simpler proof using (LP-dual) and a martingale argument. A similar lemma (but for a technically different setting of online stochastic packing problems) was proved in Devanur et al. [26].

Proof of Lemma 3.1. Let $\eta^* = (\eta_1^*, \ldots, \eta_d^*)$ denote an optimal solution to (LP-dual). Interpret each η_i^* as a unit cost for the corresponding resource i. By strong LP duality, we have $B\sum_i \eta_i^* = \mathtt{OPT}_{\mathtt{LP}}$. Dual feasibility implies that for each arm x, the expected cost of resources consumed when x is pulled exceeds the expected reward produced. Thus, if we let Z_t denote the sum of rewards gained in rounds $1, \ldots, t$ of the optimal dynamic policy, plus the cost of the remaining resource endowment after round t, then the stochastic process $Z_0, Z_1, \ldots, Z_T$ is a supermartingale. Let τ be the stopping time of the algorithm, i.e., the total number of rounds. Note that $Z_0 = B\sum_i \eta_i^* = \mathtt{OPT}_{\mathtt{LP}}$, and $Z_{\tau-1}$ equals the algorithm's total payoff, plus the cost of the remaining (non-negative) resource supply at the start of round τ. By Doob's optional stopping theorem, $Z_0 \geq \mathbb{E}[Z_{\tau-1}]$ and the lemma is proved. □

Remark 3.2. Implicit in this proof is a simple, but powerful observation that for any algorithm,

$$\mathtt{OPT}_{\mathtt{LP}} - \mathtt{REW} \geq \mathbb{E}\left[\sum_t r(x_t,\mu) - c(x_t,\mu)\cdot\eta^*\right].$$

Each summand on the right-hand side is non-negative, and equals 0 if and only if the arm x_t lies in the support of the primal solution. We use this observation to motivate the design of our primal-dual algorithm.

Remark 3.3. For each of the two main algorithms, we prove a regret bound of the form

$$\mathtt{OPT}_{\mathtt{LP}} - \mathtt{REW} \leq f(\mathtt{OPT}_{\mathtt{LP}}), \tag{6}$$

where $\mathtt{REW}$ is the expected total reward of the algorithm, and $f()$ depends only on parameters (B, m, d). This regret bound has an optimal scaling property, highlighted in the Introduction: if all budget constraints, including the time horizon, are increased by the factor of α, then the regret bound $f(\mathtt{OPT_{LP}})$ scales as $\sqrt{\alpha}$.

Regret bound Equation (6) implies the claimed regret bounds relative to $\mathtt{OPT}$, because

$$\mathtt{REW} \geq \mathtt{OPT_{LP}} - f(\mathtt{OPT_{LP}}) \geq \mathtt{OPT} - f(\mathtt{OPT}), \tag{7}$$

where the second inequality follows trivially, because $g(x) = \max(x - f(x), 0)$ is a non-decreasing function of x for $x \geq 0$, and $\mathtt{OPT_{LP}} \geq \mathtt{OPT}$.

Let us apply a similar LP-relaxation to a time-invariant policy that uses distribution $\mathcal{D}$ over arms. We approximate the expected total reward of this policy in a similar way: we define a linear program in which the only variable t represents the expected stopping time of the algorithm.

$$\begin{array}{llcll} \max & t\, r(\mathcal{D}, \mu) & & & \text{in } t \in \mathbb{R} \\ \text{s.t.} & t\, c_i(\mathcal{D}, \mu) & \leq & B & \text{for each resource } i \\ & t & \geq & 0. & \end{array} \tag{LP-distr}$$

The optimal value to (LP-distr), which we call the *LP-value* of $\mathcal{D}$, is

$$\mathtt{LP}(\mathcal{D}, \mu) = r(\mathcal{D}, \mu)\ \min_i \frac{B}{c_i(\mathcal{D}, \mu)}. \tag{8}$$

Observe that t is feasible for (LP-distr) if and only if $\xi = t\mathcal{D}$ is feasible for (LP-primal). Therefore,

$$\mathtt{OPT_{LP}} = \sup_{\mathcal{D}} \mathtt{LP}(\mathcal{D}, \mu).$$

This supremum is attained by any distribution $\mathcal{D}^* = \xi / \|\xi\|_1$ such that $\xi = (\xi_x : x \in X)$ is an optimal solution to (LP-primal). A distribution $\mathcal{D}^* \in \operatorname{argmax}_{\mathcal{D}} \mathtt{LP}(\mathcal{D}, \mu)$ is called *LP-optimal* for μ.

Claim 3.4. *For any latent structure μ, there exists a distribution $\mathcal{D}$ over arms that is LP-optimal for μ and moreover satisfies the following three properties:*

(a) $c_i(\mathcal{D}, \mu) \leq B/T$ for each resource i.
(b) $\mathcal{D}$ has a support of size at most d.
(c) If $\mathcal{D}$ has a support of size exactly 2, then for some resource i, we have $c_i(\mathcal{D}, \mu) = B/T$.

(Such distribution $\mathcal{D}$ will be called ***LP-perfect*** *for μ.)*

Proof. Fix the latent structure μ. It is a well-known fact that for any linear program there exists an optimal solution whose support has size that is exactly equal to the number of constraints that are tight for this solution. Take any such optimal solution $\xi = (\xi_x : x \in X)$ for (LP-primal), and take the corresponding LP-optimal distribution $\mathcal{D} = \xi / \|\xi\|_1$. Since there are d constraints in (LP-primal), distribution $\mathcal{D}$ has support of size at most d. If it satisfies (a), then it also satisfies (c) (else it is not optimal), and we are done.

Suppose property (a) does not hold for $\mathcal{D}$. Then there exists a resource i such that $c_i(\mathcal{D}, \mu) > B/T$. Since the ith constraint in (LP-primal) can be restated as $\|\xi\|_1\, c_i(\mathcal{D}, \mu) \leq B$, it follows that $\|\xi\|_1 < T$. Therefore, the constraint in (LP-primal) that expresses the time horizon is not tight. Consequently, at most $d - 1$ constraints in (LP-primal) are tight for ξ, so the support of $\mathcal{D}$ has size at most $d - 1$.

Let us modify $\mathcal{D}$ to obtain another LP-optimal distribution $\mathcal{D}'$ that satisfies properties (a-c). W.l.o.g., pick i to maximize $c_i(\mathcal{D}, \mu)$ and let $\alpha = \frac{B}{T} / c_i(\mathcal{D}, \mu)$. Define $\mathcal{D}'(x) = \alpha\, \mathcal{D}(x)$ for each non-null arm x and place the remaining probability in $\mathcal{D}'$ on the null arm. This completes the definition of $\mathcal{D}'$.

Note that $c_j(\mathcal{D}', \mu) = \alpha\, c_j(\mathcal{D}, \mu) \leq B/T$ for each resource j, with equality for $j = i$. Hence, $\mathcal{D}'$ satisfies properties (a) and (c). Also, $r(\mathcal{D}', \mu) = \alpha\, r(\mathcal{D}, \mu)$, and so

$$\mathsf{LP}(\mathcal{D}', \mu) = r(\mathcal{D}', \mu)\ \frac{B}{c_i(\mathcal{D}', \mu)} = r(\mathcal{D}, \mu)\ \frac{B}{c_i(\mathcal{D}, \mu)} = \mathsf{LP}(\mathcal{D}, \mu).$$

Thus, $\mathcal{D}'$ is LP-optimal. It satisfies property (b), because it adds at most 1 to the support of $\mathcal{D}$. □

4 ALGORITHM BALANCEDEXPLORATION

This section presents and analyzes BalancedExploration, one of the two main algorithms. The design principle behind BalancedExploration is to explore as much as possible while avoiding obviously suboptimal strategies. On a high level, the algorithm is very simple. The goal is to converge on an LP-perfect distribution. The time is divided into phases of $|X|$ rounds each. In the beginning of each phase p, the algorithm prunes away all distributions $\mathcal{D}$ over arms that with high confidence are not LP-perfect given the observations so far. The remaining distributions over arms are called *potentially perfect*. Throughout the phase, the algorithm chooses among the potentially perfect distributions. Specifically, for each arm x, the algorithm chooses a potentially perfect distribution $\mathcal{D}_{p,x}$, which approximately maximizes $\mathcal{D}_{p,x}(x)$, and "pulls" an arm sampled independently from this distribution. This choice of $\mathcal{D}_{p,x}$ is crucial; we call it the *balancing step*. The algorithm halts as soon as the time horizon is met, or any of the constraints is exhausted. The pseudocode is given in Algorithm 1.

ALGORITHM 1: BalancedExploration

```
For each phase $p = 0, 1, 2, \ldots$ do
    Recompute the set $\Delta_p$ of potentially perfect distributions $\mathcal{D}$ over arms.
    Over the next $|X|$ rounds, for each $x \in X$:
        pick any distribution $\mathcal{D} = \mathcal{D}_{p,x} \in \Delta_p$ such that $\mathcal{D}(x) \geq \frac{1}{2} \max_{\mathcal{D}' \in \Delta_p} \mathcal{D}'(x)$.
        choose an arm to "pull" as an independent sample from $\mathcal{D}$.
        halt if time horizon is met or one of the resources is exhausted.
```

We believe that BalancedExploration, like UCB1 [9], is a very general design principle and has the potential to be a meta-algorithm for solving stochastic online learning problems.

THEOREM 4.1. *Consider an instance of* BwK *with d resources, $m = |X|$ arms, and the smallest budget $B = \min_i B_i$. Algorithm* BalancedExploration *achieves regret*

$$\mathsf{OPT}_{\mathsf{LP}} - \mathsf{REW} \leq O(\log(T)\log(T/m)) \left(\sqrt{dm\mathsf{OPT}_{\mathsf{LP}}} + \mathsf{OPT}_{\mathsf{LP}}\sqrt{\frac{dm}{B}}\right). \tag{9}$$

Moreover, Equation (7) holds with $f(\mathsf{OPT}_{\mathsf{LP}})$ equal to the right-hand side of Equation (9).

Remark 4.2. The specification of BalancedExploration involves a mathematically well-defined step—approximate optimization over potentially perfect distributions—for which we do not provide a specific implementation. Yet, BalancedExploration is a bandit algorithm in the sense that it is a well-defined mapping from histories to actions. We prove an "information-theoretic" statement: there is an algorithm with the claimed regret. Such results are not uncommon in the literature, e.g., References [2, 40, 41], typically as first solutions for new, broad problem formulations, and are meaningful as proof-of-concept for the corresponding regret bounds and techniques.

Remaining details of the specification. In the beginning of each phase p, the algorithm recomputes a "confidence interval" I_p for the latent structure μ, so that (informally) $\mu \in I_p$ with high probability. Then the algorithm determines which distributions $\mathcal{D}$ over arms can potentially be LP-perfect given that $\mu \in I_p$. Specifically, let Δ_p be set of all distributions $\mathcal{D}$ that are LP-perfect for some latent structure $\mu' \in I_p$; such distributions are called *potentially perfect* (for phase p).

It remains to define the confidence intervals I_p. For phase $p = 0$, the confidence interval I_0 is simply $\mathcal{M}_{\texttt{feas}}$, the set of all feasible latent structures. For each subsequent phase $p \geq 1$, the confidence interval I_p is defined as follows. For each arm x, consider all rounds before phase p in which this arm has been chosen. Let $N_p(x)$ be the number of such rounds, let $\widehat{r}_p(x)$ be the time-averaged reward in these rounds, and let $\widehat{c}_{p,i}(x)$ be the time-averaged consumption of resource i in these rounds. We use these averages to estimate $r(x, \mu)$ and $c_i(x, \mu)$ as follows:

$$|r(x,\mu) - \widehat{r}_p(x)| \leq \texttt{rad}(\widehat{r}_p(x), N_p(x)), \tag{10}$$

$$|c_i(x,\mu) - \widehat{c}_{p,i}(x)| \leq \texttt{rad}(\widehat{c}_{p,i}(x), N_p(x)), \quad \text{for each resource } i. \tag{11}$$

The confidence interval I_p is the set of all latent structures $\mu' \in I_{p-1}$ that are consistent with these estimates. This completes the specification of `BalancedExploration`.

For each phase of `BalancedExploration`, the round in which an arm is sampled from distribution $\mathcal{D}_{p,x}$ will be called *designated* to arm x. We need to use approximate maximization to choose $\mathcal{D}_{p,x}$, rather than exact maximization, because an exact maximizer $\text{argmax}_{\mathcal{D}\in\Delta_p} \mathcal{D}(x)$ is not guaranteed to exist.

Proof overview. We start with some properties of the algorithm that follow immediately from the specification and hold deterministically (with probability 1). Then, we identify several properties that the algorithm satisfies with very high probability. The rest of the analysis focuses on a "clean execution" of the algorithm: an execution in which all these properties hold. We analyze the "error terms" that arise due to the uncertainty on the latent structure, and use the resulting "error bounds" to argue about the algorithm's performance.

4.1 Deterministic Properties of `BalancedExploration`

First, we show that any two latent structures in the confidence interval I_p correspond to similar consumptions and rewards, for each arm x. This follows deterministically from the specification of I_p.

Claim 4.3. *Fix any phase p, any two latent structures $\mu', \mu'' \in I_p$, an arm x, and a resource i. Then,*

$$|c_i(x,\mu') - c_i(x,\mu'')| \leq 6\ \texttt{rad}(c_i(x,\mu'), N_p(x)), \tag{12}$$

$$|r(x,\mu') - r(x,\mu'')| \leq 6\ \texttt{rad}(r(x,\mu'), N_p(x)). \tag{13}$$

Proof. We prove Equation (12); Equation (13) is proved similarly.

Let $N = N_p(x)$. By specification of `BalancedExploration`, any $\mu' \in I_p$ is consistent with estimate Equation (11):

$$|c_i(x,\mu') - \widehat{c}_{p,i}(x)| \leq \texttt{rad}(\widehat{c}_{p,i}(x), N).$$

It follows that

$$|c_i(x,\mu') - c_i(x,\mu'')| \leq 2\ \texttt{rad}(\widehat{c}_{p,i}(x), N).$$

Finally, we observe that by Claim 2.2,

$$\texttt{rad}(\widehat{c}_{p,i}(x), N) \leq 3\,\texttt{rad}\,(c_i(x,\mu'), N). \qquad \square$$

For each phase p and arm x, let $\bar{\mathcal{D}}_{p,x} = \frac{1}{p}\sum_{q<p} \mathcal{D}_{q,x}(x)$ be the average of probabilities for arm x among the distributions in the preceding phases that are designated to arm x. Because of the balancing step in `BalancedExploration`, we can compare this quantity to $\mathcal{D}(x)$, for any $\mathcal{D} \in \Delta_p$.

(Here, we also use the fact that the confidence intervals I_p are non-increasing from one phase to another.)

Claim 4.4. *$\bar{\mathcal{D}}_{p,x} \geq \frac{1}{2}\,\mathcal{D}(x)$ for each phase p, each arm x and any distribution $\mathcal{D} \in \Delta_p$.*

Proof. Fix arm x. Recall that $\bar{\mathcal{D}}_{p,x} = \frac{1}{p}\sum_{q<p} \mathcal{D}_{q,x}(x)$, where $\mathcal{D}_{q,x}$ is the distribution chosen in the round in phase q that is designated to arm x. Fix any phase $q < p$. Because of the balancing step, $\mathcal{D}_{q,x}(x) \geq \frac{1}{2}\,\mathcal{D}'(x)$ for any distribution $\mathcal{D}' \in \Delta_q$. Since the confidence intervals I_q are non-increasing from one phase to another, we have $I_p \subset I_q$ for any $q \leq p$, which implies that $\Delta_p \subset \Delta_q$. Consequently, $\mathcal{D}_{q,x}(x) \geq \frac{1}{2}\,\mathcal{D}(x)$ for each $q < p$, and the claim follows. □

4.2 High-Probability Events

We keep track of several quantities: the averages $\widehat{r}_p(x)$ and $\widehat{c}_{p,i}(x)$ defined above, as well as several other quantities that we define below.

Fix phase p and arm x. Recall that $N_p(x)$ is the number of rounds before phase p in which arm x is chosen. Now, let us consider all rounds before phase p that are *designated* to arm x. Let $n_p(x)$ denote the number of times arm x has been chosen in these rounds. Let $\widehat{\mathcal{D}}_{p,x} = n_t(x)/p$ be the corresponding empirical probability of choosing x. We compare this to $\bar{\mathcal{D}}_{p,x}$.

Further, consider all rounds in phases $q < p$. There are $N = p|X|$ such rounds. The average distribution chosen by the algorithm in these rounds is $\bar{\mathcal{D}}_p = \frac{1}{N}\sum_{q<p,\, x\in X} \mathcal{D}_{q,x}$. We are interested in the corresponding quantities $r(\bar{\mathcal{D}}_p, \mu)$ and $c_i(\bar{\mathcal{D}}_p, \mu)$, We compare these quantities to $\widehat{r}_p = \frac{1}{N}\sum_{t=1}^{N} r_t$ and $\widehat{c}_{p,i} = \frac{1}{N}\sum_{t=1}^{N} c_{t,i}$, the average reward and the average resource-i consumption in phases $q < p$.

We consider several high-probability events that follow from applying Theorem 2.1 to the various quantities defined above. All these events have a common shape: some quantities $v, \widehat{v}$ satisfy Equation (5) for some N. If this is the case, then we know that $\widehat{v}$ is an N*-strong estimator* for v.

Lemma 4.5. *For each phase p, arm x, and resource i, with probability $e^{-\Omega(C_{\mathrm{rad}})}$ it holds that:*

- *(a) $\widehat{r}_p(x)$ and $\widehat{c}_{p,i}(x)$ are $N_p(x)$-strong estimators for $r(x,\mu)$ and $c_i(x,\mu)$, respectively.*
- *(b) $\bar{\mathcal{D}}_{p,x}$ is an p-strong estimator for $\widehat{\mathcal{D}}_{p,x}$.*
- *(c) $r(\bar{\mathcal{D}}_p,\ \mu)$ and $c_i(\bar{\mathcal{D}}_p,\ \mu)$ are $(p|X|)$-strong estimators for $\widehat{r}_p$ and $\widehat{c}_{p,i}$, respectively.*

We rely on several properties of the confidence radius $\mathtt{rad}()$, which we summarize below. (We omit the easy proofs.)

Claim 4.6. *The confidence radius $\mathtt{rad}(v,N)$, defined in Equation (3), satisfies the following properties:*

- *(a) monotonicity: $\mathtt{rad}(v,N)$ is non-decreasing in v and non-increasing in N.*
- *(b) concavity: $\mathtt{rad}(v,N)$ is concave in v, for any fixed N.*
- *(c) $\max(0,\ v - \mathtt{rad}(v,N))$ is non-decreasing in v.*
- *(d) $v - \mathtt{rad}(v,N) \geq \frac{1}{4}\,v$ whenever $4\frac{C_{\mathrm{rad}}}{N} \leq v \leq 1$.*
- *(e) $\mathtt{rad}(v,N) \leq 3\frac{C_{\mathrm{rad}}}{N}$ whenever $v \leq 4\frac{C_{\mathrm{rad}}}{N}$.*
- *(f) $\mathtt{rad}(v,\alpha N) = \frac{1}{\alpha}\,\mathtt{rad}(\alpha v,\ N)$, for any $\alpha \in (0,1]$.*
- *(g) $\frac{1}{N}\sum_{\ell=1}^{N} \mathtt{rad}(v,\ell) \leq O(\log N)\ \mathtt{rad}(v,N)$.*

4.3 Clean Execution Analysis

It is convenient to focus on a *clean execution* of the algorithm: an execution in which all events in Lemma 4.5 hold. We assume a clean execution in what follows. Also, we fix an arbitrary phase p in such execution.

Clean execution analysis falls into two parts. First, we analyze the "error terms": we look at the LP-value (respectively, expected reward, or expected resource consumption) of a given distribution, and upper-bound the difference in this quantity between different latent structures μ, μ' in the confidence interval I_p, or between different potentially perfect distributions $D', D'' \in \Delta_p$. The culmination is Lemma 4.12, which upper-bounds the difference $|\mathsf{LP}(\mathcal{D}', \mu') - \mathsf{LP}(\mathcal{D}'', \mu'')|$ in terms of parameters $\frac{p}{d}$, B, T, and $\mathtt{OPT}_{\mathtt{LP}}$. Second, we apply these error bounds to reason about the algorithm itself. The key quantities of interest are LP-values of the chosen distributions, average reward/consumption, and the stopping time.

4.3.1 Bounding the Error Terms. Since a clean execution satisfies the event in Claim 4.5(a), it immediately follows that:

Claim 4.7. *The confidence interval I_p contains the (actual) latent structure μ. Therefore, $\mathcal{D}^* \in \Delta_p$ for any distribution $\mathcal{D}^*$ that is LP-perfect for μ.*

Claim 4.8. *Fix any latent structures $\mu', \mu'' \in I_p$ and distribution $\mathcal{D} \in \Delta_p$. Then, for each resource i,*

$$|c_i(\mathcal{D}, \mu') - c_i(\mathcal{D}, \mu'')| \le O(1)\ \mathtt{rad}\,(c_i(\mathcal{D}, \mu'),\ p/d), \tag{14}$$

$$|r(\mathcal{D}, \mu') - r(\mathcal{D}, \mu'')| \le O(1)\ \mathtt{rad}\,(r(\mathcal{D}, \mu'),\ p/d). \tag{15}$$

Proof. We prove Equation (14); Equation (15) is proved similarly. Let us first prove the following:

$$\forall x \in X, \quad \mathcal{D}(x)\,|c_i(x, \mu') - c_i(x, \mu'')| \le O(1)\ \mathtt{rad}(\mathcal{D}(x)\, c_i(x, \mu'), p). \tag{16}$$

Intuitively, to argue that we have good estimates on quantities related to arm x, it helps to prove that this arm has been chosen sufficiently often. Using the definition of clean execution and Claim 4.4, we accomplish this as follows:

$$\begin{aligned} \frac{1}{p}\, N_p(x) \ge \frac{1}{p}\, n_p(x) &= \widehat{\mathcal{D}}_{p,x} \\ &\ge \bar{\mathcal{D}}_{p,x} - \mathtt{rad}(\bar{\mathcal{D}}_{p,x}, p) && \text{(by clean execution)} \\ &\ge \frac{1}{2}\,\mathcal{D}(x) - \mathtt{rad}\left(\frac{1}{2}\,\mathcal{D}(x), p\right) && \text{(by Claim 4.4 and Claim 4.6(c)).} \end{aligned}$$

Consider two cases depending on $\mathcal{D}(x)$. For the first case, assume $\mathcal{D}(x) \ge 8\frac{C_{\mathtt{rad}}}{p}$. Using Claim 4.6(d) and the previous equation, it follows that $N_p(x) \ge \frac{1}{8}\, p\, \mathcal{D}(x)$. Therefore:

$$\begin{aligned} \mathcal{D}(x)\ |c_i(x, \mu') - c_i(x, \mu'')| &\le 6\ \mathcal{D}(x)\,\mathtt{rad}(c_i(x, \mu'),\ N_p(x)) && \text{(by Claim 4.3)} \\ &\le 6\ \mathcal{D}(x)\,\mathtt{rad}(c_i(x, \mu'),\ \frac{1}{8}\, p\, \mathcal{D}(x)) && \text{(by monotonicity of } \mathtt{rad}\text{)} \\ &= 48\ \mathtt{rad}(\mathcal{D}(x)\, c_i(x, \mu'_x), p) && \text{(by Claim 4.6(f)).} \end{aligned}$$

The second case is that $\mathcal{D}(x) < 8\frac{C_{\mathtt{rad}}}{p}$. Then Equation (16) follows simply because $\frac{C_{\mathtt{rad}}}{p} \le \mathtt{rad}(\cdot\,, p)$.

We have proved Equation (16). We complete the proof of Equation (14) using concavity of $\mathtt{rad}(\cdot, p)$ and the fact that, by the specification of $\mathtt{BalancedExploration}$, $\mathcal{D}$ has support of size at most d.

$$\begin{aligned}|c_i(\mathcal{D},\mu') - c_i(\mathcal{D},\mu'')| &\le \sum_{x\in X} \mathcal{D}(x)\, |c_i(x,\mu') - c_i(x,\mu'')| \\ &\le \sum_{x\in X,\, \mathcal{D}(x)>0} O(1)\ \mathsf{rad}(\mathcal{D}(x)\, c_i(x,\mu'), p) \\ &\le O(d)\ \mathsf{rad}\left(\frac{1}{d}\sum_{x\in X} \mathcal{D}(x)\, c_i(x,\mu'),\ p\right) \\ &= O(d)\ \mathsf{rad}\left(\frac{1}{d}\, c_i(\mathcal{D},\mu'),\ p\right) \\ &\le O(1)\ \mathsf{rad}\left(c_i(\mathcal{D},\mu'),\ \frac{p}{d}\right) \qquad \text{(by Claim 4.6(f)).}\end{aligned}$$ □

In what follows, we will denote $\mathfrak{M}_p = \max_{\mathcal{D}\in\Delta_p,\ \mu\in I_p} \mathsf{LP}(\mathcal{D},\mu)$.

Claim 4.9. *Fix any latent structures $\mu', \mu'' \in I_p$ and any distribution $\mathcal{D} \in \Delta_p$. Then,*

$$|\mathsf{LP}(\mathcal{D},\mu') - \mathsf{LP}(\mathcal{D},\mu'')| \le O(T)\,\mathsf{rad}\left(\mathfrak{M}_p/T,\ \frac{p}{d}\right) + O\left(\mathfrak{M}_p \frac{T}{B}\right)\ \mathsf{rad}\left(\frac{B}{T}, \frac{p}{d}\right). \tag{17}$$

Proof. Since $\mathcal{D} \in \Delta_p$, it is LP-perfect for some latent structure μ. Then $\mathsf{LP}(\mathcal{D},\mu) = T\, r(\mathcal{D},\mu)$. Therefore:

$$\begin{aligned}\mathsf{LP}(\mathcal{D},\mu') - \mathsf{LP}(\mathcal{D},\mu) \le&\ \ T\ \ (r(\mathcal{D},\mu') - r(\mathcal{D},\mu)) \\ \le&\ \ O(T)\ \mathsf{rad}\left(r(\mathcal{D},\mu), \frac{p}{d}\right) \qquad \text{(by Claim 4.8).}\end{aligned} \tag{18}$$

We need a little more work to bound the difference in the LP values in the other direction.

Consider $t_0 = \mathsf{LP}(\mathcal{D},\mu')/r(\mathcal{D},\mu')$; this is the value of the variable t in the optimal solution to the linear program (LP-distr). Let us obtain a lower bound on this quantity. Assume $t_0 < T$. Then one of the budget constraints in (LP-distr) must be tight, i.e., $t_0\, c_i(\mathcal{D},\mu') = B$ for some resource i.

$$\begin{aligned}c_i(\mathcal{D},\mu') &\le c_i(\mathcal{D},\mu) + O(1)\,\mathsf{rad}\left(c_i(\mathcal{D},\mu), \frac{p}{d}\right) \qquad \text{(by Claim 4.8)} \\ &\le \frac{B}{T} + O(1)\,\mathsf{rad}\left(\frac{B}{T}, \frac{p}{d}\right)\end{aligned}$$

Let $\Psi = \mathsf{rad}(\frac{B}{T}, \frac{p}{d})$. It follows that $t_0 = B/c_i(\mathcal{D},\mu') \ge T(1 - O(\frac{T}{B}\,\Psi))$. Therefore:

$$\begin{aligned}\mathsf{LP}(\mathcal{D},\mu) - \mathsf{LP}(\mathcal{D},\mu') =&\ \ T\, r(\mathcal{D},\mu) - t_0\, r(\mathcal{D},\mu') \\ &\le T\, r(\mathcal{D},\mu) - \left[T\left(1 - O\left(\frac{T}{B}\,\Psi\right)\right)\right]\ r(\mathcal{D},\mu') \\ &\le T\ \left[r(\mathcal{D},\mu) - r(\mathcal{D},\mu')\right] + O\left(\frac{T}{B}\,\Psi\right)\, T\, r(\mathcal{D},\mu) \\ &\le O(T)\,\mathsf{rad}\left(r(\mathcal{D},\mu), \frac{p}{d}\right) + O\left(\frac{T}{B}\,\Psi\right)\, T\, r(\mathcal{D},\mu) \qquad \text{(by Claim 4.8).}\end{aligned}$$

Using Equation (18) and noting that $r(\mathcal{D},\mu) = \mathsf{LP}(\mathcal{D},\mu)/T \le \mathfrak{M}_p/T$, we conclude that

$$|\mathsf{LP}(\mathcal{D},\mu) - \mathsf{LP}(\mathcal{D},\mu')| \le O(T)\,\mathsf{rad}\left(\mathfrak{M}_p/T,\ \frac{p}{d}\right) + O(\mathfrak{M}_p \frac{T}{B})\,\mathsf{rad}\left(\frac{B}{T}, \frac{p}{d}\right).$$

We obtain the same upper bound on $|\mathsf{LP}(\mathcal{D},\mu) - \mathsf{LP}(\mathcal{D},\mu'')|$, and the claim follows. □

We will use $\Phi_p(\mathfrak{M}_p)$ to denote the right-hand side of Equation (17) as a function of $\mathfrak{M}_p$.

Claim 4.10.

(a) *Fix any latent structure $\mu^* \in I_p$, and any distributions $\mathcal{D}', \mathcal{D}'' \in \Delta_p$. Then,*

$$|\mathsf{LP}(\mathcal{D}', \mu^*) - \mathsf{LP}(\mathcal{D}'', \mu^*)| \leq 2\Phi_p(\mathfrak{M}_p).$$

(b) *Fix any latent structure $\mu', \mu'' \in I_p$, and any distributions $\mathcal{D}', \mathcal{D}'' \in \Delta_p$. Then,*

$$|\mathsf{LP}(\mathcal{D}', \mu') - \mathsf{LP}(\mathcal{D}'', \mu'')| \leq 3\Phi_p(\mathfrak{M}_p).$$

Proof. **(a).** Since $\mathcal{D}', \mathcal{D}'' \in \Delta_p$, it holds that $\mathcal{D}'$ and $\mathcal{D}''$ are LP-perfect for some latent structures μ' and μ''. Further, pick a distribution $\mathcal{D}^*$ that is LP-perfect for μ^*. Then:

$$\begin{aligned}
\mathsf{LP}(\mathcal{D}', \mu^*) &\geq \mathsf{LP}(\mathcal{D}', \mu') - \Phi_p(\mathfrak{M}_p) && \text{(by Lemma 4.9 with} \mathcal{D} = \mathcal{D}'\text{)}\\
&\geq \mathsf{LP}(\mathcal{D}^*, \mu') - \Phi_p(\mathfrak{M}_p)\\
&\geq \mathsf{LP}(\mathcal{D}^*, \mu^*) - 2\Phi_p(\mathfrak{M}_p) && \text{(by Lemma 4.9 with} \mathcal{D} = \mathcal{D}^*\text{)}\\
&\geq \mathsf{LP}(\mathcal{D}'', \mu^*) - 2\Phi_p(\mathfrak{M}_p).
\end{aligned}$$

(b). Follows easily from part (a) and Lemma 4.9. □

The following claim will allow us to replace $\Phi_p(\mathfrak{M}_p)$ by $\Phi_p(\mathsf{OPT}_{\mathsf{LP}})$.

Claim 4.11. $\Phi_p(\mathsf{OPT}_{\mathsf{LP}}) \geq \Omega(\min(\mathsf{OPT}_{\mathsf{LP}}, \Phi_p(\mathfrak{M}_p)))$.

Proof. Consider the two summands in $\Phi_p(\mathfrak{M}_p)$:

$$\begin{aligned}
S_1(\mathfrak{M}_p) &= O(T)\,\mathsf{rad}\left(\mathfrak{M}_p/T,\ \frac{p}{d}\right),\\
S_2(\mathfrak{M}_p) &= O(\mathfrak{M}_p\,\frac{T}{B})\,\mathsf{rad}\left(\frac{B}{T}, \frac{p}{d}\right).
\end{aligned}$$

We consider the following three cases. The first case is that $S_1(\mathfrak{M}_p) \geq \mathfrak{M}_p/12$. Solving for $\mathfrak{M}_p$, we obtain $\mathfrak{M}_p \leq O(\frac{TdC_{\mathsf{rad}}}{p})$, which implies that

$$\Phi_p(\mathsf{OPT}_{\mathsf{LP}}) \geq \Omega(\mathfrak{M}_p) \geq \Omega(\mathsf{OPT}_{\mathsf{LP}}).$$

The second case is that $S_2(\mathfrak{M}_p) \geq \mathfrak{M}_p/12$. Then,

$$\Phi_p(\mathsf{OPT}_{\mathsf{LP}}) \geq S_2(\mathsf{OPT}_{\mathsf{LP}}) \geq \mathsf{OPT}_{\mathsf{LP}}/12.$$

In remaining case, $\Phi_p(\mathfrak{M}_p) \leq \frac{\mathfrak{M}_p}{6}$. Then from Claim 4.10(b), we get that $\mathfrak{M}_p \leq 2\,\mathsf{OPT}_{\mathsf{LP}}$. Noting that $\Phi_p(M)$ is a non-decreasing function of M, we obtain

$$\Phi_p(\mathfrak{M}_p) \leq \Phi_p(2\,\mathsf{OPT}_{\mathsf{LP}}) \leq 2\,\Phi_p(\mathsf{OPT}_{\mathsf{LP}}). \qquad \square$$

Claim 4.11 and Claim 4.10 imply our main bound on the error terms:

Lemma 4.12. *Fix any latent structure $\mu', \mu'' \in I_p$, and any distributions $\mathcal{D}', \mathcal{D}'' \in \Delta_p$. Then,*

$$|\mathsf{LP}(\mathcal{D}', \mu') - \mathsf{LP}(\mathcal{D}'', \mu'')| \leq O(\Phi_p(\mathsf{OPT}_{\mathsf{LP}})).$$

4.3.2 Performance of the Algorithm. The remainder of the analysis deals with rewards and resource consumption of the algorithm. We start with lower-bounding the LP-value for the chosen distributions.

Claim 4.13. *For each distribution $\mathcal{D}_{p,x}$ chosen by the algorithm in phase p,*

$$\mathsf{LP}(\mathcal{D}_{p,x}, \mu) \geq \mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p(\mathsf{OPT}_{\mathsf{LP}})).$$

Proof. The claim follows easily from Lemma 4.12, noting that $\mathcal{D}_{p,x} \in \Delta_p$. □

The following corollary lower-bounds the average reward; once we have it, it essentially remains to lower-bound the stopping time of the algorithm.

Corollary 4.14. $\widehat{r}_p \geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\log p)\,\Phi_p(\mathsf{OPT}_{\mathsf{LP}})\right)$.

Proof. Throughout this proof, denote $\Phi_p \triangleq \Phi_p(\mathsf{OPT}_{\mathsf{LP}})$. By Claim 4.13, for each distribution $\mathcal{D}_{q,x}$ chosen by the algorithm in phase $q < p$ it holds that

$$r(\mathcal{D}_{q,x}, \mu) \geq \frac{1}{T}\,\mathsf{LP}(\mathcal{D}_{q,x}, \mu) \geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_q)\right).$$

Averaging the above equation over all rounds in phases $q < p$, we obtain

$$\begin{aligned} r(\bar{\mathcal{D}}_p, \mu) &\geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - \frac{1}{p}\sum_{q<p} O(\Phi_q)\right) \\ &\geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p \log p)\right). \end{aligned}$$

For the last inequality, we used Claim 4.6(fg) to average the confidence radii in Φ_q.

Using the high-probability event in Claim 4.5(c):

$$\widehat{r}_p \geq r(\bar{\mathcal{D}}_p, \mu) - \mathsf{rad}(r(\bar{\mathcal{D}}_p, \mu), p|X|).$$

Now, using the monotonicity of $\nu - \mathsf{rad}(\nu, N)$ (Claim 4.6(c)), we obtain

$$\begin{aligned} \widehat{r}_p &\geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p)\right) - \mathsf{rad}\left(\frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p)\right),\ p|X|\right) \\ &\geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p)\right) - \mathsf{rad}\left(\ \mathsf{OPT}_{\mathsf{LP}}/T,\ p|X|\ \right) \\ &\geq \frac{1}{T}\left(\mathsf{OPT}_{\mathsf{LP}} - O(\Phi_p)\right). \end{aligned}$$

For the last equation, we use the fact that $\Phi_p/T \geq \Omega(\mathsf{rad}(\mathsf{OPT}_{\mathsf{LP}}/T, \frac{p}{d})) \geq \Omega(\mathsf{rad}(\mathsf{OPT}_{\mathsf{LP}}/T, p|X|))$. □

The following two claims help us to lower-bound the stopping time of the algorithm.

Claim 4.15. $c_i(\mathcal{D}_{p,x}, \mu) \leq \frac{B}{T} + O(1)\,\mathsf{rad}(\frac{B}{T},\ \frac{p}{d})$ *for each resource i.*

Proof. By the algorithm's specification, $\mathcal{D}_{p,x} \in \Delta_p$, and moreover there exists a latent structure $\mu' \in I_p$ such that $\mathcal{D}_{p,x}$ is LP-perfect for μ'. Apply Claim 4.8, noting that $c_i(\mathcal{D}_{p,x}, \mu') \leq \frac{B}{T}$ by LP-perfectness. □

Corollary 4.16. $\widehat{c}_{p,i} \leq \frac{B}{T} + O(\log p)\ \mathsf{rad}(\frac{B}{T}, \frac{p}{d})$ *for each resource i.*

Proof. Using a property of the clean execution, namely the event in Claim 4.5(c), we have

$$\widehat{c}_{p,i} \leq c_i(\bar{\mathcal{D}}, \mu) + \mathsf{rad}(c_i(\bar{\mathcal{D}}, \mu),\ p). \tag{19}$$

Consider all rounds preceding phase p.

$$\begin{aligned} c_i(\bar{\mathcal{D}}_p, \mu) &= \frac{1}{p|X|} \sum_{q<p,\ x\in X} c_i(\mathcal{D}_{q,x}, \mu) \\ &\leq \frac{B}{T} + \frac{O(1)}{p|X|} \sum_{q<p,\ x\in X} \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) && \text{(by Claim 4.15)} \\ &\leq \frac{B}{T} + O(\log p)\, \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) && \text{(by Claim 4.6(fg)).} \qquad (20) \end{aligned}$$

For the last inequality, we used Claim 4.6(fg) to average the confidence radii.

Using the upper bound on $c_i(\bar{\mathcal{D}}, \mu)$ that we derived above,

$$\mathtt{rad}\left(c_i(\bar{\mathcal{D}}, \mu),\ \frac{p}{d}\right) \leq O(\log p)\ \mathtt{rad}\left(\frac{B}{T} + \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right),\ \frac{p}{d}\right).$$

Using a general property of the confidence radius that

$$\mathtt{rad}(\nu + \mathtt{rad}(\nu, N),\ N) \leq O(\mathtt{rad}(\nu, N)),$$

we conclude that

$$\mathtt{rad}\left(c_i(\bar{\mathcal{D}}, \mu),\ \frac{p}{d}\right) \leq O(\log p)\ \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right). \qquad (21)$$

We obtain the claim by plugging the upper bounds Equations (20) and (21) into Equation (19). □

We are ready to put the pieces together and derive the performance guarantee for a clean execution of `BalancedExploration`.

Lemma 4.17. *Consider a clean execution of* `BalancedExploration`. *Then, the total reward*

$$\mathtt{REW} \geq \mathtt{OPT}_{\mathtt{LP}} - O\left(\log \frac{T}{|X|}\right)\ \Phi_{T/|X|}(\mathtt{OPT}_{\mathtt{LP}}).$$

Proof. Throughout this proof, denote $\Phi_p \triangleq \Phi_p(\mathtt{OPT}_{\mathtt{LP}})$. Let p be the last phase in the execution of the algorithm, and let T_0 be the stopping time. Letting $m = |X|$, note that $pm < T_0 \leq (p+1)m$.

We can use Corollary 4.14 to bound `REW` from below:

$$\mathtt{REW} = T_0\, \widehat{r}_{p+1} > p\, m\, \widehat{r}_{p+1} \geq \frac{p\, m}{T}\, (\mathtt{OPT}_{\mathtt{LP}} - O(\Phi_p \log p)). \qquad (22)$$

Let us bound $\frac{p\,m}{T}$ from below. The algorithm stops either when it runs out of time or if it runs out of resources during phase p. In the former case, $p = \lfloor T/m \rfloor$. In the latter case, $B = T_0\, \widehat{c}_{p+1,\, i}$ for some resource i, so $B \leq m(p+1)\, \widehat{c}_{p+1,\, i}$. Using Corollary 4.16, we obtain the following lower bound on p:

$$\frac{p\, m}{T} \geq 1 - O\left(\frac{p\, m \log p}{B}\right)\ \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right).$$

Plugging this into Equation (22), we conclude:

$$\begin{aligned} \mathtt{REW} &\geq \mathtt{OPT}_{\mathtt{LP}} - O\left(\frac{p\, m \log p}{B}\right)\ \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) \mathtt{OPT}_{\mathtt{LP}} - O\left(\frac{p\, m \log p}{T}\right) \Phi_p \\ &\geq \mathtt{OPT}_{\mathtt{LP}} - O\left(\frac{p\, m \log p}{T}\right) \left(\Phi_p + \frac{T}{B}\, \mathtt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) \mathtt{OPT}_{\mathtt{LP}}\right) \\ &\geq \mathtt{OPT}_{\mathtt{LP}} - \frac{p\, m \log p}{T}\ O(\Phi_p). \end{aligned}$$

To complete the proof, we observe that $(p\,\Phi_p \log p)$ is increasing in p (by definition of Φ_p) and plug in a trivial upper bound $p \leq T/m$. □

To finish the proof of Theorem 4.1, we write down the definition of $\Phi_{T/m}(\mathtt{OPT_{LP}})$, $m = |X|$, and plug in the definition of the confidence radius Equation (3):

$$\Phi_{T/m}(\mathtt{OPT_{LP}}) \triangleq O(T)\,\mathtt{rad}\left(\mathtt{OPT_{LP}}/T,\ \frac{T}{d|X|}\right) + O\left(\mathtt{OPT_{LP}}\,\frac{T}{B}\right)\mathtt{rad}\left(\frac{B}{T}, \frac{T}{dm}\right)$$
$$\leq O(\log(T))\left(\sqrt{dm\mathtt{OPT_{LP}}} + \mathtt{OPT_{LP}}\sqrt{\frac{dm}{B}}\right).$$

5 ALGORITHM PRIMALDUALBWK

This section develops an algorithm, called PrimalDualBwK, that solves the BwK problem using a very natural and intuitive idea: greedily select arms with the greatest estimated "bang per buck," i.e., reward per unit of resource consumption. One of the main difficulties with this idea is that there is no such thing as a known "unit of resource consumption": there are d different resources, and it is unclear how to trade off consumption of one resource versus another. The dual LP in Section 3 gives some insight into how to quantify this trade-off: an optimal dual solution η^* can be interpreted as a vector of unit costs for resources, such that for every arm the expected reward is less than or equal to the expected cost of resources consumed. Then the bang-per-buck ratio for a given arm x can be defined as $r(x,\mu)/(\eta^* \cdot c(x,\mu))$, where the denominator represents the expected cost of pulling this arm. The arms in the support of the optimal distribution ξ^* are precisely the arms with a maximal bang-per-buck ratio (by complimentary slackness), and pulling any other arm necessarily increases regret relative to $\mathtt{OPT_{LP}}$ (by Remark 3.2).

To estimate the bang-per-buck ratios, our algorithm will try to learn an optimal dual vector η^* in tandem with learning the latent structure μ. Borrowing an idea from References [8, 32, 50], we use the multiplicative weights update method to learn the optimal dual vector. This method raises the cost of a resource exponentially as it is consumed, which ensures that heavily demanded resources become costly, and thereby promotes balanced resource consumption. Meanwhile, we still have to ensure (as with any multi-armed bandit problem) that our algorithm explores the different arms frequently enough to gain adequately accurate estimates of the latent structure. We do this by estimating rewards and resource consumption as optimistically as possible, i.e., using upper confidence bound (UCB) estimates for rewards and lower confidence bound (LCB) estimates for resource consumption. Although both of these techniques—multiplicative weights and confidence bounds—have been successfully applied in previous online learning algorithms, it is far from obvious that this particular hybrid of the two methods should be effective. In particular, the use of multiplicative updates on dual variables, rather than primal ones, distinguishes our algorithm from other bandit algorithms that use multiplicative weights (e.g., the Exp3 algorithm [10]) and brings it closer in spirit to the literature on stochastic packing algorithms, especially Reference [26].

The pseudocode is presented as Algorithm 2. When we refer to the UCB or LCB for a latent parameter (the reward of an arm, or the amount of some resource that it utilizes), these are computed as follows. Letting $\hat{v}$ denote the empirical average of the observations of that random variable[10] and letting N denote the number of times the random variable has been observed, the lower confidence bound (LCB) and upper confidence bound (UCB) are the left and right endpoints, respectively, of

[10] Note that we initialize the algorithm by pulling each arm once, so empirical averages are always well-defined.

the *confidence interval* $[0, 1] \cap [\hat{v} - \texttt{rad}(\hat{v}, N),\ \hat{v} + \texttt{rad}(\hat{v}, N)]$. The UCB or LCB for a vector or matrix are defined componentwise.

ALGORITHM 2: PrimalDualBwK with Parameter $\epsilon \in (0, 1)$

```
1:  **Initialization**
2:      In the first $m$ rounds, pull each arm once.
3:      $v_1 = \mathbf{1} \in [0, 1]^d$.
4:          {$v_t \in [0, 1]^d$ is the round-$t$ estimate of the optimal solution $\eta^*$ to (LP-dual) in Section 3.}
5:          {We interpret $v_t(i)$ as an estimate of the (fictional) unit cost of resource $i$, for each $i$.}
6:      Set $\epsilon = \sqrt{\ln(d)/B}$.
7:  **for** rounds $t = m + 1, \ldots, \tau$  *(i.e., until resource budget exhausted)* **do**
8:      For each arm $x \in X$,
9:          Compute UCB estimate for the expected reward, $u_{t,x} \in [0, 1]$.
10:         Compute LCB estimate for the resource consumption vector, $L_{t,x} \in [0, 1]^d$.
11:         *Expected cost* for one pull of arm $x$ is estimated by $\texttt{EstCost}_x = L_{t,x} \cdot v_t$.
12:     Pull arm $x = x_t \in X$ that maximizes $u_{t,x}/\texttt{EstCost}_x$, the optimistic *bang-per-buck* ratio.
13:     Update estimated unit cost for each resource $i$:
            $v_{t+1}(i) = v_t(i)\,(1 + \epsilon)^{\ell},\ \ell = L_{t,x}(i)$.
```

The algorithm is fast: with machine word size of $\log T$ bits or more, the per-round running time is $O(md)$. Moreover, if each arm x consumes only d_x resources that are known in advance, then $L_{t,x}$ can be implemented as a d_x-dimensional vector, and $\texttt{EstCost}_x$ can be computed in $O(d_x)$ time. Then the per-round running time is $O(m + d + \sum_x d_x)$.

Discussion 5.1. The cost update in step 13 requires some explanation. Let us interpret this step as a separate algorithm that solves a particular problem. The problem is to optimize the total expected payoff when in each round $t > m$, one chooses a distribution $y_t = v_t/\|v_t\|_1$ over resources, and receives expected payoff $y_t \cdot L_{t,x_t}$. This is the well-known "best-expert" problem in which actions correspond to resources, and each action i is assigned payoff $L_{t,x_t}(i)$. Step 13 implements a multiplicative-weights algorithm for solving this problem. In fact, we could have used any other algorithm for this problem with a similar performance guarantee, as in Proposition 5.4.

But why does solving this particular best-experts problem make sense for PrimalDualBwK? Particularly, why does it make sense to maximize this notion of expected payoffs? Let us view distribution y_t as a vector of *normalized costs* of resources. Consider the total expected normalized cost consumed by the algorithm after round m, denote it W. Then $W = \sum_{t=m+1}^{\tau} y_t\ c(x_t, \mu)$. A lower confidence bound on this quantity is $W_{\mathsf{LCB}} = \sum_{t=m+1}^{\tau} y_t \cdot L_{t,x_t}$, which is precisely the total expected payoff in the best-experts problem. In the analysis, we relate W_{LCB} and the upper confidence bound on the total expected reward in the same rounds, $\mathsf{REW}_{\mathsf{UCB}} = \sum_{t=m+1}^{\tau} u_{t,x_t}$ Specifically, we prove that for any implementation of step 13, we have

$$\mathsf{REW}_{\mathsf{UCB}} \geq W_{\mathsf{LCB}}\ \mathsf{OPT}_{\mathsf{LP}}/B \quad \text{with high probability.} \tag{23}$$

(This follows from Equation (31).) Thus, maximizing W_{LCB} is a reasonable goal for the cost update rule.

Step 13 can also be seen as a variant of the Garg-Könemann width reduction technique [32]. The ratio $u_{t,x}/\texttt{EstCost}_x$ that we optimize in step 12 may be unboundedly large, so in the multiplicative update in step 13, we rescale this value to $L_{t,x}(i)$, which is guaranteed to be at most 1; this rescaling is mirrored in the analysis of the algorithm. Interestingly, unlike the Garg-Könemann algorithm, which applies multiplicative updates to the dual vectors and weighted averaging to the primal ones, in our algorithm the multiplicative updates and weighted averaging are *both* applied to the dual vectors.

Discussion 5.2. From the primal-dual point of view, we could distinguish a "primal" problem in which one chooses among arms, and a "dual" problem in which one updates the cost vector. In the primal problem, the choice of costs is deemed adversarial, and the goal is to ensure Equation (23). In the dual problem, the choice of arms is deemed adversarial, and the goal is to maximize W_{LCB} to obtain Proposition 5.4. In both problems, one is agnostic as to how the upper/lower confidence bounds u_t and $L_{t,x}$ are updated over time. As mentioned above, the dual problem falls under a standard setting of the "best-expert" problem, and is solved via a standard algorithm for this problem. Meanwhile the primal problem is solved via bang-per-buck ratios and an ad hoc application of the "optimism under uncertainty" principle.

When the rewards and consumptions are deterministic,[11] the analysis is completely modular: it works no matter which algorithm is used to solve the primal (respectively, dual) problem. In the general case, the primal algorithm also needs to ensure that the "error terms" come out suitably small.

The following theorem expresses the regret guarantee for `PrimalDualBwK`.

Theorem 5.3. *Consider an instance of* `BwK` *with d resources, $m = |X|$ arms, and the smallest budget $B = \min_i B_i$. The regret of algorithm* `PrimalDualBwK` *with parameter $\epsilon = \sqrt{\ln(d)/B}$ satisfies*

$$\mathtt{OPT}_{\mathtt{LP}} - \mathtt{REW} \le O\left(\sqrt{\log(dT)}\right)\left(\sqrt{m\,\mathtt{OPT}_{\mathtt{LP}}} + \mathtt{OPT}_{\mathtt{LP}}\sqrt{\frac{m}{B}}\right) + O(m)\,\log(dT)\log(T). \tag{24}$$

Moreover, Equation (7) holds with $f(\mathtt{OPT}_{\mathtt{LP}})$ equal to the right-hand side of Equation (24).

The rest of the section proves this theorem. Throughout, it will be useful to represent the latent values as matrices and vectors. For this purpose, we will number the arms as $X = \{1, \ldots, m\}$ and let $r \in \mathbb{R}^m$ denote the vector whose x-th component is $r(x, \mu)$, the expected reward, for each arm $x \in X$. Similarly, we will let $C \in \mathbb{R}^{d\times m}$ denote the matrix whose (i, x) entry is $c_i(x, \mu)$, the expected resource consumption, for each resource i and each arm x. Let $\mathbf{e}_j^d \in \{0,1\}^d$ denote the d-dimensional j-th coordinate vector.

While `PrimalDualBwK` uses multiplicative weights update as a general technique, we make use of a specific performance guarantee in our analysis. To this end, let us recall algorithm Hedge [31] from online learning theory, also known as the multiplicative weights algorithm. It is an online algorithm for maintaining a d-dimensional probability vector y while observing a sequence of d-dimensional payoff vectors $\pi_1, \ldots, \pi_\tau$. The version presented below, along with the following performance guarantee, is adapted from Kleinberg [38]; a self-contained proof appears in Appendix C.

Proposition 5.4. *Fix any parameter $\epsilon \in (0,1)$ and any stopping time τ. For any sequence of payoff vectors $\pi_1, \ldots, \pi_\tau \in [0,1]^d$, we have*

$$\forall y \in \Delta[d] \quad \sum_{t=1}^{\tau} y_t^\mathsf{T} \pi_t \ge (1-\epsilon)\sum_{t=1}^{\tau} y^\mathsf{T}\pi_t - \frac{\ln d}{\epsilon}.$$

[11] Then the dual problem maximizes W rather than W_{LCB}, and the primal problem ensures Equation (25) rather than Equation (23); see Section 5.1.

ALGORITHM 3: Hedge with Parameter $\epsilon \in (0, 1)$

```
$v_1 = \mathbf{1}$    $\{v_t \in \mathbb{R}_+^d$ for each round $t$.$\}$
**for** each round $t = 1, 2, 3, \ldots$ **do**
    **Output** distribution $y_t = v_t / \|v_t\|_1$.
    **Input** payoff vector $\pi_t \in [0, 1]^d$.
    **for** each resource $i$ **do**
        $v_{t+1}(i) = v_t(i)\,(1 + \epsilon)^\ell$, $\ell = \pi_t(i)$.
```

5.1 Warm-Up: The Deterministic Case

To present the application of Hedge to BwK in its purest form, we first consider the "deterministic case" in which the rewards of the various arms are deterministically equal to the components of a vector $r \in \mathbb{R}^m$, and the resource consumption vectors are deterministically equal to the columns of a matrix $C \in \mathbb{R}^{d \times m}$. Then there is no need to use upper/lower confidence bounds, so the algorithm can be simplified considerably, see Algorithm 4. In the remainder of this subsection, we discuss this algorithm and analyze its regret.

ALGORITHM 4: Algorithm PrimalDualBwK for Deterministic Outcomes, with Parameter $\epsilon \in (0, 1)$

```
1:  **Initialization**
2:      In the first $m$ rounds, pull each arm once.
3:      For each arm $x \in X$, let $r_x \in [0, 1]$ and $C_x \in [0, 1]^d$
4:          denote the reward and the resource consumption vector revealed in Step 2.
5:      $v_1 = \mathbf{1} \in [0, 1]^d$.
6:          $\{v_t \in [0, 1]^d$ is the round-$t$ estimate of the optimal solution $\eta^*$ to (LP-dual) in Section 3.$\}$
7:          {We interpret $v_t(i)$ as an estimate of the (fictional) unit cost of resource $i$, for each $i$.}
8:      Set $\epsilon = \sqrt{\ln(d)/B}$.
9:  **for** rounds $t = m + 1, \ldots, \tau$  *(i.e., until resource budget exhausted)* **do**
10:     For each arm $x \in X$,
11:         *Expected cost* for one pull of arm $x$ is estimated by $\texttt{EstCost}_x = C_x \cdot v_t$.
12:     Pull arm $x = x_t \in X$ that maximizes $r_x / \texttt{EstCost}_x$, the *bang-per-buck* ratio.
13:     Update estimated unit cost for each resource $i$:
                        $v_{t+1}(i) = v_t(i)\,(1 + \epsilon)^\ell,\ \ell = C_x(i)$.
```

Algorithm 4 is an instance of the multiplicative-weights update method for solving packing linear programs. Interpreting it through the lens of online learning, as in the survey by Arora et al. [8], it is updating a vector $y_t = v_t / \|v_t\|_1$ using the Hedge algorithm, where the payoff vector in any round $t > m$ is given by $\pi_t = C_{x_t}$ and the goal is to optimize the total (expected) payoff $W = \sum_{t=m+1}^{\tau} y_t \cdot C_{x_t}$. Note that W is also the total cost consumed by Algorithm 4.

To see why W is worth maximizing, let us relate it to the total reward collected by the algorithm in rounds $t > m$; denote this quantity by $\texttt{REW} = \sum_{t=m+1}^{\tau-1} r_t$. We will prove that

$$\texttt{REW} \geq W \cdot \texttt{OPT}_{\texttt{LP}} / B \text{ for any implementation of Step 13.} \tag{25}$$

For this reason, maximizing W also helps maximize REW. Proving it is a major step in the analysis.

Let ξ^* denote an optimal solution of the primal linear program (LP-primal) from Section 3, and let $\mathsf{OPT}_{\mathsf{LP}} = r^\intercal \xi^*$ denote the optimal value of that LP.

For each round t, let $z_t = \mathbf{e}^m_{x_t}$ denote the x_tth coordinate vector. We claim that

$$z_t \in \operatorname*{argmax}_{z \in \Delta[X]} \left\{ \frac{r^\intercal z}{y_t^\intercal C z} \right\}. \tag{26}$$

In words: z_t maximizes the "bang-per-buck ratio" among all distributions z over arms. Indeed, the argmax in Equation (26) is well-defined as that of a continuous function on a compact set. Say it is attained by some distribution z over arms, and let $\rho \in \mathbb{R}$ be the corresponding max. By maximality of ρ, the linear inequality $\rho\, y_t^\intercal C z \geq r^\intercal z$ also holds at some extremal point of the probability simplex $\Delta[X]$, i.e., at some point-mass distribution. For any such point-mass distribution, the corresponding arm maximizes the bang-per-buck ratio in the algorithm. Claim proved.

Proof of Equation (25). It follows that

$$y_t^\intercal \pi_t = y_t^\intercal C z_t \leq r_t \left(y_t^\intercal C \xi^* \right) / \mathsf{OPT}_{\mathsf{LP}},$$

$$W = \sum_t y_t^\intercal \pi_t \leq \frac{1}{\mathsf{OPT}_{\mathsf{LP}}} \sum_t r_t \left(y_t^\intercal C \xi^* \right) = \frac{1}{\mathsf{OPT}_{\mathsf{LP}}} \left(\sum_t r_t\, y_t^\intercal \right) C \xi^*.$$

Here the sums are over rounds t with $m < t < \tau$. Now, letting $\bar{y} = \frac{1}{\mathsf{REW}} \sum_t r_t\, y_t \in [0,1]^d$ be the rewards-weighted average of distributions $y_{m+1}, \ldots, y_\tau$, it follows that

$$W \leq \frac{\mathsf{REW}}{\mathsf{OPT}_{\mathsf{LP}}}\, \bar{y}^T C \xi^* \leq \frac{\mathsf{REW}}{\mathsf{OPT}_{\mathsf{LP}}}\, B.$$

The last inequality follows because all components of $C\xi^*$ are at most B by the primal feasibility of ξ^*. □

Now, combining Equation (25) and the regret bound for Hedge, we obtain

$$\mathsf{REW} \geq W \cdot \mathsf{OPT}_{\mathsf{LP}}/B \geq \left[(1-\epsilon) \sum_{m<t<\tau} y\, \pi_t - \frac{\ln d}{\epsilon} \right] \cdot \frac{\mathsf{OPT}_{\mathsf{LP}}}{B} \quad \forall y \in \Delta[d]. \tag{27}$$

To continue this argument, we need to choose an appropriate vector y to make the right-hand side large. Recall that $\pi_t = C z_t$, so $\sum_{m<t<\tau} \pi_t$ is simply the total consumption vector in all rounds $m < t < \tau$. We know some resource i must be exhausted by the time the algorithm stops, so the consumption of this resource is at least B. In a formula: $\sum_{t=1}^{\tau}\ y\, \pi_t \geq B$, where $y = \mathbf{e}^d_i$ is the identity vector for resource i. Plugging in this y into Equation (27), we obtain

$$\begin{aligned} \mathsf{REW} &\geq \left[(1-\epsilon)(B-m-1) - \frac{\ln d}{\epsilon} \right] \cdot \frac{\mathsf{OPT}_{\mathsf{LP}}}{B} \\ &\geq \mathsf{OPT}_{\mathsf{LP}} - \left[\epsilon B + m + 1 + \frac{\ln d}{\epsilon} \right] \cdot \frac{\mathsf{OPT}_{\mathsf{LP}}}{B} \\ &= \mathsf{OPT}_{\mathsf{LP}} - O(\sqrt{B \ln d} + m) \cdot \frac{\mathsf{OPT}_{\mathsf{LP}}}{B} \qquad \text{if } \epsilon = \sqrt{\tfrac{\ln d}{B}}. \end{aligned}$$

This completes regret analysis for the deterministic case.

5.2 Analysis Modulo Error Terms

We now commence the analysis of Algorithm PrimalDualBwK. In this subsection, we show how to reduce the problem of bounding the algorithm's regret to a problem of estimating two error terms that reflect the difference between the algorithm's confidence-bound estimates of its own reward and resource consumption with the empirical values of these random variables. The error terms will be treated in Section 5.3.

Recall that the algorithm computes LCBs on expected resource consumption $L_{t,x} \in [0,1]^d$ and UCBs on expected rewards $u_{t,x} \in [0,1]$, for each round t and each arm x. We also represent the LCBs as a matrix $L_t \in [0,1]^{d\times m}$ whose xth column equals $L_{t,x}$, for each arm x. We also represent the UCBs as a vector $u_t \in [0,1]^m$ over arms whose xth component equals $u_{t,x}$. Let C_t be the resource-consumption matrix for round t. That is, $C_t \in [0,1]^{d\times m}$ denotes the matrix whose (i,x) entry is the actual consumption of resource i in round t if arm x were chosen in this round.

As in the previous subsection, let $z_t = \mathbf{e}^m_{x_t}$ denote the x_tth coordinate vector, and let $y_t = v_t/\|v_t\|_1$ be the vector of normalized costs. Similar to Equation (26), z_t maximizes the "bang-per-buck ratio" among all distributions z over arms:

$$z_t \in \operatorname*{argmax}_{z\in\Delta[X]} \left\{ \frac{u_{t,x}^{\mathsf{T}} z}{y_t^{\mathsf{T}} L_{t,x}\, z} \right\}. \tag{28}$$

By Theorem 2.1 and our choice of C_{rad}, it holds with probability at least $1-T^{-1}$ that the confidence interval for every latent parameter, in every round of execution, contains the true value of that latent parameter. We call this high-probability event a *clean execution* of PrimalDualBwK. Our regret guarantee will hold deterministically assuming that a clean execution takes place. The regret can be at most T when a clean execution does not take place, and since this event has probability at most T^{-1} it contributes only $O(1)$ to the regret. We will henceforth assume a clean execution of PrimalDualBwK.

Claim 5.5. *In a clean execution of Algorithm* PrimalDualBwK *with parameter* $\epsilon = \sqrt{\ln(d)/B}$, *the algorithm's total reward satisfies the bound*

$$\mathsf{OPT}_{\mathsf{LP}} - \mathsf{REW} \le \left[2\mathsf{OPT}_{\mathsf{LP}} \left(\sqrt{\frac{\ln d}{B}} + \frac{m+1}{B} \right) + m + 1 \right] + \frac{\mathsf{OPT}_{\mathsf{LP}}}{B} \left\| \sum_{m<t<\tau} E_t z_t \right\|_\infty + \left| \sum_{m<t<\tau} \delta_t^{\mathsf{T}} z_t \right|, \tag{29}$$

where $E_t = C_t - L_t$ *and* $\delta_t = u_t - r_t$ *for each round* $t > m$.

Proof. The claim is proven by mimicking the analysis of Algorithm 4 in the preceding section, incorporating error terms that reflect the differences between observable values and latent ones. As before, let ξ^* denote an optimal solution of the primal linear program (LP-primal), and let $\mathsf{OPT}_{\mathsf{LP}} = r^{\mathsf{T}}\xi^*$ denote the optimal value of that LP. Let $\mathsf{REW}_{\mathsf{UCB}} = \sum_{m<t<\tau} u_t^{\mathsf{T}} z_t$ denote the total payoff the algorithm would have obtained, after its initialization phase, if the actual payoff at time t were replaced with the upper confidence bound. Let $y = \mathbf{e}^d_i$, where i is a resource exhausted by the algorithm when it stops; then $y^{\mathsf{T}}(\sum_{t=1}^{\tau} C_t z_t) \ge B$. As before,

$$y^{\mathsf{T}} \left(\sum_{m<t<\tau} C_t z_t \right) \ge B - m - 1. \tag{30}$$

Finally, let

$$\bar{y} = \frac{1}{\mathsf{REW}_{\mathsf{UCB}}} \sum_{m<t<\tau} \left(u_t^{\mathsf{T}} z_t \right) y_t .$$

Assuming a clean execution, we have

$$\begin{align*}
B &\geq \bar{y}^\mathsf{T} C \xi^* && \textit{(ξ^* is primal feasible)} \\
&= \frac{1}{\mathsf{REW}_\mathsf{UCB}} \sum_{m<t<\tau} (u_t^\mathsf{T} z_t)(y_t^\mathsf{T} C \xi^*) && \textit{(definition of $\bar{y}$)} \\
&\geq \frac{1}{\mathsf{REW}_\mathsf{UCB}} \sum_{m<t<\tau} (u_t^\mathsf{T} z_t)(y_t^\mathsf{T} L_t \xi^*) && \textit{(clean execution)} \\
&\geq \frac{1}{\mathsf{REW}_\mathsf{UCB}} \sum_{m<t<\tau} (u_t^\mathsf{T} \xi^*)(y_t^\mathsf{T} L_t z_t) && \textit{(by Equation (28))} \\
&\geq \frac{1}{\mathsf{REW}_\mathsf{UCB}} \sum_{m<t<\tau} (r^\mathsf{T} \xi^*)(y_t^\mathsf{T} L_t z_t) && \textit{(clean execution)} \tag{31} \\
&\geq \frac{\mathsf{OPT}_\mathsf{LP}}{\mathsf{REW}_\mathsf{UCB}} \left[(1-\epsilon) y^\mathsf{T} \left(\sum_{m<t<\tau} L_t z_t \right) - \frac{\ln d}{\epsilon} \right] && \textit{(Hedge guarantee)} \\
&> (1-\epsilon) \frac{\mathsf{OPT}_\mathsf{LP}}{\mathsf{REW}_\mathsf{UCB}} \left[y^\mathsf{T} \left(\sum_{m<t<\tau} C_t z_t \right) - y^\mathsf{T} \left(\sum_{m<t<\tau} E_t z_t \right) - \frac{\ln d}{\epsilon} \right] \\
&\geq \frac{\mathsf{OPT}_\mathsf{LP}}{\mathsf{REW}_\mathsf{UCB}} \left[(1-\epsilon) B - m - 1 - (1-\epsilon) y^\mathsf{T} \left(\sum_{m<t<\tau} E_t z_t \right) - \frac{\ln d}{\epsilon} \right] && \textit{(definition of y; see Equation (30))}
\end{align*}$$

$$\mathsf{REW}_\mathsf{UCB} \geq \mathsf{OPT}_\mathsf{LP} \left[1 - \epsilon - \frac{m+1}{B} - \frac{1}{B} \left\| \sum_{m<t<\tau} E_t z_t \right\|_\infty - \frac{\ln d}{\epsilon B} \right]. \tag{32}$$

The algorithm's actual payoff, $\mathsf{REW} = \sum_{t=1}^{\tau} r_t^\mathsf{T} z_t$, satisfies the inequality

$$\mathsf{REW} \geq \mathsf{REW}_\mathsf{UCB} - \sum_{m<t<\tau} (u_t - r_t)^\mathsf{T} z_t = \mathsf{REW}_\mathsf{UCB} - \sum_{m<t<\tau} \delta_t^\mathsf{T} z_t.$$

Combining this with Equation (32), and plugging in $\epsilon = \sqrt{\ln(d)/B}$, we obtain the bound Equation (29), as claimed. □

5.3 Error Analysis

We complete the proof of Theorem 5.3 by proving upper bounds on the terms $\|\sum_{m<t<\tau} E_t z_t\|_\infty$ and $|\sum_{m<t<\tau} \delta_t z_t|$ that appear on the right side of Equation (29). Both bounds follow from a more general lemma, which we present below.

The general lemma considers a sequence of vectors $a_1, \ldots, a_\tau$ in $[0,1]^m$ and another vector $a_0 \in [0,1]^m$. Here $a_{t,x} \in [0,1]$ represents a numerical outcome (i.e., a reward or a consumption of a given resource) if arm x is pulled in round t, and $a_{0,x}$ represents the corresponding expected outcome. Further, for each round $t > m$, we have an estimate $b_t \in [0,1]^m$ for the outcome vector a_t. We only assume a clean execution of the algorithm, and we derive an upper bound on $|\sum_{m<t<\tau} (b_t - a_t)^\mathsf{T} z_t|$.

Lemma 5.6. *Consider two sequences of vectors $a_1, \ldots, a_\tau$ and $b_1, \ldots, b_\tau$, in $[0,1]^m$, and a vector $a_0 \in [0,1]^m$. For each arm x and each round $t > m$, let $\overline{a}_{t,x} \in [0,1]$ be the average observed outcome up to round t, i.e., the average outcome $a_{s,x}$ over all rounds $s \leq t$ in which arm x has been chosen by the algorithm; let $N_{t,x}$ be the number of such rounds. Assume that for each arm x and all rounds t with $m < t < \tau$, we have*

$$\begin{aligned}
|b_{t,x} - a_{0,x}| &\leq 2\,\mathsf{rad}(\overline{a}_{t,x}, N_{t,x}) \leq 6\,\mathsf{rad}(a_{0,x}, N_{t,x}), \\
|\overline{a}_{t,x} - a_{0,x}| &\leq \mathsf{rad}(\overline{a}_{t,x}, N_{t,x}).
\end{aligned}$$

Let $A = \sum_{t=1}^{\tau-1} a_{t,x_t}$ be the total outcome collected by the algorithm. Then,

$$\left| \sum_{m<t<\tau} (b_t - a_t)^\top z_t \right| \leq O\left(\sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m \, \log T\right). \tag{33}$$

Before proving the lemma, we need to establish a simple fact about confidence radii.

Claim 5.7. *For any two vectors $a, M \in \mathbb{R}_+^m$, we have*

$$\sum_{x=1}^{m} \mathsf{rad}(a_x, M_x)\, M_x \leq \sqrt{C_{\mathsf{rad}}\, m(a^\top M)} + C_{\mathsf{rad}}\, m. \tag{34}$$

Proof. The definition of $\mathsf{rad}(\cdot,\cdot)$ implies that $\mathsf{rad}(a_x, M_x)\, M_x \leq \sqrt{C_{\mathsf{rad}}\, a_x M_x} + C_{\mathsf{rad}}$. Summing these inequalities and applying Cauchy-Schwarz,

$$\sum_{x=1}^{m} \mathsf{rad}(a_x, M_x)\, M_x \leq \sum_{x=1}^{m} \sqrt{C_{\mathsf{rad}}\, a_x M_x} + C_{\mathsf{rad}}\, m \leq \sqrt{m} \cdot \sqrt{\sum_{x \in X} C_{\mathsf{rad}}\, a_x M_x} + C_{\mathsf{rad}}\, m,$$

and the lemma follows by rewriting the expression on the right side. □

Proof of Lemma 5.6. For convenience, denote $\mathbf{N}_t = (N_{t,1}, \ldots, N_{t,m})$, and observe that

$$\mathbf{N}_t = \sum_{s=1}^{t} z_s \quad \text{and} \quad A = \overline{a}_{\tau-1}^\top \mathbf{N}_{\tau-1} = \sum_{s=1}^{\tau-1} a_s^\top\, z_s.$$

We decompose the left side of Equation (33) as a sum of three terms,

$$\sum_{m<t<\tau} (b_t - a_t)^\top z_t = \sum_{t=1}^{m} (a_t - b_t)^\top z_t + \sum_{t=1}^{\tau-1} (b_t - a_0)^\top z_t + \sum_{t=1}^{\tau-1} (a_0 - a_t)^\top z_t, \tag{35}$$

then bound the three terms separately. The first sum is clearly bounded above by m. We next work on bounding the third sum. Let $s = \tau - 1$.

$$\begin{aligned} |(a_0 - \overline{a}_s)^\top\ \mathbf{N}_s| &\leq \sum_{x \in X} \mathsf{rad}(\overline{a}_{s,x}, N_{s,x})\, N_{s,x} && \textit{(assuming clean execution)} \\ &\leq \sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m. && \textit{(by Claim 5.7)} \end{aligned} \tag{36}$$

$$\sum_{t=1}^{s} (a_0 - a_t)^\top\, z_t = a_0^\top\, \mathbf{N}_s - \sum_{t=1}^{s} a_t^\top\, z_t = (a_0 - \overline{a}_s)^\top\, \mathbf{N}_s.$$

$$\left| \sum_{t=1}^{s} (a_0 - a_t)^\top\, z_t \right| = |(a_0 - \overline{a}_s)^\top\ \mathbf{N}_s| \leq \sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m.$$

Finally, we bound the middle sum in Equation (35).

$$\begin{aligned} \left| \sum_{t=1}^{s} (b_t - a_0)^\top z_t \right| &\leq 6 \sum_{t=1}^{s} \sum_{x \in X} \mathsf{rad}(a_{0,x}, N_{t,x}) z_{t,x} \\ &= 6 \sum_{x \in X} \sum_{\ell=1}^{N_{s,x}} \mathsf{rad}(a_{0,x}, \ell) \\ &= O\left(\sum_{x \in X} \sqrt{C_{\mathsf{rad}}\, a_{0,x}\, N_{s,x}} + C_{\mathsf{rad}} \log(N_{s,x}) \right) \\ &\leq O\left(\sqrt{C_{\mathsf{rad}}\, m\, a_0^\top\, \mathbf{N}_s} + C_{\mathsf{rad}}\, m \, \log T \right). \end{aligned} \tag{37}$$

We would like to replace the expression $a_0^\intercal\, \mathbf{N}_s$ on the last line with the expression $\overline{a}_s{}^\intercal\, \mathbf{N}_s = A$. To do so, recall Equation (36) and apply the following calculation:

$$\begin{aligned} a_0^\intercal\, \mathbf{N}_s &\leq \overline{a}_s{}^\intercal \mathbf{N}_s + \sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m \\ &= A + \sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m \\ &\leq \left(\sqrt{A} + \sqrt{C_{\mathsf{rad}}\, m}\right)^2 \\ \sqrt{C_{\mathsf{rad}}\, m a_0^\intercal\, \mathbf{N}_s} &\leq \sqrt{C_{\mathsf{rad}}\, m}\left(\sqrt{A} + \sqrt{C_{\mathsf{rad}}\, m}\right) = \sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m. \end{aligned}$$

Plugging this into Equation (37), we bound the middle sum in Equation (35) as

$$\left|\sum_{t=1}^{s}(b_t - a_0)^\intercal z_t\right| \leq O\left(\sqrt{C_{\mathsf{rad}}\, mA} + C_{\mathsf{rad}}\, m\, \log T\right). \tag{38}$$

Summing up the upper bounds for the three terms on the right side of Equation (35), we obtain Equation (33). □

Corollary 5.8. *In a clean execution of* `PrimalDualBwK`,

$$\left|\sum_{m<t<\tau} \delta_t z_t\right| \leq O\left(\sqrt{C_{\mathsf{rad}}\, m\mathsf{REW}} + C_{\mathsf{rad}}\, m\, \log T\right)$$

and

$$\left\|\sum_{m<t<\tau} E_t z_t\right\|_\infty \leq O\left(\sqrt{C_{\mathsf{rad}}\, mB} + C_{\mathsf{rad}}\, m\, \log T\right).$$

Proof. The first inequality is obtained by applying Lemma 5.6 with vector sequences $a_t = r_t$ and $b_t = u_t$, and vector $a_0 = r$. In other words, a_0 is the vector of expected rewards across all arms.

The second inequality is obtained by applying the same lemma separately for each resource i, with vector sequences $a_t = (\mathbf{e}_i^d)^\intercal\, C_t$ and $b_t = \mathbf{e}_i^d\, L_t$, and vector a_0 being the ith row of matrix C. In other words, a_0 is the vector of expected consumption of resource i across all arms. □

Proof of Theorem 5.3: If $m \geq B/\log(dT)$, then the regret bound in Theorem 5.3 is trivial. Therefore, we can assume without loss of generality that $m \leq B/\log(dT)$. Therefore, recalling Equation (29), we observe that

$$2\,\mathsf{OPT}_{\mathsf{LP}}\left(\sqrt{\frac{\ln d}{B}} + \frac{m+1}{B}\right) = O\left(\sqrt{m\log(dmT)}\,\frac{\mathsf{OPT}_{\mathsf{LP}}}{\sqrt{B}}\right).$$

The term $m+1$ on the right side of Equation (29) is bounded above by $m\log(dmT)$. Finally, using Corollary 5.8 we see that the sum of the final two terms on the right side of Equation (29) is bounded by

$$O\left(\sqrt{C_{\mathsf{rad}}\, m}\left(\frac{\mathsf{OPT}_{\mathsf{LP}}}{\sqrt{B}} + \sqrt{\mathsf{OPT}_{\mathsf{LP}}}\right) + C_{\mathsf{rad}}\, m\, \log T\right).$$

The theorem follows by plugging in $C_{\mathsf{rad}} = \Theta(\log(dmT)) = O(\log(dT))$ (because $m \leq B \leq T$). □

6 LOWER BOUND

We prove that regret Equation (1) obtained by algorithm `PrimalDualBwK` is optimal up to polylog factors. Specifically, we prove that any algorithm for BwK must, in the worst case, incur regret

$$\Omega\left(\min\left(\mathtt{OPT},\ \mathtt{OPT}\sqrt{\frac{m}{B}} + \sqrt{m\,\mathtt{OPT}}\right)\right), \tag{39}$$

where $m = |X|$ is the number of arms and $B = \min_i B_i$ is the smallest budget.

Theorem 6.1. *Fix any $m \geq 2$, $d \geq 1$, $\mathtt{OPT} \geq m$, and $(B_1, \ldots, B_d) \in [2, \infty)$. Let $\mathcal{G}$ be the family of all* BwK *problem instances with m arms, d resources, budgets $(B_1, \ldots, B_d)$ and optimal reward* OPT. *Then any algorithm for* BwK *must incur regret Equation (39) in the worst case over $\mathcal{G}$.*

We treat the two summands in Equation (39) separately:

Claim 6.2. *Consider the family $\mathcal{G}$ from Theorem 6.1, and let* ALG *be some algorithm for* BwK.

(a) ALG *incurs regret $\Omega(\min(\mathtt{OPT}, \sqrt{m\,\mathtt{OPT}}))$ in the worst case over $\mathcal{G}$.*

(b) ALG *incurs regret $\Omega(\min(\mathtt{OPT}, \mathtt{OPT}\sqrt{\frac{m}{B}}))$ in the worst case over $\mathcal{G}$.*

Theorem 6.1 follows from Claim 6.2(ab). For part (a), we use a standard lower-bounding example for MAB. For part (b), we construct a new example, specific to BwK, and analyze it using KL-divergence.

Proof of Claim 6.2(a). Fix $m \geq 2$ and $\mathtt{OPT} \geq m$. Let $\mathcal{G}_0$ be the family of all MAB problem instances with m arms and time horizon $T = \lfloor 2\,\mathtt{OPT} \rfloor$, where the "best arm" has expected reward $\mu^* = \mathtt{OPT}/T$ and all other arms have reward $\mu^* - \epsilon$ with $\epsilon = \frac{1}{4}\sqrt{m/T}$. Note that $\mu^* \in [\frac{1}{2}, \frac{3}{4}]$ and $\epsilon \leq \frac{1}{4}$. It is well-known [10] that any MAB algorithm incurs regret $\Omega(\sqrt{m\,\mathtt{OPT}})$ in the worst case over $\mathcal{G}_0$.

To ensure that $\mathcal{G}_0 \subset \mathcal{G}$, let us treat each MAB instance in $\mathcal{G}_0$ as a BwK instance with d resources, budgets $(B_1, \ldots, B_d)$, and no resource consumption. □

6.1 The New Lower-Bounding Example: Proof Of Claim 6.2(b)

Our lower-bounding example is very simple. There are m arms. Each arm gives reward 1 deterministically. There is a single resource with budget B.[12] The resource consumption, for each arm and each round, is either 0 or 1. The expected resource consumption is $p - \epsilon$ for the "best arm" and p for all other arms, where $0 < \epsilon < p < 1$. There is time horizon $T < \infty$. Let $\mathcal{G}_{(p,\epsilon)}$ denote the family of all such problem instances, for fixed parameters (p, ϵ). We analyze this family in the rest of this section.

We rely on the following fact about stopping times of random sums. For the sake of completeness, we provide a proof in Section D.

Fact 6.3. *Let S_t be the sum of t i.i.d. 0-1 variables with expectation q. Let τ^* be the first time this sum reaches a given number $B \in \mathbb{N}$. Then $\mathbb{E}[\tau^*] = B/q$. Moreover, for each $T > \mathbb{E}[\tau^*]$ it holds that*

$$\sum_{t>T} \Pr[\tau^* \geq t] \leq \mathbb{E}[\tau^*]^2/T.$$

Infinite time horizon. It is convenient to consider the family of problem instances, which is the same as $\mathcal{G}_{(p,\epsilon)}$ except that it has the *infinite* time horizon; denote it $\mathcal{G}^{\infty}_{(p,\epsilon)}$. We will first prove the desired lower bound for this family, then extend it to $\mathcal{G}_{(p,\epsilon)}$.

[12] More formally, other resources in the setting of Theorem 6.1 are not consumed. For simplicity, we leave them out.

The two crucial quantities that describe algorithm's performance on an instance in $\mathcal{G}^{\infty}_{(p,\epsilon)}$ is the stopping time and the total number of plays of the best arm. (Note that the total reward is equal to the stopping time minus 1.) The following claim connects these two quantities.

Claim 6.4 (Stopping time). *Fix an algorithm* ALG *for* BwK *and a problem instance in* $\mathcal{G}^{\infty}_{(p,\epsilon)}$. *Consider an execution of* ALG *on this problem instance. Let τ be the stopping time of* ALG. *For each round t, let N_t be the number of rounds $s \le t$ in which the best arm is selected. Then,*

$$p\mathbb{E}[\tau] - \epsilon\mathbb{E}[N_\tau] = \lfloor B+1 \rfloor.$$

Proof. Let C_t be the total resource consumption after round t. Note that $\mathbb{E}[C_t] = pt - \epsilon N_t$. We claim that

$$\mathbb{E}[C_\tau] = \mathbb{E}[p\tau - \epsilon N_\tau]. \tag{40}$$

Indeed, let $Z_t = C_t - (pt - \epsilon N_t)$. It is easy to see that Z_t is a martingale with bounded increments, and moreover that $\Pr[\tau < \infty] = 1$. Therefore, the Optional Stopping Theorem applies to Z_t and τ, so that $\mathbb{E}[Z_\tau] = E[Z_0] = 0$. Therefore, we obtain Equation (40).

To complete the proof, it remains to show that $C_\tau = \lfloor B+1 \rfloor$. Recall that ALG stops if and only if $C_t > B$. Since resource consumption in any round is either 0 or 1, it follows that $C_\tau = \lfloor B+1 \rfloor$. □

Corollary 6.5. *Consider the setting in Claim 6.4. Then:*

(a) If ALG *always chooses the best arm, then* $\mathbb{E}[\tau] = \lfloor B+1 \rfloor/(p-\epsilon)$.
(b) $\mathtt{OPT} = \lfloor B+1 \rfloor/(p-\epsilon) - 1$ *for any problem instance in* $\mathcal{G}^{\infty}_{(p,\epsilon)}$.
(c) $p\mathbb{E}[\tau] - \epsilon\mathbb{E}[N_\tau] = (p-\epsilon)\,(1 + \mathtt{OPT})$.

Proof. For part(b), note that we have $\mathbb{E}[\tau] \le \lfloor B+1 \rfloor/(p-\epsilon)$, so $\mathtt{OPT} \le \lfloor B+1 \rfloor/(p-\epsilon) - 1$. By part (a), the equality is achieved by the policy that always selects the best arm. □

The heart of the proof is a KL-divergence argument that bounds the number of plays of the best arm. This argument is encapsulated in the following claim, whose proof is deferred to Section 6.3.

Lemma 6.6 (Best Arm). *Assume $p \le \frac{1}{2}$ and $\frac{\epsilon}{p} \le \frac{1}{16}\sqrt{\frac{m}{B}}$. Then for any* BwK *algorithm there exists a problem instance in* $\mathcal{G}^{\infty}_{(p,\epsilon)}$ *such that the best arm is chosen at most* $\frac{3}{4}\,\mathtt{OPT}$ *times in expectation.*

Armed with this bound and Corollary 6.5(c), it is easy to lower-bound regret over $\mathcal{G}^{\infty}_{(p,\epsilon)}$.

Claim 6.7 (Regret). *If $p \le \frac{1}{2}$ and $\frac{\epsilon}{p} \le \frac{1}{16}\sqrt{\frac{m}{B}}$, then any* BwK *algorithm incurs regret* $\frac{\epsilon}{4p}\,\mathtt{OPT}$ *over* $\mathcal{G}^{\infty}_{(p,\epsilon)}$.

Proof. Fix any algorithm ALG for BwK. Consider the problem instance whose existence is guaranteed by Lemma 6.6. Let τ be the stopping time of ALG, and let N_t be the number of rounds $s \le t$ in which the best arm is selected. By Lemma 6.6, we have $\mathbb{E}[N_\tau] \le \frac{3}{4}\,\mathtt{OPT}$. Plugging this into Corollary 6.5(c) and rearranging the terms, we obtain $\mathbb{E}[\tau] \le (1 + \mathtt{OPT})(1 - \frac{\epsilon}{4p})$. Therefore, regret of ALG is $\mathtt{OPT} - (\mathbb{E}[\tau] - 1) \ge \frac{\epsilon}{4p}\,\mathtt{OPT}$. □

Thus, we have proved the lower bound for the infinite time horizon.

Finite time horizon. Let us "translate" a regret bound for $\mathcal{G}^{\infty}_{(p,\epsilon)}$ into a regret bound for $\mathcal{G}_{(p,\epsilon)}$.

We will need a more nuanced notation for OPT. Consider the family of problem instances in $\mathcal{G}_{(p,\epsilon)} \cup \mathcal{G}^{\infty}_{(p,\epsilon)}$ with a particular time horizon $T \le \infty$. Let $\mathtt{OPT}_{(p,\epsilon,T)}$ be the optimal expected total reward for this family (by symmetry, this quantity does not depend on which arm is the best arm). We will write $\mathtt{OPT}_T = \mathtt{OPT}_{(p,\epsilon,T)}$ when parameters (p,ϵ) are clear from the context.

Claim 6.8. *For any fixed (p, ϵ) and any $T > \mathsf{OPT}_\infty$ it holds that $\mathsf{OPT}_T \geq \mathsf{OPT}_\infty - \mathsf{OPT}_\infty^2/T$.*

Proof. Let τ^* be the stopping time of a policy that always plays the best arm on a problem instance in $\mathcal{G}^\infty_{(p,\epsilon)}$.

$$\begin{aligned}\mathsf{OPT}_\infty - \mathsf{OPT}_T &= \mathbb{E}[\tau^*] - \mathbb{E}[\min(\tau^*, T)]\\ &= \sum_{t>T} (t-T)\ \Pr[\tau^* = t]\\ &= \sum_{t>T} \Pr[\tau^* \geq t]\\ &\leq \mathbb{E}[\tau^*]/T^2 = \mathsf{OPT}_\infty^2/T.\end{aligned}$$

The inequality is due to Fact 6.3. □

Claim 6.9. *Fix (p, ϵ) and fix algorithm ALG. Let REG_T be the regret of ALG over the problem instances in $\mathcal{G}_{(p,\epsilon)} \cup \mathcal{G}^\infty_{(p,\epsilon)}$ with a given time horizon $T \leq \infty$. Then, $\mathsf{REG}_T \geq \mathsf{REG}_\infty - \mathsf{OPT}_\infty^2/T$.*

Proof. For each problem instance $\mathcal{I} \in \mathcal{G}^\infty_{(p,\epsilon)}$, let $\mathsf{REW}_T(\mathcal{I})$ be the expected total reward of ALG on $\mathcal{I}$, if the time horizon is $T \leq \infty$. Clearly, $\mathsf{REW}_\infty(\mathcal{I}) \geq \mathsf{REW}_T(\mathcal{I})$. Therefore, using Claim 6.8, we have:

$$\begin{aligned}\mathsf{REG}_T &= \mathsf{OPT}_T - \inf_{\mathcal{I}} \mathsf{REW}_T(\mathcal{I})\\ &\geq \mathsf{OPT}_T - \inf_{\mathcal{I}}\ \mathsf{REW}_\infty(\mathcal{I})\\ &= \mathsf{REG}_\infty + \mathsf{OPT}_T - \mathsf{OPT}_\infty\\ &\geq \mathsf{REG}_\infty - \mathsf{OPT}_\infty^2/T.\end{aligned}$$ □

Lemma 6.10 (Regret: Finite Time Horizon). *Fix $p \leq \frac{1}{2}$ and $\epsilon = \frac{p}{16}\min(1, \sqrt{m/B})$. Then for any time horizon $T > \frac{8p}{\epsilon}\mathsf{OPT}_\infty$ and any BwK algorithm ALG there exists a problem instance in $\mathcal{G}_{(p,\epsilon)}$ with time horizon T for which ALG incurs regret $\Omega(\mathsf{OPT}_T)\min(1, \sqrt{m/B})$.*

Proof. By Claim 6.7, ALG incurs regret at least $\frac{\epsilon}{4p}\mathsf{OPT}_\infty$ for some problem instance in $\mathcal{G}^\infty_{(p,\epsilon)}$. By Claim 6.9, ALG incurs regret at least $\frac{\epsilon}{8p}\mathsf{OPT}_\infty$ for the same problem instance in $\mathcal{G}_{(p,\epsilon)}$ with time horizon T. Since $\mathsf{OPT}_\infty \geq \mathsf{OPT}_T$, this regret is at least $\frac{\epsilon}{8p}\mathsf{OPT}_T = \Omega(\mathsf{OPT}_T)\min(1, \sqrt{m/B})$. □

Let us complete the proof of Claim 6.2(b). Recall that Claim 6.2(b) specifies the values for (m, B, OPT) that our problem instance must have. Since we have already proved Claim 6.2(a) and $\mathsf{OPT}\sqrt{\frac{m}{B}} \leq O(\sqrt{m\,\mathsf{OPT}})$ for $\mathsf{OPT} < 3B$, it suffices to assume $\mathsf{OPT} \geq 3B$.

Let $\epsilon(p) = \frac{p}{16}\min(1, \sqrt{m/B})$, as prescribed by Lemma 6.10. Then taking $\epsilon = \epsilon(p)$, we obtain regret $\Omega(\mathsf{OPT}_T)\min(1, \sqrt{m/B})$ for any parameter $p \leq \frac{1}{2}$ and any time horizon $T > \frac{8p}{\epsilon}\mathsf{OPT}_{(p,\epsilon,\infty)}$. It remains to pick such p and T to ensure that $f(p,T) = \mathsf{OPT}$, where $f(p,T) = \mathsf{OPT}_{(p,\epsilon(p),T)}$.

Recall from Corollary 6.5(b) that $\mathsf{OPT}_{(p,\epsilon,\infty)} = \frac{\Gamma}{p} - 1$, where

$$\Gamma = \lfloor B+1 \rfloor / \left(1 - \frac{1}{16}\min(1, \sqrt{m/B})\right)$$

is a "constant" for the purposes of this argument, in the sense that it does not depend on p or T. So, we can state the sufficient condition for proving Claim 6.2(b) as follows:

$$\text{Pick } p \leq \frac{1}{2} \text{ and } T \geq \frac{8\Gamma}{\epsilon(p)} \text{ such that } f(p,T) = \mathsf{OPT}. \tag{41}$$

Recall that $\mathtt{OPT}_{(p,\epsilon,\infty)} \geq \mathtt{OPT}_{(p,\epsilon,T)}$ for any T, and $\mathtt{OPT}_{(p,\epsilon,T)} \geq \frac{1}{2}\mathtt{OPT}_{(p,\epsilon,\infty)}$ for any $T > 2\,\mathtt{OPT}_{(p,\epsilon,\infty)}$ by Claim 6.8. We summarize this as follows: for any $T > 2(\frac{\Gamma}{p} - 1)$,

$$\frac{\Gamma}{p} - 1 \geq \mathtt{OPT}_{(p,\epsilon,T)} \geq \frac{1}{2}\left(\frac{\Gamma}{p} - 1\right). \tag{42}$$

Define $p_0 = \Gamma/\mathtt{OPT}$. Since $\mathtt{OPT} \geq 3B$, $\Gamma \leq \frac{16}{15}(B+1)$ and $B \geq 4$, it follows that $p_0 \geq \frac{1}{2}$. Let $T = \frac{8\Gamma}{\epsilon(p_0)}$. Then, Equation (42) holds for all $p \in [p_0/4, \frac{1}{2}]$. In particular,

$$f(p_0, T) \leq \Gamma/p_0 = \mathtt{OPT} \leq f(p_0/4, T).$$

Since $f(p,T)$ is continuous in p, there exists $p \in [p_0/4, p_0]$ such that $f(p,T) = \mathtt{OPT}$. Since $p \leq p_0$, we have $T \geq \frac{8\Gamma}{\epsilon(p)}$, satisfying all requirements in Equation (41). This completes the proof of Claim 6.2(b), and therefore the proof of Theorem 6.1.

6.2 Background on KL-Divergence (For the Proof of Lemma 6.6)

The proof of Lemma 6.6 relies on the concept of KL-divergence. Let us provide some background to make on KL-divergence to make this proof self-contained. We use a somewhat non-standard notation that is tailored to the needs of our analysis.

The *KL-divergence* (a.k.a. *relative entropy*) is defined as follows. Consider two distributions μ, ν on the same finite universe Ω.[13] Assume $\mu \ll \nu$ (in words, μ is *absolutely continuous* with respect to ν), meaning that $\nu(w) = 0 \Rightarrow \mu(w) = 0$ for all $w \in \Omega$. Then KL-divergence of μ given ν is

$$\mathsf{KL}\,(\mu \,\|\, \nu) \triangleq \mathop{\mathbb{E}}_{w \sim (\Omega, \mu)} \log\left(\frac{\mu(w)}{\nu(w)}\right) = \sum_{w \in \Omega} \log\left(\frac{\mu(w)}{\nu(w)}\right)\mu(w).$$

In this formula, we adopt a convention that $\frac{0}{0} = 1$. We will use the fact that

$$\mathsf{KL}\,(\mu \,\|\, \nu) \geq \frac{1}{2}\,\|\mu - \nu\|_1^2. \tag{43}$$

Henceforth, let μ, ν be distributions on the universe Ω^∞, where Ω is a finite set. For $\vec{w} = (w_1, w_2, \ldots) \in \Omega^\infty$ and $t \in \mathbb{N}$, let us use the notation $\vec{w}_t = (w_1, \ldots, w_t) \in \Omega^t$. Let μ_t be a restriction of μ to Ω^t: that is, a distribution on Ω^t given by

$$\mu_t(\vec{w}_t) \triangleq \mu\,(\{\vec{u} \in \Omega^\infty : \vec{u}_t = \vec{w}_t\}).$$

The *next-round conditional distribution* of μ given $\vec{w}_t$, $t < T$ is defined by

$$\mu\,(w_{t+1} \,|\, \vec{w}_t) \triangleq \frac{\mu_{t+1}(\vec{w}_{t+1})}{\mu_t(\vec{w}_t)}.$$

Note that $\mu(\cdot\,|\vec{w}_t)$ is a distribution on Ω for every fixed $\vec{w}_t$.

The *conditional KL-divergence* at round $t+1$ is defined as

$$\mathsf{KL}_{t+1}(\mu \,\|\, \nu) \triangleq \mathop{\mathbb{E}}_{\vec{w}_t \sim (\Omega^t,\ \mu_t)} \mathsf{KL}\,(\mu(\cdot\,|\vec{w}_t) \,\|\, \nu(\cdot\,|\vec{w}_t)).$$

In other words, this is the KL-divergence between the next-round conditional distributions $\mu(\cdot\,|\vec{w}_t)$ and $\nu(\cdot\,|\vec{w}_t)$, in expectation over the random choice of $\vec{w}_t$ according to distribution μ_t.

[13] We use μ, ν to denote distributions throughout this section, whereas μ denotes the latent structure elsewhere in the article.

We will use the following fact, known as the *chain rule* for KL-divergence:

$$\mathsf{KL}\,(\mu_T \,\|\, \nu_T) = \sum_{t=1}^{T} \mathsf{KL}_t(\mu \,\|\, \nu), \quad \text{for each } T \in \mathbb{N}. \tag{44}$$

Here, for notational convenience, we define $\mathsf{KL}_1(\mu \,\|\, \nu) \triangleq \mathsf{KL}\,(\mu_1 \,\|\, \nu_1)$.

6.3 The KL-Divergence Argument: Proof of Lemma 6.6

Fix some BwK algorithm ALG and fix parameters (p, ϵ). Let $\mathcal{I}_x$ be the problem instance in $\mathcal{G}^{\infty}_{(p,\epsilon)}$ in which the best arm is x. For the analysis, we also consider an instance $\mathcal{I}_0$, which coincides with $\mathcal{I}_x$ but has no best arm: that is, all arms have expected resource consumption p. Let $\tau(\mathcal{I})$ be the stopping time of ALG for a given problem instance $\mathcal{I}$, and let $N_x(\mathcal{I})$ be the expected number of times a given arm x is chosen by ALG on this problem instance.

Consider problem instance $\mathcal{I}_0$. Since all arms are the same, we can apply Corollary 6.5(a) (suitably modified to the non-best arm) and obtain $\mathbb{E}[\tau(\mathcal{I}_0)] = \lfloor B+1 \rfloor / p$. We focus on an arm x with the smallest $N_x(\mathcal{I}_0)$. For this arm, it holds that

$$N_x(\mathcal{I}_0) \le \frac{1}{m} \sum_{x \in X} N_x(\mathcal{I}_0) = \frac{1}{m} \mathbb{E}[\tau(\mathcal{I}_0)] \le \frac{\lfloor B+1 \rfloor}{p\, m}. \tag{45}$$

In what follows, we use this inequality to upper-bound $N_x(\mathcal{I}_x)$. Informally, if arm x is not played sufficiently often in $\mathcal{I}_0$, then ALG cannot tell apart $\mathcal{I}_0$ and $\mathcal{I}_x$.

The *transcript* of ALG on a given problem instance $\mathcal{I}$ is a sequence of pairs $\{(x_t, c_t)\}_{t \in \mathbb{N}}$, where for each round $t \le \tau(\mathcal{I})$ it holds that x_t is the arm chosen by ALG and c_t is the realized resource consumption in that round. For all $t > \tau(\mathcal{I})$, we define $(x_t, c_t) = (\mathtt{null}, 0)$. To map this to the setup in Section 6.2, denote $\Omega = (X \cup \{\mathtt{null}\}) \times \{0,1\}$. Then the set of all possible transcripts is a subset of Ω^{∞}.

Every given problem instance $\mathcal{I}$ induces a distribution over Ω^{∞}. Let μ, ν be the distributions over Ω^{∞} that are induced by $\mathcal{I}_0$ and $\mathcal{I}_x$, respectively. We will use the following shorthand:

$$\mathtt{diff}[T_0, T_*] \triangleq \sum_{t=T_0}^{T_*} \nu(x_t = x) - \mu(x_t = x), \quad \text{where } 1 \le T_0 \le T_* \le \infty.$$

For any $T \in \mathbb{N}$ (which we will fix later), we can write

$$N_x(\mathcal{I}_x) - N_x(\mathcal{I}_0) = \mathtt{diff}[1, \infty] = \mathtt{diff}[1, T] + \mathtt{diff}[T+1, \infty]. \tag{46}$$

We will bound $\mathtt{diff}[1, T]$ and $\mathtt{diff}[T+1, \infty]$ separately.

Upper bound on $\mathtt{diff}[1, T]$. This is where we use KL-divergence. Namely, by Equation (43), we have

$$\mathtt{diff}[1, T] \le \frac{T}{2}\, \|\mu_T - \nu_T\|_1 \le T \sqrt{\frac{1}{2}\, \mathsf{KL}\,(\mu_T \,\|\, \nu_T)}. \tag{47}$$

Now, by the chain rule (Equation (44)), we can focus on upper-bounding the conditional KL-divergence $\mathsf{KL}_t(\mu \,\|\, \nu)$ at each round $t \le T$.

Claim 6.11. *For each round $t \le T$, it holds that*

$$\mathsf{KL}_t(\mu \,\|\, \nu) = \mu(x_t = x)\, \left(p \log\left(\frac{p}{p-\epsilon}\right) + (1-p) \log\left(\frac{1-p}{1-p+\epsilon}\right)\right). \tag{48}$$

Proof. The main difficulty here is to carefully "unwrap" the definition of $\mathsf{KL}_t(\mu \,\|\, \nu)$.

Fix $t \leq T$ and let $\vec{w}_t \in \Omega^t$ be the partial transcript up to and including round t. For each arm y, let $f(y|\vec{w}_t)$ be the probability that ALG chooses arm y in round t, given the partial transcript $\vec{w}_t$. Let $c(y|\mathcal{I})$ be the expected resource consumption for arm y under a problem instance $\mathcal{I}$. The transcript for round $t+1$ is a pair $w_{t+1} = (x_{t+1}, c_{t+1})$, where x_{t+1} is the arm chosen by ALG in round $t+1$, and $c_{t+1} \in \{0,1\}$ is the resource consumption in that round. Therefore, if $c_{t+1} = 1$, then

$$\begin{aligned}\mu(w_{t+1}\,|\vec{w}_t) &= f(x_{t+1}|\vec{w}_t)\; c(x_{t+1}|\mathcal{I}_0) = f(x_{t+1}|\vec{w}_t)\; p,\\ \nu(w_{t+1}\,|\vec{w}_t) &= f(x_{t+1}|\vec{w}_t)\; c(x_{t+1}|\mathcal{I}_x) = f(x_{t+1}|\vec{w}_t)\;\; (p - \epsilon\, \mathbf{1}_{\{x_{t+1}=x\}}).\end{aligned}$$

Similarly, if $c_{t+1} = 0$, then

$$\begin{aligned}\mu(w_{t+1}\,|\vec{w}_t) &= f(x_{t+1}|\vec{w}_t)\; (1 - c(x_{t+1}|\mathcal{I}_0)) = f(x_{t+1}|\vec{w}_t)\; (1-p),\\ \nu(w_{t+1}\,|\vec{w}_t) &= f(x_{t+1}|\vec{w}_t)\; (1 - c(x_{t+1}|\mathcal{I}_x)) = f(x_{t+1}|\vec{w}_t)\;\; (1 - p + \epsilon\, \mathbf{1}_{\{x_{t+1}=x\}}).\end{aligned}$$

It follows that

$$\log \frac{\mu(w_{t+1}\,|\vec{w}_t)}{\nu(w_{t+1}\,|\vec{w}_t)} = \mathbf{1}_{\{x_t=x\}}\;\left(\log\left(\frac{p}{p-\epsilon}\right)\;\mathbf{1}_{\{c_{t+1}=1\}} + \log\left(\frac{1-p}{1-p+\epsilon}\right)\;\mathbf{1}_{\{c_{t+1}=0\}}\right).$$

Taking expectations over $w_{t+1} = (x_t, c_t) \sim \mu(\cdot\,|\vec{w}_t)$, we obtain

$$\mathsf{KL}\left(\mu(\cdot\,|\vec{w}_t)\,\|\,\nu(\cdot\,|\vec{w}_t)\right) = f(x|\vec{w}_t)\;\left(p\,\log\left(\frac{p}{p-\epsilon}\right) + (1-p)\,\log\left(\frac{1-p}{1-p+\epsilon}\right)\right).$$

Taking expectations over $\vec{w}_t \sim \mu_t$, we obtain the conditional KL-divergence $\mathsf{KL}_t(\mu\,\|\,\nu)$. Equation (48) follows, because

$$\mathop{\mathbb{E}}_{\vec{w}_t \sim \mu_t} f(x|\vec{w}_t) = \mu(x_t = x). \qquad \square$$

We will use the following fact about logarithms, which is proved using standard quadratic approximations for the logarithm. The proof is in Section D.

Fact 6.12. *Assume $\frac{\epsilon}{p} \leq \frac{1}{2}$ and $p \leq \frac{1}{2}$. Then,*

$$p\,\log\left(\frac{p}{p-\epsilon}\right) + (1-p)\,\log\left(\frac{1-p}{1-p+\epsilon}\right) \leq \frac{2\epsilon^2}{p}.$$

Now, we can put everything together and derive an upper bound on $\mathtt{diff}[1,T]$.

Claim 6.13. *Assume $\frac{\epsilon}{p} \leq \frac{1}{2}$ and $p \leq \frac{1}{2}$. Then, $\mathtt{diff}[1,T] \leq T\,\frac{\epsilon}{p}\,\sqrt{\frac{B+1}{m}}$.*

Proof. By Claim 6.11 and Fact 6.12, for each round $t \leq T$, we have

$$\mathsf{KL}_t(\mu\,\|\,\nu) \leq \frac{2\epsilon^2}{p}\;\mu(x_t = x).$$

By the chain rule (Equation (44)), we have

$$\mathsf{KL}\left(\mu_T\,\|\,\nu_T\right) \leq \frac{2\epsilon^2}{p}\;\sum_{t=1}^{T}\mu(x_t = x) \leq \frac{2\epsilon^2}{p}\;N_x(\mathcal{I}_0) \leq 2\;\frac{B+1}{m}\;\left(\frac{\epsilon}{p}\right)^2.$$

The last inequality is the place where we use our choice of x, as expressed by Equation (45).

Plugging this back into Equation (47), we obtain $\mathtt{diff}[1,T] \leq T\,\frac{\epsilon}{p}\,\sqrt{\frac{B+1}{m}}$. $\square$

Upper bound on $\mathtt{diff}[T,\infty]$. Consider the problem instance $\mathcal{I}_x$, and consider the policy that always chooses the best arm. Let ν^* be the corresponding distribution over transcripts Ω^∞, and let τ be the corresponding stopping time. Note that $\nu^*(x_t = x)$ if and only if $\tau > t$. Therefore:

$$\mathtt{diff}[T,\infty] \le \sum_{t=T}^{\infty} \nu(x_t = x) \le \sum_{t=T}^{\infty} \nu^*(x_t = x) = \sum_{t=T}^{\infty} \nu^*(\tau > t) \le \mathtt{OPT}^2/T.$$

The second inequality can be proved using a simple "coupling argument." The last inequality follows from Fact 6.3, observing that $\mathbb{E}[\tau] = \mathtt{OPT}$.

Putting the pieces together. Assume $p \le \frac{1}{2}$ and $\frac{\epsilon}{p} \le \frac{1}{2}$. Denote $\gamma = \frac{\epsilon}{p}\sqrt{\frac{B+1}{m}}$. Using the upper bounds on $\mathtt{diff}[1,T]$ and $\mathtt{diff}[T+1,\infty]$ and plugging them into Equation (46), we obtain

$$N_x(\mathcal{I}_x) - N_x(\mathcal{I}_0) \le \gamma T + \mathtt{OPT}^2/T \le \mathtt{OPT}\sqrt{\gamma}$$

for $T = \mathtt{OPT}/\sqrt{\gamma}$. Recall that $N_x(\mathcal{I}_0) < \mathtt{OPT}/m$. Thus, we obtain

$$N_x(\mathcal{I}_x) \le (\frac{1}{m} + \sqrt{\gamma})\,\mathtt{OPT}.$$

Recall that we need to conclude that $N_x(\mathcal{I}_x) \le \frac{3}{4}\mathtt{OPT}$. For that, it suffices to have $\gamma \le \frac{1}{16}$.

7 BWK WITH PREADJUSTED DISCRETIZATION

In this section, we develop a general technique for preadjusted discretization, and apply it to dynamic pricing with a single product and dynamic procurement with a single budget. For both applications, our regret bounds significantly improve over prior work. While the dynamic pricing application is fairly straightforward given the general result, the dynamic procurement application takes some work and uses a non-standard mesh of prices. We also obtain an initial result for dynamic pricing with multiple products. The main technical challenge is to upper-bound the discretization error; we can accomplish this whenever the expected resource to expected consumption ratio of each arm can be expressed in a particularly simple way.

7.1 Preadjusted Discretization as a General Technique

The high-level idea behind preadjusted discretization is to apply an existing BwK algorithm with a restricted, finite action space $S \subset X$ that is chosen in advance. Typically S is, in some sense, "uniformly spaced" in X, and its "granularity" is tuned in advance to minimize regret.

Consider a problem instance with action space restricted to S. Let $\mathtt{REW}(S)$ be the algorithm's reward on this problem instance, and let $\mathtt{OPT}_{\mathtt{LP}}(S)$ be the corresponding value of $\mathtt{OPT}_{\mathtt{LP}}$, as defined in Section 3. $\mathtt{OPT}(X)$ and $\mathtt{OPT}_{\mathtt{LP}}(X)$ will refer to the corresponding quantities for the original action space X. The key two quantities in our analysis of preadjusted discretization are

$$\begin{aligned} R(S) &= \mathtt{OPT}_{\mathtt{LP}}(S) - \mathtt{REW}(S) && (S\textit{-regret}) \\ \mathtt{Err}(S|X) &= \mathtt{OPT}_{\mathtt{LP}}(X) - \mathtt{OPT}_{\mathtt{LP}}(S) && (\textit{discretization error of } S \textit{ relative to } X). \end{aligned} \tag{49}$$

Note that algorithm's regret can be expressed as

$$\mathtt{OPT}(X) - \mathtt{REW}(S) \le \mathtt{OPT}_{\mathtt{LP}}(X) - \mathtt{REW}(S) = R(S) + \mathtt{Err}(S|X).$$

Now, suppose S is parameterized by $\epsilon > 0$, which controls its "granularity." Adjusting the ϵ involves balancing $R(S)$ and $\mathtt{Err}(S|X)$: indeed, decreasing ϵ tends to increase $R(S)$ but decrease $\mathtt{Err}(S|X)$. We upper-bound the S-regret via our main algorithmic result[14]; the challenge is to upper-bound $\mathtt{Err}(S|X)$.

[14] We need to use the regret bound in terms of the best known upper bound on OPT, rather than OPT itself, because the latter is not known to the algorithm. For example, for dynamic pricing one can use $\mathtt{OPT} \le B$.

A typical scenario where one would want to apply preadjusted discretization is when an algorithm chooses among prices. More formally, each arm includes a real-valued vector of prices in $[0,1]$ (and perhaps other things, such as the maximal number of items for sale). The restricted action set S consists of all arms such that all prices belong to a suitably chosen mesh $M \subset [0,1]$ with granularity ϵ. There are several types of meshes one could consider, depending on the particular BwK domain. The most natural ones are the *ϵ-additive mesh*, with prices that are integer multiples of ϵ, and *ϵ-multiplicative mesh mesh*, with prices of the form $(1-\epsilon)^\ell$, $\ell \in \mathbb{N}$. Both have been used in the prior work on MAB in metric spaces [36, 37, 41, 45]) and dynamic pricing (e.g., References [12, 16, 18, 39]). Somewhat surprisingly, for dynamic procurement, we find it optimal to use a very different mesh, called *ϵ-hyperbolic mesh*, in which the prices are of the form $\frac{1}{1+\epsilon\ell}$, $\ell \in \mathbb{N}$.

While in practice the action set X is usually finite (although possibly very large), it is mathematically more elegant to consider infinite X. For example, we prefer to allow arbitrary fractional prices, even though in practice they may have to be rounded to whole cents. However, recall that $\mathtt{OPT}_{\mathtt{LP}}$ in Section 3 is only defined for a finite action space X. To handle infinite X, we define

$$\mathtt{OPT}_{\mathtt{LP}}(X) = \sup_{\text{finite } X' \subset X} \mathtt{OPT}_{\mathtt{LP}}(X'). \tag{50}$$

In line with Lemma 3.1, let us argue that $\mathtt{OPT}_{\mathtt{LP}}(X) \geq \mathtt{OPT}(X)$ even when X is infinite. Specifically, we prove this for all versions of dynamic pricing and dynamic procurement, and more generally for any $\mathtt{BwK}$ domain such that for each arm there are only finitely many possible outcome vectors.

Lemma 7.1. *Consider a* $\mathtt{BwK}$ *domain with infinite action space X, such that for each arm there are only finitely many possible outcome vectors. Then* $\mathtt{OPT}_{\mathtt{LP}}(X) \geq \mathtt{OPT}(X)$.

Proof. Fix a problem instance, and consider an optimal dynamic policy for this instance. W.l.o.g. this policy is deterministic.[15] For each round, this policy defines a deterministic mapping from histories to arms to be played in this round. Since there are only finitely many possible histories, the policy can only use a finite subset of arms, call it $X' \subset X$. By Lemma 3.1, we have

$$\mathtt{OPT}_{\mathtt{LP}}(X) \geq \mathtt{OPT}_{\mathtt{LP}}(X') \geq \mathtt{OPT}(X') = \mathtt{OPT}(X). \qquad \square$$

7.2 A General Bound on Discretization Error

We develop a general bound on discretization error $\mathtt{Err}(S|X)$, as defined in Equation (49). To this end, we consider the expected reward to expected consumption ratios of arms (and the differences between them), whereas in the work on MAB in metric spaces it suffices to consider the difference in expected rewards.

To simplify notation, we suppress μ, the (actual) latent structure: e.g., we will write $c_i(\mathcal{D}) = c_i(\mathcal{D},\mu)$, $r(\mathcal{D}) = r(\mathcal{D},\mu)$, and $\mathtt{LP}(\mathcal{D},\mu) = \mathtt{LP}(\mathcal{D})$ for distributions $\mathcal{D}$ and resources i.

Definition 7.2. We say that arm x *ϵ-covers* arm y if the following two properties are satisfied for each resource i such that $c_i(x) + c_i(y) > 0$:

(i) $r(x)/c_i(x) \geq r(y)/c_i(y) - \epsilon$.
(ii) $c_i(x) \geq c_i(y)$.

A subset $S \subset X$ of arms is called an *ϵ-discretization* of X if each arm in X is ϵ-covered by some arm in S.

Theorem 7.3 (Preadjusted Discretization). *Fix a BwK domain with action space X. Let $S \subset X$ be an ϵ-discretization of X, for some $\epsilon \geq 0$. Then the discretization error* $\mathtt{Err}(S|X)$ *is at most ϵdB. Consequently, for any algorithm with S-regret $R(S)$, we have* $\mathtt{OPT}_{\mathtt{LP}}(X) - \mathtt{REW}(S) = R(S) + \epsilon dB$.

[15] A randomized policy can be seen as a distribution over deterministic policies, so one of these deterministic policies must have same or better expected total reward.

Proof. We need to prove that $\mathtt{Err}(S|X) \leq \epsilon dB$. If X is infinite, then (by Equation (50)) it suffices to prove $\mathtt{Err}(S|X') \leq \epsilon dB$ for any finite subset of $X' \subset X$. Let $\mathcal{D}$ be the distribution over arms in X', which maximizes $\mathsf{LP}(\mathcal{D}, \mu)$. We use $\mathcal{D}$ to construct a distribution $\mathcal{D}_S$ over S, which is nearly as good.

We define $\mathcal{D}_S$ as follows. Since S is an ϵ-discretization of X, there exists a family of subsets $(\mathtt{cov}(x) \subset X : x \in S)$ so that each arm $x \in S$ ϵ-covers all arms in $\mathtt{cov}(x)$, the subsets are disjoint, and their union is X. Fix one such family of subsets, and define

$$\mathcal{D}_S(x) = \sum_{y \in \mathtt{cov}(x)} \mathcal{D}(y) \min_{i:\, c_i(x)>0} \frac{c_i(y)}{c_i(x)}, \qquad x \in S.$$

Note that $\sum_{x \in S} \mathcal{D}_S(S) \leq 1$ by Definition 7.2(ii). With the remaining probability, the null arm is chosen (i.e., the algorithm skips a given round).

To argue that $\mathsf{LP}(\mathcal{D}_S, \mu)$ is large, we upper-bound the resource consumption $c_i(\mathcal{D}_S)$, for each resource i, and lower-bound the reward $r(\mathcal{D}_S)$:

$$\begin{aligned} c_i(\mathcal{D}_S) &= \sum_{x \in S} c_i(x)\, \mathcal{D}_S(x) \\ &\leq \sum_{x \in S} c_i(x) \sum_{y \in \mathtt{cov}(x):\, c_i(x)>0} \mathcal{D}(y) \frac{c_i(y)}{c_i(x)} \\ &= \sum_{x \in S} \sum_{y \in \mathtt{cov}(x):\, c_i(x)>0} \mathcal{D}(y)\, c_i(y) \\ &= \sum_{y \in X} \mathcal{D}(y)\, c_i(y) \\ &= c_i(\mathcal{D}). \end{aligned} \tag{51}$$

(Note that the above argument did not use the property (i) in Definition 7.2.)

In what follows, for each arm x define $I_x = \{i : c_i(x) > 0\}$:

$$\begin{aligned} r(\mathcal{D}_S) &= \sum_{x \in S} r(x)\, \mathcal{D}_S(x) \\ &= \sum_{x \in S} r(x) \sum_{y \in \mathtt{cov}(x)} \mathcal{D}(y) \min_{i \in I_x} \frac{c_i(y)}{c_i(x)} \\ &= \sum_{x \in S} \sum_{y \in \mathtt{cov}(x)} \mathcal{D}(y) \min_{i \in I_x} \frac{c_i(y)\, r(x)}{c_i(x)} \\ &\geq \sum_{x \in S} \sum_{y \in \mathtt{cov}(x)} \mathcal{D}(y) \min_{i \in I_x} r(y) - \epsilon c_i(y) \qquad \text{(by Definition 7.2(i))} \\ &= \sum_{y \in X} \mathcal{D}(y) \min_{i} r(y) - \epsilon c_i(y) \\ &\geq \sum_{y \in X} \mathcal{D}(y) \left(r(y) - \epsilon \sum_i c_i(y) \right) \\ &= r(\mathcal{D}) - \epsilon \sum_i c_i(\mathcal{D}). \end{aligned} \tag{52}$$

Let $\tau(\mathcal{D}) = \min_i \frac{B}{c_i(\mathcal{D})}$ be the stopping time in the linear relaxation, so that $\mathsf{LP}(\mathcal{D}) = \tau(\mathcal{D})\, r(\mathcal{D})$. By Equation (51), we have $\tau(\mathcal{D}_S) \geq \tau(\mathcal{D})$. We are ready for the final computation:

$$\begin{aligned}
\mathsf{LP}(\mathcal{D}_S) &= \tau(\mathcal{D}_S)\, r(\mathcal{D}_S) \\
&\geq \tau(\mathcal{D})\, r(\mathcal{D}_S) \\
&\geq \tau(\mathcal{D}) \left(r(\mathcal{D}) - \epsilon \sum_i c_i(\mathcal{D}) \right) && \text{(by Equation (52))} \\
&\geq r(\mathcal{D})\, \tau(\mathcal{D}) - \epsilon\, \tau(\mathcal{D}) \sum_i c_i(\mathcal{D}) \\
&\geq \mathsf{LP}(\mathcal{D}) - \epsilon\, d\, B. && \square
\end{aligned}$$

7.3 Preadjusted Discretization for Dynamic Pricing

We apply the machinery developed above to handle the basic version of dynamic pricing, as defined in Section 1.1. In fact, our technique easily generalizes to multiple products, in a particular scenario that we call *dynamic bundle-pricing*. We present the more general result directly.

The dynamic bundle-pricing problem is defined as follows. There are d products, with limited supply of each, and T rounds. In each round, a new buyer arrives, an algorithm chooses a bundle of products and a price, and offers this bundle for this price. The offer is either accepted or rejected. The bundle is a vector $(b_1, \ldots, b_d)$, so that $b_i \in \mathbb{N}$ units of each product i are offered. We assume that the bundle must belong to a fixed collection $\mathcal{F}$ of allowed bundles. Buyers' valuations over bundles can be arbitrary (in particular, not necessarily additive); they are drawn independently from a fixed distribution over valuations. For normalization, we assume that each buyer's valuation for any bundle of ℓ units lies in the interval $[0, \ell]$; accordingly, the offered price for such bundle can w.l.o.g. be restricted to the same interval.

Theorem 7.4. *Consider the dynamic bundle-pricing problem such that there are d products, each with supply B. Assume each allowed bundle consists of at most ℓ items, and prices are in $[0, \ell]$. Algorithm* `PrimalDualBwK` *with an ϵ-additive mesh, for some $\epsilon = \epsilon(B, |\mathcal{F}|, \ell)$, has regret $\widetilde{O}(d\, B^{2/3}\, (|\mathcal{F}|\ell)^{1/3})$.*

The basic version from Section 1.1 is a special case with a single product and a single allowed bundle that consists of one unit of this product. Taking $d = \ell = |\mathcal{F}| = 1$ in Theorem 7.4, we obtain regret $\widetilde{O}(B^{2/3})$. This regret bound is optimal for any pair (B, T), as proved in Babaioff et al. [12].

Corollary 7.5. *Consider the dynamic pricing problem, as defined Section 1.1. Algorithm* `PrimalDualBwK` *with an ϵ-additive mesh, for a suitably chosen $\epsilon = \epsilon(B)$, has regret $\widetilde{O}(B^{2/3})$.*

Proof of Theorem 7.4. First, let us cast this problem as a BwK domain. To ensure that per-round rewards and per-round resource consumptions lie in $[0, 1]$, we scale them down by the factor of ℓ. Accordingly, the rescaled supply constraint is $B' = B/\ell$. In what follows, consider the scaled-down problem instance.

An arm is a pair $x = (b, p)$, where $b \in \mathcal{F}$ is a bundle and $p \in [0, 1]$ is the offered price. Let $F(x)$ be the probability of a sale for this arm, divided by ℓ; this probability is non-increasing in p for a fixed bundle b. Then expected per-round reward is $r(x) = p\, F(x)$, and expected per-round consumption of product i is $c_i(x) = b_i\, F(x)$. Therefore,

$$\frac{r(x)}{c_i(x)} = \frac{p}{b_i}, \quad \text{for each arm } x = (b, p) \text{ and product } i. \tag{53}$$

This is a crucial domain-specific property that enables preadjusted discretization. It follows that for any arm $x = (b, p)$, this arm ϵ-covers any arm $x' = (b, p')$ such that $p' - \epsilon \le p \le p'$. Therefore, an ϵ-additive mesh S is an ϵ-discretization, for any $\epsilon > 0$.

Consider algorithm `PrimalDualBwK` with action space S. Using Theorem 5.3 and observing that $\mathtt{OPT}_{\mathtt{LP}} \le dB'$, we obtain S-regret

$$R(S) \le \widetilde{O}(d\sqrt{B'\,|S|}) = \widetilde{O}(d\sqrt{B'\,|\mathcal{F}|/\epsilon}).$$

By Theorem 7.3 discretization error is $\mathtt{Err}(S|X) \le \epsilon dB'$. So, regret relative to $\mathtt{OPT}_{\mathtt{LP}}$ is

$$R(S) + \mathtt{Err}(S|X) \le \widetilde{O}(d\sqrt{B'\,|\mathcal{F}|/\epsilon} + \epsilon dB') \le \widetilde{O}(d)\,(B/\ell)^{2/3}\,|\mathcal{F}|^{1/3}$$

for a suitably chosen $\epsilon = (B/\ell)^{-1/3}\,|\mathcal{F}|^{1/3}$. Recall that this is regret for the rescaled problem instance. For the original problem instance, rewards are scaled up by the factor of ℓ, so regret is scaled up by ℓ, too. □

One can easily extend Theorem 7.4 to a setting where in each round an algorithm offers several copies of the same bundle for the same per-bundle price, and an agent can choose how many copies to buy (if any). More precisely, in each round an algorithm chooses two things: a bundle from $\mathcal{F}$ and the number of copies of this bundle. The latter is restricted to be at most Λ, where Λ is a known parameter. We call this setting *dynamic bundle-pricing with multiplicity* Λ. The algorithm and analysis is essentially the same.

Theorem 7.6. *Consider dynamic bundle-pricing with multiplicity Λ. Assume that each product has supply B, and each allowed bundle consists of at most ℓ items. Algorithm* `PrimalDualBwK` *with an ϵ-additive mesh, for a suitably chosen $\epsilon = \epsilon(B, |\mathcal{F}|, \ell, \Lambda)$, has regret $\widetilde{O}(d\,(B\Lambda)^{2/3}\,(|\mathcal{F}|\ell)^{1/3})$.*

7.4 Preadjusted Discretization for Dynamic Procurement

Application to dynamic procurement takes a little more work and results in a weaker regret bound, compared to the application to dynamic pricing. The main reason is that the natural mesh for dynamic procurement is ϵ-hyperbolic (rather than ϵ-additive). One needs to bound this mesh from below to make it finite, which increases the mesh size and the discretization error.

While our main goal here is to handle the basic version of dynamic procurement, as defined in Section 1.1, the same technique easily extends to a generalization where the algorithm can buy multiple items in each round. The generalization is defined as follows. In each round t, the algorithm offers to buy up to Λ units at price p_t per unit, where $p_t \in [0, 1]$ is chosen by the algorithm. The outcome is summarized by the number k_t of items bought, where k_t is an independent sample from some fixed (but unknown) distribution parameterized by p_t and Λ. The algorithm is constrained by the time horizon T, budget B, and per-round supply constraint Λ. We prove the following:

Theorem 7.7. *Consider dynamic procurement with up to Λ items bought per round. Algorithm* `PrimalDualBwK` *with a suitably chosen action space S yields regret $\tilde{O}(\Lambda^{5/4}T/B^{1/4})$. Specifically, $S = [p_0, 1] \cap M$, where M is the ϵ-hyperbolic mesh, for some parameters $\epsilon, p_0 \in (0, 1)$ that depend only on B, T, and Λ.*

Let us model this problem as a `BwK` domain. The action space is $X = [0, 1]$: the arms correspond to all possible prices. (The zero price corresponds to the "null arm".) To ensure that rewards and consumptions lie in $[0, 1]$, we scale them down by a factor of Λ, as in Section 7.3, so that reward in any round t is k_t/Λ, and budget consumption is $p_t\,k_t/\Lambda$. Accordingly, budget is rescaled to $B' = B/\Lambda$. Henceforth, consider the scaled-down problem instance, unless specified otherwise.

Let $F(p)$ be the expected per-round number of items sold for a given price p, divided by Λ; note that it is non-decreasing in p. Then expected budget consumption is $c(p) = p\,F(p)$, and expected reward is simply $r(p) = F(p)$. It follows that

$$\frac{r(p)}{c(p)} = \frac{1}{p}, \quad \text{for each price } p.$$

Like Equation (53), this is a crucial domain-specific property that enables preadjusted discretization.

By Definition 7.2 price p ϵ-covers price arm q if and only if $q < p$ and $\frac{1}{p} \geq \frac{1}{q} - \epsilon$. This makes the hyperbolic mesh a natural mesh for this problem, rather than additive or multiplicative ones. It is easy to see that the ϵ-hyperbolic mesh S on X is an ϵ-discretization of X: namely, each price q is ϵ-covered by the smallest price $p \geq q$ that lies in S.

Unfortunately, this mesh has infinitely many points. In fact, it is easy to see that any ϵ-discretization on X must be infinite, even for $\Lambda = 1$. To obtain a finite ϵ-discretization, we only consider prices $p \geq p_0$, for some parameter p_0 to be tuned later. Below we argue that this restriction is not too damaging:

Claim 7.8. *Consider dynamic procurement with non-unit supply. Then, for any $p_0 \in (0, 1)$ it holds that*

$$\mathtt{OPT}_{\mathtt{LP}}([p_0, 1]) \geq \mathtt{OPT}_{\mathtt{LP}}([0, 1]) - p_0\, T^2/B'.$$

Proof. When $p_0 > B'/T$, the bound is trivial, and for the rest of the proof we assume that $p_0 \leq B'/T$.

By Equation (50) it suffices to replace $\mathtt{OPT}_{\mathtt{LP}}([0, 1])$ in the claim with $\mathtt{OPT}_{\mathtt{LP}}(X_0)$, for any given finite subset $X_0 \subset [0, 1]$. Let $\mathcal{D}$ be an LP-perfect distribution for the problem instance restricted to X_0; such $\mathcal{D}$ exists by Claim 3.4. Thus, $\mathsf{LP}(\mathcal{D}) = \mathtt{OPT}_{\mathtt{LP}}(X)$ and $c(\mathcal{D}) \leq \frac{B'}{T}$. Furthermore, $\mathcal{D}$ has a support of size at most 2; denote it as arms $p_1, p_2 \in [0, 1]$, $p_1 \leq p_2$, where the null arm would correspond to $p_1 = 0$. If $p_1 \geq p_0$, then $\mathcal{D}$ has support in the interval $[p_0, 1]$, and we are done; so from here on, we assume $p_1 < p_0$. Note that

$$\mathsf{LP}(\mathcal{D}) = r(\mathcal{D})\, \min\left(\frac{B'}{c(\mathcal{D})}, T\right) = T\, r(\mathcal{D}).$$

To prove the desired lower bound on $\mathtt{OPT}_{\mathtt{LP}}([p_0, 1])$, we construct a distribution $\mathcal{D}'$ with support in $\{0\} \cup [p_0, 1]$ and a sufficiently large LP-value. (Here the zero price corresponds to the null arm.)

Suppose $p_2 \leq \frac{B'}{T}$. Define $\mathcal{D}'$ by putting probability mass on price $\frac{B'}{T}$. Since $c(\frac{B'}{T}) \leq \frac{B'}{T}$, we have

$$\mathsf{LP}(\mathcal{D}') = T\, r(\mathcal{D}') = T\, F\left(\frac{B'}{T}\right) \geq T\, F(p_2) \geq T\, r(\mathcal{D}) = \mathsf{LP}(\mathcal{D}),$$

and we are done. From here on, assume $p_2 > \frac{B'}{T}$.

Now consider the main case: $p_1 \leq p_0 \leq \frac{B'}{T} < p_2$. Define distribution $\mathcal{D}'$ as follows:

$$\begin{aligned} \mathcal{D}'(p_0) &= \mathcal{D}(p_1), \\ \mathcal{D}'(p_2) &= \max\left(0, \mathcal{D}(p_2) - p_0/p_2\right), \\ \mathcal{D}'(0) &= 1 - \mathcal{D}'(p_0) - \mathcal{D}'(p_2). \end{aligned}$$

We claim that $c(\mathcal{D}') \leq \frac{B'}{T}$. If $\mathcal{D}'(p_2) = 0$, then $c(\mathcal{D}') = c(p_0) \leq p_0 \leq \frac{B'}{T}$. If $\mathcal{D}'(p_2) > 0$, then $\mathcal{D}'(p_2) = \mathcal{D}(p_2) - p_0/p_2$, and therefore,

$$\begin{aligned} c(\mathcal{D}') - c(\mathcal{D}) &= \mathcal{D}(p_1)\ (p_0 F(p_0) - p_1 F(p_1)) - p_2 F(p_2) \frac{p_0}{p_2} \\ &\leq p_0 F(p_0) - p_0 F(p_2) \leq 0. \end{aligned}$$

Then, $c(\mathcal{D}') \leq c(\mathcal{D}) \leq \frac{B'}{T}$. Claim proved.

Therefore, $\mathsf{LP}(\mathcal{D}') = T\, r(\mathcal{D}')$. To complete the proof:

$$\begin{aligned} r(\mathcal{D}') - r(\mathcal{D}) &\geq \mathcal{D}(p_1)\,(F(p_0) - F(p_1)) - F(p_2)\, p_0/p_2 \\ &\geq -p_0/p_2 \geq -p_0 T/B'. \\ \mathsf{OPT}_{\mathsf{LP}}([p_0, 1]) - \mathsf{OPT}_{\mathsf{LP}}(X_0) &= \mathsf{LP}(\mathcal{D}') - \mathsf{LP}(\mathcal{D}) \\ &= T(r(\mathcal{D}') - r(\mathcal{D})) \leq -p_0 T^2/B'. \end{aligned}$$

□

Suppose algorithm PrimalDualBwK is applied to a problem instance with a finite action space S. Then by Theorem 5.3 the S-regret is

$$R(S) = \widetilde{O}(\sqrt{mT} + T\sqrt{m/B'}), \quad m = |S|.$$

Let $S = [p_0, 1] \cap M$, where M is the ϵ-hyperbolic mesh, for some $\epsilon, p_0 \in (0, 1)$. Then, $m = |S| \leq \frac{1}{\epsilon p_0}$. Moreover, S is an ϵ-discretization for action space $X' = [p_0, 1]$, for the same reason that M is an ϵ-discretization for the original action space $X = [0, 1]$. Therefore:

$$\begin{aligned} \mathsf{Err}(S|X') &\leq \epsilon B' && \text{(by Theorem 7.3)} \\ \mathsf{Err}(X'|X) &\leq p_0\, T^2/B' && \text{(by Claim 7.8)} \\ \mathsf{OPT}_{\mathsf{LP}}(X) - \mathsf{REW}(S) &= R(S) + \mathsf{Err}(S|X') + \mathsf{Err}(X'|X) \\ &\leq R(S) + \epsilon B' + p_0\, T^2/B'. \end{aligned}$$

Optimizing the choice of ϵ and p_0, we obtain the final regret bound of $\tilde{O}(T\,(B')^{-1/4})$. Recall that this is the regret bound for the rescaled problem instance. Going back to the original problem instance, regret is multiplied by a factor of Λ. This completes the proof of Theorem 7.7.

8 APPLICATIONS AND COROLLARIES

We systematically overview various applications of BwK and corresponding corollaries. This section can be read independently of the technical material in the rest of the article.

Some technicalities. In applications with very large or infinite action space X, we apply a BwK algorithm with a restricted, finite action space $S \subset X$, where S is chosen in advance. Immediately, we obtain a bound on the S-*regret*: regret with respect to the value of $\mathsf{OPT}_{\mathsf{LP}}$ on the restricted action space (such bound depends on $|S|$). Instantiating such regret bounds is typically straightforward once one precisely defines the setting. In some applications, we can choose S using preadjusted discretization, as discussed in Section 7.

In some of the applications, per-round reward and resource consumption may be larger than 1. Then one needs to scale them down to fit the definition of BwK and apply our regret bounds, and scale them back up to obtain regret for the original (non-rescaled) version. We encapsulate this argument as follows:

Lemma 8.1. *Consider a version of* BwK *with finite action set S, in which per-round rewards are upper-bounded by r_0, and per-round consumption of each resource is at most c_0. Then one can achieve* *regret*

$$\widetilde{O}\left(\sqrt{r_0\,|S|\,\mathtt{OPT}} + \mathtt{OPT}\sqrt{c_0\,|S|/B}\ \right) \tag{54}$$

by applying algorithm `PrimalDualBwK` *with suitably rescaled rewards, resource consumption, and budgets.*

Proof. Denote $R(\mathtt{OPT}, B) = \sqrt{|S|\,\mathtt{OPT}} + \mathtt{OPT}\sqrt{|S|/B}$, as in the main regret bound.

To cast this problem as an instance of BwK, consider a rescaled problem instance in which all rewards are divided by r_0, and all consumptions and budgets are divided by c_0. Now, we can apply regret bound Equation (1) for the scaled-down problem instance; we obtain regret $\widetilde{O}(R(\mathtt{OPT}/r_0, B/c_0))$. Multiply this regret bound by r_0 to obtain a regret bound for the original problem instance. □

8.1 Dynamic Pricing with Limited Supply

In dynamic pricing, the algorithm is a monopolist seller that interacts with T agents (potential buyers) arriving one by one. In each round, a new agent arrives, the algorithm makes an offer, the agent chooses among the offered alternatives, and leaves. The offer specifies which goods are offered for sale at which prices. The agent has valuations over the offered bundles of goods, and chooses an alternative that maximizes her utility: value of the bundle minus the price. An agent is characterized by her *valuation function*: function from all possible bundles of goods that can be offered to their respective valuations. For each arriving agent, the valuation function is *private*: not known to the algorithm. It is assumed to be drawn from a fixed (but unknown) distribution over the possible valuation functions, called the *demand distribution*. Algorithm's objective is to maximize the total revenue; there is no bonus for left-over inventory.

Basic version. In the basic version from Section 1.1, the algorithm has B identical items for sale. In each round, the algorithm chooses a price p_t and offers one item for sale at this price, and an agent either buys or leaves. The agent has a fixed private value $v_t \in [0, 1]$ for an item, and buys if and only if $p_t \geq v_t$. Recall from Corollary 7.5 that we obtain regret $\widetilde{O}(B^{2/3})$, which is optimal according to Reference [12].

Extension: non-unit demands. Agents may be interested in buying more than one unit of the product, and may have valuations that are non-linear in the number of products bought. Accordingly, let us consider an extension where an algorithm can offer each agent multiple units. More specifically: in each round t, the algorithm offers up to λ_t units at a fixed price p_t per unit, where the pair (p_t, λ_t) is chosen by the algorithm, and the agent then chooses how many units to buy, if any. We restrict $\lambda_t \leq \Lambda$, where Λ is a fixed parameter. We obtain regret $\widetilde{O}(B\,\Lambda)^{2/3}$ by Theorem 7.6 (considering a special case when there is a single product and a single allowed bundle with one unit of this product). One can also consider a version with $\lambda_t = \Lambda$, so that the algorithm only chooses prices; then a very similar argument gives regret $\widetilde{O}(B^{2/3}\Lambda^{1/3})$.

Extension: multiple products. When multiple products are offered for sale, it often makes sense to price them jointly. Formally, the algorithm has d products for sale, with B_i units of each product i. (To simplify regret bounds, let us assume $B_i = B$.) In each round t, the algorithm chooses a vector of prices $(p_{t,1}, \ldots, p_{t,d}) \in [0, 1]^d$ and offers at most one unit of each product i at price $p_{t,i}$. The agent then chooses the subset of products to buy. We allow arbitrary demand distributions; we do not restrict correlations between valuations of different products and/or subsets of products.

Given a finite set S of allowed price vectors, such as an ϵ-additive mesh for some specific $\epsilon > 0$, we obtain S-regret $\widetilde{O}(d\sqrt{B\,|S|})$. This follows from Lemma 8.1, observing that per-round rewards are at most $r_0 = d$, per-round consumption of each resource is at most $c_0 = 1$, and the optimal value is $\texttt{OPT} \leq dB$.

There may also be a fixed collection of subsets that agents are allowed to buy, e.g., agents may be restricted to buying at most three items in total. This does not affect our analysis and the regret bound.

Joint pricing is *not* needed in the special case when each agent can buy an arbitrary subset I of products, and her valuations are additive: $v(I) = \sum_{i \in I} v(i)$. Then she buys each product i if and only if the offered price for this product exceeds $v(i)$. Therefore, the problem is equivalent to a collection of d separate per-product problems, and one can run a separate $\texttt{BwK}$ algorithm for each product. Using Corollary 7.5 separately for each product, one obtains regret $\widetilde{O}(d\,B^{2/3})$.

Extension: network revenue management. More generally, an algorithm may have d products for sale that may be produced on demand from limited *primitive resources*, so that each unit of each product i consumes a fixed and known amount $c_{ij} \in [0, 1]$ of each primitive resource j. This generalization is known as network revenue management problem (see Besbes and Zeevi [17] and references therein). All other details are the same as above; for simplicity, let us focus on a version in which each agent buys at most one item. Given a finite set S of allowed price vectors, we obtain S-regret given by Equation (54) with $r_0 = c_0 = d$.

In particular, if all resource constraints (including the time horizon) are scaled up by factor γ, regret scales as $\sqrt{\gamma}$. This improves over the main result in Besbes and Zeevi [17], where (essentially) regret is stated in terms of γ and scales as $\gamma^{2/3}$.

Extension: bundling and volume pricing. When selling to agents with non-unit demands, an algorithm may use discounts and/or surcharges for buying multiple units of a product (the latter may make sense for high-valued products such as tickets to events at the Olympics). More generally, an algorithm can may use discounts and/or surcharges for some *bundles* of products, where each bundle can include multiple units of multiple products, e.g., two beers and one snack. In full generality, there is a collection $\mathcal{F}$ of allowed bundles. In each round an algorithm offers a *menu* of options that consist of a price for every allowed bundle in $\mathcal{F}$ (and the "none" option), and the agent chooses one option from this menu. Thus, in each round the algorithm needs to choose a price vector over the allowed bundles.

For a formal result, assume there is a finite set S of allowed price vectors, each bundle in $\mathcal{F}$ can contain at most ℓ units total, and the per-bundle prices are restricted to lie in the range $[0, \ell]$. Then, we obtain S-regret $\widetilde{O}(d\ell\sqrt{\ell\,B\,|S|})$. This follows from Lemma 8.1, observing that per-round rewards are at most $r_0 = \ell$, per-round consumption of each resource is at most $c_0 = \ell$, and the optimal value is $\texttt{OPT} \leq d\ell B$.

The action space here is $|\mathcal{F}|$-dimensional, which may result in a prohibitively large number of allowed price vectors. One can reduce the "dimensionality" of the action space by restricting how the bundles may be priced. For example, each bundle may be priced at a volume discount $x\%$ compared to buying each unit separately, where x depends only on the number of items in the bundle.

Moreover, we can analyze preadjusted discretization for a version where in each round the algorithm chooses only one bundle to offer. By Theorem 7.4, we obtain regret $\widetilde{O}(B^{2/3}\,(|\mathcal{F}|\ell)^{1/3})$.

Extension: buyer targeting. Suppose there are ℓ different types of buyers (say *men* and *women*), and the demand distribution of a buyer depends on her type. The buyer type is modeled as a sample from a fixed but unknown distribution. In each round the seller observes the type of the current buyer (e.g., using a *cookie* or a user profile), and can choose the price depending on this type.

This can be modeled as a BwK domain where arms correspond to functions from buyer types to prices. For example, with ℓ buyer types and a single product, the (full) action space is $X = [0,1]^{\ell}$. Assuming we are given a restricted action space $S \subset X$, we obtain S-regret $\widetilde{O}(\sqrt{B\,|S|})$.

8.2 Dynamic Procurement and Crowdsourcing Markets

A "dual" problem to dynamic pricing is *dynamic procurement*, where the algorithm is buying rather than selling. In the basic version, the algorithm has a budget B to spend, and is facing T agents (potential sellers) that are arriving sequentially. In each round t, a new agent arrives, the algorithm chooses a price $p_t \in [1]$ and offers to buy one item at this price. The agent has private value $v_t \in [0,1]$ for an item (unknown to the algorithm), and sells if and only if $p_t \geq v_t$. The value is an independent sample from some fixed (but unknown) distribution. Algorithm's goal is to maximize the number of items bought. Recall from Theorem 7.7 that we obtain regret $\widetilde{O}(T/B^{1/4})$ for this version.

Application to crowdsourcing markets. The problem is particularly relevant to the emerging domain of *crowdsourcing*, where agents correspond to the (relatively inexpensive) workers on a crowdsourcing platform such as Amazon Mechanical Turk, and "items" bought/sold correspond to simple jobs ("microtasks") that can be performed by these workers. The algorithm corresponds to the "requester": an entity that submits jobs and benefits from them being completed. The (basic) dynamic procurement model captures an important issue in crowdsourcing that a requester interacts with multiple users with unknown values-per-item, and can adjust its behavior (such as the posted price) over time as it learns the distribution of users. While this basic model ignores some realistic features of crowdsourcing environments (see a survey Slivkins and Vaughan [52] for background and discussion), some of these limitations are addressed by the generalizations that we present below.

Extension: non-unit supply. We consider an extension where agents may be interested in more than one item, and their valuations may be non-linear. For example, a worker may be interested in performing several jobs. In each round t, the algorithm offers to buy up to Λ units at a fixed price p_t per unit, where the price p_t is chosen by the algorithm and Λ is a fixed parameter. The tth agent then chooses how many units to sell. Recall from Theorem 7.7 that we obtain regret $\tilde{O}(\Lambda^{5/4}T/B^{1/4})$ for this extension.

Extension: multiple types of jobs. We can handle an extension in which there are d types of jobs requested on the crowdsourcing platform, with a separate budget B_i for each type. Each agent t has a private cost $v_{t,i} \in [0,1]$ for each type i; the vector of private costs comes from a fixed but unknown distribution over d-dimensional vectors (note that arbitrary correlations are allowed). The algorithm derives reward $u_i \in [0,1]$ from each job of type i. In each round t, the algorithm offers a vector of prices $(p_{t,1},\ldots,p_{t,d})$, where $p_{t,i}$ is the price for one job of type i. For each type i, the agent performs one job of this type if and only if $p_{t,i} \geq v_{t,i}$, and receives payment $p_{t,i}$ from the algorithm.

Here arms correspond to the d-dimensional vectors of prices, so that the action space is $X = [0,1]^d$. Given the restricted action space $S \subset X$, we obtain S-regret $\widetilde{O}(d)(\sqrt{T\,|S|} + T\sqrt{d\,|S|/B})$, where B is the smallest budget. This follows from Lemma 8.1, observing that per-round rewards are at most $r_0 = d$, per-round consumption of each budget is at most $c_0 = 1$, and the optimal value is $\mathtt{OPT} \leq dT$.

Extension: additional features. We can also model more complicated "menus" so that each agent can perform several jobs of the same type. Then in each round, for each type i, the algorithm specifies the maximal offered number of jobs of this type and the price per one such job. We can also incorporate constraints on the maximal number of jobs of each type that is needed by the requester, and/or the maximal amount of money spend on each type.

Extension: competitive environment. There may be other requesters in the system, each offering its own vector of prices in each round. (This is a realistic scenario in crowdsourcing, for example.) Each seller/worker chooses the requester and the price that maximize her utility. One standard way to model such a competitive environment is to assume that the "best offer" from the competitors is a vector of prices that comes from a fixed but unknown distribution. This can be modeled as a BwK instance with a different distribution over outcomes that reflect the combined effects of the demand distribution of agents and the "best offer" distribution of the environment.

8.3 Other Applications to Electronic Markets

Ad allocation with unknown click probabilities. Consider *pay-per-click* (PPC) advertising on the web (in particular, this is a prevalent model in sponsored search auctions). The central premise in PPC advertising is that an advertiser derives value from her ad only when the user clicks on this ad. The ad platform allocates ads to users that arrive over time.

Consider the following simple (albeit highly idealized) model for PPC ad allocation. Users arrive over time, and the ad platform needs to allocate an ad to each arriving user. There is a set X of available ads. Each ad x is characterized by the payment-per-click π_x and click probability μ_x; the former quantity is known to the algorithm, whereas the latter is not. If an ad x is chosen, then it is clicked on with probability μ_x, in which case payment π_x is received. The goal is to maximize the total payment. This setting and various extensions thereof that incorporate user/webpage context have received a considerable attention in the past several years (starting with References [43, 47, 48]). In fact, the connection to PPC advertising has been one of the main motivations for the recent surge of interest in MAB.

We enrich the above setting by incorporating advertisers' *budgets*. In the most basic version, for each ad x there is a budget B_x—the maximal amount of money that can be spent on this ad. More generally, an advertiser can have an ad campaign that consists of a subset S of ads, so that there is a per-campaign budget S. Even more generally, an advertiser can have a more complicated budget structure: a family of overlapping subsets $S \subset X$ and a separate budget B_S for each S. For example, BestBuy can have a total budget for the ad campaign, and also separate budgets for ads about TVs and ads about computers. Finally, in addition to budgets (i.e., constraints on the number of times ads are clicked), an advertiser may wish to have similar constraints on the number of times ads are shown. BwK allows us to express all these constraints.

Adjusting a repeated auction. An auction is held in every round, with a fresh set of participants. The number of participants and a vector of their types come from a fixed but unknown distribution. The auction is *adjustable*: it has some parameter that the auctioneer adjust over time to optimize revenue. For example, Cesa-Bianchi et al. [21] studies a repeated second price auction with an adjustable reserve price, with unlimited inventory of a single product. BwK framework allows to incorporate limited inventory of items to be sold at the auction, possibly with multiple products.

Repeated bidding. A bidder participates in a repeated auction, such as a sponsored search auction. In each round t, the bidder can adjust her bid b_t based on the past performance. The outcome for this bidder is a vector (p_t, u_t), where p_t is the payment and u_t is the utility received. We assume that this vector comes from a fixed but unknown distribution. The bidder has a fixed budget. Similar settings have been studied in References [7, 55], for example.

We model this as a BwK problem where arms correspond to the possible bids, and the single resource is money. Note that (the basic version of) dynamic procurement corresponds to this setting with two possible outcome vectors (p_t, u_t): $(0, 0)$ and $(b_t, 1)$.

The BwK setting also allows to incorporate more complicated constraints. For example, an action can result in several different types of outcomes that are useful for the bidder (e.g., an ad shown to a *male* or an ad shown to a *female*), but the bidder is only interested in a limited quantity of each outcome.

8.4 Application to Network Routing and Scheduling

In addition to applications to Electronic Markets, we describe two applications to network routing and scheduling. In both applications an algorithm chooses between different feasible policies to handle arriving "service requests," such as connection requests in network routing and jobs in scheduling.

Adjusting a routing protocol. Consider the following stylized application to routing in a communication network. Connection requests arrive one by one. A connection request consists of a pair of terminals; assume the pair comes from a fixed but unknown distribution. The system needs to choose a *routing protocol* for each connection, out of several possible routing protocols. The routing protocol defines a path that connects the terminals; abstractly, each protocol is simply a mapping from terminal pairs to paths. Once the path is chosen, a connection between the terminals is established. Connections persist for a significant amount of time. Each connection uses some amount of bandwidth. For simplicity, we can assume that this amount is fixed over time for every connection, and comes from a fixed but unknown distribution (although even a deterministic version is interesting). Each edge in the network (or perhaps each node) has a limited capacity: the total bandwidth of all connections that pass though this edge or node cannot exceed some value. A connection that violates any capacity constraint is terminated. The goal is to satisfy a maximal number of connections.

We model this problem as BwK as follows: arms correspond to the feasible routing protocols, each edge/node is a limited resource, each satisfied connection is a unit reward.

Further, if the time horizon is partitioned in epochs, we can model different bandwidth utilization in each phase; then a resource in BwK is a pair (edge,epoch).

Adjusting a scheduling policy. An application with a similar flavor arises in the domain of scheduling long-running jobs to machines. Suppose jobs arrive over time. Each job must be assigned to one of the machines (or dropped); once assigned, a job stays in the system forever (or for some number of "epochs"), and consumes some resources. Jobs have multiple "types" that can be observed by the scheduler. For each type, the resource utilization comes from a fixed but unknown distribution. Note that there may be multiple resources being consumed on each machine: for example, jobs in a datacenter can consume CPU, RAM, disk space, and network bandwidth. Each satisfied job of type i brings utility u_i. The goal of the scheduler is to maximize utility given the constrained resources.

The mapping of this setting to BwK is straightforward. The only slightly subtle point is how to define the arms: in BwK terms, arms correspond to all possible mappings from job types to machines.

One can also consider an alternative formulation where there are several allowed scheduling policies (mappings from types and current resource utilizations to machines), and in every round the scheduler can choose to use one of these policies. Then the arms in BwK correspond to the allowed policies.

APPENDIXES

A THE OPTIMAL DYNAMIC POLICY BEATS THE BEST FIXED ARM

Let us provide additional examples of BwK problem instances in which the optimal dynamic policy (in fact, the best fixed distribution over arms) beats the best fixed arm.

Dynamic pricing. Consider the basic setting of “dynamic pricing with limited supply”: in each round a potential buyer arrives, and the seller offers him one item at a price; there are k items and $n > k$ potential buyers. One can easily construct distributions for which offering a mixture of two prices is strictly superior to offering any fixed price. In fact, this situation arises whenever the “revenue curve” (the mapping from prices to expected revenue) is non-concave and its value at the quantile k/n lies below its concave hull.

Consider a simple example: fix $\epsilon = k^{\delta - 1/2}$ with $\delta \in (0, \frac{1}{2})$, and assume that the buyer's value for an item is $v = 1$ with probability $\epsilon \, \frac{k}{n}$ and $v = \epsilon$ with the remaining probability, for some fixed $\epsilon \in (0, 1)$.

To analyze this example, let $\texttt{REW}(\mathcal{D})$ be the expected total reward (i.e., the expected total revenue) from using a fixed distribution $\mathcal{D}$ over prices in each round; let $\texttt{REW}(p)$ be the same quantity when $\mathcal{D}$ deterministically picks a given price p.

- Clearly, if one offers a fixed price in all rounds, it only makes sense to offer prices $p = \epsilon$ and $p = 1$. It is easy to see that $\texttt{REW}(\epsilon) = \epsilon k$ and $\texttt{REW}(1) \leq n \cdot \Pr[\text{sale at price } 1\ ] = \epsilon k$.
- Now consider a distribution $\mathcal{D}$ that picks price ϵ with probability $(1 - \epsilon)\frac{k}{n}$, and picks price 1 with the remaining probability. It is easy to show that $\texttt{REW}(D) \geq \epsilon k (2 - o(1))$.

So, we see that $\texttt{REW}(\mathcal{D})$ is essentially twice as large compared to the total expected revenue of the best fixed arm.

Dynamic procurement. A similar example can be constructed in the domain of dynamic procurement. Consider the basic setting thereof: in each round a potential seller arrives, and the buyer offers to buy one item at a price; there are T sellers and the buyer is constrained to spend at most budget B. The buyer has no value for left-over budget and each sellers value for the item is drawn i.i.d. from an unknown distribution. Then a mixture of two prices is strictly superior to offering any fixed price whenever the “sales curve” (the mapping from prices to probability of selling) is non-concave and its value at the quantile B/T lies below its concave hull.

Let us provide a specific example. Fix any constant $\delta > 0$, and let $\epsilon = B^{1/2+\delta}$. Each seller has the following two-point demand distribution: the seller's value for item is $v = 0$ with probability $\frac{B}{T}$, and $v = 1$ with the remaining probability. We use the notation $\texttt{REW}(\mathcal{D})$ and $\texttt{REW}(p)$ as defined above.

- Clearly, if one offers a fixed price in all rounds, it only makes sense to offer prices $p = 0$ and $p = 1$. It is easy to see that $\texttt{REW}(0) \leq T \cdot \Pr[\text{selling at price } 0] = B$ and $\texttt{REW}(1) = B$.
- Now consider a distribution $\mathcal{D}$ that picks price 0 with probability $1 - \frac{B-\epsilon}{T}$, and picks price 1 with the remaining probability. It is easy to show that $\texttt{REW}(\mathcal{D}) \geq (2 - o(1))\, B$.

Again, we see that $\texttt{REW}(\mathcal{D})$ is essentially twice as large compared to the total expected sales of the best fixed arm.

B BALANCEDEXPLORATION BEATS PRIMALDUALBWK SOMETIMES

We provide a simple example in which BalancedExploration achieves much better regret than (what we can prove for) PrimalDualBwK. The reason is that BalancedExploration is aware of the BwK domain, whereas PrimalDualBwK is not. More precisely, BalancedExploration is parameterized by $\mathcal{M}_{\texttt{feas}}$, the set of all latent structures that are feasible for the BwK domain.

The example is a version of the deterministic example from Section 1.1. There is a time horizon T and two other resources, both with budget $B < T/2$. There are m arms, partitioned into two same-size subsets, X_1 and X_2. Per-round rewards and per-round resource consumptions are deterministic for all arms. All arms get per-round reward 1. For each resource i, each arm in X_i only consumes this resource. Letting $c_i(x) = c_i(x, \mu)$ denote the (expected) per-round consumption of resource i by arm x, one of the following holds:

(i) $c_1(x_1) = 1$ and $c_2(x_2) = \frac{1}{2}$ for all arms $x_1 \in X_1, x_2 \in X_2$, or
(ii) $c_1(x_1) = \frac{1}{2}$ and $c_2(x_2) = 1$ for all arms $x_1 \in X_1, x_2 \in X_2$.

Analysis. Note that an optimal dynamic policy alternates the two arms in proportion, 1 : 2 or 2 : 1, depending on the case, and an LP-optimal distribution over arms samples them in the same proportion.

The key argument is that, informally, `BalancedExploration` can tell (i) from (ii) after the initial $O(m \log T)$ rounds. In the specification of `BalancedExploration`, consider the confidence radius for the per-round consumption of resource i, as defined in Equation (11). After any one arm x is played at least $C \log T$ rounds, for a sufficiently large absolute constant C, this confidence radius goes below 1/4. Therefore, any two latent structures μ, μ' in the confidence interval satisfy $|c_i(x, \mu) - c_i(x, \mu')| < \frac{1}{2}$. It follows that the confidence interval consists of a single latent structure, either the one corresponding to (i) or the one corresponding to (ii), which is the correct latent structure for this problem instance. Accordingly, the chosen distribution over arms, being “potentially perfect” by design, is LP-optimal. Thus, `BalancedExploration` uses the LP-optimal distribution over arms after the initial $O(m \log T)$ rounds.

The resulting regret is $\tilde{O}(m + \sqrt{B})$, where the $\sqrt{B}$ term arises, because the empirical frequencies of the two arms can deviate by $O(\sqrt{B})$ from the optimal values. Whereas with algorithm `PrimalDualBwK`, we can only guarantee regret $\tilde{O}(\sqrt{mB})$.

C ANALYSIS OF THE HEDGE ALGORITHM

We provide a self-contained proof of Proposition 5.4, the performance guarantee for the Hedge algorithm from Freund and Schapire [31]. The presentation is adapted from Kleinberg [38].

For the sake of convenience, we restate the algorithm and the proposition. It is an online algorithm for maintaining a d-dimensional probability vector y while observing a sequence of d-dimensional payoff vectors $\pi_1, \dots, \pi_\tau$. The algorithm is initialized with a parameter $\epsilon \in (0, 1)$.

ALGORITHM: Hedge(ϵ)

```
$v_1 = \mathbf{1}$
for $t = 1, 2, \dots, \tau$ do
    $y_t = v_t / (\mathbf{1}^\intercal v_t)$.
    $v_{t+1} = \text{Diag}\{(1+\epsilon)^{\pi_{ti}}\} v_t$.
```

The performance guarantee of the algorithm is expressed by the following proposition.

Proposition (Proposition 5.4, Restated). *For any $0 < \epsilon < 1$ and any sequence of payoff vectors $\pi_1, \dots, \pi_\tau \in [0,1]^d$, we have*

$$\forall y \in \Delta[d] \quad \sum_{t=1}^{\tau} y_t^\intercal \pi_t \geq (1-\epsilon) \sum_{t=1}^{\tau} y^\intercal \pi_t - \frac{\ln d}{\epsilon}.$$

Proof. The analysis uses the potential function $\Phi_t = \mathbf{1}^\intercal v_t$. We have

$$\begin{aligned}
\Phi_{t+1} &= \mathbf{1}^\intercal \mathrm{Diag}\{(1+\epsilon)^{\pi_{ti}}\} v_t \\
&= \sum_{i=1}^{d} (1+\epsilon)^{\pi_{ti}} v_{t,i} \\
&\le \sum_{i=1}^{d} (1+\epsilon\pi_{ti}) v_{t,i} \\
&= \Phi_t \left(1 + \epsilon y_t^\intercal \pi_t\right) \\
\ln(\Phi_{t+1}) &\le \ln(\Phi_t) + \ln\left(1 + \epsilon y_t^\intercal \pi_t\right) \le \ln(\Phi_t) + \epsilon y_t^\intercal \pi_t.
\end{aligned}$$

On the third line, we have used the inequality $(1+\epsilon)^x \le 1 + \epsilon x$, which is valid for $0 \le x \le 1$. Now, summing over $t = 1, \ldots, \tau$, we obtain

$$\sum_{t=1}^{\tau} y_t^\intercal \pi_t \ge \frac{1}{\epsilon}\left(\ln \Phi_{\tau+1} - \ln \Phi_1\right) = \frac{1}{\epsilon} \ln \Phi_{\tau+1} - \frac{\ln d}{\epsilon}.$$

The maximum of $y^\intercal(\sum_{t=1}^{\tau} \pi_t)$ over $y \in \Delta[d]$ must be attained at one of the extreme points of $\Delta[d]$, which are simply the standard basis vectors of $\mathbb{R}^d$. Say that the maximum is attained at $\mathbf{e}_i$. Then, we have

$$\begin{aligned}
\Phi_{\tau+1} &= \mathbf{1}_\intercal v_{\tau+1} \ge v_{\tau+1,i} = (1+\epsilon)^{\pi_{1i} + \cdots + \pi_{\tau i}} \\
\ln \Phi_{\tau+1} &\ge \ln(1+\epsilon) \sum_{t=1}^{\tau} \pi_{ti} \\
\sum_{t=1}^{\tau} y_t^\intercal \pi_t &\ge \frac{\ln(1+\epsilon)}{\epsilon} \sum_{t=1}^{\tau} \pi_{ti} - \frac{\ln d}{\epsilon} \\
&\ge (1-\epsilon) \sum_{t=1}^{\tau} y^\intercal \pi_t - \frac{\ln d}{\epsilon}.
\end{aligned}$$

The last line follows from two observations. First, our choice of i ensures that $\sum_{t=1}^{\tau} \pi_{ti} \ge \sum_{t=1}^{\tau} y^\intercal \pi_t$ for every $y \in \Delta[d]$. Second, the inequality $\ln(1+\epsilon) > \epsilon - \epsilon^2$ holds for every $\epsilon > 0$. In fact,

$$\begin{aligned}
-\ln(1+\epsilon) &= \ln\left(\frac{1}{1+\epsilon}\right) = \ln\left(1 - \frac{\epsilon}{1+\epsilon}\right) < -\frac{\epsilon}{1+\epsilon}, \\
\ln(1+\epsilon) &> \frac{\epsilon}{1+\epsilon} > \frac{\epsilon(1-\epsilon^2)}{1+\epsilon} = \epsilon - \epsilon^2.
\end{aligned}$$

□

D FACTS FOR THE PROOF OF THE LOWER BOUND

For the sake of completeness, we provide self-contained proofs for the two facts used in Section 6.

Fact (Fact 6.3, restated). *Let S_t be the sum of t i.i.d. 0-1 variables with expectation q. Let τ be the first time this sum reaches a given number $B \in \mathbb{N}$. Then $\mathbb{E}[\tau] = B/q$. Moreover, for each $T > \mathbb{E}[\tau]$ it holds that*

$$\sum_{t>T} \Pr[\tau \ge t] \le \mathbb{E}[\tau]^2/T. \tag{55}$$

Proof. $\mathbb{E}[\tau] = B/q$ follows from the martingale argument presented in the proof of Claim 6.4. Formally, take $q = p - \epsilon$ and $N_\tau = \tau$.

Assume $T > \mathbb{E}[\tau]$. The proof of Equation (55) uses two properties, one being that a geometric random variable is memoryless and other being Markov's inequality. Let us first bound the random variable $\tau - T$ conditional on the event that $\tau > T$:

$$\begin{aligned}\mathbb{E}[\tau - T|\tau > T] &= \sum_{t=1}^{B} \Pr[S_T = t]\ \mathbb{E}[\tau - T|\tau > T, S_T = t]\\ &= \sum_{t=1}^{B} \Pr[S_T = t]\ \mathbb{E}[\tau - T|S_T = t]\\ &\leq \sum_{t=1}^{B} \Pr[S_T = t]\ \mathbb{E}[\tau - T|S_T = 0]\\ &\leq \mathbb{E}[\tau - T|S_T = 0] = \mathbb{E}[\tau].\end{aligned}$$

By Markov's inequality, we have $\Pr[\tau \geq T] \leq \frac{\mathbb{E}[\tau]}{T}$. Combining the two inequalities, we have

$$\begin{aligned}\sum_{t>T} \Pr[\tau \geq t] &= \sum_{t>T} \Pr[\tau \geq T]\quad \Pr[\tau \geq t|\tau > T]\\ &= \Pr[\tau \geq T]\quad \mathbb{E}[\tau - T|\tau > T]\\ &\leq \mathbb{E}[\tau]^2/T.\end{aligned}$$

□

Fact (Fact 6.12, restated). *Assume $\frac{\epsilon}{p} \leq \frac{1}{2}$ and $p \leq \frac{1}{2}$. Then,*

$$p\ \log\left(\frac{p}{p-\epsilon}\right) + (1-p)\ \log\left(\frac{1-p}{1-p+\epsilon}\right) \leq \frac{2\epsilon^2}{p}.$$

Proof. To prove the inequality, we use the following standard inequalities:

$$\begin{aligned}&\log(1+x) \geq x - x^2/2 && \forall x \in [0,1]\\ &\log(1-x) \geq -x - x^2 && \forall x \in [0, \frac{1}{2}].\end{aligned}$$

It follows that

$$\begin{aligned}p\ \log\left(\frac{p}{p-\epsilon}\right) + (1-p)\ \log\left(\frac{1-p}{1-p+\epsilon}\right) &= -p\ \log\left(1 - \frac{\epsilon}{p}\right) - (1-p)\ \log\left(1 + \frac{\epsilon}{1-p}\right)\\ &\leq p\left(\frac{\epsilon}{p} + \frac{\epsilon^2}{p^2}\right) + (1-p)\left(-\frac{\epsilon}{1-p} + \frac{\epsilon^2}{(1-p)^2}\right)\\ &= \frac{\epsilon^2}{p} + \frac{\epsilon^2}{1-p} \leq \frac{2\epsilon^2}{p}.\end{aligned}$$

□

ACKNOWLEDGMENTS

The authors thank Moshe Babaioff, Peter Frazier, Luyi Gui, Chien-Ju Ho, and Jennifer Wortman Vaughan for helpful discussions related to this work. In particular, the application to routing protocols generalizes a network routing problem that was communicated to us by Luyi Gui. The application of dynamic procurement to crowdsourcing have been suggested to us by Chien-Ju Ho and Jennifer Wortman Vaughan. We are grateful to anonymous JACM referees for their thorough and insightful feedback.

REFERENCES

[1] Ittai Abraham, Omar Alonso, Vasilis Kandylas, and Aleksandrs Slivkins. 2013. Adaptive crowdsourcing algorithms for the bandit survey problem. In *Proceedings of the 26th COLT*.

[2] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. 2014. Taming the Monster: A Fast and Simple Algorithm for Contextual Bandits. In *Proceedings of the 31st ICML*.

[3] Shipra Agrawal and Nikhil R. Devanur. 2014. Bandits with concave rewards and convex knapsacks. In *Proceedings of the 15th ACM EC*.

[4] Shipra Agrawal and Nikhil R. Devanur. 2016. Linear Contextual Bandits with Knapsacks. In *Proceedings of the 29th NIPS*.

[5] Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. 2014. A Dynamic Near-Optimal Algorithm for Online Linear Programming. *Oper. Res.* 62, 4 (2014), 876–890.

[6] Shipra Agrawal, Nikhil R. Devanur, and Lihong Li. 2016. An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives. In *Proceedings of the 29th COLT*.

[7] Kareem Amin, Michael Kearns, Peter Key, and Anton Schwaighofer. 2012. Budget Optimization for Sponsored Search: Censored Learning in MDPs. In *Proceedings of the 28th UAI*.

[8] Sanjeev Arora, Elad Hazan, and Satyen Kale. 2012. The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. *Theory Comput.* 8, 1 (2012), 121–164.

[9] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Mach. Learn.* 47, 2–3 (2002), 235–256.

[10] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The Nonstochastic Multiarmed Bandit Problem. *SIAM J. Comput.* 32, 1 (2002), 48–77. Preliminary version in *Proceedings of the 36th IEEE FOCS*, 1995.

[11] Moshe Babaioff, Yogeshwer Sharma, and Aleksandrs Slivkins. 2014. Characterizing Truthful Multi-armed Bandit Mechanisms. *SIAM J. Comput.* 43, 1 (2014), 194–230. Preliminary version in *Proceedings of the 10th ACM EC*, 2009.

[12] Moshe Babaioff, Shaddin Dughmi, Robert D. Kleinberg, and Aleksandrs Slivkins. 2015. Dynamic Pricing with Limited Supply. *ACM Trans. Econ. Comput.* 3, 1 (2015), 4. Special issue for *Proceedings of the 13th ACM EC*, 2012.

[13] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Yaron Singer. 2012. Learning on a budget: posted price mechanisms for online procurement. In *Proceedings of the 13th ACM EC*. 128–145.

[14] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. 2013. Bandits with Knapsacks. In *Proceedings of the 54th IEEE FOCS*.

[15] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. 2014. Resourceful Contextual Bandits. In *Proceedings of the 27th COLT*.

[16] Omar Besbes and Assaf Zeevi. 2009. Dynamic Pricing Without Knowing the Demand Function: Risk Bounds and Near-Optimal Algorithms. *Oper. Res.* 57 (2009), 1407–1420. Issue 6.

[17] Omar Besbes and Assaf J. Zeevi. 2012. Blind Network Revenue Management. *Oper. Res.* 60, 6 (2012), 1537–1550.

[18] Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. 2003. Online learning in online auctions. In *Proceedings of the 14th ACM-SIAM SODA*. 202–204.

[19] Arnoud V. Den Boer. 2015. Dynamic pricing and learning: Historical origins, current research, and new directions. *Surv. Oper. Res. Manage. Sci.* 20, 1 (June 2015).

[20] Sébastien Bubeck and Nicolo Cesa-Bianchi. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Found.Trends Mach. Learn.* 5, 1 (2012).

[21] Nicoló Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. 2013. Regret Minimization for Reserve Prices in Second-Price Auctions. In *Proceedings of the ACM-SIAM SODA*.

[22] Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. 2011. Contextual Bandits with Linear Payoff Functions. In *Proceedings of the 14th AISTATS*.

[23] Varsha Dani, Thomas P. Hayes, and Sham Kakade. 2008. Stochastic Linear Optimization under Bandit Feedback. In *Proceedings of the 21th COLT*. 355–366.

[24] Nikhil Devanur and Sham M. Kakade. 2009. The Price of Truthfulness for Pay-Per-Click Auctions. In *Proceedings of the 10th ACM EC*. 99–106.

[25] Nikhil R. Devanur and Thomas P. Hayes. 2009. The AdWords problem: Online keyword matching with budgeted bidders under random permutations. In *Proceedings of the 10th ACM EC*. 71–78.

[26] Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. 2011. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings of the 12th ACM EC*. 29–38.

[27] Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. 2013. Multi-Armed Bandit with Budget Constraint and Variable Costs. In *Proceedings of the 27th AAAI.*

[28] Miroslav Dudíik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. 2011. Efficient Optimal Leanring for Contextual Bandits. In *Proceedings of the 27th UAI.*

[29] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. 2002. PAC bounds for multi-armed bandit and Markov decision processes. In *Proceedings of the 15th COLT*. 255–270.

[30] Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. 2010. Online Stochastic Packing Applied to Display Ad Allocation. In *Proceedings of the 18th ESA*. 182–194.

[31] Yoav Freund and Robert E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* 55, 1 (1997), 119–139.

[32] Naveen Garg and Jochen Könemann. 2007. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM J. Comput.* 37, 2 (2007), 630–652.

[33] Sudipta Guha and Kamesh Munagala. 2007. Multi-armed Bandits with Metric Switching Costs. In *Proceedings of the 36th ICALP*. 496–507.

[34] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. 2011. Approximation Algorithms for Correlated Knapsacks and Non-martingale Bandits. In *Proceedings of the 52nd IEEE FOCS*. 827–836.

[35] András György, Levente Kocsis, Ivett Szabó, and Csaba Szepesvári. 2007. Continuous Time Associative Bandit Problems. In *Proceedings of the 20th IJCAI.* 830–835.

[36] Elad Hazan and Nimrod Megiddo. 2007. Online Learning with Prior Information. In *Proceedings of the 20th COLT.* 499–513.

[37] Robert Kleinberg. 2004. Nearly Tight Bounds for the Continuum-Armed Bandit Problem. In *Proceedings of the 18th NIPS.*

[38] Robert Kleinberg. 2007. Lecture notes for CS 683 (week 2), Cornell University. Retrieved from http://www.cs.cornell.edu/courses/cs683/2007sp/lecnotes/week2.pdf.

[39] Robert Kleinberg and Tom Leighton. 2003. The Value of Knowing a Demand Curve: Bounds on Regret for Online Posted-Price Auctions. In *Proceedings of the 44th IEEE FOCS*. 594–605.

[40] Robert Kleinberg and Aleksandrs Slivkins. 2010. Sharp Dichotomies for Regret Minimization in Metric Spaces. In *Proceedings of the 21st ACM-SIAM SODA.*

[41] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. 2008. Multi-Armed Bandits in Metric Spaces. In *Proceedings of the 40th ACM STOC*. 681–690.

[42] Tze Leung Lai and Herbert Robbins. 1985. Asymptotically efficient Adaptive Allocation Rules. *Adv. Appl. Math.* 6 (1985), 4–22.

[43] John Langford and Tong Zhang. 2007. The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits. In *Proceedings of the 21st NIPS.*

[44] Nick Littlestone and Manfred K. Warmuth. 1994. The Weighted Majority Algorithm. *Info. Comput.* 108, 2 (1994), 212–260.

[45] Tyler Lu, Dávid Pál, and Martin Pál. 2010. Showing Relevant Ads via Lipschitz Context Multi-Armed Bandits. In *Proceedings of the 14th AISTATS.*

[46] Marco Molinaro and R. Ravi. 2012. Geometry of Online Packing Linear Programs. In *Proceedings of the 39th ICALP.* 701–713.

[47] Sandeep Pandey, Deepak Agarwal, Deepayan Chakrabarti, and Vanja Josifovski. 2007. Bandits for Taxonomies: A Model-based Approach. In *Proceedings of the SDM.*

[48] Sandeep Pandey, Deepayan Chakrabarti, and Deepak Agarwal. 2007. Multi-armed Bandit Problems with Dependent Arms. In *Proceedings of the 24th ICML.*

[49] Christos H. Papadimitriou and John N. Tsitsiklis. 1999. The complexity of optimal queuing network control. *Math. Oper. Res.* 24, 2 (1999), 293–305.

[50] Serge A. Plotkin, David B. Shmoys, and Eva Tardos. 1995. Fast Approximation Algorithms for Fractional Packing and Covering Problems. *Math. Oper. Res.* 20 (1995), 257–301.

[51] Adish Singla and Andreas Krause. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *Proceedings of the 22nd WWW.* 1167–1178.

[52] Aleksandrs Slivkins and Jennifer Wortman Vaughan. 2013. Online Decision Making in Crowdsourcing Markets: Theoretical Challenges. *SIGecom Exch.* 12, 2 (December 2013).

[53] Long Tran-Thanh, Archie Chapman, Enrique Munoz de Cote, Alex Rogers, and Nicholas R. Jennings. 2010. ϵ-first policies for budget-limited multi-armed bandits. In *Proceedings of the 24th AAAI.* 1211–1216.

[54] Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas R. Jennings. 2012. Knapsack based optimal policies for budget-limited multi-armed bandits. In *Proceedings of the 26th AAAI.* 1134–1140.

[55] Long Tran-Thanh, Lampros C. Stavrogiannis, Victor Naroditskiy, Valentin Robu, Nicholas R. Jennings, and Peter Key. 2014. Efficient regret bounds for online bid optimisation in budget-limited sponsored search auctions. In *Proceedings of the 30th UAI.*

[56] Zizhuo Wang, Shiming Deng, and Yinyu Ye. 2014. Close the Gaps: A Learning-While-Doing Algorithm for Single-Product Revenue Management Problems. *Oper. Res.* 62, 2 (2014), 318–331.

[57] Peter Whittle. 1980. Multi-armed Bandits and the Gittins Index. *J. Roy. Stat. Soc., Ser. B* 42, 2 (1980), 143–149.

Received July 2015; revised September 2017; accepted November 2017