ABSTRACT

Title of dissertation: SEQUENTIAL DECISION MAKING WITH LIMITED RESOURCES

Karthik Abinav Sankararaman
Doctor of Philosophy, 2019

Dissertation directed by: Professor Aravind Srinivasan
Department of Computer Science

One of the goals of Artificial Intelligence (AI) is to enable multiple agents to interact, co-ordinate and compete with each other to realize various goals. Typically, this is achieved via a system which acts as a mediator to *control* the agents' behavior via incentives. Such systems are ubiquitous and include online systems for shopping (*e.g.,* Amazon), ride-sharing (*e.g.,* Uber, Lyft) and Internet labor markets (*e.g.,* Mechanical Turk). The main algorithmic challenge in such systems is to ensure that they can operate under a variety of informational constraints such as uncertainty in the input, committing to actions based on partial information or being unaffected by noisy input. The mathematical framework used to study such systems are broadly called *sequential decision making* problems where the algorithm does not receive the entire input at once; it obtains parts of the input by interacting (also called "actions") with the environment.

In this thesis, we answer the question, under what informational constraints can we design efficient algorithms for sequential decision making problems.

The first part of the thesis deals with the *Online Matching* problem. Here, the algorithm deals with two prominent constraints: uncertainty in the input and choice of actions being restricted by a combinatorial constraint. We design several new algorithms for many variants of this problem and provide provable guarantees. We also show their efficacy on the ride-share application using a real-world dataset. In the second part of the thesis, we consider the *Multi-armed* bandit problem with additional informational constraints. In this setting, the algorithm does not receive the entire input and needs to make decisions based on partial observations. Additionally, the set of possible actions is controlled by global resource constraints that bind across time. We design new algorithms for multiple variants of this problem that are worst-case optimal. We provide a general reduction framework to the classic multi-armed bandits problem without any constraints. We complement some of the results with preliminary numerical experiments.

SEQUENTIAL DECISION MAKING WITH LIMITED RESOURCES

by

Karthik Abinav Sankararaman

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2019

Advisory Committee:
Professor Aravind Srinivasan, Chair
Professor John P. Dickerson
Professor Tom Goldstein
Professor Furong Huang
Professor Prakash Narayan
Dr. Aleksandrs Slivkins

Dedicated to my parents.

Acknowledgments

"Although an act of help done timely, might be small in nature, it is truly larger than the world itself."

– Thiruvalluvar, a classic Tamil poet

I thank Aravind Srinivasan for the guidance, support and help throughout these years. He is a true genius and it was a privilege to work beside him and absorb some of it. I have learnt many interesting mathematical ideas from him. He is very kind and has the ability to connect his students to new topics (and associated experts) that he thinks are important. One such connection was with John Dickerson who joined UMD and has since been a very close collaborator. His keen insight on applications of the theoretical work is commendable. I am indebted for the opportunity to work on many projects with him and learn a great deal.

In Fall 2015, I was fortunate to meet Alex Slivkins through his class on bandits at UMD. Alex is an extremely prolific researcher and I am thankful for the gazillion hours he has spent over these years to mentor me: writing, talks, research, life. I am extremely impressed by the amount of energy he has towards doing research (and life) and hope that some of this has rubbed off on me. He is always available via Skype/email/phone for brainstorming and this has helped me a lot while thinking about problems. I am also thankful to him (and Microsoft) for hosting me at what was a fun and productive summer at MSR NYC along with Rob Schapire and Nicole Immorlica. Chapter 7 was the outcome of this internship; I learnt a great deal about bandits, online learning and game theory in this process from Alex, Rob and Nicole.

Aravind connected me with Anand Louis at IISc and Navin Goyal at MSR India which was the start of our collaboration on causal inference. I am extremely thankful to them for the numerous weekly meetings (and accommodating their schedule to the 9+ hours of time-difference). I learnt a great deal about many topics including linear algebra, probability and statistics. I am also grateful to Anand for reading through all the proofs carefully and giving critical feedback. I would like to thank IISc for hosting me in the summer, Microsoft for the space and food and Anand and Navin for making this collaboration possible (and having me visit on my trips back to India).

Many thanks to my committee members Prakash Narayan, Tom Goldstein and Furong Huang; I am grateful to the many interactions during my time here on various problems in machine learning. Bill Gasarch mentored me on how to be a great instructor and I am thankful for his advice on classroom teaching. And I thank my professors from IITM especially Jayalal Sarma and Narayanaswamy who first introduced me to theoretical computer science and algorithms.

I thank Prithvi Sen for hosting me at IBM Almaden and Anil Kamath for hosting me at Adobe during my first two summers. It was a period of lot of learning for me. Recently, Soham De introduced me to the world of neural nets and has since taught me many things about the theory and practice of deep learning. I am extremely thankful to him for those discussions. The initial parts of the online matching research was done with my friends Pan Xu and Brian Brubach. I would always cherish the numerous meetings we had to understand, discuss and eventually solve some of the questions. I continued collaborating with Pan and most of my

later research on this topic has been jointly with him. We have worked on several problems together and it has been extremely fulfilling. Science is a team-sport and the research done during my PhD would not have been possible without my co-authors who are some of the smartest people I have interacted with. I thank all my office mates at AV Williams and Iribe Center for the company, coffee/tea breaks and discussion about various topics over these years. I am also extremely fortunate to have had a huge army of friends right from my middle school days through the undergrad years and grad school; I am thankful to each one of them for *long* phone calls, planning numerous trips, giving me advice and company (also a place to crash). I am also very grateful for the company of all the friends I made in grad school and at internships; they have kept me happy in the last few years.

Home is where the heart is: I thank my parents, grand parents, my brother for the support, affection and happiness in all these years (and more recently my sister-in-law). The "Indian" way of returning this love is to make them proud and hopefully this is a step towards that direction. I am also thankful to many of my relatives who have frequently kept in touch via calls and meetings in all these years.

I gratefully acknowledge NSF Awards CNS 1010789, CCF 1422569, CCF-1749864, a gift from Google, Inc., and research awards from Adobe, Inc. and Amazon, Inc that has supported the research during my PhD. Chapter 3 is based on [42] with Brian Brubach, Aravind Srinivasan and Pan Xu. Chapters 4, 5 and 6 are based on [76, 78] with John Dickerson, Aravind Srinivasan and Pan Xu. Chapter 7 is based on [109] with Nicole Immorlica, Rob Schapire and Alex Slivkins and Chapter 8 is based on [163] with Alex Slivkins.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

"We can only see a short distance ahead, but we can see plenty there that needs to be done."

– Alan Turing (possibly on sequential decision making)

The last decade has seen a rise in large-scale applications of artificial intelligence and machine learning techniques in various domains ranging from e-commerce, recommender systems to healthcare, medicine and climate change efforts. The two biggest contributors to this phenomenon are the availability of large amounts of datasets to perform supervised learning and cheap computing resources. This led to efficient application of large scale supervised learning algorithms on massive datasets. From a theoretical standpoint, these applications gave rise to a host of research questions on newer models of computation for algorithm design. One such model is the stochastic input model, where the algorithm receives input as a sample from some distribution. Assumptions about this distribution are motivated from those that are commonly observed across multiple datasets. For a significant portion of this thesis, we study algorithm design questions that use the stochastic input model.

Although supervised learning on "offline data" has been very successful, the current goal is to tackle applications from a wider range of domains where the data is inherently "real-time". This implies that although one may be able to use the vast historical data to pre-compute predictions about the future the algorithm's decision has to be done "online". Often the time-scales at which these algorithms operate are in the order of milli-seconds; thus it vastly reduces the available compute time to make decisions. Moreover, the algorithm needs to optimize over multiple objectives and operate in domains where the decision affects human-beings. Concretely, consider a ride-share platform such as Uber or Lyft. A central problem is to efficiently match a driver to a request. However, the platform does not get all the requests before it makes the matching decisions. Based on historical observational data, the platform can obtain an estimate of the future requests, in order to optimize the allocation policy; however, the actual match has to be done real-time. The goal is to match requests to drivers while learning the best strategy and using them to optimize for various criteria such as revenue, welfare and fairness. In fact, this general setup is ubiquitous in many other applications such as crowdsourcing, Internet advertising markets and recommender systems. These applications motivate the study of a class of problems called "sequential decision making" (SDM) where the algorithm does not receive the entire input at once; the algorithm makes a sequence of decisions to obtain the input sequence (often the inputs are stochastic and/or depends on past decisions).

In many of these applications, the algorithm is further restricted in the choice of future decisions it can make based on the current actions. This imposes infor-

mation bottlenecks for the algorithm to learn about the future based on the past; we abstractly call these the *resource constraints*. Thus, the central thesis of this work is to *understand under what informational constraints can we design efficient algorithms for sequential decision making problems.*

We make progress towards this by considering two broad classes of SDM problems and design efficient algorithms for variants thereof. The first class of problems is called the *online matching* problem where the goal of the algorithm is to find an optimal *graph matching* when the vertices of the graph are provided sequentially. Variants of this problem have many applications such as matching riders to drivers, keywords to advertisements and recommending movies to users. The algorithm is usually faced with one or more objectives it has to optimize for such as revenue, diversity and fairness. The second class we consider is a variant of the classic multi-armed bandit problem called *bandits with knapsacks*. This is a sequential learning problem where the algorithm wants to learn parameters of the system under various information constraints such as the number of samples it is allowed to collect. Many applications such as ride-share, crowdsourcing and recommender systems have tunable parameters whose optimal value is not known apriori and the algorithm needs to explore several choices based on real-time inputs. However, it is also usually constrained by external constraints such as limited inventory. Thus variants of this model is useful in such application domains.

Our Contributions and Outline: The rest of this thesis is structured as follows. Chapter 2 describes the SDM framework formally and provides the benchmark and

performance metric we use for the algorithms. Then the thesis is divided into two parts. Part I deals with problems in Online Matching. In Chapter 3 we consider the weighted version of the online matching problem under stochastic inputs and show algorithms with improved competitive ratio. In Chapter 4 we consider an extension of the problem where the matched vertices may become available after certain number of time-steps from the time it is matched. We design a new algorithm for this problem and show that the competitive ratio obtained is tight among all algorithms that use a class of techniques. In Chapter 5 we consider another extension of the problem where the goal of the algorithm is to maximize multiple objectives, namely total reward and the diversity of *edge types*. Abstractly, we solve the online matching problem with submodular objective functions. In the last chapter 6 of this part we consider the real-world application of matching drivers to riders on a ride-share platform. We model this problem as an Online Matching problem and use a real-world dataset to empirically evaluate the proposed algorithms. In Part II of the thesis we consider the variants of the bandit with knapsack problem. In Chapter 7 we consider the adversarial version of the basic bandits with knapsacks problem and give an optimal algorithm via a reduction to the classic multi-armed bandit problem. Along the way we also give a new algorithm to the stochastic bandits with knapsacks problem. We show many extensions that can be obtained as a corollary of this reduction. In Chapter 8 we consider an extension of this problem to the combinatorial semi-bandits with knapsacks problem in the stochastic setting and give optimal algorithm for this problem. In this chapter, we also perform experiments on simulated datasets to evaluate the performance of this algorithm in

practice and compare it with other algorithms proposed in the literature. Finally in Chapter 9 we conclude the thesis with a few open problems and broader future directions for research.

Chapter 2: Preliminaries

In this chapter, we give a formal definition of the sequential decision making protocol. The sequential decision making framework is used across multiple communities (*e.g.,* reinforcement learning, experimental design, online algorithms) with (slightly) different terminologies. Here we describe a general version of the sequential decision making protocol that captures all problems considered in this thesis.

2.1 Sequential Decision Making (SDM) protocol

The SDM protocol can be viewed as a repeated game between the *algorithm* and the *nature*. The protocol proceeds in rounds with the total number of rounds T known to both the players a-priori. At each time-step $t \in [T]$ the nature first presents part of the input $\mathbf{x}_t$ to the algorithm. The algorithm then chooses an action $a_t \in \mathcal{A}_t$, where $\mathcal{A}_t$ represents the set of feasible actions available to the algorithm at time t. The choice a_t yields a *reward* r_t to the algorithm and modifies the set of available actions $\mathcal{A}_{t+1}$ to the algorithm in the next round. The goal of the algorithm is to maximize the total reward obtained in the T rounds. Critically, the algorithm's action a_t is binding and cannot be revoked at a later time. Moreover, it does not

receive the full input beforehand and thus has to optimize based on observed history until time t and any *predictions* of the future the algorithm may have. The set of actions $\mathcal{A}_{t+1}$ could potentially be a strict subset of $\mathcal{A}_t$; abstractly we call this as *resource* constraints. Resource constraints make the problem significantly harder and understanding under what constraints can we design efficient algorithms is the central thesis of this work. Figure 2.1 describes the protocol formally. Throughout this thesis, we use the term *online algorithm* interchangeably to refer to any algorithm for the SDM protocol.

Given: number of time-steps T.

In each round $t \in [T]$,

1. Nature reveals part of the input $\mathbf{x}_t$.
2. The algorithm chooses action $a_t \in \mathcal{A}_t$.
3. Algorithm receives reward r_t which is a function of a_t.
4. The set of available actions $\mathcal{A}_{t+1}$ is updated based on a_t.

Figure 2.1: SDM protocol

2.1.1 Models of the Nature

The literature makes various assumption on the power of the nature. In the *adversarial* model, the nature is assumed to be an arbitrary adversary and the goal of the algorithm is to deal with the worst-case input sequence. In the *adaptive*

adversary model, this adversary is assumed to be adaptive to the choices of the algorithm. In other words, the input $\mathbf{x}_t$, the reward r_t and the new set of actions $\mathcal{A}_{t+1}$ can be set by the nature *after* observing the set of actions $a_1, a_2, \ldots, a_{t-2}$ [1]. In the *oblivious adversary* model, the adversary has lesser power and the map from action a_t to input $\mathbf{x}_{t+1}$, reward r_t and action set $\mathcal{A}_{t+1}$ should be fixed before the start of the game. Thus, the adversary cannot adapt its strategy to the actions taken by the algorithm. We also consider *stochastic* models of the adversary where the input $\mathbf{x}_t$ and the reward r_t is drawn from a joint distribution $\mathcal{D}$ over all possible tuples $(\mathbf{x}_t, r_t)$, at each time-step. Additionally, in all problems considered in this thesis, this draw is independent across time-steps (however, the distribution from which the tuples are sampled may vary over time). In the *known* distribution model, we assume that the algorithm knows the distribution(s) before the game starts while in the *unknown* distribution model, the algorithm does not have access to the distribution(s) and has to learn the relevant information over time.

2.2 Landscape of Problems

Many problems fall under the category of the SDM protocol described above. In this thesis, we focus on two specific models of computation namely *Online Matching* and *Multi-armed Bandits with Knapsacks*. We briefly describe the basic version of these problems and show how they fit into the broader SDM framework. Table 2.1 summarizes how the two problems fit into the *SDM* protocol described in Figure 2.1.

[1] In fact, in some problems we allow the adversary to also observe the action a_{t-1}.

2.2.1 Online Matching

In this problem the algorithm makes a sequence of decisions to solve a *graph matching* problem. We are given a bipartite graph $G = (U, V, E)$ with the vertex set U given before the start of the game. The vertices in the set V are presented sequentially at each time-step. The algorithm either matches the presented vertex v_t to any (available) neighbor in U or rejects v irrevocably. Once a vertex $u \in U$ is matched to some v, it becomes unavailable for the rest of the time-steps. The goal is to maximize the total size of the matching. Mapping this to the SDM protocol, the input $\mathbf{x}_t$ is the vertex v_t presented at time-step t. The reward r_t is 1 if v_t was matched and 0 otherwise. The set of actions $\mathcal{A}_t$ is the available vertices in U at time-step t. This problem was first introduced by Karp et al. [116] and many variants of this basic problem has since been studied. In this thesis, we consider many variants of this basic problem which has applications in many modern large-scale matching markets such as ride-share, recommender systems, job scheduling and online advertising.

2.2.2 Multi-armed Bandits with Knapsacks

In this problem the algorithm has K arms to choose from at each time-step. There are d resources each with a global budget B. The algorithm first chooses an arm $a_t \in [K]$. The nature then reveals a reward $r_t(a_t)$ for the chosen arm and reveals the consumption of each of the d resources $c_{1,t}(a_t), \ldots, c_{d,t}(a_t)$. We allow the algorithm to skip any round (denoted by an action $a(\text{null}) \in [K]$) which yields no

reward. The moment some resource is exhausted the algorithm is allowed to play only the action $a(\text{null})$ in all future time-steps. The goal is to maximize the total reward collected by the algorithm. In the SDM protocol, the input $\mathbf{x}_t$ is the outcome vector $\mathbf{o}_{t-1} := (r_{t-1}(a_{t-1}); c_{1,t-1}(a_{t-1}), \ldots, c_{d,t-1}(a_{t-1}))$ for all time-steps $t \geq 2$ and for $t = 1$ the nature does not provide any input to the algorithm. The reward r_t is just the scalar $r_t(a_t)$. The set of actions $\mathcal{A}_t$ remains the same set of K arms until one resource is exhausted after which the set just reduces to the singleton set $\{a(\text{null})\}$. This problem was introduced as an extension to the classic multi-armed bandit problem (Thompson [175]) to bandits with resource constraints by Badanidiyuru et al. [33] and has since garnered a lot of attention.

Problem/Parameter	Input $\mathbf{x}_t$	Reward r_t	Action set $\mathcal{A}_t$
Online Matching	Vertex $v_t \in V$	1 if v_t matched 0 otherwise	Available vertices in U
Bandits with Knapsacks	$t = 1$: No input $t \geq 2$: $\mathbf{o}_{t-1}$	$r_t(a_t)$	$[K]$ if resource not exhausted $a(\text{null})$ otherwise

Table 2.1: Various problems in the SDM framework

2.3 Benchmark

We compare the performance of any online algorithm against a benchmark (denoted by OPT). We primarily deal with two definitions of the benchmark, namely, *best fixed distribution* and *best dynamic policy* which we now define.

Definition 1 (Best dynamic policy)**.** *Given an instance $\mathcal{I}$ of the problem, the best dynamic policy is the total reward obtained by the best possible algorithm in hindsight given the realizations of all the randomness associated with the input. Thus,* OPT *in this case is a random variable and we compare against* $\text{OPT}_{DP} := \mathbb{E}[\text{OPT}]$.

Algorithms described in Chapters 3, 4, 5 and 8 use this benchmark.

Definition 2 (Best fixed distribution)**.** *Given an instance $\mathcal{I}$ of the problem, the best fixed distribution is the total expected reward obtained by sampling from the best possible fixed distribution. We use the notation* $\text{OPT}_{FD} := \mathbb{E}[\text{OPT}]$ *to denote this benchmark.*

This benchmark is used in the algorithms described in Chapter 7. More discussion of this benchmark is deferred to Chapter 7.

2.4 Performance Metric

To measure the performance of online algorithms the following notions are commonly used, namely *competitive ratio* and *regret*. We define both these metrics now.

Definition 3 (Competitive ratio)**.** *Let* $\mathbb{E}[\text{REW}]$ *denote the expected reward obtained by any algorithm* ALG *and let* $\mathbb{E}[\text{OPT}]$ *denote the expected reward of the benchmark. Then competitive ratio of the algorithm* ALG *is the value of the ratio* $\frac{\mathbb{E}[\text{REW}]}{\mathbb{E}[\text{OPT}]}$.

Note that in Definition 3, the ratio is atmost 1 and the goal is to maximize this quantity. To keep it consistent with the related literature, in Chapter 7, we use the

reciprocal $\frac{\mathbb{E}[\text{OPT}]}{\mathbb{E}[\text{REW}]}$ as the performance metric. In that case, the goal is to minimize the quantity.

Usually, competitive ratio is used in problems when the best dynamic policy benchmark is used (Buchbinder and Naor [46]). However, for the bandits with knapsacks problem we consider in Chapter 7 we show that even against the best fixed distribution benchmark, the notion of competitive ratio is *necessary*. We are not aware of other problems that exhibit this behavior.

Definition 4 (Regret)**.** *Let* $\mathbb{E}[\text{REW}]$ *denote the expected reward obtained by any algorithm* ALG *and let* $\mathbb{E}[\text{OPT}]$ *denote the expected reward of the benchmark. Then regret of the algorithm* ALG *is the value of the difference* $\mathbb{E}[\text{OPT}] - \mathbb{E}[\text{REW}]$.

Regret of any algorithm is non-negative and the goal is to minimize regret. Regret is usually defined against the weaker best fixed distribution benchmark. However, in Chapter 7 we show that for the stochastic version of the bandits with knapsacks problem we can obtain non-trivial regret bounds even against the best dynamic policy benchmark. Regret is the primary metric in Chapter 8.

Remark 1. *Competitive ratio is invariant to the scale of the rewards* $\{r_t\}_{t\in[T]}$ *while regret scales linearly with the scale of the rewards. Thus, for consistency throughout the thesis we assume that* $r_t \in [0,1]$ *for all* $t \in [T]$.

Table 2.2 summarizes the metric and the benchmark used for the various problems considered in this thesis.

Upper-bound on OPT **by a mathematical program.** A common strategy (*e.g.,* see Buchbinder and Naor [46], Mehta [139]) in algorithm design is to not work

Problem/Metric	Competitive Ratio	Regret
Online Matching	Dynamic Policy	-
Stochastic Bandits with Knapsacks	-	Dynamic Policy
Adversarial Bandits with Knapsacks	Fixed Distribution	-

Table 2.2: Summary of benchmark and metrics for various problems

directly with OPT but to use a suitable upper-bound on its value. This upper-bound is typically the optimal solution to an appropriate mathematical program. The advantage of this method is that, the optimal solution of such mathematical programs have additional interesting properties which can be exploited in the analysis. Moreover, from the definition of the performance metrics it can be seen that algorithms that are "good" (either low competitive ratio or regret) against the optimal solution of this program are also good against OPT. For all problems considered in this thesis, we always use the optimal solution of an appropriate mathematical program (typically a linear program) as the proxy for OPT.

Part I

Online Matching

Chapter 3: Online Matching

3.1 Introduction

In this chapter, we consider the model of Online Algorithms. The central problem of study will be the *Online Weighted Matching* problem under *distributional* arrival sequences. The model of computation has many motivational examples such as Online advertising, ride-share, crowdsourcing and kidney-exchange. In the simplest setting of this problem, we start with an weighted *bipartite* graph $G = (U, V, E)$. The protocol proceeds in T rounds with the set U available before the start of the game. In each round $t \in [T]$, a vertex $v_t \in V$ is presented to the algorithm (henceforth we interchangeably use the term *arrives*) along with all its neighboring edges in U (denoted by $\delta(v_t)$) and their corresponding weights. The algorithm has two *irrevocable* choices: either choose an edge $e = (u, v_t) \in \delta(v_t)$, where u is an unmatched vertex, or not match $v_t \in V$. Thus, a choice to match an edge implies that the algorithm cannot un-match it in a future time-step. Likewise, choosing to drop v_t implies that it cannot re-visit this vertex at a later time. The goal of the algorithm is to maximize the total weight of the matching obtained after the T rounds of the protocol. Figure 3.1 formally describes the protocol for the simplest version of this problem.

Given: number of time-steps T and offline vertex set U.

In each round $t \in [T]$,

1. A vertex $v_t \in V$ and the set of neighbors $\delta(v_t)$ is presented to the algorithm.

2. The algorithm either chooses an unmatched vertex $u \in U$ to match v_t or drops the request v_t.

Figure 3.1: Online Matching Protocol: A basic version

3.1.1 Arrival sequence.

The landscape of problems can be divided based on the assumptions placed on the manner in which v_t is chosen to arrive at time t. The earliest version of this problem, studied by Karp et al. [116], assumed that the vertices $v_t \in V$ is chosen by an *oblivious adversary* and that all edge-weights are identical. Following this, more recent works proposed two other models of arrival, namely, the *unknown* and the *known* distribution models. In both these cases, we assume that the arrival sequence is stochastic and sampled from a product distribution $\mathcal{D}$ over the set of possible arrivals. In other words, at every time-step t, the vertex v_t is sampled from a distribution $\mathcal{D}_t$ independent of all other time-steps.

The focus of this work will be on *known* product distributions. We show some future directions where we consider models that interpolates between the unknown and known product distributions.

3.1.2 Competitive ratio.

The performance of any online algorithm for this problem is compared against the *best dynamic policy* (Definition 1) . The competitive ratio with respect to this benchmark is defined as follows. It is the ratio of the size of the matching obtained by the algorithm to that of the optimal matching if the entire arrival sequence was known (we interchangeably use the phrase *the optimal matching of the offline instance*). Formally, let $\mathrm{ALG}(\mathcal{I})$ (a random variable) denote the size of the matching obtained by any online algorithm ALG on instance $\mathcal{I}$. Let $\mathrm{OPT}(\mathcal{I})$ denote the size of the optimal matching (a random variable) in instance $\mathcal{I}$ if the arrival sequence of vertices in V was available a-priori. Then the competitive ratio of ALG is defined as $\max_{\mathcal{I}} \mathbb{E}[\mathrm{ALG}(\mathcal{I})]/\mathbb{E}[\mathrm{OPT}(\mathcal{I})]$.

3.2 Related work

The seminal paper of Karp et al. [116] introduced this problem where they gave the optimal competitive ratio of $1-1/\mathsf{e}$ when the vertex arrival is set by an oblivious adversary. Following this work, this problem has garnered much interest due to its various applications (see Mehta [139] for a comprehensive survey). The vertex-weighted version of this problem was introduced by Aggarwal et al. [8], where they give an optimal $\left(1-\frac{1}{\mathsf{e}}\right)$ ratio for the adversarial arrival model. The edge-weighted setting has been studied in the adversarial model by Feldman et al. [82], where they consider an additional relaxation of "free-disposal".

In addition to the adversarial and known I.I.D. models, online matching is

also studied under several other variants such as random arrival order, unknown distributions, and known adversarial distributions. In the setting of random arrival order, the arrival sequence is assumed to be a random permutation over all online vertices, see *e.g.,* Devanur and Hayes [69], Kesselheim et al. [119], Korula and Pál [122], Mahdian and Yan [136]. In the case of unknown distributions, in each round an item is sampled from a fixed but unknown distribution. If the sampling distributions are required to be the same during each round, it is called unknown I.I.D. (Devanur et al. [73, 75]); otherwise, it is called adversarial stochastic input (Devanur et al. [75]). As for known adversarial distributions, in each round an item is sampled from a known distribution, which is allowed to change over time (Alaei et al. [16, 17]). Another variant of this problem is when the edges have stochastic rewards. Models with stochastic rewards have been previously studied by Mehta and Panigrahi [140], Mehta et al. [142] among others, but not in the known I.I.D. model.

In the known i.i.d. setting prior work has considered unweighted, vertex-weighted and the edge-weighted versions of the problem. The vertex-weighted and unweighted setting (often studied in tandem) have many results starting with Feldman et al. [83] who were the first to beat $1 - 1/\mathsf{e}$ with a competitive ratio of 0.67 for the unweighted problem. This was improved by Manshadi et al. [137] to 0.705 with an adaptive algorithm. In addition, they showed that even in the unweighted variant with integral arrival rates, no algorithm can achieve a ratio better than $1 - \mathsf{e}^{-2} \approx 0.86$. Finally, Jaillet and Lu [111] presented an adaptive algorithm which used a clever LP to achieve 0.725 and $1 - 2\mathsf{e}^{-2} \approx 0.729$ for the vertex-weighted and unweighted problems, respectively. In the edge-weighted setting model Haeupler

et al. [102] were the first to beat $1 - 1/\mathsf{e}$ by achieving a competitive ratio of 0.667. They use a *discounted LP* with tighter constraints and employ the *power of two choices* paradigm to guide their online algorithm.

A closely related problem is the Adwords problem where every edge has a *bid* and every offline vertex has a *budget*. The goal is to match an online vertex to any offline vertex that has not exhausted its budget. This problem was introduced by Mehta et al. [141] and subsequently studied by Buchbinder et al. [49], Devanur and Jain [71], Devanur et al. [74], Goel and Mehta [95]. A series of works (Aggarwal et al. [8], Chan et al. [56], Devanur et al. [74], Goel and Mehta [95]) has attempted to simplify the RANKING algorithm proposed in Karp et al. [116] which has been useful to develop algorithms for related problems such as vertex-weighted matching and Adwords.

Another related model is the *two-sided online* matching problem where all vertices in the graph come online and the goal is to match the vertices that are *currently* present. A series of recent works (*e.g.,* Dickerson et al. [77], Huang et al. [107, 108], Truong and Wang [179], Wang and Wong [183]) consider many variants of this model and design improved algorithms. Online matching has also been considered in the *edge-arrival* model where the edges arrive instead of the vertices. Representative works include Badanidiyuru [29], Buchbinder et al. [51], McGregor [138].

3.3 Online Weighted Matching

3.3.1 Formal Description of the Problem

Before the start of the game, we are given a weighted bipartite graph $G = (U, V, E)$ and a corresponding weight function $\mathbf{w} : E \to \mathbb{R}^+$ that maps each edge to a positive real number. Additionally, we are given a rate function $\mathbf{r} : V \to \mathbb{Z}^+$ which denotes the number of times a vertex $v \in V$ would arrive in expectation across all the rounds in the game. The game proceeds in T rounds, where in each round a vertex $v \in V$ is sampled i.i.d. from a distribution $\mathcal{D}$ such that $\mathcal{D}(v) = r(v)/T$. In particular, the distributions $\mathcal{D}_t$ in the product distribution is the same in every time-step. As noted in Haeupler et al. [102], we can assume that $r(v) = 1$ for every vertex wlog. Indeed, any vertex v such that $r(v) > 1$ can be replaced by $r(v)$ different vertices each an arrival rate of 1. Thus, this assumption implies that the size of the new set of online vertices is exactly T, since the sum of expectations equals the total number of online rounds. The goal of the algorithm is to match an incoming vertex $v \in V$ to any unmatched neighbor (or drop) irrevocably in each round, such that the sum of the weights of the matched edges after T rounds is maximized.

Asymptotic assumption and notation. We will always assume T is large and analyze algorithms as T goes to infinity: e.g., if $x \leq 1 - (1 - 2/T)^T$, we will just write this as "$x \leq 1 - 1/\mathsf{e}^2$" instead of the more-accurate "$x \leq 1 - 1/\mathsf{e}^2 + o(1)$". These suppressed $o(1)$ terms will subtract at most $o(1)$ from our competitive ratios.

3.3.2 Linear Relaxation to Upper-bound OPT

As discussed in Chapter 2, a common strategy in online algorithms is to find a suitable upper-bound on OPT using a mathematical program (typically a LP) and compare the performance of the algorithm to this stronger benchmark. One of the challenges in the design of algorithms is to ensure that the gap between OPT and this upper-bound is as small as possible. In particular, one of the contributions of our work is in constructing a stronger upper-bound compared to prior work. In Lemma 1, we prove that the following LP is an upper-bound on OPT.

$$\begin{array}{lllll} \text{maximize} & \sum_{e\in E} w(e)\, x(e) & \text{such that} & & \\ & \sum_{e\in\delta(u)} x(e) \leq 1 & \forall u \in U & & \\ & \sum_{e\in\delta(v)} x(e) \leq 1 & \forall v \in U & & (3.1) \\ & 0 \leq x(e) \leq 1 - 1/\mathsf{e} & \forall e \in E & & \\ & x(e) + x(e') \leq 1 - 1/\mathsf{e}^2 & \forall e, e' \in \delta(u), \forall u \in U & & \end{array}$$

Lemma 1. *Let* $\mathbf{x}^*$ *denote an optimal solution to LP* (3.1)*. Then we have* $\text{OPT} \leq \sum_{e\in E} w(e)x^*(e)$.

Proof. Let $Y(e)$ denote the indicator random variable for the event that edge $e \in E$ is matched in the optimal solution for a given arrival sequence $\mathcal{A}$. Let $y(e) := \mathbb{E}_{\mathcal{A}}[Y(e)]$ for every edge $e \in E$. We will now argue that the vector $\mathbf{y} := (y(e))_{e\in E}$ is a feasible solution to the LP (3.1). Consider a vertex $u \in U$. We have that $\sum_{e\in\delta(u)} Y(e) \leq 1$. Taking expectations on both sides and using the linearity of expectation we have $\sum_{e\in\delta(u)} y(e) \leq 1$. This shows that $\mathbf{y}$ is feasible to the first constraint. Let $R(v)$

denote the random variable for the number of times a vertex $v \in V$ arrived in a given arrival sequence $\mathcal{A}$. Then we have, for every $v \in V$, $\sum_{e \in \delta(v)} Y(e) \leq R(v)$. From the integral arrival rates assumption, $\mathbb{E}_{\mathcal{A}}[R(v)] = 1$ for every $v \in V$. Thus, from linearity of expectation we obtain $\sum_{e \in \delta(v)} y(e) \leq 1$. This shows that $\mathbf{y}$ is feasible to the second constraint. For any edge $e = (u, v)$, let $\mathbb{I}[R(v) = 0]$ be an indicator for the event that a vertex $v \in V$ never arrives in the T rounds. Thus, for any arrival sequence $\mathcal{A}$, we have $Y(e) \leq \mathbb{I}[R(v) \neq 0]$. Taking expectations on both sides we get $y(e) \leq \mathbb{E}_{\mathcal{A}}[\mathbb{I}[R(v) \neq 0]$. The probability that a vertex v never arrives in T rounds is $\left(1 - \frac{1}{T}\right)^T \leq \frac{1}{\mathsf{e}}$. Thus, $\mathbb{E}_{\mathcal{A}}[\mathbb{I}[R(v) \neq 0] \leq 1 - \frac{1}{\mathsf{e}}$. This shows that $\mathbf{y}$ is feasible to the third constraint. Consider two edges $e, e' \in \delta(u)$ for some $u \in U$. Let $e = (u, v)$ and $e' = (u, v')$ and as before let $\mathbb{I}[R(v) \neq 0]$ and $\mathbb{I}[R(v') \neq 0]$ denote the indicator for the events that v, v' arrives at least once in the T rounds, respectively. For any arrival sequence $\mathcal{A}$ we have that $Y(e) + Y(e') \leq \mathbb{I}[R(v) \neq 0] \wedge \mathbb{I}[R(v') \neq 0]$. Taking expectations on both sides we get $y(e) + y(e') \leq \mathbb{E}_{\mathcal{A}}[\mathbb{I}[R(v) \neq 0] \wedge \mathbb{I}[R(v') \neq 0]]$. The probability that both v and v' never arrive in the T rounds is given by $\left(1 - \frac{2}{T}\right)^T \leq \frac{1}{\mathsf{e}^2}$. Thus, we get $y(e) + y(e') \leq 1 - \frac{1}{\mathsf{e}^2}$ which shows that $\mathbf{y}$ is feasible to the fourth constraint.

The expected weight of the optimal solution is $\mathbb{E}_{\mathcal{A}}[\sum_{e \in E} w(e) Y(e)]$ which from linearity of expectation gives $\sum_{e \in E} w(e) y(e)$. Since $\mathbf{y}$ is a feasible solution we have that the optimal value to LP (3.1) is at least as large as the expected optimal solution. □

3.4 Our Contribution

Our main contribution to this problem is to improve the competitive ratio of 0.667 due to Haeupler et al. [102] to 0.7. We do so by designing a new algorithm based on the *dependent-rounding* schemes of Gandhi et al. [92]. We describe in detail the key observation we make in the algorithm of Haeupler et al. [102] which we exploit in our algorithm design.

Haeupler et al. [102] used two matchings, M_1 and M_2 of the offline graph $G = (U, V, E)$ to guide the online algorithm and leverage the *power of two choices*. When a vertex v arrives for the first time, an attempt to match to its neighbor in M_1 is made and on its second arrival an attempt to match to its neighbor in M_2 is made. However, these two matchings may not be edge disjoint, leaving some arriving vertices with only one choice (or possibly none). In fact, choosing two matchings as a guide that maximize both the edge weights and the number of disjoint edges is a major challenge that arises in applying the *power of two choices* to this setting.

When the same edge (u, v) is included in both matchings M_1 and M_2, the copy of (u, v) in M_2 offers no additional benefit and a second arrival of v is wasted (which happens with a substantial probability). Concretely, Haeupler et al. [102] choose the two matchings in the following manner. M_1 is obtained by solving an LP similar to (3.1) and rounding this to an integral solution. M_2 is constructed by first finding a maximum-weight matching and then removing all the edges which have already been included in M_1. A key element of their proof is to show that the probability of an edge being removed from M_2 is at most $1 - 1/\mathsf{e} \approx 0.63$.

We make the observation that generating the two matchings in a slightly correlated manner can drastically decrease this probability which translates to improved competitive ratio. Moreover, we show a general technique to construct an ordered set of k matchings where k is a tunable parameter. For $k = 2$, we show that the probability of an edge appearing in both M_1 and M_2 is at most $1 - 2/\mathsf{e} \approx 0.26$.

3.5 Overview of our algorithm

We provide a overview of both the warm-up and the final algorithm. The algorithm has two components, namely the *offline* phase and the *online* phase. The offline phase is as follows. The algorithm first solves the LP (3.1) to obtain the fractional solution $\mathbf{x}^*$. Then we use a rounding scheme described in subsection 3.5.1 to obtain a integral vector $\mathbf{F}$ where each coordinate in this vector is in the set $\{0, 1, 2\}$. Then we construct two matchings M_1 and M_2 of the graph induced by the vector $\mathbf{F}$ (repeating an edge whenever any coordinate is 2). The online phases uses the two matchings M_1 and M_2 to guide its action. When a vertex type $v \in V$ arrives for the first time, we attempt to match it to the neighbor in M_1. When it arrives for the second time, we attempt to match it to the neighbor in M_2. In its third and succeeding arrivals, we simply drop the vertex.

3.5.1 LP rounding technique

We round $\mathbf{x}^*$ to an integral solution $\mathbf{F}$ using a two step process we call $\mathsf{DR}[\mathbf{x}^*, k]$. The first step is to multiply $\mathbf{x}^*$ by k. The second step is to apply the

dependent rounding techniques of Gandhi et al. [92] to this new vector. For most parts of this section we use $k = 2$. In the conclusion, we briefly mention an improved algorithm that uses $k = 3$ to achieve the best known competitive ratio of 0.705.

While dependent rounding is typically applied to values between 0 and 1, the useful properties extend naturally to our case in which $kx^*(e)$ may be greater than 1 for some edge $e \in E$. To understand this process, it is easiest to imagine splitting each $kx^*(e)$ into two edges with the integer value $x'(e) = \lfloor kx^*(e) \rfloor$ and fractional value $x''(e) = kx^*(e) - \lfloor kx^*(e) \rfloor$. The former will remain unchanged by the dependent rounding procedure since it is already an integer while the latter will be rounded to 1 with probability $x''(e)$ and 0 otherwise. The final value $F(e)$ would be the sum of the two rounded values. The two properties of dependent rounding we use are:

(P1) **Marginal distribution:** For every edge $e \in E$, let $p(e) = kx^*(e) - \lfloor kx^*(e) \rfloor$. Then, $\Pr[F(e) = \lceil kx^*(e) \rceil] = p(e)$ and $\Pr[F(e) = \lfloor kx^*(e) \rfloor] = 1 - p(e)$.

(P2) **Degree-preservation:** For any vertex $w \in U \cup V$, let its fractional degree $kx^*(w) := \sum_{e \in \delta(w)} kx^*(e)$ and integral degree be the random variable $F(w) = \sum_{e \in \delta(w)} F(e)$. Then $F(w) \in \{\lfloor kx^*(w) \rfloor, \lceil kx^*(w) \rceil\}$.

3.6 Warmup: 0.688 competitive algorithm

As a warm-up, we describe a simple algorithm which achieves a competitive ratio of 0.688 and introduces the key ideas in our approach. We begin by solving the LP (3.1) to get a fractional solution vector $\mathbf{x}^*$ and applying $\mathsf{DR}[\mathbf{x}^*, 2]$ as described

in Subsection 3.5.1 to get an integral vector $\mathbf{F}$. We construct a bipartite graph $G_{\mathbf{F}}$ with $F(e)$ copies of each edge $e \in E$. Note that $G_{\mathbf{F}}$ will have max degree 2 since for all $w \in U \cup V$, $F(w) \leq \lceil 2x^*(w) \rceil \leq 2$ and thus we can decompose it into two matchings using *Hall's Theorem*. The exact choice of the two matchings is not critical to the algorithm as long as the union contains all edges in $G_{\mathbf{F}}$. Finally, we randomly permute the two matchings into an ordered pair of matchings, $[M_1, M_2]$. These matchings serve as a guide for the online phase of the algorithm, similar to Haeupler et al. [102]. The warm-up algorithm, denoted by EW_0, is summarized in Algorithm 1.

Algorithm 1: EW_0

1 Construct and solve the benchmark LP (3.1) for the input instance.

2 Let $\mathbf{x}^*$ be an optimal fractional solution vector. Invoke $\mathsf{DR}[\mathbf{x}^*, 2]$ to get a random integral vector $\mathbf{F}$.

3 Create the graph $G_{\mathbf{F}}$ with $F(e)$ copies of each edge $e \in E$ and decompose it into two matchings as described in text.

4 Randomly permute the matchings to get a *random ordered* pair of matchings, say $[M_1, M_2]$.

5 When a vertex type v arrives for the first time, attempt to match v to u_1 if $(u_1, v) \in M_1$; when v arrives for the second time, attempt to match v to u_2 if $(u_2, v) \in M_2$.

6 When a vertex type v arrives for the third time or after, do nothing in that step.

3.6.0.1 Analysis of EW_0

We will show that EW_0 (Algorithm 1) achieves a competitive ratio of 0.688. Let $[M_1, M_2]$ be the randomly ordered pair of matchings. Note an edge $e \in E$ appears in both matchings if and only if $\mathbf{x}^*(e) > 1/2$. We consider three types of edges. We say an edge $e \in E$ is of type ψ_1, denoted by $e \in \psi_1$, if and only if e appears *only* in M_1. Similarly $e \in \psi_2$, if and only if e appears *only* in M_2. Finally, $e \in \psi_b$, if and only if e appears in *both* M_1 and M_2. Let P_1, P_2, and P_b be the probabilities that an edge $e \in \psi_1$, $e \in \psi_2$, and $e \in \psi_b$ is matched, respectively. Haeupler et al. [102] proved the following Lemma 2 that bounds P_1, P_2, P_b for *any* given matching M_1 and M_2.

Lemma 2 (Section 3 of Haeupler et al. [102])**.** *For any two matchings M_1 and M_2 steps (5) and (6) in Algorithm 1 implies that we have (1) $P_1 > 0.5808$; (2) $P_2 > 0.14849$ and (3) $P_b > 0.632$.*

We use Lemma 2 to prove that the warm-up algorithm EW_0 achieves a ratio of 0.688 by examining the probability that a given edge e is becomes type ψ_1, ψ_2, or ψ_b.

Analysis of EW_0. Consider the following two cases.

- **Case 1:** $0 \leq x^*(e) \leq 1/2$: By the marginal distribution property of dependent rounding (P1), there can be at most one copy of e in $G_{\mathbf{F}}$. Moreover, the probability of including e in $G_{\mathbf{F}}$ is $2f_e$. Since an edge in $G_{\mathbf{F}}$ can appear in either M_1 or M_2 with equal probability $1/2$, we have that $\Pr[e \in \psi_1] = \Pr[e \in$

$\psi_2] = x^*(e)$. Thus, the probability that edge e is added to the matching is $(x^*(e)P_1 + x^*(e)P_2) = 0.729x^*(e)$.

- **Case 2:** $1/2 \leq x^*(e) \leq 1 - 1/\mathsf{e}$: Similarly, by the marginal distribution (P1), $\Pr[e \in \psi_b] = \Pr[F(e) = \lceil 2x^*(e) \rceil] = 2x^*(e) - \lfloor 2x^*(e) \rfloor = 2x^*(e) - 1$. It follows that $\Pr[e \in \psi_1] = \Pr[e \in \psi_2] = (1/2)(1 - (2x^*(e) - 1)) = 1 - x^*(e)$. Thus,the probability that edge e is added to the matching is (noting that the first term is from case 1 while the second term is from case 2) $((1 - x^*(e))(P_1 + P_2) + (2x^*(e) - 1)P_b) \geq 0.688x^*(e)$, where the worst-case occurs for an edge e with $x^*(e) = 1 - 1/\mathsf{e}$.

  To prove the competitive ratio, we first show the following Lemma.

Lemma 3. *Let* $\mathbf{x}^*$ *denote the optimal solution to* LP (3.1)*. Suppose we have that for every edge* $e \in E$, $\Pr[e$ *is included in matching*$] \geq \alpha x^*(e)$ *then the competitive ratio is at least* α*.*

Proof. From Linearity of Expectation, we have that the total expected size of the matching is $\mathbb{E}[\sum_{e \in E} w(e)\mathbb{I}[\text{e is matched}]] = \sum_{e \in E} w(e) \Pr[\text{e is matched}]$. From the premise we have that for every edge $e \in E$, Pr[e is included in matching] $\geq \alpha x^*(e)$. Thus, the total expected size of the matching is $\alpha \sum_{e \in E} w(e)x^*(e)$. From Lemma 1 we have $\sum_{e \in E} w(e)x^*(e) \geq \text{OPT}$. Thus, the competitive ratio is at least α. □

In the two cases above, we proved that for every $e \in E$ we have Pr[e is matched] $\geq 0.688x^*(e)$. Thus from Lemma 3 we obtain a competitive ratio of 0.688.

3.7 0.7 competitive algorithm

In this section, we describe an improvement to the warm-up algorithm to get a competitive ratio of 0.7. We start by making an observation about the performance of the warm-up algorithm. After solving LP (3.1), let edges with $x^*(e) > 1/2$ be called *large* and edges with $x^*(e) \leq 1/2$ be called *small*. Let L and S, be the set of large and small edges, respectively. In the warm-up analysis, small edges contributed a much higher value of 0.729 towards the ratio as opposed to the worst-case 0.688 by the large edges. This is primarily due to the fact that we may potentially get two copies of a large edge in $G_{\mathbf{F}}$. In this case, the copy in M_1 has a better chance of being matched, since there is no edge which can "block" it (*i.e.,* an edge with the same offline neighbor that gets matched first), but the copy in M_2 has no chance of being matched.

To correct for this imbalance, we make an additional modification to the vector $\mathbf{x}^*$ *before* applying $\mathsf{DR}[\mathbf{x}^*, k]$. The rest of the algorithm is exactly the same. Let η be a parameter to be optimized in the analysis. For all large edges $\ell \in L$ such that $x^*(\ell) > 1/2$, we set $\tilde{x}^*(\ell) = x^*(\ell) + \eta$. For all small edges $s \in S$ which are adjacent to some large edge, let $\ell \in L$ be the largest edge adjacent to s such that $x^*(\ell) > 1/2$. Note that it is possible for s to have two large neighbors, but we only care about the larger of the two. We set $\tilde{x}^*(s) = x^*(s)\left(\frac{1-\tilde{x}^*(\ell)}{1-x^*(\ell)}\right)$.

In other words, we increase the values of large edges while ensuring that for all $w \in U \cup V$, $x^*(w) \leq 1$ by reducing the values of neighboring small edges proportional to their original values. Note that it is not possible for two large edges to be

adjacent since they must both have $x^*(e) > 1/2$. For all other small edges which are not adjacent to any large edges, we leave their values unchanged. We then apply $\mathsf{DR}[\mathbf{x}^*, 2]$ to this new vector, multiplying by 2 and applying dependent rounding as before. Algorithm 2 formally describes the algorithm.

Algorithm 2: $\mathsf{EW}(\eta)$

1 Construct and solve the benchmark LP (3.1) for the input instance.

2 Let $\mathbf{x}^*$ be an optimal fractional solution vector.

3 For all edges $e \in E$ with $x^*(e) > 1/2$ let $\tilde{x}^*(\ell) = x^*(\ell) + \eta$

4 For all edges $e \in E$ with $x^*(e) \leq 1/2$ do the following. Let $\ell \in L$ be the largest edge adjacent to e such that $x^*(\ell) > 1/2$. Let $\tilde{x}^*(e) = x^*(e)\left(\frac{1-\tilde{x}^*(\ell)}{1-x^*(\ell)}\right)$.

5 Invoke $\mathsf{DR}[\tilde{\mathbf{x}}, 2]$ to get a random integral vector $\mathbf{F}$.

6 Run steps (2)-(6) of Algorithm 1

3.7.0.1 Analysis

Theorem 1. *For edge-weighted online stochastic matching with integral arrival rates,* $\mathsf{EW}(0.0142)$ *achieves a competitive ratio of at least* 0.7.

Proof. As in the warm-up analysis, we'll consider large and small edges separately

- **Scenario 1:** $0 \leq x^*(s) \leq \frac{1}{2}$:

  Here we have two cases

  - **Case 1:** s is not adjacent to any large edges.

In this case, the analysis is the same as Case 1 in the warm-up analysis. Thus, the probability that edge s is added to the matching is $0.729x^*(e)$.

– **Case 2:** s is adjacent to some large edge ℓ.

For this case, let $x^*(\ell)$ be the value of the largest neighboring edge in the original LP solution. Then the probability that edge s is added to the matching is

$$x^*(s)\left(\frac{1-(x^*(\ell)+\eta)}{1-x^*(\ell)}\right)(0.1484+0.5803).$$

This follows from Lemma 2; in particular, the first two terms are the result of how we set $\tilde{x}(s)$ in the algorithm, while the two numbers, 0.1484 and 0.5803, are the probabilities that s is matched when it is in M_2 and M_1, respectively. Note that for $x^*(\ell) \in [0,1)$ this is a decreasing function in $x^*(\ell)$. So the worst case is when $x^*(\ell) = 1-1/\mathsf{e}$ (due to third constraint in LP (3.1)) Thus, the probability that edge s is added to the matching is

$$x^*(s)\left(\frac{1-(1-1/\mathsf{e}+\eta)}{1-(1-1/\mathsf{e})}\right)(0.1484+0.5803).$$

Since $\eta = 0.0142$, this evaluates to,

$$0.701x^*(s). \tag{3.2}$$

- $\frac{1}{2} < x^*(\ell) \leq 1-\frac{1}{\mathsf{e}}$: Here, the probability that ℓ is added to the matching is, $[1-(x^*(\ell)+\eta)][P_1+P_2]+[2(x^*(\ell)+\eta)-1]P_b$. This can re-arranged to obtain

$$(P_1+P_2)(1-\eta)+(2\eta-1)P_b+x^*(\ell)[2P_b-P_1-P_2]. \tag{3.3}$$

Since $\eta = 0.0142$ using Lemma 2 we have $(P_1+P_2)(1-\eta)+(2\eta-1)P_b = 0.1048$. Similarly, using Lemma 2 we have $2P_b - P_1 - P_2 = 0.535$. Thus, Eq. (3.3) simplifies to,

$$0.1048 + x^*(\ell)0.535 \tag{3.4}$$

We can write Eq. (3.4) as $x^*(\ell)[0.1048/x^*(\ell) + 0.535]$. Note that $\frac{1}{2} < x^*(\ell) \leq 1 - \frac{1}{\mathsf{e}}$. Thus, Eq. (3.4) can be lower-bounded by

$$0.701x^*(\ell). \tag{3.5}$$

Thus combining Eq. (3.2) and (3.5) with Lemma 3 we get a competitive ratio of 0.7.

We now show that the chosen value of $\eta = 0.0142$ ensures that both $\tilde{x}^*(\ell)$ and $\tilde{x}^*(s)$ are less than 1 *after* modification. Since $x^*(\ell) \leq 1 - 1/\mathsf{e}$ we have that $x^*(\ell) + \eta \leq 1 - 1/\mathsf{e} + 0.0142 \leq 1$. Note that $x^*(\ell) \geq 1/2$. Hence, the modified value $\tilde{x}^*(s)$ is always less than or equal to the original value, since $\left(\frac{1-(x^*(\ell)+\eta)}{1-x^*(\ell)}\right)$ is decreasing in the range $x^*(\ell) \in [1/2, 1 - 1/\mathsf{e}]$ and has a value less than 0.98 at $x^*(\ell) = 1/2$. □

3.8 Conclusion

In this chapter, we considered the *Online (Edge) Weighted Matching* problem under the known product distribution arrival assumption. The key contribution is a new algorithm based on randomized (dependent) rounding schemes of Gandhi et al. [92] that improves the competitive ratio to 0.701. For simplicity in exposition, we describe a simpler algorithm in this thesis. In Brubach et al. [42], we give

another algorithm which uses similar ideas and $\mathsf{DR}[\mathbf{x}^*, 3]$ to obtain the best-known competitive ratio of 0.705. The analysis is significantly involved with many more cases and we refer the reader to the paper for more details.

All algorithms in prior work and the one described in this thesis fall under the class of online algorithms known as *non-adaptive* algorithms. In other words, after realizing part of the randomness in the input, the algorithm does not change it's strategy. It is an interesting open problem to consider *adaptive* algorithms for this problem and understand if significant improvements to competitive ratios can be made. This is called *adaptivity* gap in the literature Gupta et al. [98, 99]. The best known lower-bound for this problem is 0.823 which holds even when all edge weights are identical. Thus, the immediate open question is to bridge the gap between the upper-bound of 0.705 and the lower-bound of 0.823. Adaptive algorithm is one possible approach towards bridging this gap.

Chapter 4: Extension 1: Online Weighted Matching with Reusable Vertices

4.1 Introduction

In this chapter, we consider an extension of the Online Matching problem where the offline vertices are *reusable*. Online Matching problems are motivated by market design problems where agents on one side of a market are paired with agents, contracts, or transactions on the other. In many of these applications such as matching drivers to riders, jobs to servers, organs to patients the process is dynamic where one side of the market arrives in an *online* fashion and is matched sequentially to the other side. Moreover, in these applications the match has a temporal component; after a certain period of time the offline agent is *freed* and can be used for other matches. We motivate this problem using the example of ride-share below.

Taxi Dispatching Services and Ride-Sharing Systems. Traditional taxi services and rideshare systems like Uber and Didi Chuxing match drivers to would-be riders Lee et al. [128], Lowalekar et al. [132], Tong et al. [176]. Here, the offline Service Providers are different vehicle drivers. Once an online request (potential rider) arrives, the system matches it to a nearby driver instantly such that the

rider's waiting time is minimized. In most cases, the driver will rejoin the system and can be matched again once she finishes the service. Additionally, the arrival rates of requests changes dramatically across the day. Consider the online arrivals during peak hours and off-peak hours for example: the arrival rates in the former case can be much larger than the latter.

Motivated by these applications, we consider the *Online Matching with Reusable Resources* problem. As before we are given a weighted bipartite graph $G = (U, V, E)$. The vertex set U is available offline and at each time-step $t \in [T]$ a vertex $v \in V$ is sampled from a known product distribution. When a vertex v arrives, we have to match it to one of the *available* vertices in U or drop it; this action is irrevocable. The key difference is that the vertices in U are *reusable*. Once we assign v to u, the vertex u will *rejoin* the system after C_e rounds with $e = (u, v)$, where $C_e \in \{0, 1, \ldots, T\}$ is an integral random variable with known distribution. In other words, after C_e time-steps the vertex u becomes *available* for future matches. The random variable C_e is called the *occupation time* of u w.r.t. e. The goal is to maximize the total weight of the successful matches at the end of T time-steps. Here we make no assumption on the individual distributions $\mathcal{D}_t$ in the product distribution and can vary (adversarially) with time.

4.2 Our Contributions

The main contribution in this chapter is to define the Online Weighted Matching with Reusable Vertices problem and provide a new algorithm that has tight

theoretical guarantees. This model capture a wide range of real-world applications related to online scheduling, organ allocation, rideshare dispatch, among others. We extend the ideas from the previous chapter to design a new algorithm that achieves a competitive ratio of $\frac{1}{2} - \epsilon$ for any given constant $\epsilon > 0$. The key new ingredient in the algorithm is the notion of *attenuation* — using Monte-Carlo simulations to approximately estimate the probability of certain safe event and weight actions inversely proportional to this estimate. We also show that our algorithm is nearly optimal among all non-adaptive algorithms.

4.3 Main Model

In this section, we present a formal statement of our main model. Consider a weighted bipartite graph $G = (U, V, E)$ where U and V represent the offline and online vertices respectively. Let $w : E \to \mathbb{R}^+$ denote the weight function that assigns a weight to every edge in the graph. We have a finite time horizon T (known beforehand) and for each time $t \in [T]$, a vertex v is sampled (we use the term v *arrives*) from a known probability distribution $\{p_t(v)\}$ such that $\sum_{v \in V} p_t(v) \leq 1$[1] This sample is independent for each round t. The expected number of times v arrives across the T rounds, $\sum_{t \in [T]} p_t(v)$, is called the *arrival rate* for vertex v. Once a vertex v arrives, we need to make an *irrevocable decision* immediately: either reject v or assign v to one of its neighbors in U. For each $u \in U$, once it is assigned to some v, it becomes unavailable for C_e rounds with $e = (u, v)$, and subsequently rejoins the system. Here C_e is an integral random variable taking values in $\{0, 1, \ldots, T\}$

[1]Thus, with probability $1 - \sum_{v \in V} p_t(v)$, none of the vertices from V will arrive at t.

with the distribution known in advance. The goal is to design an online assignment policy such that the total (expected) weight of the assignments made is maximized. In this work we assume that $|V| \gg |U|$ and $T \gg 1$.

4.4 LP benchmark

We use the following LP to upper-bound the optimal online algorithm. The linear program has variables $x_t(e)$ for every $e \in E$ and $t \in [T]$.

$$
\begin{array}{lll}
\text{maximize} & \sum_{t\in[T]} \sum_{e\in E} w(e)\, x_t(e) & \text{such that} \\
& \sum_{e\in\delta(u)} x_t(e) \le p_t(v) & \forall u \in U, t \in [T] \\
& \sum_{t'<t} \sum_{e\in\delta(u)} x_{t'}(e) \Pr[C_e > t - t'] + \sum_{e\in\delta(u)} x_t(e) \le 1 & \forall u \in U, t \in [T] \\
& 0 \le x_t(e) \le 1 & \forall e \in E, t \in [T]
\end{array}
\tag{4.1}
$$

As before we have the following Lemma which states that the optimal value of the LP is an upper-bound to the optimal online algorithm.

Lemma 4. *Let $\{\mathbf{x}^*_t\}_{t\in[T]}$ denote an optimal solution to LP (4.1). Then we have* $\text{OPT} \le \sum_{t\in[T]} \sum_{e\in E} w(e) x^*_t(e)$.

Proof. The proof of this is similar to that of Lemma 1. Let $Y_t(e)$ denote an indicator random variable to denote if an edge $e = (u, v)$ was matched at time t in the optimal solution for an online sequence $\mathcal{A}$. Let $y_t(e) := \mathbb{E}_{\mathcal{A}}[Y_t(e)]$ for every $t \in [T]$ and $e \in E$. We will now show that the vector $\mathbf{y} := (y_t(e))_{e\in E, t\in[T]}$ is a feasible solution to the LP (4.3). Consider a vertex $u \in U$. Let $R_t(v)$ denote the indicator

random variable to denote if vertex v arrived at time t in $\mathcal{A}$. Thus, $\sum_{e\in\delta(u)} Y_t(e) \leq R_t(v)$. Taking expectation on both sides and using linearity of expectation we get $\sum_{e\in\delta(u)} y_t(e) \leq p_t(v)$. This shows that $\mathbf{y}$ is feasible to the first constraint. Since $Y_t(e)$ is an indicator random variable we have $0 \leq Y_t(e) \leq 1$. Taking expectation, we obtain $0 \leq y_t(e) \leq 1$ and thus $\mathbf{y}$ is feasible to the third constraint. We will now show that it is feasible to the second constraint. Consider a $u \in U$ and time-step $t \in [T]$. u is either matched at some time-step $t' < t$ and is not yet available at t or that u got matched at time t. Let $\mathbb{I}[C_e > t-t']$ denote the indicator for the re-appear time of edge e to be larger than $t - t'$. Thus, we have $\sum_{t'<t}\sum_{e\in\delta(u)} Y_{t'}(e)\mathbb{I}[C_e > t - t'] + \sum_{e\in\delta(u)} Y_t(e) \leq 1$. Taking expectation on both sides and using linearity of expectation we get that $\sum_{t'<t}\sum_{e\in\delta(u)} y_{t'}(e)\Pr[C_e > t - t'] + \sum_{e\in\delta(u)} y_t(e) \leq 1$. The expected weight of the optimal solution is thus $\mathbb{E}_{\mathcal{A}}[\sum_{t\in[T]}\sum_{e\in E} w(e)Y_t(e)]$ which is equal to $\sum_{t\in[T]}\sum_{e\in E} w(e)y_t(e)$ from linearity of expectation. Since $\mathbf{y}$ is feasible to the LP, the optimal value to LP (4.3) is at least as large as the expected optimal solution. □

In fact, it is suffices to have a finite sample estimate of $\Pr[C_e > t-t']$ for every edge $e \in E$. In particular, we have the following Lemma. The proof follows from the fact that $\{\mathbf{x}_t\}_{t\in[T]}$ is scaled down by a factor $(1+\delta)$ and hence the objective is scaled down by a factor $(1+\delta)$.

Lemma 5. *Suppose for some $\delta \geq 0$, we have an estimate $f(e, y)$ of $\Pr[C_e > y]$ for all edges e and $y >= 0$, where $f(e, y)/\Pr[C_e > y]$ always lies in $[1/(1+\delta), 1+\delta]$. Then, by using $f(e, t-t')$ in the LP instead of $\Pr[C_e > t - t']$ and scaling down*

the resultant vector $\{\mathbf{x}_t\}_{t\in[T]}$ *by* $(1+\delta)$*, we only get a further loss of* $(1+\delta)$ *in the competitive ratio.*

4.5 $1/2-\epsilon$-competitive algorithm

The main idea of the algorithm is as follows. Let $\{\mathbf{x}^*_t\}_{t\in[T]}$ denote an optimal solution to LP (4.1). Suppose we aim to develop an online algorithm achieving a ratio of $\gamma \in [0,1]$. Consider an assignment $e=(u,v)$ when some v arrived at time t. Let $\mathrm{SF}_t(e)$ be the event that e is safe at t (*i.e.,* u is available at t). By using Monte-Carlo simulations on the algorithm's strategy up to t, we can get an estimate of $\Pr[\mathrm{SF}_t(e)]$, denoted by $\beta_t(e)$, within an arbitrary small error. Therefore when $\mathrm{SF}_t(e)$ holds, we assign v to u with probability $\frac{x^*_t(e)}{p_t(v)}\frac{\gamma}{\beta_t(e)}$ when v arrives, which leads to the fact that e is assigned with probability exactly equal to $\gamma x^*_t(e)$ unconditionally. We call any strategy that satisfies $\gamma \le \beta_t(e)$ as *valid*. At the outset, this looks similar to the Inverse Propensity Scoring (IPS) used in the multi-armed bandit literature Auer et al. [24]. However, there is a key difference between IPS estimates and these Monte-Carlo estimates. In the bandit literature, one usually scales the value by the probability of playing an action, since this is the *cost* of observing only bandit feedback. However, here we scale by a quantity that depends on the probability of a certain event happening during the *run* of the algorithm, because of playing other actions. The linear program gives a distribution over the edges assuming that all the neighbors are available. Hence this scaling can be interpreted as the *cost* the algorithm needs to incur when some neighbors are already matched.

The above strategy of using Monte-Carlo simulations to weight the probability, called simulation-based attenuation, has been used previously for other problems, such as stochastic knapsack Ma [133] and stochastic matching Adamczyk et al. [2].

Throughout the analysis, we assume that we know the exact value of $\beta_t(e) := \Pr[\mathrm{SF}_t(e)]$ for all t and e. This is because, for any given accuracy $\epsilon > 0$ using $\Theta(\frac{1}{\Pr[\mathrm{SF}_t(e)]\epsilon^2} \cdot \log(\frac{1}{\delta}))$ samples, we can ensure that $\beta_t(e) \in \left[\Pr[\mathrm{SF}_t(e)]\frac{1}{1+\epsilon}, \Pr[\mathrm{SF}_t(e)]\right]$ with probability at least $1-\delta$, via a standard Chernoff-bound argument. Thus, this leads to a loss of at most $(1 - \Theta(\epsilon))$ multiplicative factor in the competitive ratio, which is negligible when $\epsilon = o(1)$[2]. Hence, ignoring it leads to a cleaner presentation. Algorithm 3 gives a formal description of our algorithm.

Algorithm 3: Simulation-based algorithm: ALG-REUSE(γ)

1 Solve LP (4.1) to obtain optimal solution $\{\mathbf{x}_t^*\}$.

2 For each time t, let v_t denote the request arriving at time t.

3 If $\delta_t(v_t) = \emptyset$, then reject v_t; otherwise choose $e \in \delta_t(v_t)$ with prob. $\frac{x_t^*(e)}{p_t(v_t)} \cdot \frac{\gamma}{\beta_t(e)}$ where $e = (u, v)$.

First, we show that when $\gamma = \frac{1}{2}$, Algorithm 3 is a valid strategy (*i.e.*, $\gamma \leq \beta_t(e)$ for every $t \in [T], e \in E_t$).

Lemma 6. *For every $t \in [T]$ and $e \in E_t$, we have $\beta_t(e) \geq \frac{1}{2}$.*

Proof. We prove this by induction on t. The base case is when $t = 1$. In this case we have $\beta_t(e) = 1$ for all $e = (u, *)$ trivially and thus, we are done.

[2] $o(1)$ is a vanishing term when both C_e and T/C_e are sufficiently large

Consider the inductive case. For all $t' < t$, assume that $\beta_{t'}(e) \geq \frac{1}{2}$. Consider time t and a given edge $e = (u, v_t)$. Assignment e is unsafe at t iff u is already assigned to some v' at $t' < t$ and that the assignment $e' = (u, v')$ makes u unavailable at time t. Therefore we have,

$$1 - \beta_t(e) = 1 - \Pr[\mathrm{SF}_t(e)] = \sum_{t'<t} \sum_{e \in \delta_t(u)} \frac{x^*_{t'}(e)}{2} \Pr[C_e > t - t']. \tag{4.2}$$

Using the constraints of the LP (4.1) with Eq. (4.2) we have,

$$\beta_t(e) = 1 - \sum_{t'<t} \sum_{e \in \delta_t(u)} \frac{x^*_{t'}(e)}{2} \Pr[C_e > t - t'] \geq \frac{1}{2} + \frac{1}{2} \sum_{e \in \delta_t(u)} x^*_t(e) \geq \frac{1}{2}.$$

Hence we have $\sum_{e \in \delta_t(v_t)} \frac{x^*_t(e)}{p_t(v_t)} \cdot \frac{\gamma}{\beta_t(e)} \leq \sum_{e \in \delta_t(v_t)} \frac{x^*_t(e)}{p_t(v_t)} \leq 1$ and thus we are done. □

We will now prove that the competitive ratio of Algorithm 3 is at least $\frac{1}{2} - o(1)$. In particular, we prove the following theorem.

Theorem 2. *Algorithm* ALG-REUSE($\frac{1}{2}$) *achieves a competitive ratio of* $\frac{1}{2} - o(1)$.

Proof. Consider any edge e in the offline-graph. Using Lemma 6 and the definition of $\beta_t(e)$ we have that the probability that this assignment is chosen in the T rounds is at least $\sum_{t \in [T]} x^*_t(e)\gamma(1 - o(1))$. Thus, combining this with Lemma 3 we have that the competitive ratio is at least $\gamma(1 - o(1)) = \frac{1}{2}(1 - o(1))$. □

4.6 Lower-bounds

In this section we show that the analysis in the previous section is tight. In particular, any algorithm that uses the optimal solution of LP (4.1) as a guide for the online actions cannot get a competitive ratio better than $\frac{1}{2} + o(1)$. In particular, we show that the worst-case occurs for the following instance.

Construction 1. *Consider a complete bipartite graph $G = (U, V, E)$ where $|U| = K$, $|V| = T^2$. Let $p_t(v) = \frac{1}{T^2}$ for each $v \in V$ and $t \in [T]$. In other words, in each round $t \in [T]$, each $v \in V$ is sampled uniformly. For each e, let C_e be the same value K, which implies that any $u \in U$ will be unavailable for K rounds after an assignment. All assignments have the same weight (*i.e., *$w(e) = 1$ for all $e \in E$).*

Theorem 3. *No non-adaptive algorithm that uses the optimal solution of LP (4.1) as a guide for online actions, can achieve a competitive ratio better than $\frac{1}{2} + o(1)$ on the instance in Construction 1.*

Proof. First, split the T rounds in the online phase into $T - K + 1$ consecutive windows $\mathcal{W} = \{W_\ell\}_{\ell \in [T-K+1]}$ such that $W_\ell = \{\ell, \ell + 1, \ldots, \ell + K - 1\}$ for each $1 \leq \ell \leq T - K + 1$. Consider the benchmark LP (4.1) for the instance in construction 1. It can be written as follows.

$$\begin{array}{lllll} \text{maximize} & \sum_{t\in[T]} \sum_{e\in E} x_t(e) & \text{such that} & & \\ & \sum_{e\in\delta(u)} x_t(e) \leq \frac{1}{T^2} & \forall u \in U, t \in [T] & & \\ & \sum_{t\in W_\ell} \sum_{e\in\delta(u)} x_t(e) \leq 1 & & \forall u \in U, 1 \leq \ell \leq T - K + 1 & \\ & 0 \leq x_t(e) \leq 1 & \forall e \in E, t \in [T] & & \end{array} \tag{4.3}$$

We can verify that an optimal solution to LP (4.3) is as follows: $x_t^*(e) = 1/(T^2K)$ for all $e \in E$ and $t \in [T]$ with the optimal objective value of T. Consider any optimal non-adaptive algorithm. The expected number of arrivals of any $v \in V$ after T steps is $1/T$. Thus, for any non-adaptive algorithm ALG, it needs to specify the allocation distribution $\mathcal{D}_v$ for each v during the first arrival. Consider a given

non-adaptive algorithm; it can be described by a set of parameters $\{\alpha_{u,v} \in [0,1]\}$ for each pair $v \in V, u \in \delta(v)$ with $\sum_{u \in \delta(v)} \alpha_{u,v} \leq 1$, for each $v \in V$. In other words, the algorithm can be described by the probability that a $v \in V$ will be assigned to $u \in U$ in the run of the algorithm.

Let $\beta_u := \sum_{v \in \delta(u)} \alpha_{u,v} * \frac{1}{T^2}$, which is the probability that u is matched in each round if it is safe at the beginning of that round, when running ALG. Hence,

$$\sum_{u \in U} \beta_u = \sum_{u \in U} \sum_{v \in \delta(u)} \alpha_{u,v} \cdot \frac{1}{T^2} = \sum_{v \in V} \sum_{u \in \delta(v)} \alpha_{u,v} \cdot \frac{1}{T^2} \leq 1.$$

Consider a given u and the corresponding β_u. Let $\gamma_t(u)$ be the probability that u is available at time t. Then the expected number of matches of u after the T rounds is $\sum_{t \in [T]} \beta_u \gamma_t(u)$. Then the variables $\gamma_t(u)$ and β_u satisfy a recurrence, given by Lemma 7, at each time-step $t \in [T]$

Lemma 7. *$\forall 1 < t \leq T$, we have*

$$\gamma_t(u) + \beta_u \sum_{t-K+1 \leq t' < t} \gamma_{t'}(u) = 1$$

Proof. When $t = 1$, this follows from the fact that u is safe at time $t = 1$. For each time $t > 1$, let $\mathrm{SF}_t(u)$ be the event that u is safe at time t and $A_t(u)$ be the event that u is matched at time t. In each window of K time slots, we have that the events $\{\mathrm{SF}_t(u), A_{t'}(u), t - K + 1 \leq t' < t\}$ are all mutually exclusive and that together they cover the entire sample space. Therefore, we have,

$$\begin{aligned} 1 &= \Pr[\mathrm{SF}_t(u)] + \sum_{t-K+1 \leq t' < t} \Pr[A_{t'}(u)] \\ &= \gamma_t(u) + \beta_u \sum_{t-K+1 \leq t' < t} \gamma_t(u) \end{aligned}$$

□

Recall that the optimal value of the benchmark LP (4.3) is T. However, the performance of ALG is $\sum_{u\in U}\sum_{t\in[T]}\beta_u\gamma_t(u)$. Thus, the competitive ratio achieved by ALG can be captured by the following maximization program.

$$\begin{array}{lll} \text{maximize} & \frac{\sum_{u\in U}\sum_{t\in[T]}\beta_u\gamma_t(u)}{T} & \text{such that} \\ & \sum_{u\in U}\beta_u \le 1 & \forall u\in U, t\in[T] \\ & \gamma_t(u)+\beta_u\sum_{t-K+1\le t'<t}\gamma_{t'}(u)=1 & \forall 1<t\le T, u\in U \\ & \beta_u\ge 0, \gamma_1(u)=1 & \forall u\in U \end{array} \tag{4.4}$$

We now prove the following Lemma about the optimal value of the program (4.4). Theorem 3 follows directly from this lemma. □

Lemma 8. *The optimal value of the maximization program* (4.4) *is at most* $\frac{1}{2-1/K}+o(1)$ *when* $K=o(T)$.

Proof. Consider any given vertex $u\in U$. From Lemma 7 we have that $\gamma_t(u)+\beta_u\sum_{t-K+1\le t'<t}\gamma_{t'}(u)=1$ for all $1\le t\le T$. Summing both sides over $t\in[T]$, we have the following.

$$\Big(1+\beta_u(K-1)\Big)\sum_{t\in[T]}\gamma_t(u)=$$

$$T+\beta_u(K-1)\gamma_T(u)+\beta_u(K-2)\gamma_{T-1}(u)+\cdots+\beta_u\gamma_{T-K+2}(u)$$

$$\le T+K-1$$

Therefore we have,

$$\sum_{t\in[T]}\gamma_t(u)\le\frac{T}{1+\beta_u(K-1)}+\frac{K-1}{1+\beta_u(K-1)}\le\frac{T}{1+\beta_u(K-1)}+\frac{1}{\beta_u} \tag{4.5}$$

Define $H_u := \sum_{t\in[T]} \beta_u \gamma_t(u)$. From Eq. (4.5), we have that $H_u \leq \frac{T\beta_u}{1+\beta_u(K-1)} + 1$. Thus the objective value in the maximization program (4.4) can be upper-bounded as follows.

$$\frac{\sum_{u\in U}\sum_{t\in[T]} \beta_u\gamma_t(u)}{T} = \sum_{u\in U} \frac{H_u}{T} \leq \sum_{u\in U} \frac{\beta_u}{1+\beta_u(K-1)} + \frac{K}{T}.$$

Thus, the optimal value to the maximization program (4.4) can be upper bounded by the following maximization program.

$$\left\{ \max \sum_{u\in U} \frac{\beta_u}{1+\beta_u(K-1)} + \frac{K}{T} : \sum_{u\in U} \beta_u = 1, \beta_u \geq 0, \forall u \in U \right\}$$

According to our assumption we have $K = o(T)$; thus, the second term evaluates to $o(1)$. Let $g(x) := x/(1+x(K-1))$. For any $K \geq 2$, $g(x)$ is a concave function. Thus, when maximizing $g(x)$ subject to $\sum_{u\in U} \beta_u = 1$, the maximizer will be achieved when all $\beta_u = 1/K$. The resultant value is $\frac{1}{2-1/K} + o(1)$. □

Unconditional Hardness. Manshadi et al. [137] prove that for the online matching problem under known distribution (but disposable offline vertices), no algorithm can achieve a ratio better than 0.823. Since our setting generalizes this, their hardness results directly apply to our problem as well.

4.7 Conclusion

In this work, we provide a model that captures the application of assignment in ride-sharing platforms. One key aspect in our model is to consider the *reusable* aspect of the offline resources. This helps in modeling many other important applications where agents enter and leave the system *multiple* times (*e.g.,* ride-share,

organ allocation and crowdsourcing markets). We provide an LP based algorithm that achieves a tight competitive ratio of $\frac{1}{2} - o(1)$. We supplement this with a lower-bound for LP-based algorithms. In Chapter 6 we use this model to run experiments on the New York yellow cabs taxi dataset to show the application of this model for a real-world problem. We show that the proposed algorithms sometimes does better than simple heuristics that are commonly employed.

Chapter 5: Extension 2: Online Submodular Weighted Matching

5.1 Introduction

In this chapter, we consider another extension of the Online Weighted Matching problem called the *Online Submodular Matching* Problem. In the online weighted matching problem, each edge is associated with a weight and the goal of maximizing the total weight captures the notion of maximizing the total *relevance* of the match. In many applications, we also care about the *diversity* of the final matching alongside the relevance. Ahmed et al. [15] considered a motivating example of matching academic papers to potential reviewers: *simply* maximizing the relevance (the quality of each match) could potentially assign a paper to multiple scholars in a single lab due to shared expertise, which is undesirable. Instead, we want to assign each paper to *relevant* experts with diverse backgrounds to obtain comprehensive feedback. Maximizing diversity is of particular importance in various recommendation systems, ranging from recommendations of new books and movies on eBay Chen et al. [62] to returning search-engine queries Agrawal et al. [9]. A common strategy to address diversity is to first formulate a specific objective (typically maximization over a submodular function capturing the balance of diversity and relevance and then design an efficient algorithm—typically a greedy one—to solve it (*e.g.,* Ahmed

et al. [15] and references within). Inspired by the wide range of applications we propose the Online Submodular Matching problem and design algorithms that give provable guarantees. We depart from the greedy paradigm of algorithms and use tools from submodular optimization.

5.2 Our Contributions

Our contributions can be summarized as follows.We propose the Online Submodular Bipartite Matching problem, which captures the balance between *relevance* and *diversity* in the context of matching markets. We then provide a *provably* good algorithm for this model. The algorithm is based on Contention Resolution schemes (CRS, Chekuri et al. [61]) used in the offline submodular maximization literature. The key algorithmic idea is to construct an offline guide based on using CRS on the expected offline problem and use this guide in the online phase. Our algorithm leverages useful properties of CRS from prior work to argue about the total expected objective of the online algorithm.

5.3 Problem Model

Consider a given bipartite graph $G = (U, V, E)$ where U and V represent the offline and online vertices respectively. Additionally,. We have a finite time horizon T (known beforehand) and for each time (or round) $t \in [T] := \{1, 2, \ldots, T\}$, at most one vertex v is sampled—in which case we say v *arrives*—from a given *known* probability distribution $\{p(v)\}_{v \in V}$. The sampling process is independent across

different times. As in chapter 3 we consider the case of *integral* arrival rates — thus wlog the arrival rate of each vertex $v \in V$ is 1. This implies that $p(v) = \frac{1}{T}$ for every vertex $v \in V$. When a vertex v arrives, we need to make an *immediate* and *irrevocable* decision: either to reject v or assign v to one of its neighbors in U. Each u has a unit capacity: it will be unavailable in the future upon being matched.[1] We are given a *non-negative monotone submodular function* $f : 2^E \to \mathbb{R}^+$ as input. Our goal is to design an online matching algorithm such that $\mathbb{E}[f(\mathcal{M})]$ is maximized, where $\mathcal{M}$ is the final (random) matching obtained.

Related model. One important direction in addressing diversity in online algorithms has been via online convex programming. In particular, Agrawal and Devanur [10] considered the model of maximizing a concave function under convex constraints. At each time-step a random vector is drawn from an unknown distribution and the goal is to satisfy a convex constraint in expectation. Our work differs from theirs in multiple aspects. First, the offline problem of Agrawal and Devanur [10] is poly-time solvable, while our problem even in the offline version has unknown hardness (status unknown for both NP- and APX-hardness). Equivalence between discrete and continuous functions exists for submodular minimization via the Lovász extension. However, a similar continuous relaxation for submodular maximization is NP-hard to evaluate[2]. Hence, it is unclear how one would use their model to address our problem. Secondly, they consider the large budget regime while all matching

[1] The general case where each u has a given capacity $B(u)$ can be reduced to this by creating $B(u)$ copies of u.

[2] *e.g.,* Slide 26 in `https://goo.gl/HAhqaZ`

type problems differ from allocation problems in that this assumption is not true (in fact, the main challenge is small budgets). The other difference is that our "known i.i.d." gives algorithm design more power as compared to unknown distributions and therefore helps obtain improved ratios rigorously. For example, the online matching problem with linear objectives has been studied both in unknown distribution and known i.i.d. models separately since it presents a natural trade-off—knowing more information about the distribution and the competitive ratio. Based on applications, one would make assumption one-way or the other.

Special cases. Our model generalizes some well-known problems in this literature. Note that if the submodular function is just a linear function of the weights this reduces to online weighted matching in Chapter 3. Our model can also capture the Submodular Welfare Maximization (SWM) problem Kapralov et al. [114]. Given an instance of SWM we can add polynomially many extra vertices and reduce it to an instance of our problem. The SWM problem is defined as follows. We have n vertices in U which are available offline. With each vertex $u \in U$ we have a monotone submodular function $g_u : 2^V \to \mathbb{R}$ associated with it. When a vertex v arrives we need to match v to one of its neighbors in U. At the end of the online phase, let $S_1, S_2, \ldots, S_n$ be the set of vertices assigned to vertices $1, 2, \ldots, n$ respectively. The goal is to maximize the sum $\sum_{i \in [n]} w_i(S_i)$.

Note that sum of submodular functions is a submodular function. Hence to make the objective function of SWM fit our framework, we do the following. Let $E(u)$ denote the set of edges incident to u. Define a function $\tilde{g}_u : E(u) \to \mathbb{R}$. For

any subset $S \subset E(u)$, we let $\tilde{g}(S) := g(S(v))$ where $S(v)$ is the set of endpoints of S in V. The non-trivial part to handle is that in SWM any vertex in U can be matched multiple times, while in our setting we allow any U to be matched exactly once. We can overcome this by creating additional vertices as follows. For any $u \in U$, create $|\delta(u)| * T$ copies. A copy, indexed by $v \in \delta(u)$ and $t \in [T]$ has an edge only to v. Additionally, wlog we can enforce that at time t, an algorithm can be matched to vertices whose copy is indexed by t (this doesn't change the optimal value). Therefore, we now have an instance with a submodular objective and matching constraints.

Consider an arrival sequence τ. Let the optimal allocation in SWM under this arrival be $S_{\tau,1}, S_{\tau,2}, \ldots, S_{\tau,n}$ at times $T_1, T_2, \ldots, T_n$. Then note that by considering the edges $(u_{S_{\tau,i},T_{\tau,i}}, S_{\tau,i})$ yields the same value to the objective in the Online Submodular Matching model. Additionally, the above argument also holds in the other direction since there is a one-to-one mapping between the two optimal solutions.

5.4 Background on Submodular Optimization

Definition 5 (Submodular function)**.** *A function $f : 2^{[n]} \to \mathbb{R}^+$ on a ground-set of elements $[n] := \{1, 2, \ldots, n\}$ is called submodular if for every $A, B \subseteq [n]$, we have that $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ and $f(\phi) = 0$. Additionally, f is said to be monotone if for every $A \subseteq B \subseteq [n]$, we have that $f(A) \leq f(B)$.*

For our algorithms, we assume a *value-oracle* access to a submodular function. This means that, there is an oracle which on querying a subset $T \subseteq [n]$, returns the

value $f(T)$. The algorithm does not have access to f *explicitly*.

Examples. Some common examples of submodular functions include the coverage function, piece-wise linear functions, budget-additive functions among others. In our experiments section, we use the following two examples.

1. *Coverage function.* Given a universe $\mathcal{U}$ and g subsets $A_1, A_2, \ldots, A_g \subseteq \mathcal{U}$, the function $f(S) = |\cup_{i \in S} A_i|$ is called the coverage function for any $S \subseteq [g]$. This can naturally be extended to the weighted case. Given a non-negative weight function $w : \mathcal{U} \to \mathbb{R}^+$, then the weighted coverage function is defined as $f(S) = w(\cup_{i \in S} A_i)$.

2. *Budget-additive function.* For a given total budget B and a set of weights $w_i \geq 0$ on the elements $[g]$ of universe $\mathcal{U}$, for any subset $S \subseteq \mathcal{U}$ the budget-additive function is defined as $f(S) = \min\{\sum_{i \in S} w_i, B\}$.

Definition 6 (Multilinear extension)**.** *The multilinear extension of a submodular function f is the continuous function $F : [0,1]^n \to \mathbb{R}^+$ defined as $F(\mathbf{x}) := \sum_{T \subseteq [n]} (\prod_{k \in T} x(k) \prod_{k \notin T} (1 - x(k))) f(T)$.*

Note that $F(\mathbf{x}) = f(\mathbf{x})$ for every $\mathbf{x} \in \{0,1\}^n$. The multilinear extension is a useful tool in maximization of submodular objectives. In particular, the above has the following probabilistic interpretation. Let $\mathcal{R}_{\mathbf{x}} \subseteq [n]$ be a random subset of items where each item $i \in [n]$ is added into $\mathcal{R}_{\mathbf{x}}$ independently with probability x_i. We then have $F(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_{\mathbf{x}})]$.

We now describe the key theorem relating sub-modular functions to its multi-linear extension. This theorem was proved in Bansal et al. [35] and was generalized

to a procedure called *contention resolution schemes* (CRS) by Chekuri et al. [61]. In this work, we only require the following theorem, but use the term CRS to refer to the following theorem.

Theorem 4 (Bansal et al. [35])**.** *Let f be a non-negative monotone submodular function over E with $|E| = m$. Given a fractional vector $\mathbf{x} \in [0,1]^m$, let $\mathbf{X} = (X(e))_{e \in E}$ be a random binary vector such that each $X(e)$ is a Bernoulli random variable with mean $x(e)$. Suppose $\mathbf{Z} \leq \mathbf{X}$ is another random binary vector satisfying the following properties (1) & (2) for some $\alpha > 0$. Then we have that $\mathbb{E}[f(\mathbf{Z})] \geq \alpha \mathbb{E}[f(\mathbf{X})]$.*

1. ***Marginal Property***: $\Pr[Z(e) = 1 | X(e) = 1] \geq \alpha$.
2. ***Monotonicity Property***: $\Pr[Z(e) = 1 | \mathbf{X} = \mathbf{a}] \geq \Pr[Z(e) = 1 | \mathbf{X} = \mathbf{b}]$, $\forall \mathbf{a} \leq \mathbf{b} \in \{0,1\}^m, \forall e \in \text{support}(\mathbf{a}) := \{e' : a(e') = 1\}$.

By definition of the multilinear extension we have that $\mathbb{E}[f(\mathbf{X})] = F(\mathbf{x})$. Thus the theorem can be viewed as conditions when "linearity of expectation" can be applied to F.

5.5 Related Works in Submodular Optimization

The offline version of our problem is the well-studied "maximizing a monotone submodular function subject to a bipartite matching polytope constraint" problem. More generally, the constraint set can be viewed as an intersection of two partition matroids. The general area of submodular maximization is well studied; here, we

only survey algorithmic advances related to maximization of a monotone submodular function subject to various constraints. The classical work of Nemhauser et al. [151] showed that the natural greedy algorithm achieves an $(1-1/\mathsf{e})$-approximation under a cardinality constraint, which is optimal in the value oracle model assuming $P \neq NP$ Nemhauser and Wolsey [150]. Under a general matroid constraint, Calinescu et al. [52] gave an algorithm achieving the optimal ratio of $1-1/\mathsf{e}$ (in the value oracle model defined in Section 5.4) using the *pipage rounding* technique. Lee et al. [129] considered the constraint case of k matroids with $k \geq 2$ and presented a *local-search* based algorithm. Sarpatwar et al. [166] studied the case of intersection of k matroids and a single knapsack constraint and gave a $\frac{1-e^{-(k+1)}}{k+1}$-approximation algorithm for any $k \geq 1$. Recently a series of works has considered submodular maximization in the online setting. In particular, Buchbinder et al. [50] and Chan et al. [57] studied online submodular maximization in the adversarial arrival order with preemption: on arrival of an item, we should decide whether to accept it or not and *possibly rejecting a previously accepted item.* In this work, we do not allow preemption but consider a more flexible arrival assumption (*i.e.,* known i.i.d.). This makes the problem tractable and admits algorithms with non-trivial competitive ratios. Apart from the offline and online models, submodular maximization has received much attention in other models due to its applications in summarization Tschiatschek et al. [180], data subset selection and active learning Wei et al. [186], and diverse summarization Mirzasoleiman et al. [143], to name a few. It has been studied in the streaming Badanidiyuru et al. [32], Mirzasoleiman et al. [146], distributed Mirzasoleiman et al. [144, 145] and stochastic Karimi et al. [115], Stan et al. [171]

settings.

5.6 Mathematical Program to Upper-bound OPT

We use the following mathematical program to upper-bound OPT. This cannot be solved exactly in polynomial time but can be approximated to a factor of $1 - 1/\mathsf{e}$ efficiently (Adamczyk et al. [3], Calinescu et al. [52]). Let $F : [0,1]^m \to \mathbb{R}_+$ be the multilinear extension of f. Consider the following mathematical program.

$$\begin{array}{lll} \text{maximize} & F(\mathbf{x}) & \text{such that} \\ & \sum_{e \in \delta(v)} x(e) \leq r(v) & \forall v \in V \\ & \sum_{e \in \delta(u)} x(e) \leq 1 & \forall u \in U \\ & 0 \leq x(e) \leq 1 & \forall e \in E \end{array} \tag{5.1}$$

Lemma 9. *There is an efficient algorithm (running in polynomial time) which returns a feasible solution $\mathbf{x}^*$ to the program (5.1) such that $F(\mathbf{x}^*) \geq \left(1 - \frac{1}{\mathsf{e}}\right) \mathbb{E}[\mathrm{OPT}]$, where $\mathbb{E}[\mathrm{OPT}]$ is the offline optimal value.*

Proof. First, it is easy to see that the constraint set (denoted by $\mathcal{P}$) in LP (5.1) is downward closed. In other words, if $\mathbf{x}$ is a feasible solution, then $(1-\epsilon)\mathbf{x}$ for every $0 \leq \epsilon \leq 1$ is also a feasible solution. Define $f^+(\mathbf{y})$ to be the optimal solution of the following program.

$$f^+(\mathbf{y}) := \max \left\{ \sum_{A \subseteq E} \alpha_A f(A) \ : \ \sum_{A \subseteq E} \alpha_A \leq 1; \alpha_A \geq 0; \forall e \in E \ \sum_{A : e \in A} \alpha_A \leq y(e) \right\} \tag{5.2}$$

Using Lemma 6 in Adamczyk et al. [3] we have that the continuous greedy algorithm of Calinescu et al. [52] yields a feasible solution $\mathbf{x}^*$ such that $F(\mathbf{x}^*) \geq$

$(1 - 1/\mathsf{e}) \max_{\mathbf{y} \in \mathcal{P}} f^+(\mathbf{y})$. As noted in Adamczyk et al. [3], for any given $\mathbf{y} \in \mathcal{P}$, $f^+(\mathbf{y})$ is the maximum value of $\mathbb{E}[f(\mathcal{R})]$ over all possible random sets $\mathcal{R}$ such that $\Pr[e \in \mathcal{R}] \leq y(e)$ for each e. Thus, this implies that $\max_{\mathbf{y} \in \mathcal{P}} f^+(\mathbf{y})$ is an upper bound of the offline optimal. This proves that $F(\mathbf{x}^*) \geq (1 - 1/\mathsf{e}) \max_{\mathbf{y} \in \mathcal{P}} f^+(\mathbf{y}) \geq (1 - 1/\mathsf{e})\mathbb{E}[\text{OPT}]$. □

5.7 0.125-competitive Algorithm

We present an algorithm for the Online Submodular Matching problem that achieves a competitive ratio of at least 0.125. The central idea is as follows. We first start with an approximately optimal solution $\mathbf{x}^*$ to the mathematical program (5.1). We then obtain $\mathcal{R}_{\mathbf{x}^*}$, which is obtained by independently sampling each edge $e \in E$ with probability $x^*(e)$. Let $\mathbf{X} \in \{0,1\}^m$ be the indicator vector corresponding to $\mathcal{R}_{\mathbf{x}^*}$ such that $X(e) = 1$ iff $e \in \mathcal{R}_{\mathbf{x}^*}$. Let $E_{\mathbf{X}}(v) = \{e : e \in \delta(v), X(e) = 1\}$ and $E_{\mathbf{X}}(u) = \{e : e \in \delta(u), X(e) = 1\}$ be the set of sampled edges incident to u and v respectively. Now we obtain another random vector $\mathbf{Y} \in \{0,1\}^m$ from $\mathbf{X}$ by uniformly sampling an edge from $E_{\mathbf{X}}(u)$ for each $u \in U$ (the sampling process is independent across different u). We then use both $\mathbf{X}$ and $\mathbf{Y}$ to guide the online phase. Algorithm 4 describes this algorithm formally.

We now prove the following theorem about the performance of this algorithm. In particular, we prove the following theorem.

Theorem 5. *The algorithm* CR-ALG *achieves a competitive ratio of at least* $\frac{1}{2}(1 - \mathsf{e}^{-1/2})(1 - 1/\mathsf{e})$ *for Online Submodular Matching problem with integral arrival rates.*

Algorithm 4: A CR-based algorithm (CR-ALG)

Offline Phase:

1 Approximately solve the program (5.1) using the continuous greedy of Calinescu et al. [52]. Let $\mathbf{x}^* = (x(e)^*)_{e \in E}$ be the approximate solution with $F(\mathbf{x}^*) \geq (1 - 1/\mathsf{e})\mathbb{E}[\text{OPT}]$.

2 Independently sample each edge with probability $x^*(e)$. Let $\mathbf{X} = (X(e))_{e \in E} \in \{0,1\}^m$ be the resultant indicator vector.

3 For each $w \in U \cup V$, let $E_{\mathbf{X}}(w) := \{e : e \in \delta(w), X(e) = 1\}$ be the set of sampled edges incident to w. Sample one edge uniformly at random from $E_{\mathbf{X}}(u)$ for each u if $E_{\mathbf{X}}(u) \neq \emptyset$. Let $\mathbf{Y} \leq \mathbf{X}$ be the indicator vector of the edges chosen after both the sampling processes.

Online Phase:

4 When $v \in V$ arrives at time $t \in [T]$, sample an edge e uniformly from $E_{\mathbf{X}}(v)$. Match v iff $Y(e) = 1$ and $e = (u, v)$ is safe at t (*i.e.,* u is available); skip it otherwise.

Proof. Let $\mathbf{Z} = (Z(e))_{e \in E}$ be the indicator vector for any $e \in E$ being matched in Algorithm CR-ALG. Let $\mathbf{X}$ be as defined in Theorem 4 with $\mathbf{x} = \mathbf{x}^*$ where $\mathbf{x}^*$ is the approximate optimal solution to the offline program. We show that $\mathbf{Z}$ satisfies the two properties stated in Theorem 4 with $\alpha = \frac{1}{2}(1 - \mathsf{e}^{-0.5})$. Hence, we have that $\mathbb{E}[f(\mathbf{Z})] \geq \alpha\mathbb{E}[f(\mathbf{X})]$. From Lemma 9 we have that $\mathbb{E}[f(\mathbf{X})] = F(\mathbf{x}^*) \geq (1 - 1/\mathsf{e})\mathbb{E}[\text{OPT}]$. Combining these two facts proves Theorem 5.

We now show that $\mathbf{Z}$ satisfies the two properties stated in Theorem 4 with $\alpha = \frac{1}{2}(1 - \mathsf{e}^{-0.5})$. To prove this, we first prove the following Lemma.

Lemma 10. *Consider an edge $e = (u, v)$ with $X(e) = 1$. Let $|E_{\mathbf{X}}(u)| = k_u$ and $|E_{\mathbf{X}}(v)| = k_v$, where k_u and k_v are the number of edges incident to u and v in the sub-graph induced by $\mathbf{X}$, respectively. Then,*

$$\Pr[Z(e) = 1 \mid X(e) = 1, |E_{\mathbf{X}}(u)| = k_u, |E_{\mathbf{X}}(v)| = k_v] \geq \frac{1}{k_u}\Big(1 - \exp\big(-\frac{1}{k_v}\big)\Big). \quad (5.3)$$

Proof. A sufficient condition for $Z(e) = 1$ to occur given $X(e) = 1, |E_{\mathbf{X}}(u)| = k_u, |E_{\mathbf{X}}(v)| = k_v$ is that $Y(e) = 1$ in this conditional space and that $Z(e) = 1$ given that $Y(e) = 1$. Thus, we have

$$\begin{aligned}\Pr[Z(e) = 1 \mid X(e) = 1, |E_{\mathbf{X}}(u)| = k_u, |E_{\mathbf{X}}(v)| = k_v] \\ \geq \Pr[Y(e) = 1 \mid X(e) = 1, |E_{\mathbf{X}}(u)| = k_u] \\ * \Pr[Z(e) = 1 \mid Y(e) = 1, |E_{\mathbf{X}}(v)| = k_v, |E_{\mathbf{X}}(u)| = k_u]. \quad (5.4)\end{aligned}$$

Since $Y(e) = 1$ is set uniformly for every edge $e \in E_{\mathbf{X}}(u)$ we have that

$$\Pr[Y(e) = 1 \mid X(e) = 1, |E_{\mathbf{X}}(u)| = k_u] = \frac{1}{k_u}. \quad (5.5)$$

We now show that the following holds.

$$\Pr[Z(e) = 1 \mid Y(e) = 1, |E_{\mathbf{X}}(v)| = k_v, |E_{\mathbf{X}}(u)| = k_u] \geq \sum_{t \in [T]} \frac{1}{T}\frac{1}{k_v}\Big(1 - \frac{1}{T \cdot k_v}\Big)^{t-1}. \quad (5.6)$$

Consider a given $e = (u, v)$ with $Y(e) = 1$. We compute the probability that u is safe. At any time $t \in [T]$, the probability that a vertex u is matched is at most $\frac{1}{k_v T}$ (*i.e.,* the probability that a neighbor v of u arrives in this time-step). Thus, the probability that u is safe is at least $\left(1 - \frac{1}{Tk_v}\right)^{t-1}$. Thus, the probability that $e = (u, v)$ is matched in the T time-steps is at least $\sum_{t \in [T]} \frac{1}{T}\frac{1}{k_v}\left(1 - \frac{1}{T \cdot k_v}\right)^{t-1}$.

Plugging equations (5.5) and (5.6) back into Eq. (5.4) we get,

$$\begin{aligned}
&\Pr[Z(e) = 1 \mid X(e) = 1, |E_{\mathbf{X}}(u)| = k_u, |E_{\mathbf{X}}(v)| = k_v] \\
&\geq \frac{1}{k_u} \sum_{t=1}^{T} \frac{1}{T} \frac{1}{k_v} \Big(1 - \frac{1}{T \cdot k_v}\Big)^{t-1} \\
&\geq \frac{1}{k_u} \Big(1 - \exp\big(-\frac{1}{k_v}\big)\Big).
\end{aligned}$$

□

Given Lemma 10, we are now ready to prove the two properties. First, we show that the **Marginal Property** holds with $\alpha = (1 - \mathsf{e}^{-0.5})$. Taking expectation over the random sets $|E_{\mathbf{X}}(u)|$ and $|E_{\mathbf{X}}(v)|$, Eq. (5.3) in Lemma 10 becomes,

$$\Pr[Z_e = 1 | X_e = 1] \geq \mathbb{E}_{k_u, k_v}\Big[\tfrac{1}{k_u}\Big(1 - \exp\big(-\tfrac{1}{k_v}\big)\Big)\Big] \tag{5.7}$$

Since $\frac{1}{k_u}$ and $\Big(1 - \exp\big(-\frac{1}{k_v}\big)\Big)$, are convex functions in k_u and k_v by Jensen's inequality, this evaluates to,

$$\mathbb{E}_{k_u, k_v}\Big[\frac{1}{k_u}\Big(1 - \exp\big(-\frac{1}{k_v}\big)\Big)\Big] \geq \frac{1}{\mathbb{E}[|E_{\mathbf{X}}(u)|]}\Big(1 - \exp\big(-\frac{1}{\mathbb{E}(|E_{\mathbf{X}}(v)|)}\big)\Big). \tag{5.8}$$

From construction we have, $\mathbb{E}[|E_{\mathbf{X}}(u)|] = 1 + \sum_{e' \in E(u), e' \neq e} x^*(e') \leq 2$. Likewise, we have $\mathbb{E}[|E_{\mathbf{X}}(v)|] = 1 + \sum_{e' \in E(v), e' \neq e} x^*(e') \leq r(v) = 2$, since $\mathbf{x}^* = (x^*(e))_{e \in E}$ is feasible to the mathematical program (5.1). Thus, the RHS in Eq. (5.8) can be lower-bounded by

$$\frac{1}{\mathbb{E}[|E_{\mathbf{X}}(u)|]}\Big(1 - \exp\big(-\frac{1}{\mathbb{E}(|E_{\mathbf{X}}(v)|)}\big)\Big) \geq \frac{1}{2}\Big(1 - e^{-1/2}\Big) (:= \alpha). \tag{5.9}$$

Combining Eq. (5.9) and (5.7) proves the marginal property.

The **Monotonicity Property** follows from combining the fact that $\frac{1}{k_u}$ and $\Big(1 - \exp\big(-\frac{1}{k_v}\big)\Big)$ are decreasing functions in k_u and k_v respectively and Eq. (5.3). □

5.8 Conclusion

In this chapter, we proposed a new model, Online Submodular bipartite matching, which effectively captures notions such as relevance and diversity in matching markets. Many applications such as advertising, hiring diverse candidates, recommending movies or songs naturally fit within this framework. We propose an algorithm based on contention-resolution schemes and prove theoretical guarantees on their performance. In Dickerson et al. [78], we propose another algorithm that has an improved competitive ratio in the special case when $|U| = o(\sqrt{T})$ and $T \to \infty$. For this special case, this algorithm achieves a competitive ratio of $(1 - 1/\mathsf{e})^2$. Dickerson et al. [78] also has initial numerical experiments on the MovieLens dataset Harper and Konstan [103] which validates the theoretical results.

Chapter 6: Rideshare: Empirical Evaluation of Online Matching

6.1 Introduction

In this chapter, we consider a case-study of the Online Matching with Reusable Vertices problem studied in Chapter 4 in the context of ride-share. We cast the problem of matching drivers to riders as an Online Matching problem. Using the publicly available New York City yellow cabs dataset,[1] which contains the trip records for trips in Manhattan, Brooklyn, and Queens for the year 2013, we evaluate the performance of the proposed algorithm and many natural heuristics. Further, we also show that many of the assumptions made to help provable guarantees hold in practice.

6.2 Dataset and Assumptions

The dataset is split into 12 months. For each month we have numerous records each corresponding to a single trip. Each record has the following structure. We have an anonymized license number which is the primary key corresponding to a car. For privacy purposes a long string is used as opposed to the actual license number. We then have the time at which the trip was initiated, the time at which the trip ended, and the total time of the trip in seconds. This is followed by the starting

[1] `http://www.andresmh.com/nyctaxitrips/`

coordinates (*i.e.,* latitude and longitude) of the trip and the destination coordinates of the trip.

Assumptions. We make two assumptions specific to our experimental setup. Firstly, we assume that every *car* starts and ends at the same location, for *all* trips that it makes. Initially, we assign every car a location (potentially the same) which corresponds to its *docking* position. On receiving a request, the car leaves from this docking position to the point of pick-up, executes the trip and returns to this docking position. Secondly, we assume that occupation time distributions (OTD) associated with all matches are identically (and independently) distributed, *i.e.,* $\{C_e\}_{e \in E}$ follow the same distribution. Note that this is a much stronger assumption than what we made in the model, and is completely inspired by the dataset (see Section 6.4). We test our model on two specific distributions, namely a *normal* distribution and the *power-law* distribution (see Figure 6.5). The docking position of each car and parameters associated with each distribution are all learned from the training dataset (described below in the **Training** discussion).

6.3 Experimental Setup

The cars represent the set of vertices U and the request types represent the set of vertices V. We assign an edge between a car $u \in U$ and request type $v \in V$ iff the request type v can be assigned to the car u. Given this modeling, the experimental setup is as follows. We randomly select 30 cabs (each cab is denoted by u). We discretize the Manhattan map into cells such that each cell is approximately 4 miles

(increments of 0.15 degrees in latitude and longitude). For each pair of locations, say (a, b), we create a request *type* v, which represents all trips with starting and ending locations falling into a and b respectively. In our model, we have $|U| = 30$ and $|V| \approx 550$ (variations depending on day to day requests with low variance). We focus on the month of January 2013. We split the records into 31 parts, each corresponding to a day of January. We choose a random set of 12 parts for *training* purposes and use the remaining for *testing* purposes.

The edge weight $w(e)$ on $e = (u, v)$ (*i.e.,* edge from a car u to type v) is set as a function of two distances in our setup. The first is the trip distance (*i.e.,* the distance from the starting location to the ending location of v, denoted L_1) while the second is the docking distance (*i.e.,* the distance from the docking position of u to the starting/ending location of v, denoted L_2). We set $w(e) = \max(L_1 - \alpha L_2, 0)$, where α is a parameter capturing the subtle balance between the positive contribution from the trip distance and negative contribution from the docking distance to the final profit. We set $\alpha = 0.5$ for the experiments. We consider each single day as the time horizon and set the total number of rounds $T = \frac{24*60}{5} = 288$ by discretizing the 24-hour period into a time-step of 5 minutes. Throughout this section, we use time-step and round interchangeably.

Training. We use the training dataset of 12 days to learn various parameters. As for the arrival rates $\{p_t(v)\}_{t\in[T],v\in V}$, we first count the total number of appearances of each request type v at time-step t in the 12 parts (denote it by $c_t(v)$). Then, we set $p_t(v) = c_t(v)/12$ when we assume known adversarial arrivals (KAD) and

$p(v) = p_t(v) = (c/12)/T$ (*i.e.,* the arrival distributions are assumed the same across all the time-steps for each v) when we assume i.i.d. arrivals (KIID). The estimation procedure for the parameters of the two distributions for the occupation time is as follows. We first compute the average number of seconds between two *requests* in the dataset (note this was 5 minutes in the experimental setup). We then assume that each *time-step* of our online process corresponds to a time-difference of this average in seconds. We then compute the sample mean and sample variance of the trip lengths (as number of seconds taken by the trip divided by five minutes) in the 12 parts. Hence we use the normal distribution obtained by this sample mean and standard deviation as the distribution with which a car is unavailable. We assign the docking position of each car to the location (in the discretized space) in which the majority of the requests were initiated (*i.e.,* starting location of a request) and matched to this car.

6.4 Justifying The Two Important Model Assumptions

Known Adversarial Distributions. Figure 6.4 plots the number of arrivals of a particular type at various times during the day. Notice the significant increase in the number of requests in the middle of the day as opposed to the mornings and nights. This justified our arrival assumption of KAD which assumes different arrival distributions at different time-steps. Hence the LP (and the corresponding algorithm) can exploit this vast difference in the arrival rates and potentially obtain improved results compared to the assumption of Known Identical Independent Distributions

(KIID). This is confirmed by our experimental results shown in Figures 6.1 and 6.2.

Identical Occupation Time Distribution. We assume each car will be available again via an independent and identical random process regardless of the matches it received. The validity of our assumptions can be seen in Figures 6.5 and 6.6, where the x-axis represents the different occupation time and the y-axis represents the corresponding number of requests in the dataset responsible for each occupation time. It is clear that for most requests the occupation time is around 2-3 time-steps and dropping drastically beyond that with a long tail. Figure 6.6 displays occupation times for two representative (we chose two out of the many cars we plotted, at random) cars in the dataset; we see that the distributions roughly coincide with each other, suggesting that such distributions can be learned from historical data and used as a guide for future matches.

6.5 Results

Inspired by the experimental setup by Tong et al. [176], we run five different algorithms on our dataset. The first algorithm is the ALG-LP. In this algorithm, when a request v arrives, we choose a neighbor u with probability $x_t^*(e)/p_t(v)$ with $e = (u, v)$ if u is available. Here $x_t^*(e)$ is an optimal solution to the benchmark LP (4.3) and $p_t(v)$ is the arrival rate of type v at time-step t. The second algorithm is called ALG-SC-LP. Recall that $E_t(v)$ is the set of "safe" or available assignments with respect to v when the type v arrives at t. Let $x_t(v) = \sum_{e \in E_t(v)} x_t^*(e)$. In ALG-SC-LP, we sample a safe assignment for v with probability $x_t^*(e)/x_t(v)$. The

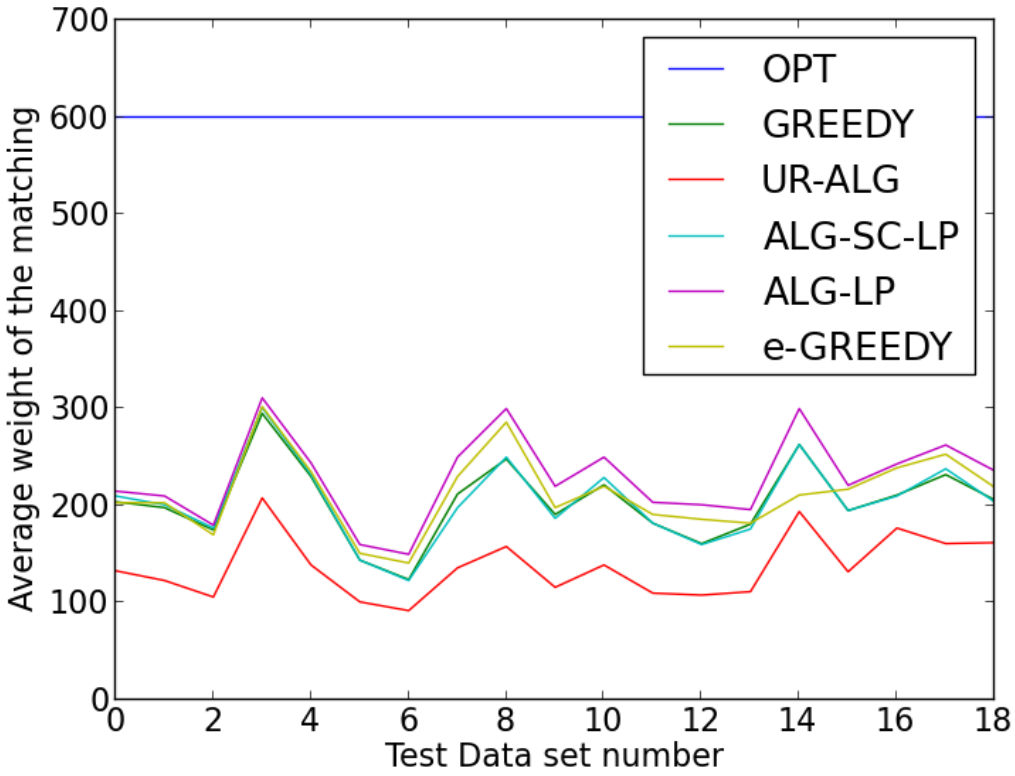

Figure 6.1: OTD is normal distribution under KIID

next two algorithms are heuristics oblivious to the underlying LP. Our third algorithm is called GREEDY which is as follows. When a request v comes, match it to the safe neighbor u with the highest edge weight. Our fourth algorithm is called UR-ALG which chooses one of the safe neighbors uniformly at random. Finally, we use a combination of LP-oblivious algorithm and LP-based algorithm called ϵ-GREEDY. In this algorithm when a type v comes, with probability ϵ we use the greedy choice and with probability $1-\epsilon$ we use the optimal LP choice. In our algorithm, we optimized the value of ϵ and set it to $\epsilon = 0.1$. We summarize our results in the following plots. Figures 6.1, 6.2, and 6.3 show the performance of the five algorithms and OPT (optimal value of the benchmark LP) under the different as-

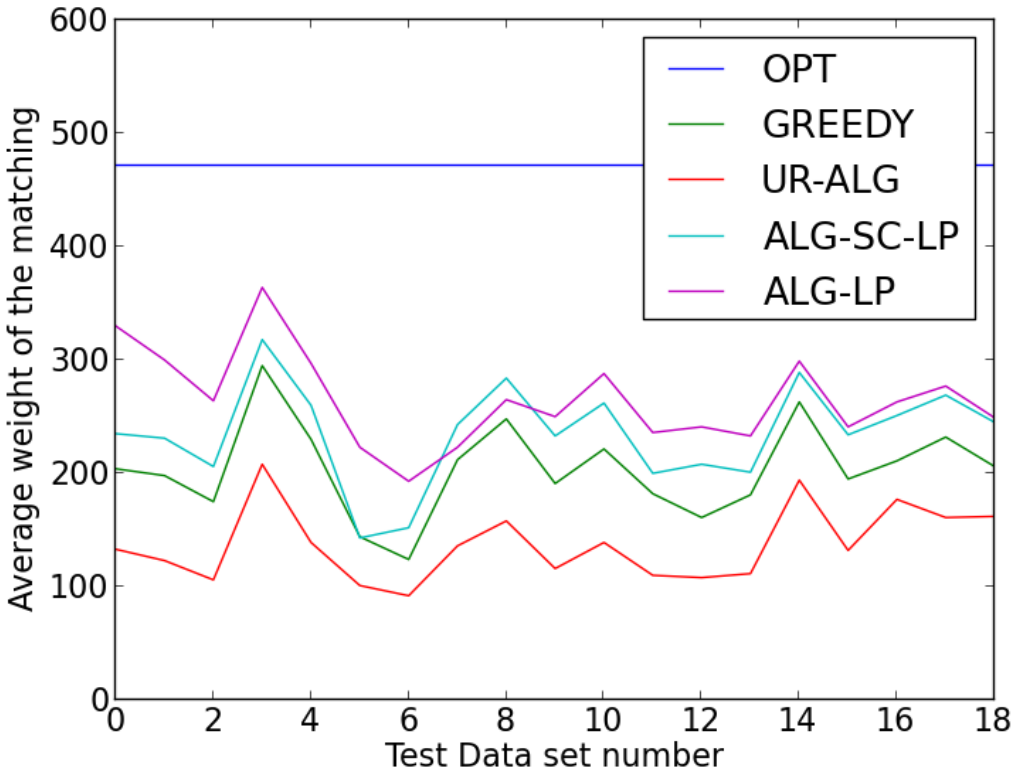

Figure 6.2: OTD is normal distribution under KAD

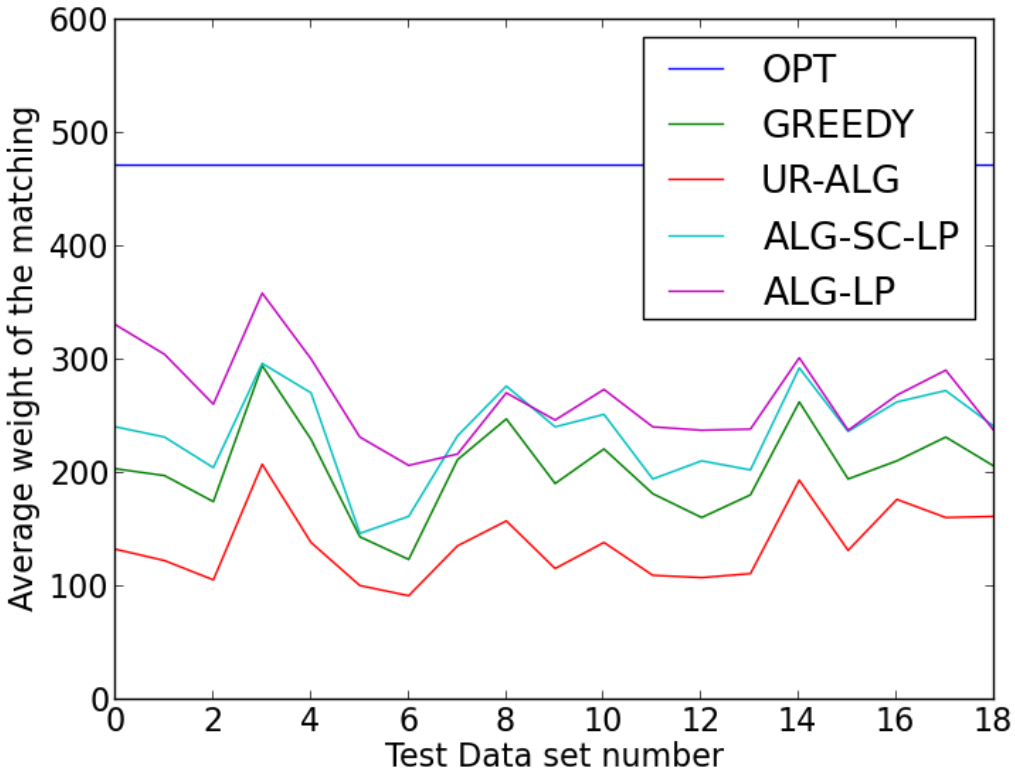

Figure 6.3: OTD is power law distribution under KAD

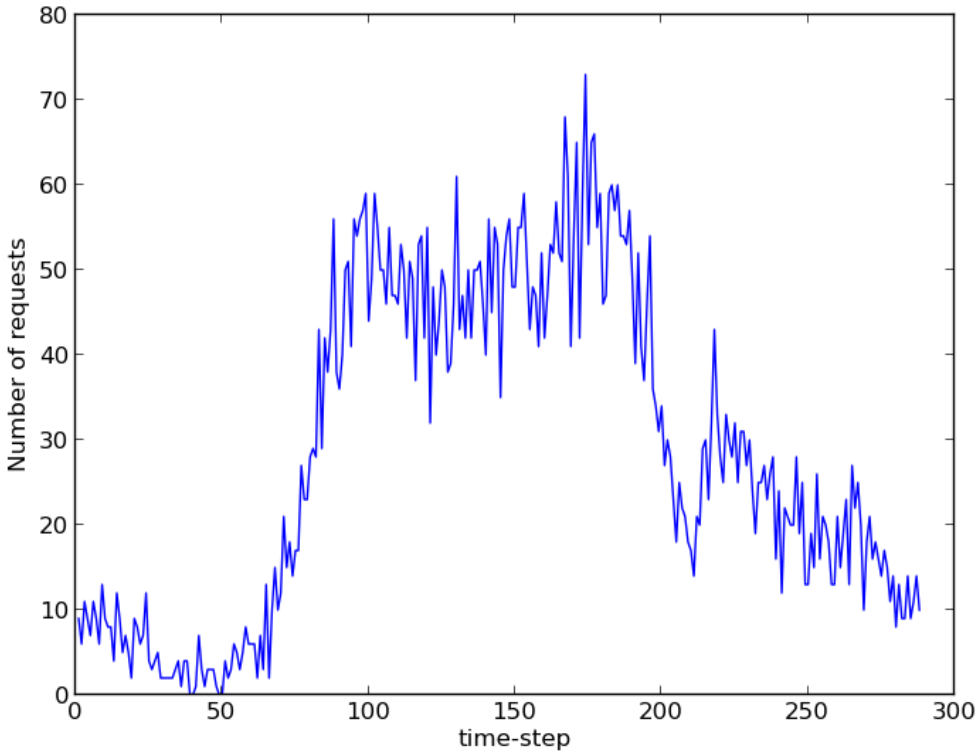

Figure 6.4: The number of requests of a given type at various time-steps. x-axis: time-step, y-aixs: number of requests

sumptions of the OTD (normal or power law) and online arrives (KIID or KAD). In all three figures the x-axis represents test data-set number and the y-axis represents average weight of matching.

Discussion. From the figures, it is clear that both the LP-based solutions, namely ALG-LP and ALG-SC-LP, do better than choosing a free neighbor uniformly at random. Additionally, with distributional assumptions the LP-based solutions outperform greedy algorithm as well. We would like to draw attention to a few interesting details in these results. Firstly, compared to the LP optimal solution, our LP-based algorithms have a competitive ratio in the range of 0.5 to 0.7. We believe

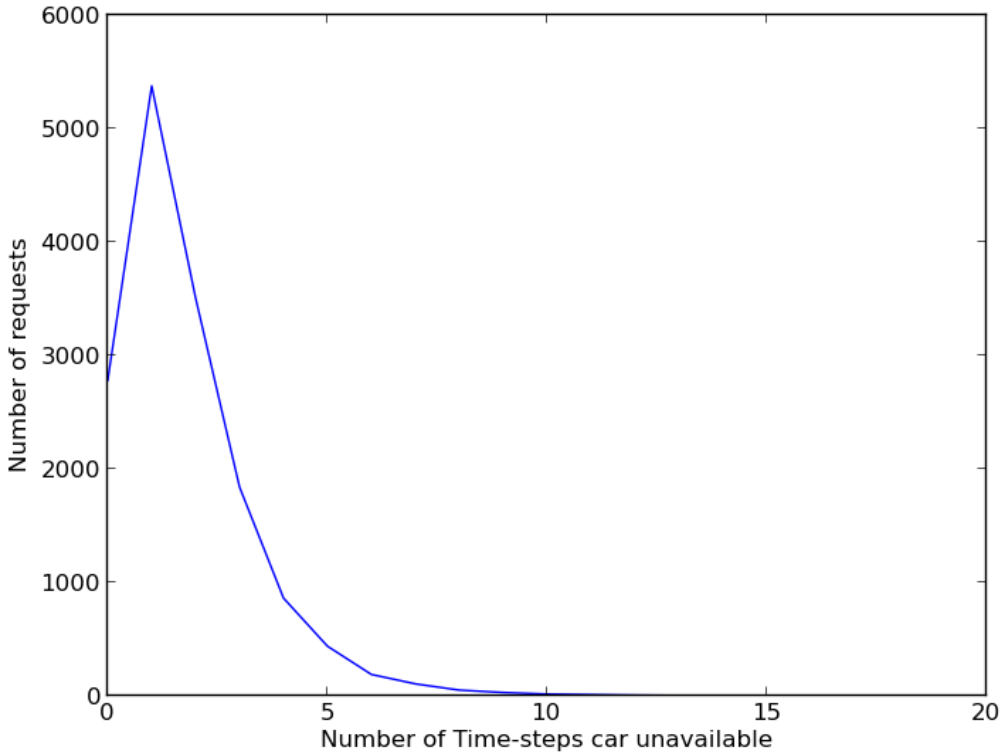

Figure 6.5: Occupation time distribution of all cars. x-axis: number of time-steps, y-axis: number of requests

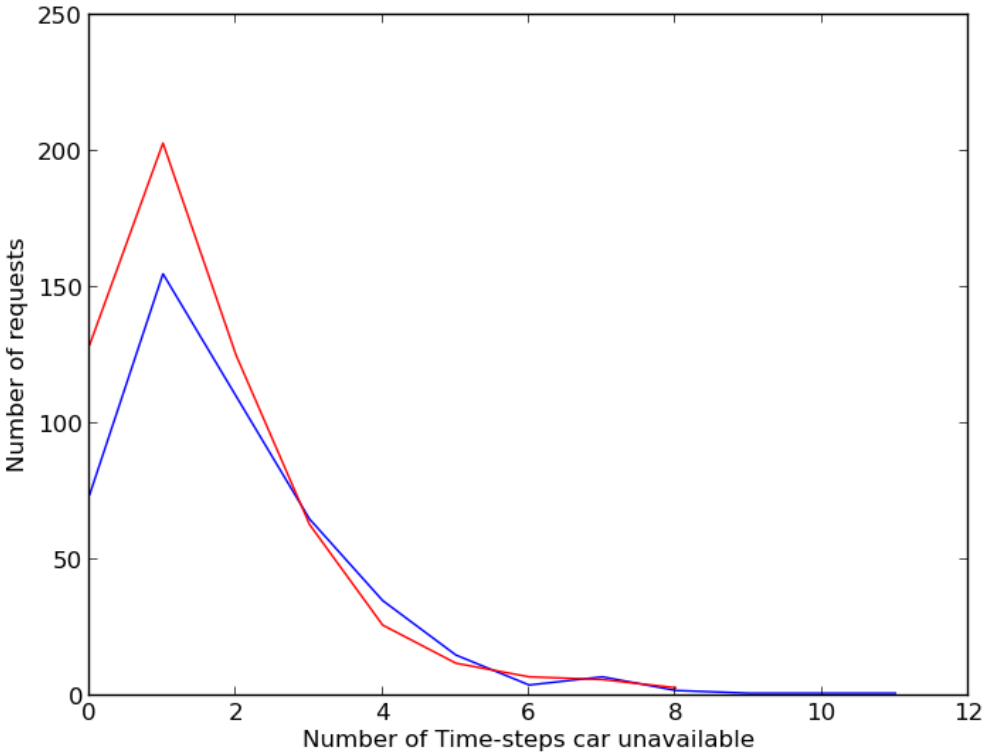

Figure 6.6: Occupation time distribution of two different cars. x-axis: number of time-steps, y-axis: number of requests

this is because of our experimental setup. In particular, we have that the rates are high (> 0.1) only in a few time-steps while in all other time-steps the rates are very close to 0. This means that it resembles the structure of the *theoretical* worst case example we showed in Chapter 4. In future experiments, running our algorithms during *peak* periods (where the request rates are significantly larger than 0) may show that competitive ratios in those cases approach 1. Secondly, it is surprising that our algorithm is fairly robust to the *actual* distributional assumption we made. In particular, from Figures 6.2 and 6.3 it is clear that the difference between the assumption of normal distribution versus power-law distribution for the unavailability of cars is *negligible*. This is important since it might not be easy to learn the *exact* distribution in many cases (*e.g.,* cases where the sample complexity is high) and this shows that a close approximation will still be as good.

Part II

Multi-armed Bandits

Chapter 7: Bandits with Knapsacks

7.1 Introduction

We now focus on another model of sequential decision making paradigm, called *multi-armed bandits*. Multi-armed bandits (*MAB*) is an elegant model for studying the tradeoff between acquisition and usage of information, a.k.a. *explore-exploit tradeoff* (Robbins [159], Thompson [175]). In each round an algorithm sequentially chooses from a fixed set of alternatives (sometimes known as *actions* or *arms*), and receives reward for the chosen action. Crucially, the algorithm does not have enough information to answer all "counterfactual" questions about what would have happened if a different action was chosen in this round. Studied over many decades, multi-armed bandits is a very active research area spanning computer science, operations research, and economics (Bergemann and Välimäki [37], Bubeck and Cesa-Bianchi [43], Cesa-Bianchi and Lugosi [54], Gittins et al. [94]).

In this work, we focus on bandit problems which feature supply or budget constraints, as is the case in many realistic applications. For example, a seller who experiments with prices may have a limited inventory, and a website optimizing ad placement may be constrained by the advertisers' budgets. This general problem is called *Bandits with Knapsacks* (BwK) since, in this model, a bandit algorithm

needs effectively to solve a *knapsack problem* (find an optimal packing of items into a limited-size knapsack) or generalization thereof. The BwK model was introduced in Badanidiyuru et al. [33] as a common generalization of numerous motivating examples, ranging from dynamic pricing to ad allocation to repeated auctions to network routing/scheduling. Various special cases with budget/supply constraints were studied previously (*e.g.,* Babaioff et al. [28], Badanidiyuru et al. [30], Besbes and Zeevi [38], Combes et al. [67], Singla and Krause [169]).

In BwK, the algorithm is endowed with $d \geq 1$ limited resources that are consumed by the algorithm. In each round, the algorithm chooses an action (*arm*) from a fixed set of K actions, and the outcome consists of a reward and consumption of each resource; all are assumed to lie in $[0, 1]$. The algorithm observes *bandit feedback*, *i.e.,* only the outcome of the chosen arm. The algorithm stops at time horizon T, or when the total consumption of some resource exceeds its budget. The goal is to maximize the total reward, denoted REW.

7.1.1 Formal Model

We now describe the formal model. Figure 7.1 describes the protocol. There are T rounds, K possible actions and d resources, indexed as $[T], [K], [d]$, respectively. In each round $t \in [T]$, the algorithm chooses an action $a_t \in [K]$ and receives an *outcome vector* $\mathbf{o}_t = (r_t; c_{t,1}, \ldots, c_{t,d}) \in [0, 1]^{d+1}$, where r_t is a reward and $c_{t,i}$ is consumption of each resource $i \in [d]$. Each resource i is endowed with budget $B_i \leq T$. The game stops early, at some round $\tau_{\text{alg}} < T$, when/if the total consumption of any resource exceeds its budget. The algorithm's objective is to

maximize its total reward. Without loss of generality all budgets are the same: $B_1 = B_2 = \ldots = B_d = B$.[1]

The outcome vectors are chosen as follows. In each round t, the adversary chooses the *outcome matrix* $\mathbf{M}_t \in [0,1]^{K\times(d+1)}$, where rows correspond to actions. The outcome vector $\mathbf{o}_t$ is defined as the a_t-th row of this matrix, denoted $\mathbf{M}_t(a_t)$. Only this row is revealed to the algorithm. The adversary is deterministic and *oblivious*, meaning that the entire sequence $\mathbf{M}_1, \ldots, \mathbf{M}_T$ is chosen before round 1. A problem instance of BwK consists of (known) parameters (d, K, T, B), and the (unknown) sequence $\mathbf{M}_1, \ldots, \mathbf{M}_T$.

In the stochastic version of BwK, henceforth termed *Stochastic BwK*, each outcome matrix $\mathbf{M}_t$ is chosen from some fixed but unknown distribution $\mathcal{D}_{\texttt{BwK}}$ over the outcome matrices. An instance of this problem consists of (known) parameters (d, K, T, B), and the (unknown) distribution $\mathcal{D}_{\texttt{BwK}}$.

In the adversarial version of BwK, henceforth termed *Adversarial BwK*, the outcome matrix $\mathbf{M}_t$ is chosen by an adversary. We use two versions of the adversary, namely *oblivious* adversary who chooses all the outcome matrices before the start of the game and *adaptive* adversary who chooses the outcome matrix at any time $t \in [T]$ after observing the actions $a_1, a_2, \ldots, a_{t-1}$ of the algorithm.

Following prior work Agrawal and Devanur [11], Badanidiyuru et al. [33], we assume, w.l.o.g., that one of the resources is a *dummy resource* similar to time;

[1] To see that this is indeed w.l.o.g., for each resource i, divide all per-round consumptions $c_{t,i}$ by B_i/B, where $B := \min_{i\in[d]} B_i$ is the smallest budget. In the modified problem instance, all consumptions still lie in $[0,1]$, and all the budgets are equal to B.

formally, each action consumes B/T units of this resource per round (we only need this for Stochastic BwK). Further, we posit that one of the actions is a *null action*, which lets the algorithm skips a round: it has 0 reward and consumes 0 amount of each resource other than the dummy resource.

Given: number of time-steps T, resources d, arms K and budget B.

In each round $t \in [T]$,

1. ALG selects an arm $a_t \in [K]$.
2. Nature reveals reward $r_t(a_t)$ and consumptions $\{c_{i,t}(a_t)\}_{i\in[d]}$.
3. ALG gets reward $r_t(a_t)$ and consumes $c_{i,t}(a_t)$ amount of each resource $i \in [d]$
4. Stop if some resource consumed more than B amount.

Figure 7.1: Bandits with Knapsacks Protocol

7.1.2 Benchmarks

Let $\text{REW(ALG)} = \sum_{t\in[\tau_{\text{alg}}]} r_t$ be the total reward of algorithm ALG in the BwK problem. Our benchmark is the *best fixed distribution* (Definition 2), a distribution over actions which maximizes $\mathbb{E}[\text{REW}(\cdot)]$ for a particular problem instance. The expected total reward of this distribution is denoted $\text{OPT}_{\texttt{FD}}$.

For Stochastic BwK, one can compete with the *best dynamic policy*: an algorithm that maximizes $\mathbb{E}[\text{REW}(\cdot)]$ for a particular problem instance. Essentially, this

algorithm knows the latent distribution $\mathcal{D}_{\texttt{BwK}}$ over outcome matrices. Its expected total reward is denoted $\text{OPT}_{\texttt{DP}}$.

We argue that the best fixed distribution over arms is an appropriate benchmark for Adversarial BwK. First, consider the total expected reward of the *best dynamic policy*, denote it $\text{OPT}_{\texttt{DP}}$. (The best dynamic policy is the best algorithm, in hindsight, that is allowed to switch arms arbitrarily across time-steps.) This is the strongest possible benchmark, but it is *too* strong for Adversarial BwK. Indeed, we show a simple example with just one resource (with budget B), where competitive ratio against this benchmark is at least $\frac{T}{B^2}$ for any algorithm. Second, consider the total expected reward of the *best fixed arm*, denote it $\text{OPT}_{\texttt{FA}}$. It is a traditional benchmark in multi-armed bandits, but is uninteresting for Adversarial BwK. We show that the competitive ratio is at least $\Omega(K)$ in the worst case, and this is matched, in expectation, by a trivial algorithm that samples one arm at random and sticks with it forever.

For Stochastic BwK, these three benchmarks are related as follows. The best fixed distribution is still the main object of interest, as far as the design and analysis of algorithms is concerned. However, all results – both ours and prior work – are almost automatically extended to compete against the best dynamic policy. The best fixed arm is a much weaker benchmark than the best fixed distribution: there are simple examples when their expected reward differs by a factor of two, in multiple special cases of interest (Badanidiyuru et al. [33]).

7.2 Related Work

The literature on regret-minimizing online learning algorithms is vast; see Bubeck and Cesa-Bianchi [43], Cesa-Bianchi and Lugosi [54], Hazan [104] for background. Most relevant are two algorithms for adversarial rewards/costs: Hedge for full feedback (Freund and Schapire [89]), and EXP3 for bandit feedback (Auer et al. [25]); both are based on the weighted majority algorithm from Littlestone and Warmuth [130].

Stochastic BwK was introduced and optimally solved in Badanidiyuru et al. [33]. Subsequent work extended these results to soft supply/budget constraints (Agrawal and Devanur [11]), a more general notion of rewards[2] (Agrawal and Devanur [11]), combinatorial semi-bandits (Sankararaman and Slivkins [163]), and contextual bandits (Agrawal and Devanur [12], Agrawal et al. [14], Badanidiyuru et al. [31]). Several special cases with budget/supply constraints were studied previously: dynamic pricing (Babaioff et al. [28], Besbes and Zeevi [38, 39], Wang et al. [184]), dynamic procurement (Badanidiyuru et al. [30], Singla and Krause [169]) (a version of dynamic pricing where the algorithm is a buyer rather than a seller), dynamic ad allocation (Combes et al. [67], Slivkins [170]), and a version with a single resource and unlimited time (Ding et al. [79], György et al. [100], Tran-Thanh et al. [177, 178]). While all this work is on regret minimization, Guha and Munagala [96], Gupta et al. [97] studied closely related Bayesian formulations.

[2]The total reward is determined by the time-averaged outcome vector, but can be an arbitrary Lischitz-concave function thereof.

Stochastic BwK was optimally solved using three different algorithms (Agrawal and Devanur [11], Badanidiyuru et al. [33]), with extremely technical and delicate analyses. All three algorithms involve inherently 'stochastic' techniques such as "successive elimination" and "optimism under uncertainty", and do not appear to extend to the adversarial version. One of them, `PrimalDualBwK` from Badanidiyuru et al. [33], is a primal-dual algorithm superficially similar to ours. Indeed, it decouples into two online learning algorithms: a "primal" algorithm which chooses among arms, and a "dual" algorithm similar to ours, which chooses among resources. However, the two algorithms are not playing a repeated game in any meaningful sense, let alone a zero-sum game. The primal algorithm operates under a much richer input: it takes the entire outcome vector for the chosen arm, as well as the "dual distribution" – the distribution over resources chosen by the dual algorithm. Further, the primal algorithm is very problem-specific: it interprets the dual distribution as a vector of costs over resources, and chooses arms with largest reward-to-cost ratios, estimated using "optimism under uncertainty".

Our approach to using regret minimization in games can be traced to Freund and Schapire [87, 89] (see Ch. 6 in Schapire and Freund [167]), who showed how a repeated zero-sum game played by two agents yields an approximate Nash equilibrium.

This approach has been used as a unifying algorithmic framework for several problems: boosting (Freund and Schapire [87]), linear programs (Arora et al. [21]), maximum flow (Christiano et al. [66]), and convex optimization (Abernethy and Wang [1], Wang and Abernethy [182]). While we use a result with the $1/\sqrt{t}$

convergence rate for the equilibrium property, recent literature obtains faster convergence for cumulative payoffs (but not for the equilibrium property) under various assumptions (e.g., Rakhlin and Sridharan [155], Syrgkanis et al. [172], Wei and Luo [185]).

Repeated Lagrangian games, in conjunction with regret minimization in games, have been used in a series of recent papers (Agarwal et al. [7], Hsu et al. [106], Kearns et al. [118], Rogers et al. [160], Roth et al. [161, 162]), as an algorithmic tool to solve convex optimization problems; application domains range from differential privacy to algorithmic fairness to learning from revealed preferences. All these papers deal with deterministic games (*i.e.,* same game matrix in all rounds). Reframing the problem in terms of repeated Lagrangian games is a key technical insight in this work.

Most related to our paper are Roth et al. [161, 162], where a repeated Lagrangian game is used as a subroutine (the "inner loop") in an online algorithm; the other papers solve an offline problem. We depart from this prior work in several respects: we use a stochastic game, we deal with some subtleties specific to Stochastic BwK, and we provide a very different analysis for our main results on Adversarial BwK, where we cannot rely on the standard machinery.

Online packing problems (*e.g.,* Buchbinder and Naor [47], Devanur et al. [72], see Buchbinder and Naor [48] for a survey) can be seen as a special case of Adversarial BwK with a much more permissive feedback model: the algorithm observes full feedback (the outcomes for all actions) before choosing an action. Online packing subsumes various *online matching* problems, including the *AdWords problem* (Mehta

et al. [141]) motivated by ad allocation (see Mehta [139] for a survey). While we derive $O(\log T)$ competitive ratio against $\mathrm{OPT}_{\mathtt{FD}}$, online packing admits a similar result against $\mathrm{OPT}_{\mathtt{DP}}$.

Another related line of work concerns online convex optimization with constraints Chen and Giannakis [63], Chen et al. [64], Mahdavi et al. [134, 135], Neely and Yu [149]. Their setting differs from ours in several important respects. First, the action set is a convex subset of $\mathbb{R}^K$ (and the algorithms rely on the power to choose arbitrary actions in this set). In particular, there is no immediate way to handle discrete action sets.[3] Second, convexity/concavity is assumed on the rewards and resource consumption. Third, in addition to bandit feedback, full feedback is observed for the resource consumption, and (in all papers except Chen and Giannakis [63]) one also observes either full feedback on rewards or the rewards gradient around the chosen action. Fourth, their algorithm only needs to satisfy the budget constraints at the time horizon (whereas in BwK the budget constraints hold for all rounds). Fifth, their fixed-distribution benchmark is weaker than ours: essentially, its time-averaged consumption must be small enough at each round t. Due to these differences, their setting admits sublinear regret in the adversarial setting, whereas we have a $\Omega(\log T)$ lower bound on the competitive ratio.

Logarithmic competitive ratios are quite common in prior work on approximation algorithms and online algorithms, *e.g.,* in the context of the set cover problem Johnson [112], Lovász [131], buy-at-bulk network design Awerbuch and Azar [26],

[3] Unless there is full feedback, in which case one can use a standard reduction whereby actions in online convex optimization correspond to distributions over actions in a K-armed bandit problem.

sparsest cut Arora et al. [20], and dial-a-ride problem Charikar and Raghavachari [58], the online k-server problem Bansal et al. [34], online packing/covering problems Azar et al. [27], online set cover Alon et al. [18], online network design Umboh [181], and online paging Fiat et al. [85].

Simultaneous work. Two very recent (and yet unpublished) manuscripts have come to our attention after the initial version of this paper has appeared on `arxiv.org`. Rivera et al. Rivera et al. [158] consider online convex optimization with knapsacks with full feedback. Focusing on the stochastic version, they design an algorithm similar to `LagrangeBwK`, and derive a regret bound similar to ours, using a similar analysis. They also claim an extension to bandit feedback, without providing any details (such as precise statement of Lemma 11 in terms of the regret property (7.4)).

Rangi et al. Rangi et al. [157] consider Adversarial BwK in the special case when there is only one constrained resource, including time. They attain sublinear regret, *i.e.,* a regret bound that is sublinear in T. They also assume a known lower bound $c_{\min} > 0$ on realized per-round consumption of each resource, and their regret bound scales as $1/c_{\min}$. They also achieve polylog(T) instance-dependent regret for the stochastic version using the same algorithm (matching results from prior work on the stochastic version). BwK with only one constrained resource (including time) is a much easier problem, compared to the general case with multiple resources studied in this paper, in the following sense. First, the single-resource version admits much stronger performance guarantees (polylog(T) vs. $\sqrt{T}$ regret bounds for Stochastic

BwK, and sublinear regret vs. approximation ratio for Adversarial BwK). Second, the optimal all-knowing time-invariant policy is the best fixed arm, rather than the best fixed distribution over arms.

7.3 Challenges and Our Contributions

7.3.1 Challenges

Adversarial BwK is a much harder problem compared to Stochastic BwK. The new challenge is that the algorithm needs to decide how much budget to save for the future, without being able to predict it. (It is also the essential challenge in online packing problems, and it drives our lower bounds.) This challenge compounds the ones already present in Stochastic BwK: that exploitation may be severely limited by the resource consumption during exploration, that optimal per-round reward no longer guarantees optimal total reward, and that the best fixed distribution over arms may perform much better than the best fixed arm. Jointly, these challenges amount to the following. An algorithm for Adversarial BwK must compete, during any given time segment $[1, \tau]$, with a distribution over arms that maximizes the total reward on this time segment. However, this distribution may behave very differently, in terms of expected per-round outcomes, compared to the optimal distribution for some other time segment $[1, \tau']$.

In more concrete terms, let $\mathrm{OPT}_{\texttt{FD}}$ be the total expected reward of the *best fixed distribution* over arms. In Stochastic BwK (as well as in adversarial bandits)

an algorithm can achieve sublinear regret: $\text{OPT}_{\texttt{FD}} - \mathbb{E}[\text{REW}] = o(T)$.[4] In contrast, in Adversarial BwK regret minimization is no longer possible, and we therefore are primarily interested in the *competitive ratio* $\text{OPT}_{\texttt{FD}} / \mathbb{E}[\text{REW}]$.

It is instructive to consider a simple example in which the competitive ratio is at least $\frac{5}{4} - o(1)$ for any algorithm. There are two arms and one resource with budget $\frac{T}{2}$. Arm 1 has zero rewards and zero consumption. Arm 2 has consumption 1 in each round, and offers reward $\frac{1}{2}$ in each round of the first half-time ($\frac{T}{2}$ rounds). In the second half-time, it offers either reward 1 in all rounds, or reward 0 in all rounds. Thus, there are two problem instances that coincide for the first half-time and differ in the second half-time. The algorithm needs to choose how much budget to invest in the first half-time, without knowing what comes in the second. Any choice leads to competitive ratio at least $\frac{5}{4}$ on one of the problem instances.

Extending this idea, we prove an even stronger lower bound on the competitive ratio:

$$\text{OPT}_{\texttt{FD}} / \mathbb{E}[\text{REW}] \geq \Omega(\log T). \tag{7.1}$$

Like the simple example above, the lower-bounding construction involves only two arms and only one resource, and forces the algorithm to make a huge commitment without knowing the future.

[4] More specifically, one can achieve regret $\tilde{O}(\sqrt{KT})$ for adversarial bandits (Auer et al. [25]), as well as for Stochastic BwK if all budgets are $\Omega(T)$ Badanidiyuru et al. [33]. One can achieve sublinear regret for Stochastic BwK if all budgets are $\Omega(T^\alpha)$, $\alpha \in (0, 1)$ (Badanidiyuru et al. [33]).

7.3.2 Our Contributions

Our main result is an algorithm which nearly matches (7.1), achieving

$$\mathbb{E}[\text{REW}] \geq \frac{1}{O(\log T)} \left(\text{OPT}_{\texttt{FD}} - o(\text{OPT}_{\texttt{FD}})\right). \tag{7.2}$$

We put forward a new algorithm for BwK, called `LagrangeBwK`, that unifies the stochastic and adversarial versions. It has a natural game-theoretic interpretation for Stochastic BwK, and admits a simpler analysis compared to the prior work. For Adversarial BwK, we use `LagrangeBwK` as a subroutine, though with a different parameter and a different analysis, to derive two algorithms: a simple one that achieves (7.2), and a more involved one that achieves the same competitive ratio with high probability. Absent resource consumption, we recover the optimal $\tilde{O}(\sqrt{KT})$ regret for adversarial bandits.

`LagrangeBwK` is based on a new perspective on Stochastic BwK. We reframe a standard linear relaxation for Stochastic BwK in a way that gives rise to a repeated zero-sum game, where the two players choose among arms and resources, respectively, and the payoffs are given by the Lagrange function of the linear relaxation. Our algorithm consists of two online learning algorithms playing this repeated game. We analyze `LagrangeBwK` for Stochastic BwK, building on the tools from regret minimization in stochastic games, and achieve a near-optimal regret bound when the optimal value and the budgets are $\Omega(T)$.[5]

Discussion. `LagrangeBwK` has numerous favorable properties. As just discussed, it

[5]This regime is of primary importance in prior work, *e.g.,* Besbes and Zeevi [38], Wang et al. [184].

is simple, unifying, modular, and yields strong performance guarantees in multiple settings. It is the first "black-box reduction" from bandits to BwK: we take a bandit algorithm and use it as a subroutine for BwK. This is a very natural algorithm for the stochastic version once the single-shot game is set up; indeed, it is immediate from prior work that the repeated game converges to the optimal distribution over arms. Its regret analysis for Stochastic BwK is extremely clean. Compared to prior work (Agrawal and Devanur [11], Badanidiyuru et al. [33]), we side-step the intricate analysis of sensitivity of the linear program to non-uniform stochastic deviations that arise from adaptive exploration.

`LagrangeBwK` has a primal-dual interpretation, as arms and resources correspond respectively to primal and dual variables in the linear relaxation. Two players in the repeated game can be seen as the respective *primal algorithm* and *dual algorithm*. Compared to the rich literature [6] on (*e.g., primal-dual algorithms* Buchbinder and Naor [48], Mehta [139], Williamson and Shmoys [188]) `LagrangeBwK` has a very specific and modular structure dictated by the repeated game.

7.4 Preliminaries

In this section, we state the required preliminaries from prior work on adversarial bandits and zero-sum games.

[6] This also includes the more recent literature on stochastic online packing problems (*e.g.,* Agrawal et al. [13], Devanur and Hayes [70], Devanur et al. [72], Feldman et al. [84], Molinaro and Ravi [147]).

7.4.1 Adversarial Online Learning

Given: action set A, payoff range $[b_{\min}, b_{\max}]$.

In each round $t \in [T]$,

1. the adversary chooses a payoff vector $\mathbf{f}_t \in [b_{\min}, b_{\max}]^K$;
2. the algorithm chooses a distribution $\mathbf{p}_t$ over A, without observing $\mathbf{f}_t$,
3. algorithm's chosen action $a_t \in A$ is drawn independently from $\mathbf{p}_t$;
4. payoff $f_t(a_t)$ is received by the algorithm.

Figure 7.2: Adversarial online learning

In this protocol (see Figure 7.2), the adversary can use previously chosen arms to choose the payoff vector $\mathbf{f}_t$, but not the algorithm's random seed. The distribution $\mathbf{f}_t$ is chosen as a deterministic function of history. (The history at round t consists, for each round $s < t$, of the chosen action a_s and the observed feedback in this round.) We focus on two feedback models: *bandit feedback* (no auxiliary feedback) and *full feedback* (the entire payoff vector $\mathbf{f}_t$). The version for costs can be defined similarly, by setting the payoffs to be the negative of costs.

We are interested in adversarial online learning algorithms with known upper bounds on *regret*,

$$R_{\texttt{AOL}}(T) := \left[\max_{a\in A} \sum_{t\in[T]} f_t(a)\right] \; - \; \left[\sum_{t\in[T]} f_t(a_t)\right]. \tag{7.3}$$

The benchmark here is the total payoff of the best arm, according to the payoff

vectors actually chosen by the adversary. More precisely, we assume high-probability regret bounds of the following form:

$$\forall \delta > 0 \qquad \Pr\left[\, R_{\texttt{AOL}}(T) \leq (b_{\max} - b_{\min})\, R_\delta(T) \,\right] \geq 1 - \delta, \tag{7.4}$$

for some function $R_\delta(\cdot)$. We will actually use a stronger version implied by (7.4),[7]

$$\forall \delta > 0 \qquad \Pr\left[\, \forall \tau \in [T] \quad R_{\texttt{AOL}}(\tau) \leq (b_{\max} - b_{\min})\, R_{\delta/T}(T) \,\right] \geq 1 - \delta. \tag{7.5}$$

Algorithms EXP3.P (Auer et al. [25]) for bandit feedback, and Hedge (Freund and Schapire [88]) for full feedback, satisfy (7.4) with, resp.,

$$R_\delta(T) = O\left(\sqrt{|A|\, T \log(T/\delta)}\right) \quad \text{and} \quad R_\delta(T) = O\left(\sqrt{T \log(|A|/\delta)}\right). \tag{7.6}$$

7.4.2 Regret minimization in games

We build on the framework of *regret minimization in games*. A *zero-sum game* $(A_1, A_2, \mathbf{G})$ is a game between two players $i \in \{1, 2\}$ with action sets A_1 and A_2 and payoff matrix $\mathbf{G} \in \mathbb{R}^{A_1 \times A_2}$. If each player i chooses an action $a_i \in A_i$, the outcome is a number $G(a_1, a_2)$. Player 1 receives this number as *reward*, and player 2 receives it as *cost*. A *repeated zero-sum game* $\mathcal{G}$ with action sets A_1 and A_2, time horizon T and game matrices $\mathbf{G}_1, \ldots, \mathbf{G}_T \in \mathbb{R}^{A_1 \times A_2}$ is a game between two algorithms, ALG_1 and ALG_2, which proceeds over T rounds such that each round t is a zero-sum game $(A_1, A_2, \mathbf{G}_t)$. The goal of ALG_1 is to maximize the total reward, and the goal of ALG_2 is to minimize the total cost.

[7] Regret bound (7.5) follows from (7.4) using a simple "zeroing-out" trick: for a given round $\tau \in [T]$, the adversary can set all future payoffs to some fixed value $x \in [b_{\min}, b_{\max}]$, in which case $R_{\texttt{AOL}}(\tau) = R_{\texttt{AOL}}(T)$.

The game $\mathcal{G}$ is called *stochastic* if the game matrix $\mathbf{G}_t$ in each round t is drawn independently from some fixed distribution. For such games, we are interested in the *expected game*, defined by the expected game matrix $\mathbf{G} = \mathbb{E}[\mathbf{G}_t]$. We can relate the algorithms' performance to the minimax value of $\mathbf{G}$.

Lemma 11. *Consider a stochastic repeated zero-sum game between algorithms* ALG_1 *and* ALG_2*, with payoff range* $[b_{\min}, b_{\max}]$*. Assume that each* ALG_j*,* $j \in \{1, 2\}$ *is an algorithm for adversarial online learning, as per Figure 7.2, which satisfies regret bound (7.4) with* $R_\delta(T) = R_{j,\delta}(T)$*.*

Let τ *be some fixed round in the game. For each algorithm* ALG_j*,* $j \in \{1, 2\}$*, let* A_j *be its action set, let* $p_{t,j} \in \Delta_{A_j}$ *be the distribution chosen in each round* t*, and let* $\bar{\mathbf{p}}_j = \frac{1}{\tau} \sum_{t \in [\tau]} \mathbf{p}_{t,j}$ *be the average play distribution at round* τ*. Let* v^* *be the minimax value for the expected game* $\mathbf{G} = \mathbb{E}[\mathbf{G}_t]$*.*

Then for each $\delta > 0$*, with probability at least* $1 - 2\delta$ *it holds that*

$$\forall \mathbf{p}_2 \in \Delta_{A_2} \quad \bar{\mathbf{p}}_1^{\mathrm{T}} \, \mathbf{G} \, \mathbf{p}_2 \geq v^* - \tfrac{1}{\tau}(b_{\max} - b_{\min}) \left(R_{1,\delta/T}(T) + R_{2,\delta/T}(T) + 4\sqrt{2T \log(T/\delta)} \right). \tag{7.7}$$

Eq. (7.7) states that the average play of player 1 is approximately optimal against any distribution chosen by player 2.[8] This lemma is well-known for the deterministic case (*i.e.*, when $\mathbf{G}_t = \mathbf{G}$ for each round t), and folklore for the stochastic case. We provide a proof in Appendix 10.3.2 for the sake of completeness.

[8] If each player j chooses distribution $p_j \in \Delta_{A_j}$, and the game matrix is $\mathbf{G}$, then expected reward/cost is $\mathbf{p}_1^{\mathrm{T}} \mathbf{G} \mathbf{p}_2$.

7.5 A new algorithm for Stochastic BwK

We present a new algorithm for Stochastic BwK, based on the framework of regret minimization in games. This is a very natural algorithm once the single-shot game is set up, and it allows for a very clean regret analysis. We will also use this algorithm as a subroutine for the adversarial version.

On a high level, we define a stochastic zero-sum game for which a mixed Nash equilibrium corresponds to an optimal solution for a linear relaxation of the original problem. Our algorithm consists of two regret-minimizing algorithms playing this game. The framework of regret minimization in games guarantees that the average primal and dual play distributions ($\bar{\mathbf{p}}_1$ and $\bar{\mathbf{p}}_2$ in Lemma 11) approximate the mixed Nash equilibrium in the expected game, which correspondingly approximates the optimal solution.

7.5.1 Linear relaxation and Lagrange functions

We start with a linear relaxation of the problem that all prior work relies on. This relaxation is stated in terms of expected rewards/consumptions, *i.e.,* implicitly, in terms of the expected outcome matrix $\mathbf{M} = \mathbb{E}[\mathbf{M}_t]$. We explicitly formulate the relaxation in terms of $\mathbf{M}$, and this is essential for the subsequent developments. For ease of notation, we write the a-th row of $\mathbf{M}$, for each action $a \in [K]$, as

$$\mathbf{M}(a) = (r^{\mathbf{M}}(a); c_1^{\mathbf{M}}(a), \ \ldots \ , c_d^{\mathbf{M}}(a)),$$

so that $r^{\mathbf{M}}(a)$ is the expected reward and $c_i^{\mathbf{M}}(a)$ is the expected consumption of each resource i.

Essentially, the relaxation assumes that each instantaneous outcome matrix $\mathbf{M}_t$ is equal to the expected outcome matrix $\mathbf{M} = \mathbb{E}[\mathbf{M}_t]$. The relaxation seeks the best distribution over actions, focusing on a single round with budgets rescaled as B/T. This leads to the following linear program (LP):

$$\begin{array}{lll} \text{maximize} & \sum_{a\in[K]} X(a)\, r^{\mathbf{M}}(a) & \text{such that} \\ & \sum_{a\in[K]} X(a) = 1 & \\ \forall i \in [d] & \sum_{a\in[K]} X(a)\, c_i^{\mathbf{M}}(a) \le B/T & \\ \forall a \in [K] & 0 \le X(a) \le 1. & \end{array} \tag{7.8}$$

We denote this LP by $\mathrm{LP}_{\mathbf{M},B,T}$. The solution $\mathbf{X}$ is the best fixed distribution over actions, according to the relaxation. The value of this LP, denoted $\mathrm{OPT}_{\texttt{LP}}(\mathbf{M}, B, T)$, is the expected per-round reward of this distribution. It is also the total reward of $\mathbf{X}$ in the relaxation, divided by T. We know from Badanidiyuru et al. [33] that

$$T \cdot \mathrm{OPT}_{\texttt{LP}}(\mathbf{M}, B, T) \ge \mathrm{OPT}_{\texttt{DP}} \ge \mathrm{OPT}_{\texttt{FD}}, \tag{7.9}$$

where $\mathrm{OPT}_{\texttt{DP}}$ and $\mathrm{OPT}_{\texttt{FD}}$ are the total expected rewards of, respectively, the best dynamic policy and the best fixed distribution. In words, $\mathrm{OPT}_{\texttt{DP}}$ is sandwiched between the total expected reward of the best fixed distribution and that of its linear relaxation.

Associated with the linear program $\mathrm{LP}_{\mathbf{M},B,T}$ is the *Lagrange function* $\mathcal{L} =$

$\mathcal{L}_{\mathbf{M},B,T}$. It is a function $\mathcal{L} : \Delta_K \times \mathbb{R}^d_{\geq 0} \to \mathbb{R}$ defined as

$$\mathcal{L}(\mathbf{X}, \lambda) := \sum_{a \in [K]} X(a)\, r^{\mathbf{M}}(a) + \sum_{i \in [d]} \lambda_i \left[1 - \frac{T}{B} \sum_{a \in [K]} X(a)\, c_i^{\mathbf{M}}(a) \right]. \tag{7.10}$$

The values $\lambda_1, \ldots, \lambda_d$ in Eq. (7.10) are called the *dual variables*, as they correspond to the variables in the dual LP. Lagrange functions are meaningful due to their max-min property (*e.g.,* Theorem D.2.2 in Ben-Tal and Nemirovski [36]):

$$\min_{\lambda \geq 0} \max_{\mathbf{X} \in \Delta_K} \mathcal{L}(\mathbf{X}, \lambda) = \max_{\mathbf{X} \in \Delta_K} \min_{\lambda \geq 0} \mathcal{L}(\mathbf{X}, \lambda) = \mathrm{OPT}_{\mathrm{LP}}(\mathbf{M}, B, T). \tag{7.11}$$

This property holds for our setting because $\mathrm{LP}_{\mathbf{M},B,T}$ has at least one feasible solution (namely, one that puts probability one on the null action), and the optimal value of the LP is bounded.

Remark 2. *We use the linear program* $\mathrm{LP}_{\mathbf{M},B,T}$ *and the associated Lagrange function* $\mathcal{L}_{\mathbf{M},B,T}$ *throughout this chapter. Both are parameterized by an outcome matrix* $\mathbf{M}$*, budget* B *and time horizon* T*. In particular, we can plug in an arbitrary* $\mathbf{M}$*, and we heavily use this ability throughout. For the adversarial version, it is essential to plug in parameter* $T_0 \leq T$ *instead of the time horizon* T*. For the analysis of the high-probability result in Adversarial BwK, we use a rescaled budget* $B_0 \leq B$ *instead of budget* B*.*

7.5.2 Our algorithm: repeated Lagrangian game

The Lagrange function $\mathcal{L} = \mathcal{L}_{\mathbf{M},B,T}$ from (7.10) defines the following zero-sum game: the *primal player* chooses an arm a, the *dual player* chooses a resource i, and

the payoff is a number

$$\mathcal{L}(a,i) = r^{\mathbf{M}}(a) + 1 - \tfrac{T}{B}\, c_i^{\mathbf{M}}(a). \tag{7.12}$$

The primal player receives this number as a reward, and the dual player receives it as cost. This game is termed the *Lagrangian game* induced by $\mathcal{L}_{\mathbf{M},B,T}$. This game will be crucial throughout this chapter.

The Lagrangian game is related to the original linear program as follows:

Lemma 12. *Assume one of the resources is the dummy resource. Consider the linear program* $\text{LP}_{\mathbf{M},B,T}$*, for some outcome matrix* $\mathbf{M}$*. Then the value of this LP equals the minimax value* v^* *of the Lagrangian game induced by* $\mathcal{L}_{\mathbf{M},B,T}$*. Further, if* $(\mathbf{X}, \lambda)$ *is a mixed Nash equilibrium in the Lagrangian game, then* $\mathbf{X}$ *is an optimal solution to the LP.*

The proof can be found in Appendix 10.3. The idea is that because of the special structure of the LP, the second equality in (7.11) also holds when the dual vector λ is restricted to distributions.

Consider a repeated version of the Lagrangian game. Formally, the *repeated Lagrangian game* with parameters $B_0 \leq B$ and $T_0 \leq T$ is a repeated zero-sum game between the *primal algorithm* that chooses among arms and the *dual algorithm* that chooses among resources. Each round t of this game is the Lagrangian game induced by the Lagrange function $\mathcal{L}_t := \mathcal{L}_{\mathbf{M}_t,B_0,T_0}$, where $\mathbf{M}_t$ is the round-t outcome matrix. Note that we use parameters B_0, T_0 instead of budget B and time horizon T.[9]

[9]These parameters are needed only for the adversarial version. For Stochastic BwK we use $B_0 = B$ and $T_0 = T$.

Remark 3. *Consider repeated Lagrangian game for Stochastic BwK (with $B_0 = B$ and $T_0 = T$). The payoffs in the expected game are defined by the expected Lagrange function $\mathcal{L} := \mathbb{E}[\mathcal{L}_t]$. By linearity, $\mathcal{L}$ is the Lagrange function for the expected outcome matrix $\mathbf{M} = \mathbb{E}[\mathbf{M}_t]$:*

$$\mathcal{L} := \mathbb{E}[\mathcal{L}_t] = \mathcal{L}_{\mathbf{M},B,T}. \tag{7.13}$$

Our algorithm, called `LagrangeBwK`, is very simple: it is a repeated Lagrangian game in which the primal algorithm receives bandit feedback, and the dual algorithm receives full feedback.

To set up the notation, let a_t and i_t be, respectively, the chosen arm and resource in round t. The payoff is therefore $\mathcal{L}_t(a_t, i_t)$. It can be rewritten in terms of the observed outcome vector $\mathbf{o}_t = (r_t; c_{t,1}, \ldots, c_{t,d})$ (which corresponds to the a_t-th row of the instantaneous outcome matrix $\mathbf{M}_t$):

$$\mathcal{L}_t(a_t, i_t) = r_t + 1 - \tfrac{T_0}{B_0}\, c_{t,i_t} \in [-\tfrac{T_0}{B_0} + 1, 2]. \tag{7.14}$$

Note that the payoff range is $[b_{\min}, b_{\max}] = [-\frac{T_0}{B_0} + 1]$.

With this notation, the pseudocode for `LagrangeBwK` is summarized in Algorithm 5. The pseudocode is simple and self-contained, without referring to the formalism of repeated games and Lagrangian functions. Note that the algorithm is implementable, in the sense that the outcome vector $\mathbf{o}_t$ revealed in each round t of the BwK problem suffices to generate full feedback for the dual algorithm.

Algorithm 5: Algorithm `LagrangeBwK` for Stochastic BwK.

input: parameters B_0, T_0, primal algorithm ALG_1, dual algorithm ALG_2.

// ALG_1, ALG_2 are adversarial online learning algorithms

// with bandit feedback and full feedback, respectively

for *round* $t = 1, 2, 3, \ldots$ **do**

1. ALG_1 returns arm $a_t \in [K]$, algorithm ALG_2 returns resource $i_t \in [d]$.
2. arm a_t is chosen, outcome vector $\mathbf{o}_t = (r_t(a_t); c_{t,1}(a_t), \ldots, c_{t,d}(a_t)) \in [0,1]^{d+1}$ is observed.
3. The payoff $\mathcal{L}_t(a_t, i_t)$ from (7.14) is reported to ALG_1 as reward, and to ALG_2 as cost.
4. The payoff $\mathcal{L}_t(a_t, i)$ is reported to ALG_2 for each resource $i \in [d]$.

7.5.3 Performance guarantees

We consider algorithm `LagrangeBwK` with parameter $T_0 = T$. We assume the existence of the dummy resource; this is to ensure that the crucial step, Eq. (7.20), works out even if the algorithm stops at time T, without exhausting any actual resources. We obtain a regret bound that is non-trivial whenever $B > \Omega(\sqrt{T})$, and is optimal, up to log factors, in the regime when $\min(\text{OPT}_{\texttt{DP}}, B) > \Omega(T)$.

Theorem 6. *Consider Stochastic BwK with K arms, d resources, time horizon T, and budget B. Assume that one resource is the dummy resource (with consumption $\frac{B}{T}$ for each arm). Fix the failure probability parameter $\delta \in (0,1)$. Consider algorithm*

LagrangeBwK with parameters $B_0 = B$, $T_0 = T$.

If EXP3.P and Hedge are used as the primal and the dual algorithms, respectively, then the algorithm achieves the following regret bound, with probability at least $1 - \delta$:

$$\mathrm{OPT}_{DP} - \mathrm{REW}(\texttt{LagrangeBwK}) \leq O\left(\frac{T}{B}\sqrt{TK\log(dT/\delta)}\right). \tag{7.15}$$

In general, suppose each algorithm ALG_j satisfies a regret bound (7.4) with $R_\delta(T) = R_{j,\delta}(T)$ and payoff range $[b_{\min}, b_{\max}] = [-\frac{T}{B} + 1, 2]$. Then with probability at least $1 - O(\delta T)$ it holds that

$$\mathrm{OPT}_{DP} - \mathrm{REW}(\texttt{LagrangeBwK}) \leq O\left(\tfrac{T}{B}\right)\left(R_{1,\delta/T}(T) + R_{2,\delta/T}(T) + \sqrt{T\log(dT/\delta)}\right). \tag{7.16}$$

Remark 4. *To obtain (7.15) from the "black-box" result (7.16), we use regret bounds in Eq. (7.6).*

Remark 5. *From Badanidiyuru et al. [33], the optimal regret bound for Stochastic BwK is*

$$\mathrm{OPT}_{DP} - \mathbb{E}[\mathrm{REW}] \leq \tilde{O}\left(\sqrt{K\,\mathrm{OPT}_{DP}}\,(1 + \sqrt{\mathrm{OPT}_{DP}/B})\right).$$

Thus, the regret bound (7.15) is near-optimal if $\min(\mathrm{OPT}_{DP}, B) > \Omega(T)$, and non-trivial if $B > \Omega(\sqrt{T})$.

We next prove the "black-box" regret bound (7.16). For the sake of analysis, consider a version of the repeated Lagrangian game that continues up to the time horizon T. In what follows, we separate the "easy steps" from what we believe is the crux of the proof.

Notation. Let $\mathbf{X}_t$ be the distribution chosen in round t by the primal algorithm ALG_1. Let $\overline{\mathbf{X}}_\tau := \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{X}_t$ be the distribution of average play up to round τ. Let $\mathbf{M} = \mathbb{E}[\mathbf{M}_t]$ be the expected outcome matrix. Let $\mathbf{r} = (r^{\mathbf{M}}(a) : \ a \in [K])$ be the vector of expected rewards over the actions. Likewise, $\mathbf{c}_i = (c_i^{\mathbf{M}}(a) : \ a \in [K])$ be the vector of expected consumption of each resource $i \in [d]$.

Using Azuma-Hoeffding inequality. Consider the first τ rounds, for some $\tau \in [T]$. The average reward and resource-i consumption over these rounds are close to $\overline{\mathbf{X}}_\tau \cdot \mathbf{r}$ and $\overline{\mathbf{X}}_\tau \cdot \mathbf{c}_i$, respectively, with high probability. Specifically, a simple usage of Azuma-Hoeffding inequality (Lemma 28) implies that

$$\frac{1}{\tau}\sum_{t\in[\tau]} r_t \geq \overline{\mathbf{X}}_\tau \cdot \mathbf{r} - R_0(\tau)/\tau, \tag{7.17}$$

$$\frac{1}{\tau}\sum_{t\in[\tau]} c_{i,t} \leq \overline{\mathbf{X}}_\tau \cdot \mathbf{c}_i + R_0(\tau)/\tau, \qquad \forall i \in [d], \tag{7.18}$$

hold with probability at least $1-\delta$, where $R_0(\tau) = O(\sqrt{\tau \log(d/\delta)})$.

Regret minimization in games. Let us apply the machinery from regret minimization in games to the repeated Lagrangian game. Consider the game matrix $\mathbf{G}$ of the expected game. Using Eq. (7.13) and Lemma 12, we conclude that the minimax value of $\mathbf{G}$ is $v^* = \text{OPT}_{\texttt{LP}}(\mathbf{M}, B, T)$.

We apply Lemma 11, with a fixed stopping time $\tau \in [T]$. Recall that the payoff range is $b_{\max} - b_{\min} = \frac{T}{B} + 1$. Thus, with probability at least $1-2\delta$ it holds that

$$\lambda \in \Delta_d : \quad \overline{\mathbf{X}}_\tau^{\mathrm{T}} \, \mathbf{G} \, \lambda \geq v^* - \tfrac{1}{\tau}(\tfrac{T}{B}+1) \cdot \texttt{reg}(T), \tag{7.19}$$

where the regret term is $\texttt{reg}(T) := R_{1,\delta/T}(T) + R_{2,\delta/T}(T) + 4\sqrt{2T\log(T/\delta)}$.

Crux of the proof. Let us condition on the event that (7.17), (7.18), and (7.19) hold for each $\tau \in [T]$. By the union bound, this event holds with probability at least $1 - 3\delta T$.

Let τ denote the *stopping time* of the algorithm, the first round when the total consumption of some resource exceeds its budget. Let i be the resource for which this happens; hence,

$$\sum_{t\in[\tau]} c_{i,t} > B. \tag{7.20}$$

Let us use Eq. (7.19) with $\lambda = \lambda^{(i)}$, the point distribution for this resource.

From Eq. (7.13), we have

$$\overline{\mathbf{X}}_\tau^{\mathrm{T}}\, \mathbf{G}\, \lambda^{(i)} = \overline{\mathbf{X}}_\tau \cdot \mathbf{r} + 1 - \tfrac{T}{B}\, \overline{\mathbf{X}}_\tau \cdot \mathbf{c}_i.$$

Plugging in (7.17) and (7.18) we get

$$\overline{\mathbf{X}}_\tau \cdot \mathbf{r} + 1 - \tfrac{T}{B}\, \overline{\mathbf{X}}_\tau \cdot \mathbf{c}_i \leq \tfrac{1}{\tau}\, \left(\left(\textstyle\sum_{t\in[\tau]} r_t \right) - \left(\tfrac{T}{B}\, \textstyle\sum_{t\in[\tau]} c_{i,t} \right) + \tau - (1 + \tfrac{T}{B})\, R_0(\tau) \right).$$

Finally, plugging in Eq. (7.20) we get this to be upper-bounded by the following.

$$\leq \tfrac{1}{\tau}\, \left(\left(\textstyle\sum_{t\in[\tau]} r_t \right) + \tau - T - (1 + \tfrac{T}{B})\, R_0(\tau) \right).$$

Plugging this into Eq. (7.19) and rearranging, we obtain

$$\sum_{t\in[\tau]} r_t \geq \tau\, v^* + T - \tau - (1 + \tfrac{T}{B}) \cdot \texttt{reg}(T).$$

Since $v^* \leq 1$ (because $v^* = \mathrm{OPT}_{\texttt{LP}}$, as we've proved above),

$$\mathrm{REW}(\texttt{LagrangeBwK}) = \sum_{t\in[\tau]} r_t \geq T\, v^* - (1 + \tfrac{T}{B}) \cdot \texttt{reg}(T).$$

The claimed regret bound (7.16) follows by Eq. (7.9), completing the proof of Theorem 6.

7.6 A simple algorithm for Adversarial BwK

We present and analyze an algorithm for Adversarial BwK which achieves $O(d^2 \log T)$ competitive ratio, in expectation, up to a low-order additive term. Our algorithm is very simple: we randomly guess the value of $\text{OPT}_{\texttt{FD}}$ and run `LagrangeBwK` with parameter T_0 driven by this guess. The analysis is very different, however, since we cannot rely on the machinery from regret minimization in stochastic games. The crux of the analysis (Lemma 14) is re-used to analyze the high-probability algorithm in the next section.

The intuition for our algorithm can be explained as follows. `LagrangeBwK` builds on adversarial online learning algorithms ALG_j, and appears plausibly applicable to Adversarial BwK. We analyze it for Adversarial BwK, with an arbitrary parameter T_0 (see Lemma 14, the crux of our analysis), and find that it performs best when T_0 is tailored to $\text{OPT}_{\texttt{FD}}$ up to a constant multiplicative factor. This is precisely what our algorithm achieves using the random guess.

Our algorithm is presented as Algorithm 6. We randomly guess the value of $\text{OPT}_{\texttt{FD}}$ from within a specified range $[g_{\min}, g_{\max}]$, up to the specified multiplicative factor of $\kappa > 0$. We consider multiplicative scales $[\kappa^u, \kappa^{u+1}]$, $u \in \mathbb{N}$, and we guess uniformly at random among all possible u. Our analysis works as long as $\text{OPT}_{\texttt{FD}} \in [g_{\min}, g_{\max}]$ and $\kappa \geq d+1$; then we obtain competitive ratio $\kappa^2 \left\lceil \log \frac{g_{\max}}{g_{\min}} \right\rceil$ up to a low-order additive term. As a corollary, we obtain competitive ratio κ^2

ceil $\log T \rceil$ with no assumptions.

Theorem 7. *Consider Adversarial BwK with K arms, d resources, time horizon T,*

Algorithm 6: A simple algorithm for Adversarial BwK.

input: scale parameter $\kappa > 0$, guess range $[g_{\min}, g_{\max}]$, primal and dual algorithms ALG_1, ALG_2

// ALG_1, ALG_2 are adversarial online learning algorithms

// with bandit feedback and full feedback, resp.

Choose u uniformly at random from $\{0, 1, \ldots, u^{\max}\}$, where $u^{\max} = \left\lceil \log_\kappa \frac{g_{\max}}{g_{\min}} \right\rceil$.

Guess the value of $\text{OPT}_{\texttt{FD}}$ as $\widehat{g} = g_{\min} \cdot \kappa^u$.

Run LagrangeBwK with algorithms ALG_1, ALG_2 and parameters $B_0 = B$ and $T_0 = \widehat{g}/\kappa$.

and budget B. Assume that one of the arms is a null arm *that has zero reward and zero resource consumption. Consider Algorithm 6 with scale parameter $\kappa \geq d + 1$. Suppose algorithms ALG_j that satisfy the regret bound Eq. (7.4) with $\delta = T^{-2}$ and regret term $R_\delta(T) = R_{j,\delta}(T)$, for any known payoff range $[b_{\min}, b_{\max}]$.*

If $\text{OPT}_{\texttt{FD}} \in [g_{\min}, g_{\max}]$ then the expected reward of Algorithm 6 satisfies

$$\mathbb{E}[\text{REW}] \geq (\text{OPT}_{\texttt{FD}} - \texttt{reg}) / \left(\kappa^2 \left\lceil \log_\kappa \tfrac{g_{\max}}{g_{\min}} \right\rceil \right), \tag{7.21}$$

where $\texttt{reg} = (1 + \frac{\text{OPT}_{\texttt{FD}}}{\kappa B}) \left(R_{1,\delta/T}(T) + R_{2,\delta/T}(T) \right)$. Taking $[g_{\min}, g_{\max}] = [1, T]$, we obtain

$$\mathbb{E}[\text{REW}] \geq (\text{OPT}_{\texttt{FD}} - \texttt{reg}) / \left(\kappa^2 \lceil \log_\kappa T \rceil \right). \tag{7.22}$$

Remark 6. *One can use algorithms EXP3.P for ALG_1 and Hedge for ALG_2, with regret bounds given by Eq. (7.6), and achieve the regret term*

$$\texttt{reg} = O\left(1 + \tfrac{\text{OPT}_{\texttt{FD}}}{\kappa B}\right) \sqrt{TK \log(Td/\delta)}.$$

We obtain a meaningful performance guarantee as long as, say, $\texttt{reg} < \text{OPT}_{FD}/2$*; this requires* OPT_{FD} *and* B *to be at least* $\widetilde{\Omega}(\sqrt{TK})$*.*

Remark 7. *We define the outcome matrices slightly differently compared to Section 7.5 in that we do not posit a dummy resource. Formally, we assume that the null arm has zero consumption in every resource. This is essential for case 1 (*i.e., *when* $\tau_{alg} \leq \sigma$*) in the analysis of Lemma 14.*

If a problem instance of Adversarial BwK is actually an instance of adversarial bandits, then we recover the optimal $\tilde{O}(\sqrt{KT})$ regret. (This easily follows by examining the proof of Lemma 14.)

Lemma 13. *Consider* `LagrangeBwK`*, with algorithms EXP3.P for* ALG_1 *and Hedge for* ALG_2*, for an instance of Adversarial BwK with zero resource consumption. This algorithm obtains* $\tilde{O}(\sqrt{KT})$ *regret, for any parameters* $B_0, T_0 > 0$*. Accordingly, so does Algorithm 6 with any scale parameter* $\kappa > 0$*.*

7.6.1 Analysis: proof of Theorem 7 and Lemma 13

Stopped linear program. Let us set up a linear relaxation that is suitable to the adversarial setting. The expected outcome matrix is no longer available. Instead, we use *average* outcome matrices:

$$\overline{\mathbf{M}}_\tau = \tfrac{1}{\tau} \sum_{t \in [\tau]} \mathbf{M}_t, \tag{7.23}$$

the average up to a given intermediate round $\tau \in [T]$. Similar to the stochastic case, the relaxation assumes that each instantaneous outcome matrix $\mathbf{M}_t$ is equal to the

average outcome matrix $\overline{\mathbf{M}}_\tau$. What is different now is that the relaxation depends on τ: using $\overline{\mathbf{M}}_\tau$ is tantamount to stopping precisely at this round.

With this intuition in mind, for a particular end-time τ we consider the linear program (7.8), parameterized by the time horizon τ and the average outcome matrix $\overline{\mathbf{M}}_\tau$. Its value, $\text{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_\tau, B, \tau)$, represents the per-round expected reward, so it needs to be scaled by the factor of τ to obtain the total expected reward. Finally, we maximize over τ. Thus, our linear relaxation for Adversarial BwK is defined as follows:

$$\text{OPT}_{\texttt{LP}}^{[T]} := \max_{\tau \in [T]} \tau \cdot \text{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_\tau, B, \tau) \geq \text{OPT}_{\texttt{FD}}\,. \tag{7.24}$$

The proof of Eq. (7.24) is similar to prior work (Badanidiyuru et al. [33], Devanur et al. [72]). Denote $\mathcal{D}_\tau$ to be the set of all distributions over the arms such that for every $\mathbf{p} \in \mathcal{D}_\tau$ we have the following: for every $i \in [d]$ we have $\sum_{t\in[\tau]} \mathbf{p} \cdot \mathbf{c}_{t,i} \leq B$. In other words, $\mathcal{D}_\tau$ denotes the set of distributions whose expected stopping time is at least τ. Thus it immediately follows that $\text{OPT}_{\text{LP}}(\tau) \geq \max_{\mathbf{p}\in\mathcal{D}_\tau} \sum_{t\in[\tau]} \mathbf{p} \cdot \mathbf{r}_t$ since for any given $\mathbf{p} \in \mathcal{D}_\tau$ it is feasible to LP(τ). Thus $\text{OPT}_{\text{LP}}(\tau)$ is at least the value of any feasible solution $\mathbf{p} \in \mathcal{D}_\tau$. Note that for every fixed distribution $\mathbf{p} \in \Delta_K$, there exists a τ such that either $\mathbf{p} \in \mathcal{D}_\tau$ and $\mathbf{p} \notin \mathcal{D}_{\tau+1}$ or $\mathbf{p} \in \mathcal{D}_T$. Moreover the total expected reward we can obtain using $\mathbf{p}$ is $\sum_{t\in[\tau]} \mathbf{p} \cdot \mathbf{r}_t$. Thus $\max_{1\leq\tau\leq T} \text{OPT}_{\text{LP}}(\tau) \geq \text{OPT}_{\texttt{FD}}$.

Regret bounds for ALG_j**.** Since each algorithm ALG_j, $j \in \{1, 2\}$ satisfies regret bound Eq. (7.4) with $\delta = T^{-2}$ and $R_\delta(T) = R_{j,\delta}(T)$, it also satisfies a stronger version (7.5) with the same parameters. Recall from Eq. (7.14) that the payoff range is $[b_{\min}, b_{\max}] = [-\frac{T_0}{B}+1, 2]$. For succinctness, let $U_j(T|T_0) = (1+\frac{T_0}{B})\, R_{j,\,\delta/T}(T)$

denote the respective regret term in Eq. (7.5).

Let us apply these regret bounds to our setting. Let $a_t \in [K]$ and $i_t \in [d]$ be, resp., the chosen arm and resource in round t. We represent the outcomes as vectors over arms: $\mathbf{r}_t, \mathbf{c}_{t,i} \in [0,1]^K$ denote, resp., reward vector and resource-i consumption vector for a given round t. Recall that the round-t payoffs in `LagrangeBwK` are given by the Lagrange function $\mathcal{L}_t := \mathcal{L}_{\mathbf{M}_t,B,T_0}$ such that

$$\mathcal{L}_t(a,i) = r_t(a) + 1 - \tfrac{T_0}{B}\, c_{t,i}(a) \tag{7.25}$$

for each arm a and resource i. Consider the total Lagrangian payoff at a given round $\tau \in [T]$:

$$\sum_{t\in[\tau]} \mathcal{L}_t(a_t, i_t) = \text{REW}_\tau + \tau - W_\tau, \tag{7.26}$$

where $\text{REW}_\tau = \sum_{t\in[\tau]} r_t(a_t)$ is the total reward up to round τ, and $W_\tau = \frac{T_0}{B} \sum_{t\in[\tau]} c_{t,i_t}(a_t)$ is the *consumption term*. The regret bounds sandwich Eq. (7.26) from above and below:

$$\left(\max_{a\in[K]} \sum_{t\in[\tau]} \mathcal{L}_t(a,i_t)\right) - U_1(T|T_0) \le \text{REW}_\tau + \tau - W_\tau \le \left(\min_{i\in[d]} \sum_{t\in[\tau]} \mathcal{L}_t(a_t,i)\right) + U_2(T|T_0). \tag{7.27}$$

This holds for all $\tau \in [T]$, with probability at least $1 - 2\delta$. The first inequality in Eq. (7.27) is due to the primal algorithm, and the second is due to the dual algorithm. Call them *primal* and *dual* inequality, respectively.

Crux of the proof. We condition on the event that Eq. (7.27) holds for all $\tau \in [T]$, which we call the *clean event*. The crux of the analysis is encapsulated in

the following lemma, which analyzes an execution of LagrangeBwK with an arbitrary parameter T_0 under the clean event.

Lemma 14. *Consider an execution of LagrangeBwK with $B_0 = B$ and an arbitrary parameter T_0 such that the clean event holds. Fix an arbitrary round $\sigma \in [T]$, and consider the LP value relative to this round:*

$$f(\sigma) := \mathrm{OPT}_{LP}(\overline{\mathbf{M}}_\sigma, B, \sigma). \tag{7.28}$$

The algorithm's reward up to round σ satisfies

$$\mathrm{REW}_\sigma \geq \min(T_0,\, \sigma \cdot f(\sigma) - dT_0) - (\, U_1(T|T_0) + U_2(T|T_0)\,). \tag{7.29}$$

Taking σ to be the maximizer in Eq. (7.24), algorithm's reward satisfies

$$\mathrm{REW} \geq \min(T_0, \mathrm{OPT}_{FD} - dT_0) - (\, U_1(T|T_0) + U_2(T|T_0)\,). \tag{7.30}$$

Proof. Let τ_{alg} be the stopping time of the algorithm. We consider two cases, depending on whether some resource is exhausted at time σ. In both cases, we focus on the round $\min(\tau_{\text{alg}}, \sigma)$.

Case 1: $\tau_{\text{alg}} \leq \sigma$ and some resource is exhausted. Let us focus on round $\tau = \tau_{\text{alg}}$. If i is the exhausted resource, then $\sum_{t\in[\tau]} c_{t,i}(a_t) > B$. Let us apply the dual inequality in Eq. (7.27) for this resource:

$$\begin{aligned}
\mathrm{REW}_\tau + \tau - W_\tau - U_2(T|T_0) &\leq \sum_{t\in[\tau]} \mathcal{L}_t(a_t, i) \\
&= \mathrm{REW}_\tau + \tau - \tfrac{T_0}{B} \sum_{t\in[\tau]} c_{t,i}(a_t) \\
&\leq \mathrm{REW}_\tau + \tau - T_0.
\end{aligned}$$

It follows that $W_\tau \geq T_0 - U_2(T|T_0)$.

Now, let us apply the primal inequality in Eq. (7.27) for the null arm. Recall that the reward and consumption for this arm is 0, so $\mathcal{L}_t(\texttt{null}, i_t) = 1$ for each round t. Therefore,

$$\text{REW}_\tau + \tau - W_\tau + U_1(T|T_0) \geq \sum_{t\in[\tau]} \mathcal{L}_t(\texttt{null}, i_t) = \tau.$$

We conclude that $\text{REW}_\tau \geq W_\tau - U_1(T|T_0) \geq T_0 - U_1(T|T_0) - U_2(T|T_0)$.

Case 2: $\tau_{\textbf{alg}} \geq \sigma$**.** Let us focus on round σ. Consider the linear program $\text{LP}_{\overline{\mathbf{M}}_\sigma, B, \sigma}$, and let $\mathbf{X}^* \in \Delta_K$ be an optimal solution to this LP. The primal inequality in Eq. (7.27) implies that

$$\begin{aligned}
\text{REW}_\sigma + \sigma - W_\sigma + U_1(\sigma) &\geq \max_{a\in[K]} \sum_{t\in[\sigma]} \mathcal{L}_t(a, i_t) \\
&\geq \sum_{t\in[\sigma]} \sum_{a\in[K]} X^*(a)\ \mathcal{L}_t(a, i_t) \\
&= \sigma + \sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{r}_t - \tfrac{T_0}{B} \sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i_t} \\
\text{REW}_\sigma &\geq \sigma \cdot f(\sigma) - \tfrac{T_0}{B} \sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i_t} - U_1(T|T_0). \qquad (7.31)
\end{aligned}$$

In the last inequality we used the fact that $\sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{r_t} = \sigma \cdot f(\sigma)$ by optimality of $\mathbf{X}^*$.

$\sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i} \leq B$ for each resource i, since $\mathbf{X}^*$ is a feasible solution for $\text{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_\sigma, B, \sigma)$. Then,

$$\sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i_t} \leq \sum_{i\in[d]} \sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i} \leq dB. \qquad (7.32)$$

Plugging Eq. (7.32) into Eq. (7.31), we conclude that $\text{REW}_\sigma \geq \sigma \cdot f(\sigma) - dT_0 - U_1(T|T_0)$.

Conclusions from the two cases imply Eq. (7.30), as claimed. □

Wrapping up. If $\mathrm{OPT}_{\texttt{FD}}$ lies in the guess range $[g_{\min}, g_{\max}]$, then some guess $\widehat{g}$ is approximately correct:

$$\mathrm{OPT}_{\texttt{FD}}/\kappa \leq \widehat{g} \leq \mathrm{OPT}_{\texttt{FD}}.$$

With such a guess $\widehat{g}$, and provided that $\kappa \geq d+1$, we have $T_0 = \widehat{g}/\kappa \geq \mathrm{OPT}_{\texttt{FD}}/\kappa^2$, and

$$\mathrm{OPT}_{\texttt{FD}} - dT_0 \geq \mathrm{OPT}_{\texttt{FD}}(1 - \tfrac{d}{\kappa}) \geq \mathrm{OPT}_{\texttt{FD}}/\kappa.$$

So, by Lemma 14, the algorithm's execution with this guess, assuming the clean event, satisfies Eq. (7.30) with $\min(T_0, \mathrm{OPT}_{\texttt{FD}} - dT_0) \geq \mathrm{OPT}_{\texttt{FD}}/\kappa^2$ and $T_0 \leq \mathrm{OPT}_{\texttt{FD}}/\kappa$. The regret term for this guess is

$$U_1(T|T_0) + U_2(T|T_0) \leq (1 + \tfrac{\mathrm{OPT}_{\texttt{FD}}}{\kappa B})\,(R_{1,\delta/T}(T) + R_{2,\delta/T}(T)).$$

To complete the proof of Theorem 7, we obtain a suitable guess $\widehat{g}$ with probability $1/\left\lceil \log_\kappa \frac{g_{\max}}{g_{\min}} \right\rceil$.

Proof Sketch of Lemma 13. Recall that in the adversarial bandit setting we have $\mathbf{c}_{i,t} = 0$ for every $i \in [d]$ and every $t \in [T]$. We re-analyze Lemma 14 with $\sigma = T$. Notice that case 1 never occurs. Thus we obtain obtain Eq. (7.31) in case 2. Note that $\frac{T_0}{B} \sum_{t\in[\sigma]} \mathbf{X}^* \cdot \mathbf{c}_{t,i_t} = 0$ since $\mathbf{c}_{i,t} = 0$. Therefore, we obtain

$$\mathrm{REW}_T \geq T \cdot f(T) - U_1(T|T_0).$$

We now argue that $T \cdot f(T) = \max_{a\in[K]} \sum_{t\in[T]} r_t(a)$. Let $\mathbf{X}^*$ be the optimal distribution over the arms. Thus $\sum_{t\in[T]} \mathbf{X}^* \cdot \mathbf{r_t} = T \cdot f(T)$. Note that since $\mathbf{c}_{i,t} = 0$

the only constraint on $\mathbf{X}^*$ is that it lies in Δ_K. Therefore the maximizer is a point distribution on $\max_{a\in[K]} \sum_{t\in[T]} r_t(a)$. This proof does not rely on any specific value for B_0, T_0. The payoff range is $[b_{\max}, b_{\min}] = [1, 2]$, so $U_1(T|T_0) = \tilde{O}\left(\sqrt{KT}\right)$.

7.7 High-probability algorithm for Adversarial BwK

In this section we recover the $O(\log T)$ approximation ratio for Adversarial BwK, but with high probability rather than merely in expectation. Our algorithm uses `LagrangeBwK` as a subroutine, and re-uses the adversarial analysis thereof (Lemma 14). We do not attempt to optimize the regret term.

The algorithm is considerably more complicated compared to Algorithm 6. Instead of making one random guess $\widehat{g}$ for the value of $\text{OPT}_{\text{LP}}^{[T]}$, we iteratively refine this guess over time. The algorithm proceeds in phases. In the beginning of each phase, we start a fresh instance of `LagrangeBwK` with parameter T_0 defined by the current value of $\widehat{g}$.[10] We update the guess $\widehat{g}$ in each round (in a way specified later), and stop the phase once $\widehat{g}$ becomes too large compared to its initial value in this phase. We invoke `LagrangeBwK` with a rescaled budget $B_0 = B/\Theta(\log T)$. Within each phase, we simulate the BwK problem with budget B_0: we stop `LagrangeBwK` once the consumption of some resource in this phase exceeds B_0. For the remainder of the phase, we play the null arm with probability $1-\gamma_0$ and do uniform exploration with the remaining probability, for some parameter $\gamma_0 \in (0, 1)$ (here and elsewhere, *uniform exploration* refers to choosing each action with equal probability). The

[10] The idea of restarting the algorithm in each phase is similar to the standard "doubling trick" in the online machine learning literature, but much more delicate in our setting.

pseudocode is summarized in Algorithm 7.

Algorithm 7: High-probability algorithm for Adversarial BwK.

input: scale parameter κ, exploration parameter γ_0, primal algorithm ALG_1, dual algorithm ALG_2

// ALG_1, ALG_2 `are adversarial online learning algorithms`
// `with bandit feedback and full feedback, resp.`

Initialize $\widehat{g} = 1$.
for *each phase* **do**
  Start a fresh instance ALG of `LagrangeBwK`
    with parameters $B_0 = B/2\lceil \log_\kappa T \rceil$ and $T_0 = \widehat{g}/(\lceil \log_\kappa T \rceil \kappa^2)$.
  for *each round in this phase* **do**
    Recompute the global estimate $\widehat{g}$
    if $\widehat{g} > T_0/\kappa$ **then** start a new phase
    if *consumption of all resources in this phase does not exceed* B_0 **then**
      Play the action chosen by ALG, observe the outcome and report it back to ALG.
    else
      Choose the null arm with probability $1 - \gamma_0$, do uniform exploration otherwise

To complete algorithm's specification, let us define how to update the guess $\widehat{g}$ in each round t. The guess, denoted $\widehat{g}_t$, is an estimate for $\text{OPT}_{\text{LP}}^{[t]}$, as defined in (7.24). We form this estimate using a standard *inverse propensity scoring* (*IPS*)

technique. Let $\mathbf{p}_t$ and a_t be, resp., the distribution and the arm chosen by the primal algorithm in round t. The instantaneous outcome matrix $\mathbf{M}_t$ is estimated by matrix $\mathbf{M}_t^{\texttt{ips}} \in [0,\infty)^{K\times d}$ such that each row $\mathbf{M}_t^{\texttt{ips}}(a)$ is defined as follows:

$$\mathbf{M}_t^{\texttt{ips}}(a) := 1_{\{a_t=a\}}\ \tfrac{1}{f_t(a_t)}\ \mathbf{M}_t(a).$$

For a given end-time τ, the average outcome matrix $\overline{\mathbf{M}}_\tau$ from (7.23) is estimated as

$$\overline{\mathbf{M}}_\tau^{\texttt{ips}} := \tfrac{1}{\tau}\ \sum_{t\in[\tau]} \mathbf{M}_t^{\texttt{ips}}.$$

Finally, we plug this estimate into (7.23) and define

$$\widehat{g}_t := \max_{\tau\in[T]}\ \tau\cdot \mathrm{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_\tau^{\texttt{ips}}, B, \tau). \tag{7.33}$$

For the analysis, we will assume that the primal algorithm does some uniform exploration:

$$p_t(a) \geq \gamma > 0 \quad \text{for each arm } a\in[K] \text{ and each round } t\in[T]. \tag{7.34}$$

Theorem 8. *Consider Adversarial BwK with K arms, d resources, time horizon T, and budget B; assume $B > 4\,T^{3/4}$. Suppose that one of the arms is a* null arm *that has zero reward and zero resource consumption. Let $\delta > 0$ be the failure probability parameter.*

Consider Algorithm 7 with parameters $\kappa \geq d+1$ and $\gamma_0 = T^{-1/4}$. Assume that each algorithm ALG_j, $j\in\{1,2\}$, satisfies the regret bound (7.4) with payoff range $[b_{\min}, b_{\max}] = [-\frac{T}{B}+1, 2]$ and regret term $R_\delta(T) = R_{j,\delta}(T)$. Assume that the primal algorithm ALG_1 satisfies (7.34) with parameter $\gamma \geq T^{-1/4}$.

Then the total reward REW *collected by Algorithm 7 satisfies*

$$\Pr\left[\ \mathrm{REW} \geq (\mathrm{OPT}_{\texttt{FD}} - \texttt{reg})/\left(2\kappa^4\,\lceil \log_\kappa T\rceil\right)\ \right] \geq 1 - O(\delta T), \tag{7.35}$$

where the regret term is $\texttt{reg} = \frac{T}{B}\left(K\, T^{3/4}\, \log^{1/2}(\frac{1}{\delta}) + R_{1,\delta/T}(T) + R_{2,\delta/T}(T) \right)$.

Remark 8. *Using algorithms EXP3.P for* ALG_1 *and Hedge for* ALG_2*, we can achieve (7.35) with*

$$\texttt{reg} = O\left(\tfrac{TK}{B}\right)\, T^{3/4}\, \sqrt{\log(T/\delta)}.$$

This is because EXP3.P, with appropriately modified uniform exploration term $\gamma = T^{-1/4}$*, satisfies the regret bound (7.4) with* $R_\delta(\tau) = O(T^{3/4})\sqrt{K \log \frac{T}{\delta}}$*, and for Hedge we can (still) use 7.6. The theorem is meaningful whenever, say,* $\texttt{reg} < \text{OPT}_{\texttt{FD}}/2$*. The latter requires* $\text{OPT}_{\texttt{FD}} \cdot \frac{B}{K} > \widetilde{\Omega}(T^{7/4})$.

Remark 9. *Like in Theorem 7, we posit that the null arm does not consume any resources.*

Proof Sketch. The proof consists of several steps. First, we argue that the guess $\widehat{g}_t$ is close to $\text{OPT}_{\text{LP}}^{[t]}$ with high probability. This argument only relies on the uniform exploration property (7.33) and the definition of IPS estimators, not on any properties of the algorithm. We immediately obtain concentration for the average outcome matrices; a somewhat subtle point is to derive concentration on the respective LP-values.

Next, we focus on a particular phase in the execution of the algorithm. We say that a phase is *full* if the stopping condition $\widehat{g}_t > T_0/\kappa$ has fired. We focus on the last full phase. We prove there is enough reward to be collected in this phase. Essentially, letting τ_1, τ_2 be, resp., the start and end time of this phase, we consider the BwK problem restricted to time interval $[\tau_1, \tau_2]$, and lower-bound the LP-value of this problem in terms of the LP-value of the original problem. Finally, we use the

adversarial analysis of `LagrangeBwK` (Lemma 14) to guarantee that our algorithm actually collects that value.

Because of the stopping condition $\widehat{g}_t > T_0/\kappa$, there can be at most $\lceil \log_\kappa T \rceil$ phases. Therefore, rescaling the budget to $B_0/2\lceil \log_\kappa T \rceil$ guarantees that the algorithm consumes at most $B/2$ of the budget. We then argue that, with high-probability, the additional uniform exploration in each phase, consumes a budget of at most $B/2$ with high-probability. Thus, the algorithm never runs out of budget. □

We now describe the full proof of Theorem 8, following the plan outlined in the proof sketch. We decompose the analysis into several distinct pieces, present them one by one, and then show how to put them together. Each piece is presented as a lemma, with appropriate notation and intuition; some of the proofs are deferred to later in this section. Throughout, we assume that

$$\min\{\mathrm{OPT}_{\texttt{FD}}, B\} > \Omega\left(KT^{3/4} \log \tfrac{T}{\delta}\right). \tag{7.36}$$

This is without loss of generality, because otherwise the guarantee in Theorem 8 is vacuous. Recall that the primal algorithm ALG_1 satisfies the uniform exploration property (7.33) with parameter $\gamma \geq T^{-1/4}$.

Extended notation. To argue about a given phase, we extend some of our notation to refer to arbitrary time intervals, not just $[1, \tau]$. In what follows, fix time interval

$[\tau_1, \tau_2]$, and let $\boldsymbol{\Delta}\tau = \tau_2 - \tau_1 + 1$. Let

$$\overline{\mathbf{M}}_{[\tau_1,\tau_2]} := \frac{1}{\boldsymbol{\Delta}\tau} \sum_{t=\tau_1}^{\tau_2} \mathbf{M}_t,$$
$$\overline{\mathbf{M}}^{\texttt{ips}}_{[\tau_1,\tau_2]} := \frac{1}{\boldsymbol{\Delta}\tau} \sum_{t=\tau_1}^{\tau_2} \mathbf{M}^{\texttt{ips}}_t$$

be, resp., the average outcome matrix and its IPS-estimate on this time interval. Define

$$\texttt{OBJ}([\tau_1, \tau_2]) := \boldsymbol{\Delta}\tau \cdot \text{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_{[\tau_1,\tau_2]}, B, \boldsymbol{\Delta}\tau), \tag{7.37}$$

$$\texttt{OBJ}^{\texttt{ips}}([\tau_1, \tau_2]) := \boldsymbol{\Delta}\tau \cdot \text{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}^{\texttt{ips}}_{[\tau_1,\tau_2]}, B, \boldsymbol{\Delta}\tau). \tag{7.38}$$

When $\tau_1 = 1$ we use the short-hand $\texttt{OBJ}(\tau_2)$ and $\texttt{OBJ}^{\texttt{ips}}(\tau_2)$. Recall that

$$\text{OPT}^{[\tau]}_{\texttt{LP}} := \max_{\tau \in [T]} \texttt{OBJ}(\tau). \tag{7.39}$$

$$\widehat{g}_\tau := \max_{\tau \in [T]} \texttt{OBJ}^{\texttt{ips}}(\tau). \tag{7.40}$$

Uniform exploration does not exhaust budget. The uniform exploration in Algorithm 7 happens for at most $\gamma_0 T$ rounds in expectation, and therefore for at most $\gamma_0 T + 3\sqrt{\gamma_0 T \ln(1/\delta)}$ rounds with probability at least $1 - \delta$.[11] It does not consume more than $B/2$ units of each resource, since $\gamma_0 = T^{-1/4}$ and $B > 4\,T^{3/4}$.

IPS estimators are good. We argue that, essentially, the guess $\widehat{g}_\tau$ is close to $\text{OPT}^{[\tau]}_{\texttt{LP}}$ with high probability. More precisely, we prove that $\texttt{OBJ}(\tau)$ is close to its IPS estimator, for any given $\tau \in [T]$. We will denote the deviation term as

$$\texttt{DEV}(\tau) := C_0 \left(\frac{K \cdot \texttt{OBJ}(\tau)}{\gamma B} \sqrt{\tau \log \tfrac{T}{\delta}} \right), \tag{7.41}$$

[11] By an easy application of Chernoff-Hoeffding bounds (Lemma 29).

for a sufficiently large absolute constant C_0. Thus, $\texttt{DEV}(\tau) \leq O\left(\frac{K\,T^{7/4}}{B}\sqrt{\log\frac{T}{\delta}}\right)$ ($:= \texttt{DEV}(T)$).

In this notation, we characterize the IPS estimators as follows:

Lemma 15. *With probability at least $1 - d\delta T$ it holds that*

$$\forall \tau \in [T] \qquad \textit{OBJ}^{ips}(\tau) - \textit{DEV}(\tau) \leq \textit{OBJ}(\tau) \leq \textit{OBJ}^{ips}(\tau) + \textit{DEV}(\tau). \tag{7.42}$$

If the event (7.42) holds, then $\widehat{g}_\tau$ and $\mathrm{OPT}_{LP}^{[\tau]}$ are indeed close:

$$\forall \tau \in [T] \qquad \left|\widehat{g}_\tau - \mathrm{OPT}_{LP}^{[\tau]}\right| \leq \max_{t\in[\tau]} \textit{DEV}(t). \tag{7.43}$$

The proof only relies on the uniform exploration property (7.33) and the definition of IPS estimators, not on anything that the algorithm does. A somewhat subtle point is to derive concentration on the respective LP-values from concentration of the average outcome matrices.

IPS estimators do not change too fast. We use the stopping condition in the algorithm to argue that the IPS estimators $\texttt{OBJ}^{\texttt{ips}}(\cdot)$ do not change too fast from one phase to another, in some specific sense. Namely, we compare $\texttt{OBJ}^{\texttt{ips}}(\cdot)$ in the first round of any full phase to the guess in the last round of the next phase.

Lemma 16. *Consider a full phase in the execution of the algorithm. Let τ be the first round in this phase, and let τ' be any round in the next phase. Then*

$$\frac{1}{\kappa^2} \leq \frac{\textit{OBJ}^{ips}(\tau)}{\widehat{g}_{\tau'}} \leq \frac{1}{\kappa} + \frac{K}{\gamma \widehat{g}_{\tau'}}. \tag{7.44}$$

Proof. There exists a $j \in \{1, 2, \ldots, \lceil \log_\kappa T \rceil\}$ such that $\texttt{OBJ}^{\texttt{ips}}(\tau) \geq \kappa^j$. Since τ is the beginning of a phase, this implies that $\texttt{OBJ}^{\texttt{ips}}(\tau - 1) < \kappa^{j-1}$. Observe that

in a single time-step the value of the estimate can increase by at most $\frac{K}{\gamma}$. Thus $\texttt{OBJ}^{\texttt{ips}}(\tau) < \kappa^{j-1} + \frac{K}{\gamma}$. Moreover since τ' belongs to the next phase, we have that $\kappa^{j+1} \leq \widehat{g}_{\tau'} \leq \kappa^{j+2}$. Putting these together we get that

$$\frac{1}{\kappa^2} \leq \frac{\texttt{OBJ}^{\texttt{ips}}(\tau)}{\widehat{g}_{\tau'}} \leq \frac{1}{\kappa} + \frac{K}{\gamma\kappa^{j+1}} \leq \frac{1}{\kappa} + \frac{K}{\gamma\widehat{g}_{\tau'}}. \qquad \square$$

This Lemma relies only on the stopping condition in the algorithm, $\widehat{g} > T_0/\kappa$, and the way the guess $\widehat{g}_\tau$ is expressed in terms of the IPS estimators $\texttt{OBJ}^{\texttt{ips}}(\cdot)$, as expressed in 7.40. It is irrelevant to this Lemma how the IPS estimators $\texttt{OBJ}^{\texttt{ips}}(\cdot)$ are actually defined.

Last full phase offers sufficient rewards. Recall that a phase in the execution of the algorithm is called *full* if the stopping condition $\widehat{g}_t > T_0/\kappa$ has fired. We focus on the last full phase; let τ_{start}, τ_{end} denote the first and last time-steps of this phase. We prove there is enough reward to be collected in this phase.

Let τ^* denote the maximizer in 7.24 which we interpret as the *optimal stopping time*. Essentially, we compare the LP value for the time interval $[\tau_{start}, \tau_{end}]$ with the LP value for the time interval $[1, \tau^*]$. The former is expressed as $\texttt{OBJ}([\tau_{start}, \tau_{end}])$ and the latter as $\text{OPT}_{\texttt{LP}}^{[T]}$. Note that the time horizon T lies in the subsequent phase (so we can apply Lemma 16).

Lemma 17. *Consider a run of the algorithm such that event* (7.42) *holds. Then*

$$\texttt{OBJ}([\tau_{start}, \tau_{end}]) \geq \left(\frac{1}{\kappa} - \frac{1}{\kappa^2}\right) \text{OPT}_{LP}^{[T]} - O(\texttt{DEV}(T)). \tag{7.45}$$

Adversarial analysis of `LagrangeBwK`. Let us plug in the adversarial analysis

of LagrangeBwK, as encapsulated in Lemma 14. We focus on the last full phase in the execution. We interpret it as an execution of algorithm LagrangeBwK with parameters B_0, T_0 on an instance of Adversarial BwK with budget B_0 that starts at round τ_{start} of the original problem. Let $\widehat{g} = \widehat{g}_{\tau_{start}}$ be the guess at the first round of the phase. Then the parameters are $B_0 = B/\texttt{ratio}$ and $T_0 = \widehat{g}/(\kappa^2 \cdot \texttt{ratio})$, where $\texttt{ratio} = \lceil \log_\kappa T \rceil$.

We apply Lemma 14 for round $\sigma = \tau_{end} - \tau_{start} + 1$ in the execution of LagrangeBwK. Restated in our notation, $f(\sigma)$ in Lemma 14 becomes

$$f(\sigma) = \mathrm{OPT}_{\texttt{LP}}(\overline{\mathbf{M}}_{[\tau_{start}, \tau_{end}]}, B_0, \sigma).$$

Thus, we obtain that with probability at least $1 - \delta$ we have

$$\mathrm{REW} \geq \sum_{t=\tau_{start}}^{\tau_{end}} r_t(a_t) \geq \min\left(\frac{\widehat{g}}{\lceil \log_\kappa T \rceil \kappa^2},\ \sigma f(\sigma) - \frac{d\widehat{g}}{\lceil \log_\kappa T \rceil \kappa^2} \right) - \texttt{reg}(T), \quad (7.46)$$

where the regret term is $\texttt{reg}(T) := (1 + \frac{T}{B}) \left(R_{1,\delta/T}(T) + R_{2,\delta/T}(T) \right)$.

Rescaling the budget. Since we use rescaled budget B_0, it is important to connect the corresponding LP-values to those for the original budget B. We have the following general fact (observed in Agrawal and Devanur [11]): for any outcome matrix $\mathbf{M}$, budget B, time horizon T, and rescaling factor $\psi \in (0, 1]$ it holds that

$$\mathrm{OPT}_{\texttt{LP}}(\mathbf{M}, \psi B, T) \geq \psi \cdot \mathrm{OPT}_{\texttt{LP}}(\mathbf{M}, B, T). \quad (7.47)$$

This holds because an optimal solution μ to $\mathrm{LP}_{\mathbf{M},B,T}$, the vector $\psi\,\mu$ is feasible to $\mathrm{LP}_{\mathbf{M},\psi B,T}$.

Putting it all together. We defer the proofs of Lemma 15 and Lemma 17 to the

following subsections, and show how to complete the proof of Theorem 8 assuming these Lemmas hold.

Proof of Theorem 8. Throughout this proof, let us condition on the high-probability events in Lemma 15 and 7.46. Moreover from 7.47 we have that $\sigma f(\sigma) \geq \frac{1}{\texttt{ratio}}\texttt{OBJ}([\tau_{start}, \tau_{end}])$ since $B_0 = \frac{B}{\texttt{ratio}}$. Thus combining this with 7.46 we get,

$$\text{REW} = \sum_{t=\tau_{start}}^{\tau_{end}} r_t(a_t) \geq \frac{1}{\texttt{ratio}} \min\left(\frac{\widehat{g}}{\kappa^2}, \texttt{OBJ}([\tau_{start}, \tau_{end}]) - \frac{d\widehat{g}}{\kappa^2}\right) - \texttt{reg}(T). \quad (7.48)$$

Further from Lemma 17 with $\kappa = d + 1$, we can re-write 7.48 as

$$\text{REW} \geq \frac{1}{\texttt{ratio}} \min\left(\frac{\widehat{g}}{\kappa^2}, \left(\frac{1}{\kappa} - \frac{1}{\kappa^2}\right)\text{OPT}_{\texttt{LP}}^{[T]} - \frac{\widehat{g}}{\kappa}\right) - \texttt{reg}(T). \quad (7.49)$$

Recall that $\widehat{g}$ for the phase $[\tau_{start}, \tau_{end}]$ is $\texttt{OBJ}^{\texttt{ips}}(\tau_{start})$. Moreover we have that T lies in the phase immediately after $[\tau_{start}, \tau_{end}]$.

Thus from 7.44 (first inequality below) and 7.43 (second inequality below) we have,

$$\widehat{g} \leq \frac{1}{\kappa}\widehat{g}_T + \frac{K}{\gamma} \leq \frac{1}{\kappa}\text{OPT}_{\texttt{LP}}^{[T]} + O(\texttt{DEV}(T)). \quad (7.50)$$

Likewise we have,

$$\widehat{g} \geq \texttt{OBJ}^{\texttt{ips}}(\tau_{start}) \geq \frac{1}{\kappa^2}\widehat{g}_T \geq \frac{1}{\kappa^2}\text{OPT}_{\texttt{LP}}^{[T]} - O(\texttt{DEV}(T)), \quad (7.51)$$

where the second inequality uses 7.44 and the last inequality uses 7.42. Plugging 7.50 and 7.51 back into 7.49 we get,

$$\text{REW} \geq \frac{1}{\texttt{ratio}} \min\left(\frac{\text{OPT}_{\texttt{LP}}^{[T]}}{\kappa^4}, \left(\frac{1}{\kappa} - \frac{2}{\kappa^2}\right)\text{OPT}_{\texttt{LP}}^{[T]}\right) - \texttt{reg}(T) - O(\texttt{DEV}(T)). \quad (7.52)$$

And from 7.24 we have $\text{OPT}_{\texttt{LP}}^{[T]} \geq \text{OPT}_{\texttt{FD}}$. Plugging this into 7.52 we get 7.35. □

7.7.1 Proof of Lemma 17 (last full phase offers sufficient rewards)

We first prove the following property of the optimal solution.

Claim 1. *Let $T_1 < T_2$ be any two time-steps. Then,*

$$\texttt{OBJ}([T_1, T_2]) \geq \texttt{OBJ}(T_2) - \texttt{OBJ}(T_1 - 1). \quad (7.53)$$

Proof. We prove this claim as follows. Let μ_{T_2} denote the optimal solution to $\text{LP}_{\overline{\mathbf{M}}_{T_2}, B, T_2}$. Since $c_{t,i}(a) \geq 0$ for every $a \in [K]$, $t \in [T]$ and $i \in [d]$ we have

$$\mu_{T_2} \cdot \sum_{t \in [T_2]} \mathbf{c}_{t,i} \leq B \implies \mu_{T_2} \cdot \sum_{t \in [T_1 - 1]} \mathbf{c}_{t,i} \leq B.$$

Thus μ_{T_2} is feasible to $\text{LP}_{\overline{\mathbf{M}}_{T_1-1}, B, T_1-1}$. This implies that $\sum_{t \in [T_1-1]} \mu_{T_2} \cdot \mathbf{r}_t \leq \texttt{OBJ}(T_1 - 1)$. Likewise μ_{T_2} is also feasible to $\text{LP}_{\overline{\mathbf{M}}_{[T_1,T_2]}, B, [T_1, T_2]}$.

Therefore we have,

$$\begin{aligned}
\texttt{OBJ}([T_1, T_2]) &\geq \sum_{t=T_1}^{T_2} \mu_{T_2} \cdot \mathbf{r}_t \\
&= \sum_{t\in[T_2]} \mu_{T_2} \cdot \mathbf{r}_t - \sum_{t\in[T_1-1]} \mu_{T_2} \cdot \mathbf{r}_t \\
&= \texttt{OBJ}(T_2) - \sum_{t\in[T_1-1]} \mu_{T_2} \cdot \mathbf{r}_t \\
&\geq \texttt{OBJ}(T_2) - \texttt{OBJ}(T_1 - 1) \qquad \square
\end{aligned}$$

Consider $\texttt{OBJ}([\tau_{start}, \tau_{end}])$. From 7.53 we have

$$\texttt{OBJ}([\tau_{start}, \tau_{end}]) \geq \texttt{OBJ}(\tau_{end}) - \texttt{OBJ}(\tau_{start} - 1). \tag{7.54}$$

From Lemma 15 we can re-write 7.54 as

$$\texttt{OBJ}(\tau_{end}) - \texttt{OBJ}(\tau_{start} - 1) \geq \texttt{OBJ}^{\texttt{ips}}(\tau_{end}) - \texttt{OBJ}^{\texttt{ips}}(\tau_{start} - 1) - O(\texttt{DEV}(T)). \tag{7.55}$$

For some $j \in \lceil \log_\kappa(T) \rceil$, at time-step τ_{end} the value $\widehat{g}_t$ exceeds κ^j for the first time. Likewise $\widehat{g}_t$ exceeds κ^{j-1} for the first time at τ_{start} and is smaller than this value at $\tau_{start} - 1$. This implies that $\texttt{OBJ}^{\texttt{ips}}(\tau_{end}) \geq \kappa^j$ and $\texttt{OBJ}^{\texttt{ips}}(\tau_{start} - 1) < \kappa^{j-1}$. Therefore 7.55 simplifies to

$$\texttt{OBJ}^{\texttt{ips}}(\tau_{end}) - \texttt{OBJ}^{\texttt{ips}}(\tau_{start} - 1) - O(\texttt{DEV}(T)) \geq \kappa^j - \kappa^{j-1} - O(\texttt{DEV}(T)). \tag{7.56}$$

Combining 7.56, 7.55 and 7.54 we get

$$\texttt{OBJ}([\tau_{start}, \tau_{end}]) \geq \kappa^j - \kappa^{j-1} - O(\texttt{DEV}(T)).$$

Dividing throughout by $\widehat{g}_T$ and noting that $\widehat{g}_T \leq \kappa^{j+1}$ we have,

$$\begin{aligned}\frac{\texttt{OBJ}([\tau_{start}, \tau_{end}])}{\widehat{g}_T} &\geq \frac{\kappa^{j-1}(\kappa-1)}{\kappa^{j+1}} - \frac{O(\texttt{DEV}(T))}{\widehat{g}_T} \\ &= \left(\frac{1}{\kappa} - \frac{1}{\kappa^2}\right) - \frac{O(\texttt{DEV}(T))}{\widehat{g}_T}\end{aligned}$$

Finally applying Lemma 15 to $\widehat{g}_T$ we get 7.45. This completes the proof of Lemma 17.

7.7.2 Proof of Lemma 15 (IPS estimators are good)

Recall that for every $t \in [T]$ and $a \in [K]$ we have that $p_t(a)$, the probability that arm a is chosen at time t is at least $\frac{\gamma}{K}$. We now prove Lemma 2 which relates linear sums of rewards and consumptions computed using the unbiased estimates and the true values. Denote $R_{\gamma,\delta}(\tau) := \frac{K}{\gamma}\sqrt{2\tau \ln(T/\delta)}$.

Claim 2. *Let $\delta > 0$, $\gamma > 0$ used by the EXP3.P(γ) be given parameters. Then we have the following statements for any fixed $\mathbf{z} \in \Delta_K$.*

$$\Pr\left[\exists \tau \in [T] \quad \left|\sum_{t\in[\tau]} \mathbf{z} \cdot [\hat{\mathbf{r}}_t - \mathbf{r}_t]\right| > R_{\gamma,\delta}(\tau)\right] \leq \delta \tag{7.57}$$

$$\forall i \in [d] \qquad \Pr\left[\exists \tau \in [T] \quad \left|\sum_{t\in[\tau]} \mathbf{z} \cdot [\hat{\mathbf{c}}_{t,i} - \mathbf{c}_{t,i}(a)]\right| > R_{\gamma,\delta}(\tau)\right] \leq \delta \tag{7.58}$$

Proof. The proof of this follows directly from the invocation of the Azuma-Hoeffding inequality. We will show this for Equation (7.57). Define $Y_t := \mathbf{z} \cdot [\hat{\mathbf{r}}_t - \mathbf{r}_t]$ (like-wise for the lower-tail use $Y_t := \mathbf{z} \cdot (\mathbf{r}_t - \hat{\mathbf{r}}_t)$). Note that this forms a martingale difference sequence since $\mathbb{E}[\mathbf{z} \cdot (\hat{\mathbf{r}}_t - \mathbf{r}_t) \mid \mathcal{H}_{t-1}] = \mathbf{z} \cdot [\mathbf{r}_t - \mathbf{r}_t] = 0$. Here we used the fact that $\mathbf{z}$ is not random and fixed before the start of the algorithm. Also we have that $|Y_t| \leq \frac{K}{\gamma}$.

Using Lemma 28 and taking a union bound over all $\tau \in [T]$ we have the desired equation. $\square$

We will now prove the two inequalities in 7.42. We will first prove the first inequality in 7.42.

Let μ^* denote the optimal solution to $\text{OPT}_{\text{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right)$. Note this is valid whenever $B > \Omega\left(\frac{K}{\gamma}\sqrt{\tau \log \frac{T}{\delta}}\right)$. From Equation (7.58) we have that with probability at least $1-\delta$ for every $i \in [d]$,

$$\begin{aligned}\sum_{t\in[\tau]} \mu^* \cdot \hat{\mathbf{c}}_{t,i} &\leq \sum_{t\in[\tau]} \mu^* \cdot \mathbf{c}_{t,i} + R_{\gamma,\delta}(\tau). \\ &\leq B \end{aligned} \tag{7.59}$$

7.59 used the fact that $\sum_{t\in[\tau]} \mu^* \cdot \mathbf{c}_{t,i} \leq B\left(1 - R_{\gamma,\delta}(\tau)\right)$.

Using Equation (7.57), we have that with probability at least $1-\delta$,

$$\sum_{t\in[\tau]} \mu^* \cdot \mathbf{r}_t \leq \sum_{t\in[\tau]} \mu^* \cdot \hat{\mathbf{r}}_t + R_{\gamma,\delta}(\tau).$$

Using the fact that,

$$\sum_{t\in[\tau]} \mu^* \cdot \mathbf{r}_t = \text{OPT}_{\text{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right),$$

we have the following.

$$\text{OPT}_{\text{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right) - R_{\gamma,\delta}(\tau) \leq \sum_{t\in[\tau]} \mu^* \cdot \hat{\mathbf{r}}_t. \tag{7.60}$$

From 7.59 we have that μ^* is feasible to $\text{OPT}^{\texttt{ips}}_{\texttt{LP}}(\tau)$ and from 7.60 this implies that

$$\texttt{OBJ}^{\texttt{ips}}(\tau) \geq \text{OPT}_{\texttt{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right) - R_{\gamma,\delta}(\tau). \tag{7.61}$$

Finally from 7.47 we have

$$\text{OPT}_{\texttt{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right) \geq \left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right)\texttt{OBJ}(\tau). \tag{7.62}$$

From 7.61 and 7.62 we have

$$\texttt{OBJ}^{\texttt{ips}}(\tau) \geq \texttt{OBJ}(\tau) - \underbrace{R_{\gamma,\delta}(\tau)\left(1 + \frac{\texttt{OBJ}(\tau)}{B}\right)}_{=O\left(\frac{K\texttt{OBJ}(\tau)}{\gamma B}\sqrt{\tau \log \frac{T}{\delta}}\right)},$$

which gives the lower-tail in 7.42.

We will now prove the second inequality in 7.42 in a similar fashion. Let $\tilde{\mu}^*$ denote the optimal solution to $\texttt{OBJ}^{\texttt{ips}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right)$.

From Equation (7.58) we have that with probability at least $1 - \delta$ for every $i \in [d]$,

$$\begin{aligned}\sum_{t\in[\tau]} \tilde{\mu}^* \cdot \mathbf{c}_{t,i} &\leq \sum_{t\in[\tau]} \tilde{\mu}^* \cdot \hat{\mathbf{c}}_{t,i} + R_{\gamma,\delta}(\tau). \\ &\leq B \end{aligned} \tag{7.63}$$

7.63 used the fact that $\sum_{t\in[\tau]} \tilde{\mu}^* \cdot \hat{\mathbf{c}}_{t,i} \leq B\left(1 - R_{\gamma,\delta}(\tau)\right)$.

Using Equation (7.57), we have that with probability at least $1 - \delta$,

$$\sum_{t\in[\tau]} \tilde{\mu}^* \cdot \hat{\mathbf{r}}_t \leq \sum_{t\in[\tau]} \tilde{\mu}^* \cdot \mathbf{r}_t + R_{\gamma,\delta}(\tau).$$

From the fact that

$$\sum_{t\in[\tau]} \tilde{\mu}^* \cdot \hat{\mathbf{r}}_t = \text{OPT}^{\texttt{ips}}_{\texttt{LP}}\left(\overline{\mathbf{M}}_\tau, B\left(1 - \frac{R_{\gamma,\delta}(\tau)}{B}\right), \tau\right),$$

we get the following.

$$\mathrm{OPT}_{\mathtt{LP}}^{\mathtt{ips}}\left(\overline{\mathbf{M}}_\tau, B\left(1-\frac{R_{\gamma,\delta}(\tau)}{B}\right),\tau\right) - R_{\gamma,\delta}(\tau) \leq \sum_{t\in[\tau]} \tilde{\mu}^* \cdot \mathbf{r}_t. \tag{7.64}$$

From 7.63 we have that $\tilde{\mu}_j^*$ is feasible to $\mathrm{OPT}_{\mathtt{LP}}(\tau)$ and from 7.64 this implies that

$$\mathrm{OPT}_{\mathtt{LP}}^{\mathtt{ips}}\left(\overline{\mathbf{M}}_\tau, B\left(1-\frac{R_{\gamma,\delta}(\tau)}{B}\right),\tau\right) \leq \mathtt{OBJ}(\tau) + R_{\gamma,\delta}(\tau). \tag{7.65}$$

Finally from 7.47 we have

$$\mathrm{OPT}_{\mathtt{LP}}^{\mathtt{ips}}\left(\overline{\mathbf{M}}_\tau, B\left(1-\frac{R_{\gamma,\delta}(\tau)}{B}\right),\tau\right) \geq \left(1-\frac{R_{\gamma,\delta}(\tau)}{B}\right)\mathtt{OBJ}^{\mathtt{ips}}(\tau). \tag{7.66}$$

Combining 7.66 and 7.65 we get,

$$\mathtt{OBJ}^{\mathtt{ips}}(\tau) \leq \mathtt{OBJ}(\tau) + \frac{R_{\gamma,\delta}(\tau)}{B - R_{\gamma,\delta}(\tau)}\left(\mathtt{OBJ}(\tau) + R_{\gamma,\delta}(\tau)\right). \tag{7.67}$$

Since $B_0 > 2R_{\gamma,\delta}(\tau)$ we get,

$$\mathtt{OBJ}^{\mathtt{ips}}(\tau) \leq \mathtt{OBJ}(\tau) + \underbrace{\frac{2R_{\gamma,\delta}(\tau)}{B}\left(\mathtt{OBJ}(\tau) + R_{\gamma,\delta}(\tau)\right)}_{=O\left(\frac{K\mathtt{OBJ}(\tau)}{\gamma B}\sqrt{\tau\log\frac{T}{\delta}}\right)},$$

and thus we get the upper-tail in 7.42.

We will now prove 7.43. Recall that $\widehat{g}_\tau := \max_{t\in[\tau]}\mathtt{OBJ}^{\mathtt{ips}}(t)$. Moreover $\mathrm{OPT}_{\mathtt{LP}}^{[\tau]} = \max_{t\in[\tau]}\mathtt{OBJ}(t)$.

Consider $\widehat{g}_\tau - \mathrm{OPT}_{\mathtt{LP}}^{[\tau]}$. We have,

$$\begin{aligned}
\widehat{g}_\tau - \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} &= \max_{t\in[\tau]} \mathtt{OBJ}^{\mathtt{ips}}(t) - \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} \\
&\leq \max_{t\in[\tau]} (\mathtt{OBJ}(t) + \mathtt{DEV}(t)) - \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} \\
&\leq \max_{t\in[\tau]} \mathtt{OBJ}(t) + \max_{t\in[\tau]} \mathtt{DEV}(t) - \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} \\
&= \max_{t\in[\tau]} \mathtt{DEV}(t).
\end{aligned}$$

Now consider $\mathrm{OPT}_{\mathtt{LP}}^{[\tau]} - \widehat{g}_\tau$. We have,

$$\begin{aligned}
\mathrm{OPT}_{\mathtt{LP}}^{[\tau]} - \widehat{g}_\tau &\leq \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} - \max_{t\in[\tau]} (\mathtt{OBJ}(t) - \mathtt{DEV}(t)) \\
&\leq \mathrm{OPT}_{\mathtt{LP}}^{[\tau]} - \max_{t\in[\tau]} \mathtt{OBJ}(t) + \max_{t\in[\tau]} \mathtt{DEV}(t) \\
&= \max_{t\in[\tau]} \mathtt{DEV}(t).
\end{aligned}$$

This completes the proof of Lemma 15.

7.8 Lower bounds

We prove the lower bounds on the competitive ratio that we have claimed in sub-section 7.3.1: the $\Omega(\log T)$ lower bound w.r.t. the best fixed distribution benchmark ($\mathrm{OPT}_{\mathtt{FD}}$), the $\Omega(T)$ lower bound w.r.t. the best dynamic policy benchmark ($\mathrm{OPT}_{\mathtt{DP}}$), and the $\Omega(K)$ lower bound w.r.t. the best fixed arm benchmark ($\mathrm{OPT}_{\mathtt{FA}}$). As a warm-up, we analyze the simple example from sub-section 7.3.1 that leads to the $\frac{5}{4}$ lower bound w.r.t. $\mathrm{OPT}_{\mathtt{FD}}$. All lower-bounds are for a randomized algorithm against an oblivious adversary. We summarize all these lower bounds in the following theorem:

Theorem 9. *Consider Adversarial BwK with a single resource ($d = 1$) and K arms. Consider any randomized algorithm for this problem, and let* REW *denote its reward. Then:*

(a) $\text{OPT}_{\texttt{FD}}/\mathbb{E}[\text{REW}] \geq \frac{5}{4} - o(1)$ *for some problem instance (from the example in the Introduction).*

(b) $\text{OPT}_{\texttt{FD}}/\mathbb{E}[\text{REW}] \geq \Omega(\log T)$ *for some problem instance.*

(c) $\text{OPT}_{\texttt{DP}}/\mathbb{E}[\text{REW}] \geq T/B^2$ *for some problem instance, for any given budget B.*

(d) $\text{OPT}_{\texttt{FA}}/\mathbb{E}[\text{REW}] \geq \Omega(K)$ *for some problem instance.*

Remark 10. *The lower bounds for parts (a,b,c) hold (even) for problem instances with $K = 2$ arms. The lower bounds in parts (a,b) hold even for a much more permissive feedback model from the online packing literature, namely, when the algorithm observes the outcome vector for all actions in a given round, and moreover does it* before *it chooses an arm in this round.*

We tweak our construction from Theorem 9(c) to obtain a strong lower bound for the contextual version of Adversarial BwK (a.k.a. *Adversarial cBwK*), as studied in Section 7.9.3. This lower bound implies that Adversarial cBwK is essentially hopeless in the regime $B < \sqrt{T}$, complementing a strong positive result (Corollary 3) for the regime $B > \tilde{\Omega}(\sqrt{T})$. It is proved in Section 7.8.3, along with Theorem 9(c).

Theorem 10. *Consider adversarial contextual bandits with knapsacks (Adversarial cBwK), with policy class Π, a single resource ($d = 1$), $K = 2$ arms, and any given*

budget $B < \sqrt{T}$. Consider any randomized algorithm for this problem, and let REW *denote its reward. Then*

$$\text{OPT}_{FD}(\Pi)/\mathbb{E}[\text{REW}] \geq T/B^2 \quad \textit{for some problem instance.}$$

Notation. In the proof of lower-bounds below, we use the following notation. Given an instance $\mathcal{I}$, we denote $\text{OPT}_{\texttt{FD}}(\mathcal{I})$, $\text{OPT}_{\texttt{FA}}(\mathcal{I})$ and $\text{OPT}_{\texttt{DP}}(\mathcal{I})$ to denote the optimal value of the best fixed distribution, best fixed arm and best dynamic policy respectively, for instance $\mathcal{I}$. Likewise let $\text{OPT}_{\texttt{LP}}^{[T]}(\mathcal{I})$ denote the optimal LP value for instance $\mathcal{I}$ and given an algorithm $\mathcal{A}$ and an instance $\mathcal{I}$, let $\mathbb{E}[\text{REW}(\mathcal{A}, \mathcal{I})]$ denote the expected reward obtained by $\mathcal{A}$ on instance $\mathcal{I}$, where the expectation is over the internal randomness of the algorithm.

7.8.1 Warm-up: example from the Introduction

As a warm-up, let us recap and analyze the example from the Introduction.

Construction 2. *There are two arms and one resource with budget $B = \frac{T}{2}$. Arm* 1 *has zero rewards and zero consumption. Arm* 2 *has consumption* 1 *in each round, and offers reward $\frac{1}{2}$ in each round of the first half-time ($\frac{T}{2}$ rounds). In the second half-time, arm* 1 *offers either reward* 1 *in all rounds, or reward* 0 *in all rounds. More formally, there are two problem instances, call them $\mathcal{I}_1$ and $\mathcal{I}_2$, that coincide for the first half-time and differ in the second half-time.*

Lemma 18. *Any algorithm suffers $\text{OPT}_{FD}/\mathbb{E}[\text{REW}] \geq \frac{5}{4} - o(1)$ on some instances in Construction 2.*

The intuition is that given a random instance as input the algorithm needs to choose how much budget to invest in the first half-time, without knowing what comes in the second, and any choice (in expectation) leads to the claimed competitive ratio.

To prove Lemma 18 (as well we the main lower bound in Theorem 9(b)) we compare algorithm's performance to $\mathrm{OPT}_{\mathtt{LP}}^{[T]}$, and invoke the following lemma:

Lemma 19. $\mathrm{OPT}_{FD} \geq \mathrm{OPT}_{LP}^{[T]} - O\left(\mathrm{OPT}_{LP}^{[T]} \cdot \sqrt{\frac{\log dT}{B}}\right)$.

Proof. Let τ^* denote the time-step at which $\mathrm{OPT}_{\mathtt{LP}}^{[T]}$ is maximized. Let $\mathbf{p}$ denote the optimal solution to $\tau^* \cdot \mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{\tau^*}, B(1-\epsilon), \tau^*)$ where $\epsilon = \sqrt{\frac{\log dT}{B}}$. Note that $\mathrm{OPT}_{\mathtt{FD}}$ is at least as large as the expected total reward obtained by the distribution $\mathbf{p}$. From the Chernoff-Hoeffding bounds (Lemma 29), with probability at least $1 - dT^{-2}$ we have

$$\forall i \in [d] \qquad \sum_{t \in [\tau^*]} \mathbf{p} \cdot \mathbf{c}_{t,i} \leq B.$$

Conditioning on this event the expected total reward obtained by $\mathbf{p}$ is

$$\sum_{t \in [\tau^*]} \mathbf{p} \cdot \mathbf{r}_t = \tau^* \cdot \mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{\tau^*}, B(1-\epsilon), \tau^*).$$

Thus the expected total reward obtained by $\mathbf{p}$ is at least $\tau^* \cdot \mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{\tau^*}, B(1-\epsilon), \tau^*)$. [12] Moreover from Eq. (7.47) we have that

$$\begin{aligned}
\mathrm{OPT}_{\mathtt{FD}} &\geq \tau^* \cdot \mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{\tau^*}, B(1-\epsilon), \tau^*) \\
&\geq (1-\epsilon)\tau^* \cdot \mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{\tau^*}, B(1-\epsilon), \tau^*) \\
&\geq \mathrm{OPT}_{\mathtt{LP}}^{[T]} - O\left(\mathrm{OPT}_{\mathtt{LP}}^{[T]} \sqrt{\frac{\log dT}{B}}\right).
\end{aligned}$$

□

[12] With probability T^{-2} we assume that $\mathbf{p}$ has an expected reward of 0.

Proof of Lemma 18. Denote the two arms by A_1 and A_0 where A_0 denotes the null arm. The consumption for arm A_1 is 1 for all rounds in both $\mathcal{I}_1$ and $\mathcal{I}_2$. Thus the only difference between the two instances is the rewards obtained for playing arm A_1 in each round. The instances have two phases where each phase lasts for $\frac{T}{2}$ rounds. In phase 1, in both $\mathcal{I}_1$ and $\mathcal{I}_2$ playing arm A_1 fetches a reward $\frac{1}{2}$. In the second phase, in $\mathcal{I}_1$, the reward for playing arm A_1 is 0 while in $\mathcal{I}_2$ the reward for playing arm A_1 is 1. Thus the *outcome* matrix for the first $\frac{T}{2}$ time-steps is the same in instances $\mathcal{I}_1$ and $\mathcal{I}_2$.

Consider a randomized algorithm $\mathcal{A}$. Let α_1 be the expected number of times arm A_1 is played by $\mathcal{A}$ in the first $\frac{T}{2}$ rounds on instances $\mathcal{I}_1$ and $\mathcal{I}_2$. Note since the outcome matrix is same, the expected number of times the arm is played should be same in both the instances. Let $\alpha_{2,1}$, $\alpha_{2,2}$ denote the expected number of times arm A_1 is played in the second phase in instances $\mathcal{I}_1$ and $\mathcal{I}_2$ respectively.

Recall that in this section we are interested in a lower-bound on the competitive ratio $\mathrm{OPT}_{\mathtt{FD}}/\mathbb{E}[\mathrm{REW}]$ for every instance. Consider $\mathrm{OPT}_{\mathtt{LP}}^{[T]}(\mathcal{I}_1)$, the optimal value of the best fixed distribution on $\mathcal{I}_1$. Using Eq. (7.24) with $\tau = \frac{T}{2}$ this equals $\frac{T}{2} \cdot \mathrm{OPT}_{\mathtt{LP}}\left(\overline{\mathbf{M}}_{\frac{T}{2}}, B, \frac{T}{2}\right)$ which evaluates to $\frac{T}{4}$. Likewise $\mathrm{OPT}_{\mathtt{LP}}^{[T]}(\mathcal{I}_2)$ equals $T \cdot \mathrm{OPT}_{\mathtt{LP}}\left(\overline{\mathbf{M}}_T, B, T\right)$, which evaluates to $\frac{3T}{8}$. Consider the performance of $\mathcal{A}$ on $\mathcal{I}_1$. We have,

$$\frac{\mathrm{OPT}_{\mathtt{LP}}^{[T]}(\mathcal{I}_1)}{\mathbb{E}[\mathrm{REW}(\mathcal{A},\mathcal{I}_1)]} \geq \left(\frac{T}{4}\right) / \left(\frac{\alpha_1}{2}\right). \tag{7.68}$$

Likewise on $\mathcal{I}_2$ we have,

$$\frac{\mathrm{OPT}_{\mathtt{LP}}^{[T]}(\mathcal{I}_2)}{\mathbb{E}[\mathrm{REW}(\mathcal{A},\mathcal{I}_2)]} \geq \left(\frac{3T}{8}\right) / \left(\frac{\alpha_1}{2} + \alpha_{2,2}\right). \tag{7.69}$$

Thus the competitive ratio of $\mathcal{A}$ is at least the maximum of the ratios in Eq. (7.68) and Eq. (7.69). Thus we want to minimize this maximum and is achieved when the two ratios are equal to each other.

Notice that the term $\alpha_{2,1}$ does not appear in Eq. (7.68) and Eq. (7.69). By setting the term in Eq. (7.68) equal to the term in Eq. (7.69) and re-arranging,

$$\alpha_1 = 4\alpha_{2,2}. \tag{7.70}$$

Moreover we have $\alpha_1 + \alpha_{2,2} \leq B$. Combining this with Eq. (7.70) we get $\alpha_1 \leq \frac{4B}{5} = \frac{2T}{5}$ and the corresponding competitive ratio is at least $\left(\frac{T}{4}\right) / \left(\frac{\alpha_1}{2}\right) \geq \frac{5}{4}$. By Lemma 19 with $d = 1$, for every $j \in [2]$,

$$\mathrm{OPT}_{\texttt{FD}}(\mathcal{I}_j)/\mathbb{E}[\mathrm{REW}(\mathcal{A}, \mathcal{I}_j)] \geq \frac{5}{4} - O\left(\frac{\mathrm{OPT}_{\texttt{LP}}^{[T]}}{T}\sqrt{\frac{\log T}{B}}\right). \qquad \square$$

7.8.2 The main lower bound: proof of Theorem 9(b)

To obtain the $\Omega(\log T)$ lower bound in Theorem 9(b), we extend Construction 2 to one with $\Omega(\log T)$ phases rather than just two. As before, the algorithm needs to decide how much budget to save for the subsequent phases; without knowing whether they would bring higher rewards or nothing. The construction is as follows, see Figure 7.3 for a pictorial representation:

Construction 3. *There is one resource with budget B, and two arms, denoted A_0, A_1. Arm A_0 is the "null arm" that has zero reward and zero consumption. The consumption of arm A_1 is 1 in all rounds. The rewards of A_1 are defined as follows. We partition the time into $\frac{T}{B}$ phases of duration B each (for simplicity, assume that*

B divides T). We consider $\frac{T}{B}$ problem instances; for each instance $\mathcal{I}_\tau$, $\tau \in \left[\frac{T}{B}\right]$ arm A_1 has positive rewards up to and including phase τ; after that all rewards are 0. In each phase $\sigma \in [\tau]$, arm A_1 has reward σ/T in each round.

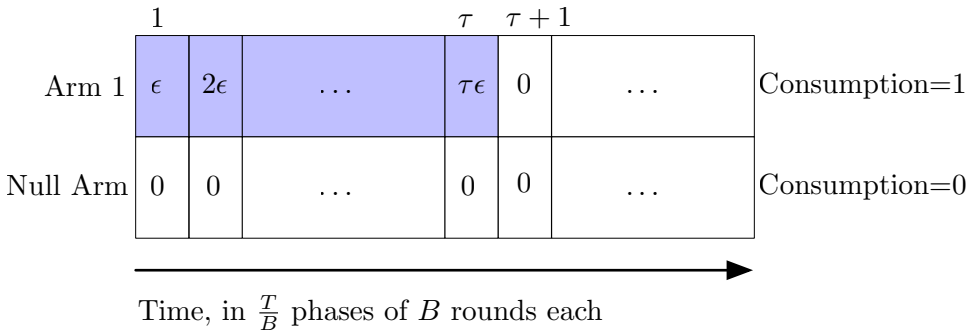

Figure 7.3: The lower-bounding construction for the $\Omega(\log T)$ lower bound.

The lower bound holds for a wide range of budgets B, as expressed by the following lemma:

Lemma 20. *Fix an absolute constant $\alpha \in (0,1)$, and budget B in the interval $[\Omega(\log^3 T), O(T^{1-\alpha})]$. Then any algorithm suffers $\mathrm{OPT}_{FD}/\mathbb{E}[\mathrm{REW}] \geq \Omega(\log T)$ for some problem instance in Construction 3.*

In the rest of this subsection we prove Lemma 20. Fix any randomized algorithm $\mathcal{A}$. As before in this sub-section we are interested in the ratio $\mathrm{OPT}_{\mathtt{FD}}/\mathbb{E}[\mathrm{REW}(\mathcal{A})]$. We argue that it has the claimed competitive ratio on at least one instance $\mathcal{I}_\tau$ in the construction 3. The proof proceeds in two parts. We first argue about the solution structure of the optimal distribution for the construction 3 (we prove this formally in Lemma 21). Next we characterize the expected number of times arm A_1 is played if $\mathcal{A}$ optimal algorithm in each of the phases. Combining the two we get Lemma 20.

Structure of the optimal solution. Define $\text{OPT}_{\text{LP}}(\overline{\mathbf{M}}_{\tau^*}, B, \tau^*)$ to be the optimal value of LP 7.8 on the instance $\mathcal{I}_\tau$. Then we have the following Lemma.

Lemma 21. *For a given instance $\mathcal{I}_\tau$ we have the following.*

1. *The optimal stopping time $\tau^* = B\tau$.*

2. $\text{OPT}_{LP}(\overline{\mathbf{M}}_{\tau^*}, B, \tau^*) \leq \frac{1}{\tau}\sum_{t\in[B\tau]} \mathcal{P}(t)\epsilon$.

Proof. Note that (2) follows from (1). Indeed, given that the stopping time is $B\tau$, the optimal solution is to set $X(1) = \frac{1}{\tau}$ and $X(0) = 1 - \frac{1}{\tau}$ thus obtaining a total reward of $\frac{1}{\tau}\sum_{t\in[B\tau]} \mathcal{P}(t)\epsilon$. Thus it remains to prove (1).

First it is easy to prove that $\tau^* \leq B\tau$. Since there are no rewards after time-step τ^*, we have

$$\forall t' > 0 \qquad \text{OPT}_{\text{LP}}(\overline{\mathbf{M}}_{\tau*+t'}, B, \tau^* + t') = \frac{1}{\tau + t'}\sum_{t\in[\tau^*]} \mathcal{P}(t)\epsilon < \frac{1}{\tau}\sum_{t\in[B\tau]} \mathcal{P}(t)\epsilon.$$

Now we will argue that the optimal stopping time cannot be strictly lesser than τ^*. To do so, first we argue that for two stopping times $t_1 < t_2$ within the same phase, the maximum objective is achieved for the stopping time t_2. This implies that the optimal stopping time has to be the last time step of some phase.

Consider times $t_1 < t_2$ such that $\mathcal{P}(t_1) = \mathcal{P}(t_2) = \tau$. Then we want to claim that

$$\frac{B}{t_1}\left(\sum_{t\in[t_1]} \mathcal{P}(t)\epsilon\right) \leq \frac{B}{t_2}\left(\sum_{t\in[t_2]} \mathcal{P}(t)\epsilon\right).$$

For contradiction assume the inequality does not hold. Then we have the following.

$$\sum_{t\in[t_1]} \mathcal{P}(t) > \frac{t_1}{t_2}\left(\sum_{t\in[t_2]} \mathcal{P}(t)\right).$$

Note that $\sum_{t\in[B(\tau-1)]}\mathcal{P}(t)=\sum_{t'\in[\tau]}Bt'=\frac{B(\tau-1)\tau}{2}$. Thus we have

$$\sum_{t\in[t_1]}\mathcal{P}(t)=\frac{B(\tau-1)\tau}{2}+(t_1-B(\tau-1))\tau,$$
$$\sum_{t\in[t_2]}\mathcal{P}(t)=\frac{B(\tau-1)\tau}{2}+(t_2-B(\tau-1))\tau.$$

Therefore we have,

$$\frac{B(\tau-1)\tau}{2}+(t_1-B(\tau-1))\tau>\frac{t_1B(\tau-1)\tau}{2t_2}+(t_2-B(\tau-1))\tau.$$

Further re-arranging, $\frac{B(\tau-1)}{2}>t_2$. This is a contradiction since t_2 is in phase τ, so $t_2\geq B(\tau-1)$.

Next we argue that the optimal value is achieved when the stopping time is in the last *non-zero rewards phase*. Consider two phases $\tau_1<\tau_2$. Thus the ending times are $B\tau_1$ and $B\tau_2$. To prove that the optimal value increases by stopping at $B\tau_2$, as opposed to $B\tau_1$, we want to show that

$$\frac{1}{\tau_1}\sum_{t\in[\tau_1]}Bt\epsilon\leq\frac{1}{\tau_2}\sum_{t\in[\tau_2]}Bt\epsilon.$$

As before assume for a contradiction that this doesn't hold. Then re-arranging we get, $\frac{\tau_1(\tau_1+1)}{2}>\frac{\tau_1(\tau_2+1)}{2}$, which implies $\tau_1>\tau_2$, contradiction. We conlude that the stopping time is $\tau^*=B\tau$. □

Expected behavior of the optimal algorithm. Consider any randomized algorithm $\mathcal{A}$. The performance of $\mathcal{A}$ is then as follows.

$$\frac{\mathrm{OPT}_{\mathtt{LP}}^{[T]}}{\mathbb{E}[\mathrm{REW}(\mathcal{A})]}\geq\max_{1\leq\tau\leq T/B}\frac{B\tau\cdot\mathrm{OPT}_{\mathtt{LP}}(\overline{\mathbf{M}}_{B\tau},B,B\tau)}{\mathbb{E}[\mathrm{REW}(\mathcal{A})]}.\tag{7.71}$$

Consider two consecutive instances $\mathcal{I}_\tau$ and $\mathcal{I}_{\tau+1}$. The outcome matrix in the phases $1, 2, \ldots, \tau$ look identical in both these instances. Thus the expected number of times arm A_1 is chosen by algorithm $\mathcal{A}$ in phases $1, 2, \ldots, \tau$ is identical. Let α_τ denote the expected number of times $\mathcal{A}$ plays arm A_1 in phase τ on instances $\mathcal{I}_1, \mathcal{I}_2, , \ldots, \mathcal{I}_\tau$. The expected number of times arm A_1 is played in phase τ on instances $\mathcal{I}_{\tau+1}, \mathcal{I}_{\tau+2}, \ldots, \mathcal{I}_{T/B}$ is set arbitrarily since this term does not appear in the objective function (*i.e.,* $\mathbb{E}[\text{REW}(\mathcal{A})]$). Thus the maximum of the ratios in Eq. (7.71) occurs when they are equal for all values τ. Therefore we want for every $1 \leq \tau \leq T/B$,

$$\frac{\frac{1}{\tau}\sum_{j\in[\tau]} Bj\epsilon}{\sum_{j\in[\tau]} j\epsilon\alpha_j} = \frac{\frac{1}{\tau+1}\sum_{j\in[\tau+1]} Bj\epsilon}{\sum_{j\in[\tau+1]} j\epsilon\alpha_j}. \tag{7.72}$$

Also note that $\alpha_1 + \alpha_2 + \ldots + \alpha_{T/B} \leq B$, since the total budget is B and the consumption is 1 in every round for arm A_1 and 0 for arm A_0. We will prove that these lead to the following recurrence for the maximizing values of α_j.

$$\forall j \geq 2 \qquad \alpha_j = \frac{1}{2j}\alpha_1. \tag{7.73}$$

We will prove the recurrence Eq. (7.73) via induction. The base case is when $j = 2$. Plugging in $\tau = 1$ in Eq. (7.72) we have

$$\frac{1}{\alpha_1} = \frac{3/2}{\alpha_1 + 2\alpha_2}.$$

This implies that $\alpha_2 = \frac{1}{4}\alpha_1$. Therefore we are done. Now consider the inductive case; let all α up to α_τ satisfy the recurrence Eq. (7.73). Consider the instance $\mathcal{I}_\tau$ and $\mathcal{I}_{\tau+1}$. From Eq. (7.72) we have,

$$\frac{\frac{1}{\tau}\sum_{j\in[\tau]} jB}{\alpha_1 + \sum_{j=2}^{\tau} \alpha_1/2} = \frac{\frac{1}{\tau+1}\sum_{j\in[\tau+1]} jB}{\alpha_1 + \sum_{j=2}^{\tau} \alpha_1/2 + (\tau+1)\alpha_{\tau+1}}.$$

Rearranging, we get that $\alpha_{\tau+1} = \frac{1}{2(\tau+1)}\alpha_1$. This completes the inductive step.

Using the fact that $\alpha_1(1 + 1/4 + 1/6 + \ldots + B/2T) \leq B$ we have that $\alpha_1 \leq \frac{B}{2H(T/B)}$ where $H(n)$ denotes the n^{th} Harmonic number. The value of Eq. (7.71) for $\tau = 1$ is the same at all other τ and has a value of $\frac{B}{\alpha_1} \geq 2H\left(\frac{T}{B}\right) \geq \Omega\left(\log \frac{T}{B}\right) \geq \Omega\left(\log T\right)$. The second last inequality uses the well-known fact that $\zeta + \log n \leq H(n) \leq \zeta + \log(n+1)$ for an absolute constant ζ and the last inequality uses the fact that $B \in \left[\Omega(\log^3 T), O(T^{1-\alpha})\right]$.

Therefore we have that for every instance $j \in \left[\frac{T}{B}\right]$,

$$\frac{\mathrm{OPT}_{\mathtt{LP}}^{[T]}(\mathcal{I}_j)}{\mathbb{E}[\mathrm{REW}(\mathcal{A}, \mathcal{I}_j)]} \geq \Omega(\log T).$$

Thus combining this with Lemma 19 we obtain

$$\frac{\mathrm{OPT}_{\mathtt{FD}}}{\mathbb{E}[\mathrm{REW}(\mathcal{A})]} \geq \Omega(\log T) - O\left(\frac{\log^{1.5} T}{\sqrt{B}}\right).$$

Since $B \geq \Omega\left(\log^3 T\right)$, we get that $\frac{\mathrm{OPT}_{\mathtt{FD}}}{\mathbb{E}[\mathrm{REW}(\mathcal{A})]} \geq \Omega(\log T)$.

7.8.3 Best dynamic policy: proof of Theorem 9(c)

Consider the following construction of the lower-bound example.

Construction 4. *There is one resource with budget B, and two arms, denoted A_0, A_1. Arm A_0 is the 'null arm' that has zero reward and zero consumption. The consumption of arm A_1 is 1 in all rounds. The rewards of A_1 are defined as follows. We partition the time into $\frac{T}{B}$ phases of duration B each (for simplicity, assume that B divides T). We consider $\frac{T}{B}$ problem instances; for each instance $\mathcal{I}_\tau$, $\tau \in [T/B]$ arm A_1 has 0 reward in all phases except phase τ; in phase τ it has a reward of 1 in each round.*

Lemma 22. *Consider Construction 4 with any given time horizon $T \geq 2$ and budget $B \leq \sqrt{T}$. Let* ALG *be an arbitrary randomized algorithm for BwK. Then for one of the problem instances,*

$$\mathrm{OPT}_{DP} / \mathbb{E}[\mathrm{REW}] \geq T/B^2. \tag{7.74}$$

Proof. Let $n = \frac{T}{B}$ be the number of phases in Construction 4. Let ALG be a deterministic algorithm. Let REW denote its total reward, and let $\mathbb{E}_\tau[\cdot]$ denote the expectation over the uniform-at-random choice of the problem instance $\mathcal{I}_\tau$. We claim that

$$\mathrm{OPT}_{\mathtt{DP}} / \mathbb{E}_\tau[\mathrm{REW}] \geq T/B^2. \tag{7.75}$$

Assume that ALG maximizes $\mathbb{E}_\tau[\mathrm{REW}]$ (over deterministic algorithms). Then it satisfies the following:

- Within each phase, if ALG ever chooses to play arm A_1, it does so in the first round of the phase. If it receives a reward of 1 in this round, it plays A_1 for the rest of the phase. Else, it never plays A_1 for the rest of this phase.

For each $\tau \in [n]$, let α_τ denote the number of times ALG chooses arm A_1 in phase τ in problem instance $\mathcal{I}_\tau$. The expected reward of ALG over the uniform-at-random choice of the problem instance $\mathcal{I}_\tau$ is $\mathbb{E}[\mathrm{REW}] = \frac{1}{n} \sum_{i \in [n]} \alpha_i$. Let $(\alpha_{\pi(1)}, \alpha_{\pi(2)}, \ \ldots \ , \alpha_{\pi(k)})$ be the subsequence of $(\alpha_1, \ \ldots \ , \alpha_n)$ which contains all elements with non-zero values.

The key observation is as follows. The problem instances $\mathcal{I}_{\pi(\tau-1)}$ and $\mathcal{I}_{\pi(\tau)}$ are identical until phase $\pi(\tau-1)-1$. Since the feedback received by ALG until the first

time it chooses arm A_1 in phase $\pi(\tau-1)$ is identical, it follows that $\alpha_{\pi(\tau-1)}-\alpha_{\pi(\tau)} = 1$. Therefore,

$$\sum_{i\in[n]} \alpha_i = \sum_{i\in[k]} \alpha_{\pi(i)} = k \cdot \alpha_{\pi(1)} - \frac{k(k-1)}{2}.$$

Noting that $\alpha_1 \leq B$ and $k \leq \min(B, n) = B$, we have:

$$\mathbb{E}[\text{REW}] \leq \tfrac{1}{n} \sum_{i\in[n]} \alpha_i < B^2/n = B^3/T.$$

Since $\text{OPT}_{\texttt{DP}} = B$ for every problem instance $\mathcal{I}_\tau$, Eq. (7.75) holds for ALG, and therefore for any other deterministic algorithm. By Yao's lemma (Motwani and Raghavan [148]), for every randomized algorithm ALG there exists a problem instance $\mathcal{I}_\tau$ such that (7.74) holds. □

We now use the same construction to prove Theorem 10.

Proof sketch of Theorem 10. We prove the Theorem by contradiction. Let $B \leq \sqrt{T}$. For contradiction, consider an algorithm ALG for cBwK on a policy set Π such that $\text{OPT}_{\texttt{FD}}(\Pi)/\mathbb{E}[\text{REW}(\text{ALG})] < T/B^2$. We will now use ALG to construct an algorithm $\mathcal{A}$ for the Construction 4 such that $\text{OPT}_{\texttt{DP}}/\mathbb{E}[\text{REW}(\mathcal{A})] < T/B^2$ for every instance. This contradicts Lemma 22.

Consider a policy set Π with $|n|$ policies. Every policy $\pi \in \Pi$ maps contexts in the range $[1, T]$ to the action set $\{A_1, A_0\}$. In particular, a policy $\pi_\tau \in \Pi$ maps contexts that lie in the range $[B*(\tau-1)+1, B*\tau]$ to arm A_1 and all other contexts to A_0. $\mathcal{A}$ invokes ALG as a sub-routine with the policy set Π. At each time-step t, $\mathcal{A}$ gives the context $x_t = t$ to ALG and plays the arm chosen by ALG.

For an instance $\mathcal{I}_\tau$ in Construction 4, $\text{OPT}_{\texttt{FD}}(\Pi)$ is the total reward obtained by choosing the action given by π_τ in all time-steps. The total reward obtained is B,

which equals $\text{OPT}_{\texttt{DP}}(\mathcal{I}_\tau)$. Therefore, $\text{OPT}_{\texttt{FD}}(\Pi)/\mathbb{E}[\text{REW}(\text{ALG})] < T/B^2$ implies we have $\text{OPT}_{\texttt{DP}}/\mathbb{E}[\text{REW}(\mathcal{A})] < T/B^2$ for every instance $\mathcal{I}_\tau$, which is a contradiction.

□

7.8.4 Best fixed arm: proof of Theorem 9(d)

We use the following construction for the lower-bound.

Construction 5. *There is one resource with budget B, and K arms denoted by $A_1, A_2, \ldots, A_K$. Arm A_K is the 'null arm' that has zero reward and zero consumption. There are K instances in the family. In instance $\mathcal{I}_j$, all arms $A_{j'}$ where $j' > j$ have 0 reward and 0 consumption in all time-steps. Consider an instance $\mathcal{I}_j$ for some $j \in [K-1]$ and an arm $j' \leq j$. Arm $A_{j'}$ has a reward of $\frac{1}{K^{K-j'}}$ and consumption of 1 in all time-steps in phase j' and has a reward of 0 and consumption of 0 in every other time-step. Thus the rewards and consumption are bounded in the interval $[0,1]$ for every arm and every time-step in all instances in this family.*

Lemma 23. *Let $T \geq 2$, $2 \leq B \leq T$, $K \geq 3$ be given parameters of the AdversarialBwK problem. We show that there exists a family of instances with $d = 1$ shared resource such that for every randomized algorithm $\mathcal{A}$ we have $\frac{\text{OPT}_{FA}}{\mathbb{E}[\text{REW}(\mathcal{A})]}$ is at least $\Omega(K)$ on one of these instances.*

Proof. First note that the best fixed arm in instance $\mathcal{I}_j$ is to pick arm A_j which yields a total reward of $\frac{B}{K^{K-j}}$.

Consider a randomized algorithm $\mathcal{A}$. Observe that in the first j phases, the instances $\mathcal{I}_{j-1}$ and $\mathcal{I}_j$ have identical outcome matrices. Thus the expected number

of times any arm A_k for $k \in [K]$ is chosen in phases $\{1, 2, \ldots, j\}$ should be the same in both the instances. Let α_k denote the expected number of times arm k is played by $\mathcal{A}$ in phase k on instances $\mathcal{I}_k, \mathcal{I}_{k+1}, \ldots, \mathcal{I}_{K-1}$ [13]. Moreover we have that $\alpha_1 + \alpha_2, \ldots, \alpha_{K-1} \leq B$.

To show the lower-bound we want to minimize the competitive ratio on every instance for all possible values of $\alpha_1, \alpha_2, \ldots, \alpha_{K-1}$. For ease of notation denote $r_j := \frac{1}{K^{K-j}}$. Let $\alpha_{\mathcal{B}}$ denote the set of values to $\{\alpha_k\}_{k\in[K-1]}$ such that $\sum_{k\in[K-1]} \alpha_k \leq B$. Thus,

$$\frac{\text{OPT}_{\text{FA}}}{\mathbb{E}[\text{REW}(\mathcal{A})]} \geq \min_{\alpha_{\mathcal{B}}} \frac{r_k B}{\sum_{j\in[k]} r_j \alpha_j}. \tag{7.76}$$

The ratio is minimized when all ratios in Eq. (7.76) are equal. We will show via induction that this yields the following recurrence,

$$\forall k \geq 2 \qquad \alpha_k = \left(1 - \frac{r_{k-1}}{r_k}\right) \alpha_1. \tag{7.77}$$

Combining this with the condition that $\sum_{k\in[K-1]} \alpha_k \leq B$, this yields the condition $\alpha_1 \leq \frac{B}{K - \frac{1}{K}}$. Moreover the minimizing value in Eq. (7.76) is $K - \frac{1}{K}$ which proves Lemma 23.

We will now prove the recurrence Eq. (7.77). Consider the base case with $k = 2$. Equalizing the first two terms in Eq. (7.76) we get

$$\frac{r_1 B}{r_1 \alpha_1} = \frac{r_2 B}{r_1 \alpha_1 + r_2 \alpha_2}.$$

Re-arranging we obtain that $\alpha_2 = \left(1 - \frac{r_1}{r_2}\right) \alpha_1$. We will now prove the inductive case. Let the recurrence be true for all $1 \leq k \leq k'$. Consider the case $k = k' + 1$.

[13] This has to be the same in all instances since the outcome matrix is identical until phase k in all these instances

Setting the k' and $k'+1$ ratios in Eq. (7.76) equal, we obtain

$$\frac{r_{k'}B}{\sum_{j\in[k']} r_j\alpha_j} = \frac{r_{k'+1}B}{\sum_{j\in[k'+1]} r_j\alpha_j}. \tag{7.78}$$

Moreover from the inductive hypothesis we have $\alpha_j = \left(1 - \frac{r_{j-1}}{r_j}\right)\alpha_1$ for every $j \le k'$. Thus we have

$$\sum_{j\in[k']} r_j\alpha_j = r_{k'}\alpha_1$$

$$\sum_{j\in[k'+1]} r_j\alpha_j = r_{k'}\alpha_1 + r_{k'+1}\alpha_{k'+1}.$$

Plugging this back in Eq. (7.78) we get

$$\frac{r_{k'}B}{r_{k'}\alpha_1} = \frac{r_{k'+1}B}{r_{k'}\alpha_1 + r_{k'+1}\alpha_{k'+1}}.$$

Rearranging we get $\alpha_{k'+1} = \left(1 - \frac{r'_k}{r_{k'+1}}\right)\alpha_1$. This completes the induction. □

7.9 Extensions

We obtain several extensions which highlight the modularity of `LagrangeBwK`: we apply Theorem 6 and Theorem 7 with appropriately chosen primal algorithm ALG_1, and immediately obtain strong performance guarantees.[14] We tackle four well-known scenarios:

- *full feedback* (*e.g.,* Arora et al. [21], Freund and Schapire [88], Littlestone and Warmuth [130]): in each round, the algorithm chooses an action and observes

[14]For these theorems to hold, ALG_1 needs to satisfy regret bound (7.4) only against adaptive adversaries that arise in the repeated Lagrange game in the corresponding extension, not against *arbitrary* adaptive adversaries.

the outcomes of all possible actions; this is a classic scenario in online machine learning.

- *combinatorial semi-bandits* (*e.g.,* Audibert et al. [23], György et al. [101], Kale et al. [113]): actions are feasible subsets of "atoms". The atoms in the chosen action have individual outcomes that are observed and add up to the action's total outcome. Typical motivating example are subsets/lists of news articles, ads, or web search results.

- *contextual bandits with policy sets* (*e.g.,* Agarwal et al. [5], Dudík et al. [80], Langford and Zhang [127]): before each round, a *context* is observed, and the algorithm competes against the best policy (mapping from context to actions) in a given policy class. In a typical application scenario, the context includes known features of the current user.

- *bandit convex optimization* (starting from Flaxman et al. [86], Kleinberg [120], with recent advances Bubeck et al. [44, 45], Hazan and Levy [105]). Here the set of actions is a convex set $\mathcal{X} \subset \mathbb{R}^K$. For each round t, the adversary chooses a concave function $f_t : \mathcal{X} \to [0,1]$ such that the reward for chosen action $\mathbf{x} \in \mathcal{X}$ is $f_t(\mathbf{x})$.

Formalities. To simplify the statements, we make the following assumptions without further mention:

- The dual algorithm, ALG_2, is always Hedge, with the associated regret bound from 7.6. For high-probability regret bounds, $\delta = \frac{1}{T}$ is a fixed and known

failure probability parameter.

- For Stochastic BwK, one resource is the dummy resource (with consumption $\frac{B}{T}$ for each arm). Algorithm `LagrangeBwK` is run with parameters $B_0 = B$ and $T_0 = T$.

- For Adversarial BwK, one of the arms is a *null arm* that has zero reward and zero resource consumption. Algorithm 6 is run with scale parameter $\kappa \geq d+1$ and range $[g_{\min}, g_{\max}] = [1, T]$.

A typical corollary. All our corollaries have the following shape, for some regret term `reg`:

(C1) In the stochastic version, algorithm `LagrangeBwK` achieves, with probability at least $1 - \frac{1}{T}$,

$$\text{OPT}_{\texttt{DP}} - \text{REW} \leq O\left(\tfrac{T}{B} \cdot \texttt{reg}\right).$$

(C2) In the adversarial version, Algorithm 6 achieves

$$\frac{\mathbb{E}[\text{REW}]}{\text{OPT}_{\texttt{FD}}} \geq \frac{1 - O(\texttt{reg})\left(\frac{1}{\text{OPT}_{\texttt{FD}}} + \frac{1}{\kappa B}\right)}{\kappa^2 \ \lceil \log_\kappa T \rceil}.$$

Corollaries similar to (C2) can be achieved for Algorithm 7, too; we omit them for ease of exposition.

7.9.1 BwK with full feedback

In the full-feedback version of BwK, the entire outcome matrix $\mathbf{M}_t$ is revealed to the algorithm after each round t. Accordingly, we can use Hedge as the primal

algorithm ALG_1. The effect is, essentially, that the dependence on K, the number of arms, in the regret term becomes logarithmic rather than $\sqrt{K}$.

Corollary 1. *Consider BwK with full feedback. Using Hedge as the primal algorithm, we obtain corollaries (C1) and (C2) with regret term* $\texttt{reg} = \sqrt{T \ln(dKT)}$.

Adversarial BwK with full feedback have not been studied before. For the stochastic version, the regret bound is unsurprising: one expects to obtain a similar improvement with each of the three other algorithms in the prior work on Stochastic BwK by tracing the "confidence terms" through the analysis. The significance here is that we obtain this result as an immediate corollary.

7.9.2 Combinatorial Semi-Bandits with Knapsacks

Following Sankararaman and Slivkins [163], we consider *Combinatorial Semi-BwK*, a common generalization of BwK and *combinatorial semi-bandits* (*e.g.,* Audibert et al. [23], György et al. [101], Kale et al. [113]). In this problem, actions correspond to subsets of some finite ground set Ω of size n, whose elements are called *atoms*. There is a fixed family $\mathcal{F} \subset 2^\Omega$ of feasible actions. For each round t, there is an outcome vector $\mathbf{o}_{t,e} \in [0, \frac{1}{n}]^{d+1}$ for each round atom $e \in \Omega$, with the same semantics as the actions' outcome vectors. If an action $S \subset \Omega$ is chosen, the outcome vectors $\mathbf{o}_{t,e}$ are observed for all atoms $e \in S$, and the action's outcome is the sum $\mathbf{M}_t(S) = \sum_{e \in S} \mathbf{o}_{t,e} \in [0,1]^d$. In the adversarial case, all outcome vectors $\mathbf{o}_{t,e}$, $t \in [T]$, $e \in \Omega$ are chosen by an adversary arbitrarily before round 1. In the stochastic case, the atomic outcome matrix $(\mathbf{o}_{t,e} : \ e \in \Omega)$ is drawn independently in

each round t from some fixed distribution. Combinatorial semi-bandits, as studied previously, is a special case with no resource constraints ($d = 0$).

Typical motivating examples involve ad/content personalization scenarios. Atoms can correspond to items news articles, ads, or web search results, and actions are subsets that satisfy some constraints on quantity, relevance, or diversity of items. One can also model ranked lists of atoms: then atoms are rank-item pairs, and each feasible action $S \subset \Omega$ satisfies a constraint that each rank between 1 and $|S|$ is present in exactly one chosen rank-item pair.

A naive application of our main results suffers from regret terms that are proportional to $\sqrt{|\mathcal{F}|}$, which may be exponential in the number of atoms n. Instead, the work on combinatorial semi-bandits features regret bounds that scale polynomially in n. This is what we achieve, too. We use an algorithm from Neu and Bartók [152] which solves combinatorial semi-bandits in the absence of resource constraints. This algorithm satisfies a high-probability regret bound (7.4) against an adaptive adversary, with $R_\delta(T) = O(\sqrt{nT \log(1/\delta)})$. [15]

Corollary 2. *Consider Combinatorial Semi-BwK with n atoms. Using the algorithm from Neu and Bartók [152] as the primal algorithm, we obtain corollaries (C1) and (C2) with regret term $\texttt{reg} = \sqrt{nT \log T}$.*

The adversarial version of Combinatorial Semi-BwK has not been studied before.

[15] Prior work (Neu and Bartók [152], Sankararaman and Slivkins [163]) posits that atoms' per-round rewards/consumptions lie in the range $[0, 1]$, rather than $[0, \frac{1}{n}]$, so their stated regret bounds should be recaled accordingly.

The stochastic version has been studied in Sankararaman and Slivkins [163] when the action set is a matroid, achieving regret

$$\tilde{O}\left(\mathrm{OPT}_{\texttt{DP}}\ \sqrt{n/B}+\sqrt{T/n}+\sqrt{\mathrm{OPT}_{\texttt{DP}}}\right).$$

This regret bound becomes $\tilde{O}(\sqrt{nT})$ in the regime when B and $\mathrm{OPT}_{\texttt{DP}}$ are $\Omega(T)$ (see Footnote 15). We achieve the same regret bound for this regime, without the matroid assumption and without any extra work. However, the regret bound in Sankararaman and Slivkins [163] can be substantially better than ours when $\mathrm{OPT}_{\texttt{DP}} \ll T$.

7.9.3 Contextual Bandits with Knapsacks

Following Agrawal et al. [14], Badanidiyuru et al. [31], we consider *Contextual Bandits with Knapsacks* (*cBwK*), a common generalization of BwK and *contextual bandits with policy sets* (*e.g.,* Agarwal et al. [5], Dudík et al. [80], Langford and Zhang [127]). The only change in the protocol, compared to BwK, is that in the beginning of each round t a context $x_t \in \mathcal{X}$ arrives and is observed by the algorithm before it chooses an action. The context set $\mathcal{X}$ is arbitrary and known. In the adversarial version (*Adversarial cBwK*) both x_t and the outcome matrix $\mathbf{M}_t$ is chosen by an adversary. In the stochastic version (*Stochastic cBwK*) the pair $(x_t, \mathbf{M}_t)$ is chosen independently from some fixed but unknown distribution over such pairs.

In cBwK one is also given a finite set Π of *policies*: deterministic mappings from contexts to actions.[16] Essentially, the algorithm competes with the best course

[16] W.l.o.g. assume that Π contains all *constant policies*, *i.e.,* all policies that always evaluate to

of action restricted to these policies. For a formal definition, let us interpret cBwK as a BwK problem with action set Π, denote this problem as $\texttt{BwK}(\Pi)$. In other words, actions in $\texttt{BwK}(\Pi)$ are policies in cBwK. An algorithm for $\texttt{BwK}(\Pi)$ is oblivious to context arrivals. It chooses a policy $\pi_t \in \Pi$ in each round t, and receives an outcome for this policy: namely, the outcome for action $\pi(x_t)$. We are interested in the usual benchmarks for this problem, the best algorithm $\text{OPT}_{\texttt{DP}}$ and the best fixed distribution $\text{OPT}_{\texttt{FD}}$ (where both benchmarks are constrained to use policies in Π); denote them $\text{OPT}_{\texttt{DP}}(\Pi)$ and $\text{OPT}_{\texttt{FD}}(\Pi)$, respectively.

Without budget constraints (*i.e.,* with $B = T$), this is exactly contextual bandits with policy set Π. Both benchmarks then reduce to the standard benchmark of the best fixed policy.

Background: algorithm EXP4.P. We use EXP4.P (Beygelzimer et al. [40]), an algorithm for the contextual version of adversarial online learning with bandit feedback. The algorithm operates according to the protocol in Figure 7.4.

We are interested in regret bounds for EXP4.P relative to the best fixed policy:

$$\text{OPT}_\Pi = \max_{\pi\in\Pi} \sum_{t\in[T]} f_t(\pi(x_t)).$$

For each round t, the pair $(x_t, \mathbf{g}_t)$ induces a payoff vector $\mathbf{f}_t \in [b_{\min}, b_{\max}]^\Pi$ on policies:

$$f_t(\pi) = g_t(\pi(x_t)) \qquad \forall \pi \in \Pi.$$

Theorem 11 (Beygelzimer et al. [40])**.** *Fix failure probability $\delta > 0$, policy set Π, and payoff range $[b_{\min}, b_{\max}]$. Then algorithm EXP4.P (appropriately tuned) satisfies*

the same action.

Given: action set $[K]$, context set $\mathcal{X}$, policy set Π, payoff range $[b_{\min}, b_{\max}]$.

In each round $t \in [T]$,

1. the adversary chooses a context $x_t \in \mathcal{X}$ and a payoff vector $\mathbf{g}_t \in [b_{\min}, b_{\max}]^K$;
2. the algorithm chooses a distribution $\mathbf{p}_t$ over Π (without seeing x_t);
3. a policy $\pi_t \in \Pi$ is drawn independently from $\mathbf{p}_t$;
4. algorithm's chosen action is defined as $a_t = \pi_t(x_t) \in [K]$;
5. payoff $g_t(a_t)$ is received by the algorithm.

Figure 7.4: Adversarial contextual bandits

the following regret bound:

$$\Pr\left[\ \mathrm{OPT}_\Pi \ - \ \textstyle\sum_{t\in[T]} f_t(\pi_t) \le (b_{\max} - b_{\min})\, R_\delta(T)\ \right] \ge 1-\delta, \tag{7.79}$$

with regret term $R_\delta(T) = O\left(\sqrt{\tau K \log(KT\,|\Pi|/\delta)}\right)$.

Our solution for cBwK. We solve cBwK by reducing it to `BwK`(Π), and treating it as a BwK problem. A naive solution simply posits $|\Pi|$ arms and directly applies the machinery developed earlier in this paper. This results in $\sqrt{|\Pi|}$ dependence in regret bounds, which is unsatisfactory, as the policy set may be very large. Instead, we use EXP4.P as the primal algorithm (ALG_1). We interpret EXP4.P as an algorithm for (non-contextual) adversarial online learning, with action set Π. It is easy to see that Theorem 11 provides regret bound (7.4) under this interpretation. Therefore,

we obtain the following:

Corollary 3. *Consider contextual bandits with knapsacks, with policy set* Π. *Using EXP4.P as the primal algorithm, we obtain corollaries (C1) and (C2) with regret term* $\texttt{reg} = \sqrt{TK \ln(dKT\,|\Pi|)}$. *The benchmarks are* $\mathrm{OPT}_{DP} = \mathrm{OPT}_{DP}(\Pi)$ *and* $\mathrm{OPT}_{FD} = \mathrm{OPT}_{FD}(\Pi)$.

Adversarial cBwK has not been studied before. The regret bound for the adversarial case is meaningful only if $B > \sqrt{T}$. This is essentially inevitable in light of the lower bound in Theorem 10.

Stochastic cBwK has been studied in Agrawal et al. [14], Badanidiyuru et al. [31], achieving regret bound

$$O(\texttt{reg})(1 + \mathrm{OPT}_{\texttt{DP}}(\Pi)/B), \tag{7.80}$$

where the `reg` term is the same as in Corollary 3. Whereas the regret bound from Corollary 3 is $O(\texttt{reg} \cdot T/B)$. Note that we match (7.80) in the regime $\mathrm{OPT}_{\texttt{DP}}(\Pi) > \Omega(T)$. Our regret bound is optimal, up to logarithmic factors, in the regime $B > \Omega(T)$.[17]

Discussion. Our algorithms are slow, as the per-round running time of EXP4.P is proportional to $|\Pi|$. The state-of-art approach to computational efficiency in contextual bandits is *oracle-efficient algorithms*, which make only a small number of calls to an oracle that finds the best policy in Π for a given data set. In particular, prior work for Stochastic cBwK (Agrawal et al. [14]) obtains an oracle-efficient algorithm

[17] This is due to the min $\left(T,\, \Omega(\sqrt{KT \log(|\Pi|)/\log(K)})\right)$ regret bound, which holds for contextual bandits Agarwal et al. [4].

with regret bound as in (7.80). To obtain oracle-efficient algorithms for cBwK in our framework, both for the stochastic and adversarial versions, it suffices to replace EXP4.P with an oracle-efficient algorithm for adversarial contextual bandits that obtains regret bound (7.79), possibly with a larger regret term R_δ. Such algorithms *almost* exist: a recent breakthrough (*e.g.,* Rakhlin and Sridharan [156], Syrgkanis et al. [173, 174]) obtains algorithms with similar regret bounds, but only for expected regret.

7.9.4 Bandit Convex Optimization with Knapsacks

We consider *Bandit Convex Optimization with Knapsacks* (*BCOwK*), a common generalization of BwK and *bandit convex optimization*. We define BCOwK as a version of BwK, where the action set $\mathcal{X}$ is a convex subset of $\mathbb{R}^K$. For each round t, there is a concave function $f_t : \mathcal{X} \to [0,1]$ and convex functions $g_{t,i} : \mathcal{X} \to [0,1]$, for each resource i, so that the reward for choosing action $\mathbf{x} \in \mathcal{X}$ in this round is $f_t(\mathbf{x})$ and consumption of each resource i is $g_{t,i}(\mathbf{x})$. In the stochastic version, the tuple of functions $(f_t; g_{t,1}, \ldots, g_{t,d})$ is sampled independently in each round t from some fixed distribution (which is not known to the algorithm). In the adversarial version, all these tuples are chosen by an adversary before the first round.

Neither stochastic nor adversarial version of BCOwK have been studied in prior work (but see the discussion of constrained online convex optimization in Section 7.2). Bandit convex optimization, as studied previously, is a special case with no resource constraints ($d = 0$).

The primal algorithm ALG_1 in `LagrangeBwK` faces an instance of BCO (with

an adaptive adversary). This is because the Lagrange function (7.10) is a concave function of the action, as sum of concave functions. For our primal algorithm, we use a recent breakthrough on BCO due to Bubeck et al. [45]. This algorithm satisfies the high-probability regret bound (7.4) against an adaptive adversary, with regret term

$$R_\delta(T) = O(K^{9.5}\log^7(T)\sqrt{T\log(1/\delta)}).$$

We assume the existence of a *null arm*: a point $\mathbf{x} \in \mathcal{X}$ such that $f_t(\mathbf{x}) = g_{t,i}(\mathbf{x}) = 0$ for each resource i except the "dummy resource". (Recall that we posit the "dummy resource" – a resource whose consumption is B/T for each arm – for the stochastic version.) Unlike elsewhere in this paper, this assumption is not without loss of generality: indeed, the null arm should be "embedded" into $\mathcal{X}$ without breaking the convexity/concavity properties. Moreover, we assume that the null arm lies in the interior of $\mathcal{X}$.

Corollary 4. *Consider BCOwK for a given convex set $\mathcal{X} \subset \mathbb{R}^K$. Using the algorithm from Bubeck et al. [45] as the primal algorithm, we obtain corollaries (C1) and (C2) with regret term* $\texttt{reg} = K^{9.5}\log^{7.5}(T)\sqrt{T}$.

Remark 11. *`LagrangeBwK` framework extends to infinite action sets: everything carries over, as long as 7.11 holds. (Essentially, we never take union bounds over actions, and we can replace* max *and sums over actions with* sup *and integrals.) For BCOwK, 7.11 is a statement about constrained convex optimization programs. According to* Slater's condition *(see Eq. (5.27) in Boyd and Vandenberghe [41]), it suffices to have a point $\mathbf{x}$ in the interior of $\mathcal{X}$ such that $g_{t,i}(\mathbf{x}) < B/T$ for each*

resource $i \in [d]$ other than the dummy resource (or any other resource whose consumption is the same in all rounds). One such point is the null arm.

Chapter 8: Better Bounds for Combinatorial Semi-bandits with Knapsacks

8.1 Introduction

In this chapter we consider one extension of Stochastic BwK, namely Combinatorial Semi-Bandits with Knapsacks. We obtain better bounds for this problem than what was achieved in Chapter 7[1]. This extension combines two lines of work related to bandits: on *bandits with knapsacks* (BwK) (Badanidiyuru et al. [33]) and on *combinatorial semi-bandits* (György et al. [101]). In combinatorial semi-bandits, actions correspond to subsets of some "ground set", rewards are additive across the elements of this ground set (*atoms*), and the reward for each chosen atom is revealed (*semi-bandit feedback*). A paradigmatic example is an online routing problem, where atoms are edges in a graph, and actions are paths.

8.1.1 Formal Model

Our model, called *Semi-Bandits with Knapsacks* (`SemiBwK`) is a generalization of multi-armed bandits (henceforth, *MAB*) with i.i.d. rewards. As such, in each round $t = 1, 2, \ldots, T$, an algorithm chooses an action S_t from a fixed set of actions

[1]This work was done and published *before* that of Chapter 7.

$\mathcal{F}$, and receives a reward $\mu_t(S_t)$ for this action which is drawn independently from a fixed distribution that depends only on the chosen action. The number of rounds T, a.k.a. the *time horizon*, is know[2].

There are d resources being consumed by the algorithm. The algorithm starts out with budget $B_j \geq 0$ of each resource j. All budgets are known to the algorithm. If in round t action $S \in \mathcal{F}$ is chosen, the outcome of this round is not only the reward $\mu_t(S)$ but the consumption $C_t(S,j)$ of each resource $j \in [d]$. We refer to $\mathbf{C}_t(S) = (C_t(S,j) : j \in [d])$ as the *consumption vector*.[3] Following prior work on BwK, we assume that all budgets are the same: $B_j = B$ for all resources j.[4] Algorithm stops as soon as any one of the resources goes strictly below 0. The round in which this happens is called the stopping time and denoted $\tau_{\texttt{stop}}$. The reward collected in this last round does not count; so the total reward of the algorithm is $\text{REW} = \sum_{t<\tau_{\texttt{stop}}} \mu_t(S_t)$.

Actions correspond to subsets of a finite ground set $\mathcal{A}$, with $n = |\mathcal{A}|$; we refer to elements of $\mathcal{A}$ as *atoms*. Thus, the set $\mathcal{F}$ of actions corresponds to the family of "feasible subsets" of $\mathcal{A}$. The rewards and resource consumption is additive over the atoms: for each round t and each atom a there is a reward $\mu_t(a) \in [0,1]$ and consumption vector $\mathbf{C}_t(a) \in [0,1]^d$ such that for each action $S \subset \mathcal{F}$ it holds that $\mu_t(S) = \sum_{a \in S} \mu_t(a)$ and $\mathbf{C}_t(S) = \sum_{a \in S} \mathbf{C}_t(a)$.

[2] As opposed to Chapter 7, throughout this chapter we use n to denote the number of arms, j to denote the index of a resource and k to denote the maximum number of atoms in any action.

[3] We use bold font to indicate vectors and matrices.

[4] This is w.l.o.g. because we can divide all consumption of each resource j by $B_j / \min_{j' \in [d]} B_{j'}$. Effectively, B is the smallest budget in the original problem instance.

We assume the i.i.d. property across rounds, but allow arbitrary correlations within the same round. Formally, for a given round t we consider the $n \times (d+1)$ "outcome matrix" $(\mu_t(a), \mathbf{C}_t(a) : a \in \mathcal{A})$, which specifies rewards and resource consumption for all resources and all atoms. We assume that the outcome matrix is chosen independently from a fixed distribution $\mathcal{D}_{\texttt{M}}$ over such matrices. The distribution $\mathcal{D}_{\texttt{M}}$ is not revealed to the algorithm. The mean rewards and mean consumption is denoted $\mu(a) := \mathbb{E}[\mu_t(a)]$ and $\mathbf{C}(a) := \mathbb{E}[\mathbf{C}_t(a)]$. We extend the notation to actions, *i.e.,* to subsets of atoms: $\mu(S) := \sum_{a \in S} \mu(a)$ and $\mathbf{C}(S) := \sum_{a \in S} \mathbf{C}(a)$.

An instance of `SemiBwK` consists of the action set $\mathcal{F} \subset 2^{[n]}$, the budgets $\mathbf{B} = (B_j : \ j \in [d])$, and the distribution $\mathcal{D}_{\texttt{M}}$. The $\mathcal{F}$ and $\mathbf{B}$ are known to the algorithm, and $\mathcal{D}_{\texttt{M}}$ is not. As explained in the introduction, `SemiBwK` subsumes *Bandits with Knapsacks* (BwK) and semi-bandits. BwK is the special case when $\mathcal{F}$ consists of singletons, and semi-bandits is the special case when all budgets are equal to $B_j = nT$ (so that the resource consumption is irrelevant).

Following the prior work on BwK, we compete against the "optimal all-knowing algorithm" (Definition 1): an algorithm that optimizes the expected total reward for a given problem instance; its expected total reward is denoted by $\text{OPT}_{\texttt{DP}}$. As observed in Badanidiyuru et al. [33], $\text{OPT}_{\texttt{DP}}$ can be much larger (*e.g.,* factor of 2 larger) than the expected cumulative reward of the best action, for a variety of important special cases of BwK. Our goal is to minimize *regret*, defined as $\text{OPT}_{\texttt{DP}}$ minus algorithm's total reward.

8.2 Challenges and Contribution

8.2.1 Our Contribution

We define a common generalization of combinatorial semi-bandits and BwK, termed *Combinatorial Semi-Bandits with Knapsacks* (`SemiBwK`). We focus on the i.i.d. environment: in each round, the "outcome" is drawn independently from a fixed distribution over the possible outcomes. Here the "outcome" of a round is the matrix of reward and resource consumption for all atoms.[5] We design an algorithm for `SemiBwK`, achieving regret rates that are comparable with those for BwK and combinatorial semi-bandits.

Specifics are as follows. As usual, we assume "bounded outcomes": for each atom and each round, rewards and consumption of each resource is non-negative and at most 1. Regret is relative to the expected total reward of the best all-knowing policy, denoted $\text{OPT}_{\texttt{DP}}$. We upper-bound the regret in terms of the relevant parameters: time horizon T, (smallest) budget B, number of atoms n, and $\text{OPT}_{\texttt{DP}}$ itself (which may be as large as nT). We obtain

$$\texttt{reg} \leq \tilde{O}(\sqrt{n})(\text{OPT}_{\texttt{DP}}/\sqrt{B} + \sqrt{T + \text{OPT}_{\texttt{DP}}}). \tag{8.1}$$

The "shape" of the regret bound is consistent with prior work: the $\text{OPT}_{\texttt{DP}}/\sqrt{B}$ ad-

[5] Our model allows arbitrary correlations within a given round, both across rewards and consumption for the same atom and across multiple atoms. Such correlations are essential in applications such as dynamic pricing and dynamic assortment. *E.g.,* customers' valuations can be correlated across products, and algorithm earns only if it sells; see Section 8.7 for details.

ditive term appears in the optimal regret bound for BwK, and the $\sqrt{T}$ and $\sqrt{\mathrm{OPT}_{\mathtt{DP}}}$ additive terms are very common in regret bounds for MAB. The per-round running time is polynomial in n, and near-linear in n for some important special cases.

Our regret bound is optimal up to polylog factors for paradigmatic special cases. BwK is a special case when actions are atoms. For $\mathrm{OPT}_{\mathtt{DP}} > \Omega(T)$, the regret bound becomes $\tilde{O}(T\sqrt{n/B} + \sqrt{nT})$, where n is the number of actions, which coincides with the lower bound from Badanidiyuru et al. [33]. Combinatorial semi-bandits is a special case with $B = nT$. If all feasible subsets contain at most k atoms, we have $\mathrm{OPT}_{\mathtt{DP}} \leq kT$, and the regret bound becomes $\tilde{O}(\sqrt{knT})$. This coincides with the $\Omega(\sqrt{knT})$ lower bound from Kveton et al. [124].

Our main result assumes that the action set, *i.e.,* the family of feasible subsets of atoms, is described by a *matroid constraint*.[6] This is a rather general scenario which includes many paradigmatic special cases of combinatorial semi-bandits such as cardinality constraints, partition matroid constraints, and spanning tree constraints. We also assume that $B > \tilde{\Omega}(n + \sqrt{nT})$.

Our model captures several application scenarios, incl. dynamic pricing, dynamic assortment, repeated auctions, and repeated bidding. We work out these applications, and explain how our regret bounds improve over prior work.

[6]Matroid is a standard notion in combinatorial optimization which abstracts and generalizes linear independence.

8.2.2 Challenges and Techniques

Generic challenges in combinatorial semi-bandits concern handling exponentially many actions (both in terms of regret and in terms of the running time), and taking advantage of the additional feedback. And in `SemiBwK`, one needs to deal with distributions over subsets of atoms, rather than "just" with distributions over actions.

Our algorithm connects a technique from BwK and a randomized rounding technique from combinatorial optimization. (With *three* existing BwK algorithms and a wealth of approaches for combinatorial optimization, choosing the techniques is a part of the challenge.)

We build on a BwK algorithm from Agrawal and Devanur [11], which combines linear relaxations and a well-known "optimism-under-uncertainty" paradigm. A generalization of this algorithm to `SemiBwK` results in a fractional solution $\mathbf{x}$, a vector over atoms. Randomized rounding converts $\mathbf{x}$ into a distribution over feasible subsets of atoms that equals $\mathbf{x}$ in expectation. It is crucial (and challenging) to ensure that this distribution contains enough randomness so as to admit concentration bounds not only across rounds, but also across atoms. Our analysis "opens up" a fairly technical proof from prior work and intertwines it with a new argument based on negative correlation.

We present our algorithm and analysis so as to "plug in" any suitable randomized rounding technique. This makes our presentation more lucid, and also leads to faster running times for important special cases.

8.2.3 Solving SemiBwK using prior work.

Solving SemiBwK using an algorithm for BwK would result in a regret bound like (8.1) with n replaced with the number of actions. The latter could be on the order of n^k if each action can consist of at most k atoms, or perhaps even exponential in n.

SemiBwK can be solved as a special case of a much more general *linear-contextual* extension of BwK from Agrawal and Devanur [11, 12]. In their model, an algorithm takes advantage of the combinatorial structure of actions, yet it ignores the additional feedback from the atoms. Their regret bounds have a worse dependence on the parameters, and apply for a much more limited range of parameters. Further, their per-round running time is linear in the number of actions, which is often prohibitively large.

To compare the regret bounds, let us focus on instances of SemiBwK in which at most one unit of each resource is consumed in each round. (This is the case in all our motivating applications, except repeated bidding.) Then Agrawal and Devanur [11, 12] assume $B > \sqrt{n}\,T^{3/4}$, and achieve regret $\tilde{O}(n\sqrt{T}\frac{\mathrm{OPT}_{\mathrm{DP}}}{B} + n^2\sqrt{T})$. [7] It is easy to see that we improve upon the range and upon both summands. In particular, we

[7] Agrawal and Devanur [11, 12] state regret bound with term $+n\sqrt{T}$ rather than $+n^2\sqrt{T}$, but they assume that per-round rewards lie in $[0, 1]$. Since per-round rewards can be as large as n in our setting, we need to scale down all rewards by a factor of n, apply their regret bound, and then scale back, which results in the regret bound with $+n^2\sqrt{T}$. When per-round consumption can be as large as n, regret bound from Agrawal and Devanur [11, 12] becomes $\tilde{O}(n^2\,\mathrm{OPT}_{\mathrm{DP}}\,\sqrt{T}/B + n^2\sqrt{T})$ due to rescaling.

improve both summands by the factor of $n\sqrt{n}$ in a lucid special case when $B > \Omega(T)$ and $\text{OPT}_{\texttt{DP}} < O(T)$.[8]

8.3 Additional Related Work

Combinatorial semi-bandits were studied by György et al. [101], in the adversarial setting. In the i.i.d. setting, in a series of works by Anantharam et al. [19], Chen et al. [65], Combes et al. [68], Gai et al. [90, 91], Kveton et al. [126], an optimal algorithm was achieved. This result was then extended to atoms with linear rewards by Wen et al. [187]. Kveton et al. [124] obtained improved results for the special case when action set is described by a matroid. Some other works studied a closely related "cascade model", where the ordering of atoms matters (Katariya et al. [117], Kveton et al. [125], Zong et al. [189]). Contextual semi-bandits have been studied in Krishnamurthy et al. [123], Wen et al. [187].

Randomized rounding schemes (RRS) come from the literature on approximation algorithms in combinatorial optimization (see Papadimitriou and Steiglitz [153], Williamson and Shmoys [188] for background). RRS were introduced in Raghavan and Tompson [154]. Subsequent work Asadpour et al. [22], Chekuri et al. [59, 60], Gandhi et al. [92] developed RRS which correlate the rounded random variables so as to guarantee sharp concentration bounds.

[8] In prior work on combinatorial bandits (without constraints), semi-bandit feedback improves regret bound by a factor of $\sqrt{n}$, see the discussion in Kveton et al. [126].

8.4 Preliminaries

8.4.1 Combinatorial constraints.

Action set $\mathcal{F}$ is given by a *combinatorial constraint*, *i.e.,* a family of subsets. Treating subsets of atoms as n-dimensional binary vectors, $\mathcal{F}$ corresponds to a finite set of points in $\mathbb{R}^n$. We assume that the convex hull of $\mathcal{F}$ forms a polytope in $\mathbb{R}^n$. In other words, there exists a set of linear constraints over $\mathbb{R}^n$ whose set of feasible *integral* solutions is $\mathcal{F}$. We call such $\mathcal{F}$ *linearizable*; the convex hull is called the polytope *induced* by $\mathcal{F}$.

Our main result is for *matroid constraints*, a family of linearizable combinatorial constraints which subsumes several important special cases such as cardinality constraints, partition matroid constraints, spanning tree constraints and transversal constraints. Formally, $\mathcal{F}$ is a matroid if it contains the empty set, and satisfies two properties: (i) if $\mathcal{F}$ contains a subset S, then it also contains every subset of S, and (ii) for any two subsets $S, S' \in \mathcal{F}$ with $|S| > |S'|$ it holds that $S' \cup \{a\} \in \mathcal{F}$ for each atom $a \in S \setminus S'$.

We incorporate prior work on randomized rounding for linear programs. Consider a linearizable action set $\mathcal{F}$ with induced polytope $P \subset [0,1]^n$. The *randomized rounding scheme* (henceforth, RRS) for $\mathcal{F}$ is an algorithm which inputs a feasible fractional solution $\mathbf{x} \in P$ and the linear equations describing P, and produces a random vector $\mathbf{Y}$ over $\mathcal{F}$. We consider RRS's such that $\mathbb{E}[\mathbf{Y}] = \mathbf{x}$ and $\mathbf{Y}$ is negatively correlated (see below for definition); we call such RRS's *negatively correlated*.

Several such RRS are known: *e.g.,* for cardinality constraints and bipartite matching (Gandhi et al. [92]), for spanning trees (Asadpour et al. [22]), and for matroids (Chekuri et al. [59]).

8.4.2 Negative correlation.

Let $\mathcal{X} = (X_1, X_2, \ldots, X_m)$ denote a family of random variables which take values in $[0, 1]$. Let $X := \frac{1}{m}\sum_{i=1}^{m} X_i$ be the average, and $\mu := \mathbb{E}[X]$.

Family $\mathcal{X}$ is called *negatively correlated* if

$$\mathbb{E}\left[\prod_{i\in S} X_i\right] \leq \prod_{i\in S} \mathbb{E}[X_i] \quad \forall S \subseteq [m] \tag{8.2}$$

$$\mathbb{E}\left[\prod_{i\in S} (1 - X_i)\right] \leq \prod_{i\in S} \mathbb{E}[1 - X_i] \; \forall S \subseteq [m] \tag{8.3}$$

Independent random variables satisfy both properties with equality. For intuition: if X_1, X_2 are Bernoulli and (8.2) is strict, then X_1 is more likely to be 0 if $X_2 = 1$.

Negative correlation is a generalization of independence that allows for similar *concentration bounds*, *i.e.,* high-probability upper bounds on $|X - \mu|$. However, our analysis does not invoke them directly. Instead, we use a concentration bound given a closely related property:

$$\mathbb{E}\left[\prod_{i\in S} X_i\right] \leq (\tfrac{1}{2})^{|S|} \quad \forall S \subseteq [m]. \tag{8.4}$$

Theorem 12 (Impagliazzo and Kabanets [110])**.** *If* (8.4)*, then for some absolute constant* c*,*

$$\Pr[X \geq \tfrac{1}{2} + \eta] \leq c \cdot e^{-2m\eta^2} \qquad (\forall \eta > 0) \tag{8.5}$$

8.4.3 Confidence radius.

We bound deviations $|X - \mu|$ in a way that gets sharper when μ is small, without knowing μ in advance. (We use the notation $\mathcal{X}, X, \mu$ as above.) To this end, we use the notion of *confidence radius* from Agrawal and Devanur [11], Babaioff et al. [28], Badanidiyuru et al. [33], Kleinberg et al. [121][9]:

$$\mathrm{Rad}_\alpha(x, m) = \sqrt{\alpha x/m} + \alpha/m. \tag{8.6}$$

If random variables $\mathcal{X}$ are independent, then event

$$|X - \mu| < \mathrm{Rad}_\alpha(X, m) < 3\,\mathrm{Rad}_\alpha(\mu, m) \tag{8.7}$$

happens with probability at least $1 - O(e^{-\Omega(\alpha)})$, for any given $\alpha > 0$. We use this notion to define upper/lower confidence bounds on the mean rewards and mean resource consumption. Fix round t, atom a, and resource j. Let $\hat{\mu}_t(a)$ and $\hat{C}_t(a, j)$ denote the empirical average of the rewards and resource-j consumption, resp., between rounds 1 and $t - 1$. Let $N_t(a)$ be the number of times atom a has been chosen in these rounds (*i.e.,* included in the chosen actions). The confidence bounds are defined as

$$\begin{aligned} C_t^{\pm}(a, j) &= \texttt{proj}(\ \hat{C}(a, j) \pm \mathrm{Rad}_\alpha(\hat{C}(a, j), N_t(a))\ ) \\ \mu_t^{\pm}(a) &= \texttt{proj}\ (\ \hat{\mu}(a) \pm \mathrm{Rad}_\alpha(\hat{\mu}(a), N_t(a))\ ) \end{aligned} \tag{8.8}$$

where $\texttt{proj}(x) := \arg\min_{y \in [0,1]} |y - x|$ denotes the projection into $[0, 1]$. We always use the same parameter $\alpha = c_{\texttt{conf}} \log(ndT)$, for an appropriately chosen absolute

[9] For instance Theorem 2.1 in Badanidiyuru et al. [33]

constant $c_{\texttt{conf}}$. We suppress α and $c_{\texttt{conf}}$ from the notation. We use a vector notation $\mu_t^{\pm}$ and $\mathbf{C}_t^{\pm}(j)$ to denote the corresponding n-dimensional vectors over all atoms a.

By (8.7), with probability $1 - O(e^{-\Omega(\alpha)})$ the following hold.

$$\mu(a) \in [\mu_t^-(a),\ \mu_t^+(a)]$$

$$C(a,j) \in [C^-(a,j),\ C(a,j)^+]$$

8.4.4 Matroid constraints

Recall that in `SemiBwK`, we have a finite ground set whose elements are called "atoms", and a family $\mathcal{F}$ of "feasible subsets" of the ground set which are the actions. To be consistent with the literature on matroids, the ground set will be denoted E. Family $\mathcal{F}$ of subsets of E is called a matroid if it satisfies the following properties:

- **Empty set**: The empty set ϕ is present in $\mathcal{F}$
- **Hereditary property**: For two subsets $X, Y \subseteq E$ such that $X \subseteq Y$, we have that

$$Y \in \mathcal{F} \implies X \in \mathcal{F}$$

- **Exchange property**: For $X, Y \in \mathcal{F}$ and $|X| > |Y|$, we have that

$$\exists e \in X \setminus Y : Y \cup \{e\} \in \mathcal{F}$$

Matroids are *linearizable*, *i.e.,* the convex hull of $\mathcal{F}$ forms a polytope in $\mathbb{R}^E$. (Here subsets of $\mathcal{F}$ are interpreted as binary vectors in $\mathbb{R}^E$.) In other words, there exists a set of linear constraints whose set of feasible *integral* solutions is $\mathcal{F}$. In

fact, the convex hull of $\mathcal{F}$, a.k.a. the *matroid polytope*, can be represented via the following linear system:

$$\begin{aligned} x(S) &\leq \text{rank}(S) \quad \forall S \subseteq E \\ x(e) &\in [0,1]^E \qquad \forall e \in E. \end{aligned} \tag{LP-Matroid}$$

Here $x(S) := \sum_{e \in S} x(e)$, and $\text{rank}(S) = \max\{|Y| : Y \subseteq S, Y \in \mathcal{F}\}$ is the "rank function" for $\mathcal{F}$.

$\mathcal{F}$ is indeed the set of all feasible integral solutions of the above system. This is a standard fact in combinatorial optimization, *e.g.,* see Theorem 40.2 and its corollaries in Schrijver [168].

We will now describe some well-studied special cases of matroids. That they indeed are special cases of matroids is well-known, we will not present the corresponding proofs here.

In all LPs presented below, we have variables $x(e)$ for each arom $e \in E$, and we use shorthand $x(S) := \sum_{e \in S} x(e)$ for $S \subset E$.

Cardinality constraints. Cardinality constraint is defined as follows: a subset S of atoms belongs to $\mathcal{F}$ if and only if $|S| \leq K$ for some fixed K. This is perhaps the simplest constraint that our results are applicable to. In the context of `SemiBwK`, each action selects at most K atoms.

The corresponding induced polytope is as follows:

$$\begin{aligned} x(E) &\leq K \\ x(e) &\in [0,1] \quad \forall e \in E. \end{aligned} \tag{LP-Cardinality}$$

Partition matroid constraints. A generalization of cardinality constraints, called partition matroid constraints, is defined as follows. Suppose we have a collection

$B_1, B_2, \ldots, B_k$ of disjoint subsets of E, and numbers $d_1, d_2, \ldots, d_k$. A subset S of atoms belongs to $\mathcal{F}$ if and only if $|S \cap B_i| \leq d_i$ for every i. Partition matroid constraints appear in several applications of `SemiBwK` such as dynamic pricing, adjusting repeated auctions, and repeated bidding. In these applications, each action selects one price/bid for each offered product. Also, partition matroid constraints can model clusters of mutually exclusive products in dynamic assortment application.

The induced polytope is as follows:

$$\begin{aligned} x(B_i) &\leq d_i \quad \forall i \in [k] \\ x(e) &\in [0,1] \quad \forall e \in E. \end{aligned} \tag{LP-PartitionMatroid}$$

8.5 Main algorithm (`SemiBwK-RRS`)

Let us define our main algorithm, called `SemiBwK-RRS`. The algorithm builds on an arbitrary RRS for the action set $\mathcal{F}$. It is parameterized by this RRS, the polytope $\mathcal{P}$ induced by $\mathcal{F}$ (represented as a collection of linear constraints), and a number $\epsilon > 0$. In each round t, it recomputes the upper/lower confidence bounds, as defined in (8.8), and solves the following linear program:

$$\begin{aligned} \text{maximize} \quad & \mu_{\mathbf{t}}^+ \cdot \mathbf{x} \\ \text{subject to} \quad & \mathbf{C}_t^-(j) \cdot \mathbf{x} \leq \tfrac{B(1-\epsilon)}{T}, \quad j \in [d] \\ & \mathbf{x} \in \mathcal{P} \end{aligned} \tag{LP_{ALG}}$$

This linear program defines a linear relaxation of the original problem which is "optimistic" in the sense that it uses upper confidence bounds for rewards and lower

confidence bounds for consumption. The linear relaxation is also "conservative" in the sense that it rescales the budget by $1-\epsilon$. Essentially, this is to ensure that the algorithm does not run out of budget with high probability. Parameter ϵ will be fixed throughout. For ease of notation, we will denote $B_\epsilon := (1-\epsilon)B$ henceforth. The LP solution $\mathbf{x}$ can be seen as a probability vector over the atoms. Finally, the algorithm uses the RRS to convert the LP solution into a feasible action. The pseudocode is given as Algorithm 8.

Algorithm 8: `SemiBwK-RRS`

input: an RRS for action set $\mathcal{F}$, induced polytope $\mathcal{P}$ (as a set of linear constraints), $\epsilon > 0$.

for $t = 1, 2, \ldots, T$ **do**

1. **Recompute Confidence Bounds** as in (8.8)
2. **Obtain fractional solution** $\mathbf{x}_t \in [0,1]^n$ by solving the linear program LP_{ALG}.
3. **Obtain a feasible action** $S_t \in \mathcal{F}$ by invoking the RRS on vector $\mathbf{x}_t$.
4. **Semi-bandit Feedback**: observe the rewards/consumption for all atoms $a \in S_t$.

If action set $\mathcal{F}$ is described by a matroid constraint, we can use the negatively correlated RRS from Chekuri et al. [59]. In particular, we obtain a complete algorithm for several combinatorial constraints commonly used in the literature on semi-bandits, such as partition matroid constraints, spanning trees.

Theorem 13. *Consider the* `SemiBwK` *problem with a linearizable action set* $\mathcal{F}$ *that admits a negatively correlated* RRS. *Then algorithm* `SemiBwK-RRS` *with this* RRS *achieves expected regret bound at most*

$$O(\log(ndT))\ \sqrt{n}\left(\mathrm{OPT}_{\texttt{DP}}/\sqrt{B}+\sqrt{T+\mathrm{OPT}_{\texttt{DP}}}\right). \tag{8.9}$$

Here T *is the time horizon,* n *is the number of atoms, and* B *is the budget. We require* $B > 3(\alpha n + \sqrt{\alpha n T})$*, where* $\alpha = \Theta(\log(ndT))$ *is the parameter in confidence radius. Parameter* ϵ *in the algorithm is set to* $\sqrt{\frac{\alpha n}{B}} + \frac{\alpha n}{B} + \frac{\sqrt{\alpha n T}}{B}$.

Corollary 5. *Consider the setting in Theorem 13 and assume that the action set* $\mathcal{F}$ *is defined by a matroid on the set of atoms. Then, using the negatively correlated* RRS *from Chekuri et al. [59], we obtain regret bound* (8.9).

Running time of the algorithm. The algorithm does two computationally intensive steps in each round: solves the linear program ($\mathrm{LP}_{\mathrm{ALG}}$) and runs the RRS. For matroid constraints, the RRS from Chekuri et al. [59] has $O(n^2)$ running time. Hence, in the general case the computational bottleneck is solving the LP, which has n variables and $O(2^n)$ constraints. Matroids are known to admit a polynomial-time seperation oracle (*e.g.,* see Schrijver [168]). It follows that the entire set of constraints in $\mathrm{LP}_{\mathrm{ALG}}$ admits a polynomial-time separation oracle, and therefore we can use the Ellipsoid algorithm to solve $\mathrm{LP}_{\mathrm{ALG}}$ in polynomial time. For some classes of matroid constraints the LP is much smaller: *e.g.,* for cardinality constraints (just $d+1$ constraints) and for traversal matroids on bipartite graphs (just $2n+d$ constraints). Then near-linear-time algorithms can be used.

Our algorithm works under any negatively correlated RRS. We can use this flexibility to improve the per-round running time for some special cases. (Making decisions extremely fast is often critical in practical applications of bandits (*e.g.,* see Agarwal et al. [6].) We obtain near-linear per-round running times for cardinality constraints and partition matroid constraints. Indeed, LP_{ALG} can be solved in near-linear time, as mentioned above, and we can use a negatively correlated RRS from Gandhi et al. [92] which runs in linear time. These classes of matroid constraints are important in our applications (see Section 8.7).

8.6 Proof of Main Theorem

8.6.1 Proof overview.

First, we argue that LP_{ALG} provides a good benchmark that we can use instead of $\text{OPT}_{\texttt{DP}}$. Specifically, at any given round, the optimal value for LP_{ALG} in each round is at least $\frac{1}{T}(1-\epsilon)\,\text{OPT}_{\texttt{DP}}$ with high probability. We prove this by constructing a series of LPs, starting with a generic linear relaxation for BwK and ending with LP_{ALG}, and showing that the optimal value does not decrease along the series.

Next we define an event that occur with high probability, henceforth called *clean event*. This event concerns total rewards, and compares our algorithm against LP_{ALG}:

$$|\sum_{t\in[T]} r_t - \sum_{t\in[T]} \mu_{\mathbf{t}}^+ \cdot \mathbf{x_t}| \leq O\left(\sqrt{\alpha n \sum_{t\in[T]} r_t} + \sqrt{\alpha n T} + \alpha n\right). \tag{8.10}$$

We prove that it is indeed a high-probability event in three steps. First, we relate the algorithm's reward $\sum_t r_t$ to its expected reward $\sum_t \mu \cdot S_t$, where we

interpret the chosen action S_t, a subset of atoms, as a binary vector over the atoms. Then we relate $\sum_t \mu \cdot S_t$ to $\sum_t \mu_{\mathbf{t}}^+ \cdot S_t$, replacing expected rewards with the upper confidence bounds. Finally, we relate $\sum_t \mu_{\mathbf{t}}^+ \cdot S_t$ to $\sum_t \mu_{\mathbf{t}}^+ \cdot \mathbf{x_t}$, replacing the output of the RRS with the corresponding expectations. Putting it together, we relate algorithm's reward to $\sum_t \mu_{\mathbf{t}}^+ \cdot \mathbf{x_t}$, as needed. It is essential to bound the deviations in the sharpest way possible; in particular, the naive $\tilde{O}(\sqrt{T})$ bounds are not good enough. To this end, we use several tools: the confidence radius from (8.6), the negative correlation property of the RRS, and another concentration bound from prior work.

A similar "clean event" (with a similar proof) concerns the total resource consumption of the algorithm. We condition on both clean events, and perform the rest of the analysis via a "deterministic" argument not involving probabilities. In particular, we use the second "clean event" to guarantee that the algorithm never runs out of resources.

We use negative correlation via a rather delicate argument. We extend the concentration bound in Theorem 12 to a random process that evolves over time, and only assumes that property (8.4) holds within each round conditional on the history. For a given round, we start with a negative correlation property of S_t and construct another family of random variables that conditionally satisfies (8.4). The extended concentration bound is then applied to this family. The net result is a concentration bound for $\sum_t \mu_{\mathbf{t}}^+ \cdot S_t$ *as if* we had $n \times T$ independent random variables there.

The rest of the section contains the full proof.

8.6.2 Linear programs

We argue that LP_{ALG} provides a good benchmark that we can use instead of $\text{OPT}_{\texttt{DP}}$. Fix round t and let $\text{OPT}_{\texttt{ALG},t}$ denote the optimal value for LP_{ALG} in this round. Then:

Lemma 24. $\text{OPT}_{\texttt{ALG},t} \geq \frac{1}{T}(1-\epsilon)\,\text{OPT}_{DP}$ *with probability at least* $1-\delta$.

We will prove this by constructing a series of LP's, starting with a generic linear relaxation for BwK and ending with LP_{ALG}. We show that along the series the optimal value does not decrease with high probability.

The first LP, adapted from Badanidiyuru et al. [33], has one decision variable for each action, and applies generically to any BwK problem.

$$\begin{aligned} \text{maximize} \quad & \textstyle\sum_{S\in\mathcal{F}} \mu(S)\,x(S) \\ \text{subject to} \quad & \textstyle\sum_{S\in\mathcal{F}} C(S,j)\,x(S) \leq B/T \quad j=1,\ldots,d \\ & 0 \leq \textstyle\sum_{S\in\mathcal{F}} x(S) \leq 1. \end{aligned} \tag{LP_{BwK}}$$

Let $\text{OPT}_{\texttt{BwK}}(B)$ denote the optimal value of this LP with a given budget B. Then:

Claim 3. $\text{OPT}_{\texttt{BwK}}(B_\epsilon) \geq (1-\epsilon)\,\text{OPT}_{\texttt{BwK}}(B) \geq \frac{1}{T}(1-\epsilon)\ \text{OPT}_{DP}$.

Proof. The second inequality in Claim 3 follows from [Lemma 3.1 in 33]. We will prove the first inequality as follows. Let $\mathbf{x}^*$ denote an optimal solution to $\text{LP}_{BwK}(\text{B})$. Consider $(1-\epsilon)x^*$; this is feasible to $\text{LP}_{BwK}(B_\epsilon)$, since for every S,

$$(1-\epsilon)x^*(S) \leq 1 \quad \text{and} \quad \sum_{S\subseteq\mathcal{A}:S\in\mathcal{S}} C(S,j)(1-\epsilon)x(S) \leq B_\epsilon/T.$$

Hence, this is a feasible solution. Now, consider the objective function. Let $\mathbf{y}$ denote an optimal solution to $\text{LP}_{BwK}(B_\epsilon)$. We have that

$$\text{OPT}_{\texttt{BwK}}(B_\epsilon) = \sum_{S \subseteq \mathcal{A}: S \in \mathcal{S}} \mu(S) y^*(S) \geq \sum_{S \subseteq \mathcal{A}: S \in \mathcal{S}} \mu(S)(1-\epsilon) x^*(S) = (1-\epsilon)\,\text{OPT}_{\texttt{BwK}}(B).$$

□

Now consider a simpler LP where the decision variables correspond to atoms. As before, $\mathcal{P}$ denotes the polytope induced by action set $\mathcal{F}$.

$$\begin{aligned} &\text{maximize} \quad \mu \cdot \mathbf{x} \\ &\text{subject to} \quad C^\dagger \cdot \mathbf{x} \preccurlyeq B_\epsilon/T \quad \mathbf{x} \in \mathcal{P} \quad \mathbf{x} \in [0,1]^n. \end{aligned} \tag{LP_{ATOMS}}$$

Here $C = (C(a,j) : \ a \in A, j \in d)$ is the $n \times d$ matrix of expected consumption, and $C^\dagger$ denotes its transpose. The notation $\preccurlyeq$ means that the inequality $\leq$ holds for for each coordinate.

Leting $\text{OPT}_{\texttt{atoms}}$ denote the optimal value for LP_{ATOMS}, we have:

Claim 4. *With probability at least* $1-\delta$ *we have,* $\text{OPT}_{\texttt{ALG},t} \geq \text{OPT}_{\texttt{atoms}} \geq \text{OPT}_{\texttt{BwK}}(B_\epsilon)$.

Proof. We will first prove the second inequality.

Consider the optimal solution vector $\mathbf{x}$ to $\text{LP}_{BwK}(B_\epsilon)$. Define $S^* := \{S : x(S) \neq 0\}$.

We will now map this to a feasible solution to LP_{ATOMS} and show that the objective value does not decrease. This will then complete the claim. Consider the following solution $\mathbf{y}$ defined as follows.

$$y(a) = \sum_{S \in S^*: a \in S} x(S).$$

We will now show that $\mathbf{y}$ is a feasible solution to the polytope $\mathcal{P}$. From the definition of $\mathbf{y}$, we can write it as $\mathbf{y} = \sum_{S \in S^*} x(S) \times \mathbb{I}[S]$. Here, $\mathbf{I}[S]$ is a binary vector, such that it has 1 at position a if and only if atom a is present in set S. Hence, $\mathbf{y}$ is a point in the polytope since it can be written as convex combination of its vertices.

Now, we will show that, $\mathbf{y}$ also satisfies the resource consumption constraint.

$$\mathbf{C(j)} \cdot \mathbf{y} = \sum_{a \in \mathcal{A}} C(a,j) \sum_{S \in S^*: a \in S} x(S) = \sum_{S \in S^*} \sum_{a \in S} C(a,j) x(S) = \sum_{S \in S^*} C(S,j) x(S) \leq B_\epsilon / T.$$

The last inequality is because in the optimal solution, the x value corresponding to subset S^* is 1 while rest all are 0. We will now show that $\mathbf{y}$ produces an objective value at least as large as $\mathbf{x}$.

$$\begin{aligned} \text{OPT}_{\texttt{atoms}} = \mu \cdot \mathbf{y}^* \geq \mu \cdot \mathbf{y} &= \sum_{a=1}^{n} \mu(a) \sum_{S \in S^*: a \in S} x(S) \\ &= \sum_{S \in S^*} \sum_{a \in S} \mu(a) x(S) = \sum_{S \in S^*} \mu(S) x(S) \\ &= \text{OPT}_{\texttt{subsets}}(B_\epsilon). \end{aligned}$$

Now we will prove the first inequality. We will assume the "clean event" that $\mu_t^+ \geq \mu$ and $\mathbf{C}_t^- \leq \mathbf{C}_t$ for all t. Hence, the inequality holds with probability at least $1 - \delta$.

Consider a time t. Given an optimal solution $\mathbf{x}^*$ to LP_{ATOMS} we will show that this is feasible to $\text{LP}_{\text{ALG},t}$. Note that, $\mathbf{x}^*$ satisfies the constraint set $\mathbf{x} \in \mathcal{P}$ since that is same for both $\text{LP}_{\text{ALG},t}$ and LP_{ATOMS}. Now consider the constraint $\mathbf{C_t^-(j)} \cdot \mathbf{x} \leq \frac{B_\epsilon}{T}$. Note that $C_t^-(a,j) \leq C(a,j)$. Hence, we have that $\mathbf{C_t^-(j)} \cdot \mathbf{x}^* \leq \mathbf{C(j)} \cdot \mathbf{x}^* \leq \frac{B_\epsilon}{T}$. The

last inequality is because $\mathbf{x}^*$ is a feasible solution to $\mathrm{LP}_{\mathrm{ATOMS}}$.

Now consider the objective function. Let $\mathbf{y}^*$ denote the optimal solution to $\mathrm{LP}_{\mathrm{ALG},t}$.

$\mathrm{OPT}_{\mathtt{ALG},\,t} = \mu_{\mathbf{t}}^{+} \cdot \mathbf{y}^* \geq \mu_{\mathbf{t}}^{+} \cdot \mathbf{x}^* \geq \mu \cdot \mathbf{y}^* = \mathrm{OPT}_{\mathtt{atoms}}$. □

Hence, combining Claim 3 and Claim 4, we obtain Lemma 24.

8.6.3 Negative correlation and concentration bounds

Our analysis relies on several facts about negative correlation and concentration bounds. First, we argue that property (8.2) in the definition of negative correlation is preserved under a specific linear transformation:

Claim 5. *Suppose* $(X_1, X_2, \ldots, X_m)$ *is a family of negatively correlated random variables with support* $[0,1]$. *Fix numbers* $\lambda_1, \lambda_2, \ldots, \lambda_m \in [0,1]$. *Consider two families of random variables:*

$$\mathcal{F}^+ = \left(\frac{1 + \lambda_i (X_i - \mathbb{E}[X_i])}{2} \,:\, i \in [m] \right) \quad \textit{and} \quad \mathcal{F}^- = \left(\frac{1 - \lambda_i (X_i - \mathbb{E}[X_i])}{2} \,:\, i \in [m] \right).$$

Then both families satisfy property (8.2).

Proof. Let us focus on family $\mathcal{F}^+$; the proof for family $\mathcal{F}^-$ is very similar.

Denote $\mu_i = \mathbb{E}[X_i]$ and $Y_i := (1 + \lambda_i (X_i - \mu_i))/2$ and $z_i := (1 - \lambda_i \mu_i)/2$ for all $i \in [m]$. Note that $Y_i = \lambda_i X_i / 2 + z_i$ and $z_i \geq 0, X_i \geq 0$. Fix a subset $S \subseteq [m]$. We

have,

$$\mathbb{E}\left[\prod_{i\in S} Y_i\right] = \mathbb{E}\left[\sum_{T\subseteq S}\prod_{i\in T}(\lambda_i X_i/2)\prod_{j\in S\setminus T} z_j\right] \tag{8.11}$$

$$= \sum_{T\subseteq S}\mathbb{E}\left[\prod_{i\in T}(\lambda_i X_i/2)\right]\prod_{j\in S\setminus T} z_j$$

$$\leq \sum_{T\subseteq S}\prod_{i\in T}(\lambda_i \mu_i/2)\prod_{j\in S\setminus T} z_j \tag{8.12}$$

$$= \prod_{i\in S}((1-\lambda_i\mu_i)/2 + \lambda_i\mu_i/2) \tag{8.13}$$

$$= (\tfrac{1}{2})^{|S|} = \prod_{i\in S}\mathbb{E}[Y_i] \qquad \square$$

Eq. (8.11) and (8.13) follow from Binomial Theorem. Eq. (8.12) is because Eq. (8.2) invariant under non-negative scaling, X_i neg. correlated

Second, we extend Theorem 12 to a random process that evolves over time, and only assumes that property (8.4) holds within each round conditional on the history.

Theorem 14. *Let $\mathcal{Z}_T = \{\zeta_{t,a} : a \in \mathcal{A}, t \in [T]\}$ be a family of random variables taking values in $[0,1]$. Assume random variables $\{\zeta_{t,a} : a \in \mathcal{A}\}$ satisfy property (8.2) given $\mathcal{Z}_{t-1}$ and have expectation $\frac{1}{2}$ given $\mathcal{Z}_{t-1}$, for each round t. Let $Z = \frac{1}{nT}\sum_{a\in\mathcal{A},t\in[T]}\zeta_{t,a}$ be the average. Then for some absolute constant c,*

$$\Pr[Z \geq \tfrac{1}{2} + \eta] \leq c\cdot e^{-2m\eta^2} \qquad (\forall \eta > 0). \tag{8.14}$$

Proof. We prove that family $\mathcal{Z}_t$ satisfies property (8.4), and then invoke Theorem 12. Let us restate property (8.4) for the sake of completeness:

$$\mathbb{E}\left[\prod_{(t,a)\in S}\zeta_{t,a}\right] \leq 2^{-|S|} \quad \text{for any subset } S \subseteq \mathcal{Z}_T. \tag{8.15}$$

Fix subset $S \subset \mathcal{Z}_T$. Partition S into subsets $S_t = \{\zeta_{t,a} \in \mathcal{Z}_T \cap S\}$, for each round t. Fix round τ and denote

$$G_\tau = \prod_{t \in [\tau]} H_t, \text{ where } H_t = \prod_{a \in S_t} \zeta_{t,a}.$$

We will now prove the following statement by induction on τ:

$$\mathbb{E}[G_\tau] \leq 2^{-k_\tau}, \text{ where } k_\tau = \sum_{t \in [\tau]} |S_t|. \tag{8.16}$$

The base case is when $\tau = 1$. Note that G_τ is just the product of elements in set ζ_1 and they are negatively correlated from the premise. Therefore we are done. Now for the inductive case of $\tau \geq 2$,

$$\mathbb{E}[H_\tau | \mathcal{Z}_{\tau-1}] \leq \prod_{a \in S_\tau} \mathbb{E}[\zeta_{\tau,a} | \mathcal{Z}_{\tau-1}] \quad \text{From property (8.2) in the conditional space} \tag{8.17}$$

$$\leq 2^{-|S_\tau|} \quad \text{From assumption in Lemma 14} \tag{8.18}$$

Therefore, we have

$$\begin{aligned}
\mathbb{E}[G_\tau] &= \mathbb{E}[\mathbb{E}[G_{\tau-1} H_\tau | \mathcal{Z}_{\tau-1}]] && \text{Law of iterated expectation} \\
&= \mathbb{E}[G_{\tau-1} \mathbb{E}[H_\tau | \mathcal{Z}_{\tau-1}]] && \text{Since } G_{\tau-1} \text{ is a fixed value conditional on } \mathcal{Z}_{\tau-1} \\
&\leq 2^{-|S_\tau|} \mathbb{E}[G_{\tau-1}] && \text{From Eq. (8.18)} \\
&\leq 2^{-k_\tau} && \text{From inductive hypothesis}
\end{aligned}$$

This completes the proof of Eq. (8.16). We obtain Eq. (8.15) for $\tau = T$. □

Third, we invoke Eq. (8.7) for rewards and resource consumptions:

Lemma 25. *With probability at least* $1 - e^{-\Omega(\alpha)}$*, we have the following:*

$$\begin{aligned} |\hat{\mu}_t(a) - \mu_t(a)| &\leq 2\,\mathrm{Rad}(\hat{\mu}_t(a), N_t(a)+1) \\ \forall j \in [d] \quad |\hat{C}_t(a,j) - C_t(a,j)| &\leq 2\,\mathrm{Rad}(\hat{C}_t(a,j), N_t(a)+1). \end{aligned} \tag{8.19}$$

Fourth, we use a concentration bound from prior work which gets sharper when the expected sum is very small, and does not rely on independent random variables:

Theorem 15 (Babaioff et al. [28])**.** *Let* $X_1, X_2, \ldots, X_m$ *denote a set of* $\{0,1\}$ *random variables. For each* t*, let* α_t *denote the multiplier determined by random variables* $X_1, X_2, \ldots, X_{t-1}$*. Let* $M = \sum_{t=1}^{m} M_t$ *where* $M_t = \mathbb{E}[X_t | X_1, X_2, \ldots, X_{t-1}]$*. Then for any* $b \geq 1$*, we have the following with probability at least* $1 - m^{-\Omega(b)}$*:*

$$|\sum_{t=1}^{m} \alpha_t (X_t - M_t)| \leq b(\sqrt{M \log m} + \log m)$$

8.6.4 Analysis of the "clean event"

Let us set up several events, henceforth called *clean events*, and prove that they hold with high probability. Then the remainder of the analysis can proceed conditional on the intersection of these events. The clean events are similar to the ones in Agrawal and Devanur [11], but are somewhat "stronger", essentially because our algorithm has access to per-atom feedback and our analysis can use the negative correlation property of the RRS.

In what follows, it is convenient to consider a version of `SemiBwK` in which the algorithm does not stop, so that we can argue about what happens w.h.p. if our

algorithm runs for the full T rounds. Then we show that our algorithm does indeed run for the full T rounds w.h.p.

Recall that $\mathbf{x_t}$ be the optimal fractional solution obtained by solving the LP in round t. Let $\mathbf{Y_t} \in \{0,1\}^n$ be the random binary vector obtained by invoking the RRS (so that the chosen action $S_t \in \mathcal{F}$ corresponds to a particular realization of $\mathbf{Y_t}$, interpreted as a subset). Let $\mathcal{G}_t := \{\mathbf{Y_{t'}} : \forall t' \leq t\}$ denote the family of RRS realizations up to round t.

8.6.4.1 "Clean event" for rewards

For brevity, for each round t let $\mu_t = (\mu_t(a) : a \in A)$ be the vector of realized rewards, and let $r_t := \mu_t(S_t) = \mu_\mathbf{t} \cdot \mathbf{Y_t}$ be the algorithm's reward at this round.

Lemma 26. *Consider* `SemiBwK` *without stopping. Then with probability at least* $1 - nT\, e^{-\Omega(\alpha)}$*:*

$$|\sum_{t\in[T]} r_t - \sum_{t\in[T]} \mu^+_\mathbf{t} \cdot \mathbf{x_t}| \leq O\left(\sqrt{\alpha n \sum_{t\in[T]} r_t} + \sqrt{\alpha n T} + \alpha n\right).$$

Proof. We prove the Lemma by proving the following three high-probability inequalities.

With probability at least $1 - nT\, e^{-\Omega(\alpha)}$: the following holds:

$$|\sum_{t\in[T]} r_t - \sum_{t\in[T]} \mu \cdot \mathbf{Y_t}| \leq 3nT\,\mathrm{Rad}\left(\frac{1}{nT}\sum_{t\in[T]} \mu^+_\mathbf{t} \cdot \mathbf{x_t}\ ,\ nT\right) \tag{8.20}$$

$$|\sum_{t\in[T]} \mu \cdot \mathbf{Y_t} - \mu^+_\mathbf{t} \cdot \mathbf{Y_t}| \leq 12\sqrt{\alpha n\left(\sum_{t\in[T]} \mu^+_\mathbf{t} \cdot \mathbf{x_t}\right)} + 12\sqrt{\alpha} n + 12\alpha n \tag{8.21}$$

$$|\sum_{t\in[T]} \mu^+_\mathbf{t} \cdot \mathbf{Y_t} - \mu^+_\mathbf{t} \cdot \mathbf{x_t}| \leq \sqrt{\alpha n T}. \tag{8.22}$$

We will use the properties of RRS to prove Eq. (8.22). Proof of Eq. (8.21) is similar to Agrawal and Devanur [11], while proof of Eq. (8.20) follows immediately from the setup of the model. Using the parts (8.20) and (8.22) we can now find an appropriate upper bound on $\sqrt{\sum_{t\in[T]} \mu_{\mathbf{t}}^{+} \cdot \mathbf{x_t}}$ and using this upper bound, we prove Lemma 26.

Proof of Eq. (8.20). Recall that $r_t = \mu_{\mathbf{t}} \mathbf{Y_t}$. Note that, $\mathbb{E}[\mu_{\mathbf{t}} \mathbf{Y_t}] = \mu \mathbf{Y_t}$ when the expectation is taken over just the independent samples of μ. By Theorem 15, with probability $1 - e^{-\Omega(\alpha)}$ we have:

$$\begin{aligned}
|\sum_{t\le T} r_t - \sum_{t\le T} \mu \cdot \mathbf{Y_t}| &\le 3nT \,\mathrm{Rad}\left(\frac{1}{nT}\sum_{t\le T} \mu \cdot \mathbf{Y_t}\ ,\ nT\right) \\
&\le 3nT \,\mathrm{Rad}\left(\frac{1}{nT}\sum_{t\le T} \mu_{\mathbf{t}}^{+} \cdot \mathbf{Y_t}\ ,\ nT\right) \\
&\le 3nT \,\mathrm{Rad}\left(\frac{1}{nT}\sum_{t\le T} \mu_{\mathbf{t}}^{+} \cdot \mathbf{x_t}\ ,\ nT\right).
\end{aligned}$$

The last inequality is because Y_t is a feasible solution to $\mathrm{LP}_{\mathrm{ALG}}$.

Proof of Eq. (8.21). For this part, the arguments similar to Agrawal and Devanur [11] follow with some minor adaptations. For sake of completeness we describe the full proof. Note that we have,

$$|\sum_{t\le T} \mu \cdot \mathbf{Y_t} - \mu_{\mathbf{t}}^{+} \cdot \mathbf{Y_t}| \le \sum_{a=1}^{n} |\sum_{t\le T} \mu(a) Y_t(a) - \mu_t^{+}(a) Y_t(a)|.$$

Now, using Lemma 25 in Appendix, we have that with probability $1 - nTe^{-\Omega(\alpha)}$

$$|\sum_{t\le T} \mu(a) Y_t(a) - \mu_t^{+}(a) Y_t(a)| \le 12 \sum_{t\le T} \mathrm{Rad}(\mu(a), N_t(a)+1).$$

Hence, we have

$$\sum_{a=1}^{n} |\sum_{t\le T} \mu(a)Y_t(a) - \mu_t^+(a)Y_t(a)| = 12\sum_{a\in\mathcal{A}} \sum_{r=1}^{N_T(a)+1} \mathrm{Rad}(\mu(a), r)$$
$$\le 12\sum_{a\in\mathcal{A}} (N_T(a)+1)\,\mathrm{Rad}(\mu(a), N_T(a)+1)$$
$$\le 12\sqrt{\alpha n\,(\mu\cdot(\mathbf{N_T}+\mathbf{1}))} + 12\alpha n.$$

The last inequality is from the definition of Rad function and using the Cauchy-Swartz inequality. Note that $\mu\mathbf{N_T} = \sum_{t\le T} \mu\cdot\mathbf{Y_t}$. Also, since we have with probability $1-e^{-\Omega(\alpha)}$, $\mu(a) \le \mu_t^+(a)$, we have,

$$12\sqrt{\alpha n\,(\mu\cdot(\mathbf{N_T}+\mathbf{1}))} + 12\alpha n \le 12\sqrt{\alpha n\left(\sum_{t\le T}\mu_\mathbf{t}^+\cdot\mathbf{Y_t}\right)} + 12\sqrt{\alpha}n + 12\alpha n.$$

Finally note that $\mathbf{Y_t}$ is a feasible solution to the semi-bandit polytope $\mathcal{P}$. Hence, we have that

$$\mu_\mathbf{t}^+\cdot\mathbf{Y_t} \le \mu_\mathbf{t}^+\cdot\mathbf{x_t}.$$

Hence,

$$12\sqrt{\alpha n\left(\sum_{t\le T}\mu_\mathbf{t}^+\cdot\mathbf{Y_t}\right)} + 12\sqrt{\alpha}n + 12\alpha n \le 12\sqrt{\alpha n\left(\sum_{t\le T}\mu_\mathbf{t}^+\cdot\mathbf{x_t}\right)} + 12\sqrt{\alpha}n + 12\alpha n.$$

Proof of Eq. (8.22)*:* Recall that for each round t, the UCB vector $\mu_\mathbf{t}^+$ is determined by the random variables $\mathcal{G}_{t-1} = \{\mathbf{Y_{t'}} : \forall t' < t\}$. Further, conditional on a realization of $\mathcal{G}_{t-1}$, the random variables $\{Y_t(a) : a\in\mathcal{A}\}$ are negatively correlated from the property of RRS. Let $\tilde{\zeta}_t(a) := \mu_t^+(a)\,Y_t(a)$, $a\in\mathcal{A}$. Note that we have $\mathbb{E}[\tilde{\zeta}_t(a)|\mathcal{G}_{t-1}] = \mu_t^+(a)\,x_t(a)$. Define

$$\zeta_t(a) := \frac{1+\mu_t^+(a)\,Y_t(a) - \mu_t^+(a)\,x_t(a)}{2}.$$

From Claim 5, we have that $\{\zeta_t(a) : a \in \mathcal{A}\}$ conditioned on $\mathcal{G}_{t-1}$ satisfy (8.2). Further, $\mathbb{E}[\zeta_t(a)|\mathcal{G}_{t-1}] = \frac{1}{2}$. Therefore, the family $\{\zeta_t(a) : t \in [T], a \in \mathcal{A}\}$ satisfies the assumptions in Theorem 14 and hence satisfies Eq. (8.14) for some absolute constant c. Plugging back the $\tilde{\zeta}_t(a)$'s, we obtain an upper-tail concentration bound:

$$\Pr\left[\frac{1}{nT}(\sum_{t\in[T]}\sum_{a\in\mathcal{A}}\tilde{\zeta}_t(a) - \mu_t^+(a)\,x_t(a)) \geq \eta\right] \leq c\cdot e^{-2nT\eta^2}.$$

To obtain a corresponding concentration bound for the lower tail, we apply a similar argument to

$$\zeta'_t(a) = \frac{1+\mu_t^+(a)\,x_t(a) - \tilde{\zeta}_t(a)}{2}.$$

Once again from Claim 5, we have that $\{\zeta'_t(a) : a \in \mathcal{A}\}$ conditioned on $\mathcal{G}_{t-1}$ satisfy (8.2). The family $\{\zeta'_t(a) : t \in [T], a \in \mathcal{A}\}$ satisfies the assumptions in Theorem 14 and hence satisfies Eq. (8.14). Plugging back the $\tilde{\zeta}_t(a)$'s, we obtain a lower-tail concentration bound:

$$\Pr\left[\frac{1}{nT}(\sum_{t\in[T]}\sum_{a\in\mathcal{A}}\mu_t^+(a)x_t(a) - \tilde{\zeta}_t(a)) \geq \eta\right] \leq c\cdot e^{-2nT\eta^2}.$$

Combining these two we have,

$$\Pr\left[\frac{1}{nT}|\sum_{t\in[T]}\sum_{a\in\mathcal{A}}\mu_t^+(a)Y_t(a) - \mu_t^+(a)x_t(a)| \geq \eta\right] \leq 2\,c\cdot e^{-2nT\eta^2}. \tag{8.23}$$

Hence setting $\eta = \sqrt{\frac{\alpha}{nT}}$, we obtain Eq. (8.22) with probability at least $1 - e^{-\Omega(\alpha)}$.

Combining Eq. (8.20), (8.21) *and* (8.22) Let us denote $H := \sqrt{\sum_{t\in[T]}\mu_{\mathbf{t}}^+\cdot\mathbf{x_t}}$. Adding the three equations we get

$$|\sum_{t\in[T]} r_t - H^2| \leq \sqrt{\alpha}H + \alpha + \sqrt{\alpha n}H + O(\alpha n) + \sqrt{\alpha nT} \tag{8.24}$$

Rearranging and solving for H, we have

$$H \leq \sqrt{\sum_{t\in[T]} r_t} + O(\sqrt{\alpha n}) + (\alpha n T)^{1/4}$$

Plugging this back into Eq. (8.24), we get Lemma 26. □

8.6.4.2 "Clean event" for resource consumption

We define a similar "clean event" for consumption of each resource j. By a slight abuse of notation, for each round t let $\mathbf{C_t(j)} = (C_t(a,j) : \ a \in \mathcal{A})$ be the vector of realized consumption of resource j. Let $\chi_t(j)$ denote algorithm's consumption for resource j at round t (*i.e.*, $\chi_t(j) = \mathbf{C_t(j)} \cdot \mathbf{Y_t}$).

Lemma 27. *Consider* `SemiBwK` *without stopping. Then with probability at least* $1 - nT\, e^{-\Omega(\alpha)}$*:*

$$\forall j \in [d] \quad |\sum_{t\in[T]} \chi_t(j) - \sum_{t\in[T]} \mathbf{C_t^-(j)} \cdot \mathbf{x_t}| \leq \sqrt{\alpha n B_\epsilon} + \alpha n + \sqrt{\alpha n T}.$$

Proof. The proof is similar to Lemma 26. We will split the proof into following three equations. Fix an arbitrary resource $j \in [d]$. With probability at least $1 - nTe^{-\Omega(\alpha)}$ the following holds:

$$|\sum_{t\leq T} \chi_t(j) - \sum_{t\leq T} \mathbf{C(j)} \cdot \mathbf{Y_t}| \leq 3nT\, \text{Rad}\left(\frac{1}{nT}\sum_{t\leq T} \mathbf{C(j)} \cdot \mathbf{Y_t}\ ,\ nT\right). \tag{8.25}$$

$$|\sum_{t\leq T} \mathbf{C(j)} \cdot \mathbf{Y_t} - \mathbf{C_t^-(j)} \cdot \mathbf{Y_t}| \leq 12\sqrt{\alpha n \left(\sum_{t\leq T} \mathbf{C(j)} \cdot \mathbf{Y_t}\right)} + 12\sqrt{\alpha} n + 12\alpha n. \tag{8.26}$$

$$|\sum_{t\le T} \mathbf{C_t^-(j)}\cdot\mathbf{Y_t} - \mathbf{C_t^-(j)}\cdot\mathbf{x_t}| \le \sqrt{\alpha n T}. \tag{8.27}$$

Using the parts (8.25), (8.26) and (8.27) we can find an upper bound on $\sqrt{\sum_{t\le T}\mathbf{C_t(j)}\cdot\mathbf{Y_t}}$. Hence, combining equations (8.25), (8.26) and (8.27) with this bound and taking an Union Bound over all the resources, we get Lemma 27.

Proof of Eq. (8.25). We have that $\{C_t(a,j) : a \in \mathcal{A}\}$ is a set of independent random variables over a probability spacee C_Ω. Note that, $\mathbb{E}_{C_\Omega} C_t(a,j)Y_t(a) = C(a,j)Y_t(a)$. Hence, we can invoke Theorem 15 on *independent random variables* to get with probability $1 - nTe^{-\Omega(\alpha)}$

$$|\sum_{t\le T}\chi_t(j) - \sum_{t\le T}\mathbf{C(j)}\cdot\mathbf{Y_t}| \le 3nT\,\mathrm{Rad}\left(\frac{1}{nT}\sum_{t\le T}\mathbf{C(j)}\cdot\mathbf{Y_t}\ ,\ nT\right).$$

Proof of Eq. (8.26). This is very similar to proof of (8.21) and we will skip the repetitive parts. Hence, we have with probability $1 - nTe^{-\Omega(\alpha)}$

$$\begin{aligned}|\sum_{t\le T}\mathbf{C(j)}\cdot\mathbf{Y_t} - \mathbf{C_t^-(j)}\cdot\mathbf{Y_t}| &\le 12\sqrt{\alpha n(\mathbf{C(j)}\cdot(\mathbf{N_T}+\mathbf{1}))} + 12\alpha n \\ &\le 12\sqrt{\alpha n\left(\sum_{t\le T}\mathbf{C(j)}\cdot\mathbf{Y_t}\right)} + 12\sqrt{\alpha}n + 12\alpha n.\end{aligned}$$

Proof of Eq. (8.27). Recall that for each round t and each resource j, the LCB vector $\mathbf{C_t^-(j)}$ is determined by the random variables $\mathcal{G}_{t-1} = \{\mathbf{Y_{t'}} : \forall t' < t\}$. Similar to the proof of Eq. (8.22), random variables $\{Y_t(a) : a \in \mathcal{A}\}$ obtained from the RRS are negatively correlated given $\mathcal{G}_{t-1}$. As before define $\tilde{\zeta}_t(a) = C_t^-(a)\,Y_t(a)$, $a \in \mathcal{A}$. We have that $\mathbb{E}[\zeta_t(a) \mid \mathcal{G}_{t-1}] = C_t^-(a)\,x_t(a)$.

By Claim 5, random variables

$$\zeta_t(a) = \frac{1 + \tilde{\zeta}_t(a) - C_t^-(a)\, x_t(a)}{2},\ a \in \mathcal{A}$$

satisfy (8.2), given $\mathcal{G}_{t-1}$. We conclude that family $\{\zeta_t(a) : t \in [T], a \in \mathcal{A}\}$ satisfies the assumptions in Theorem 14, and therefore satisfies Eq. (8.14) for some absolute constant c. Therefore, we obtain an upper-tail concentration bound for $\tilde{\zeta}_t(a)$'s:

$$\Pr\left[\ \frac{1}{nT}\Big(\sum_{t\in[T]}\sum_{a\in\mathcal{A}} \tilde{\zeta}_t(a) - C_t^-(a)\, x_t(a)\Big) \geq \eta\ \right] \leq c \cdot e^{-2nT\eta^2}.$$

To obtain a corresponding concentration bound for the lower tail, we apply a similar argument to

$$\zeta'_t(a) = \frac{1 + C_t^-(a)\, x_t(a) - \tilde{\zeta}_t(a)}{2}.$$

Once again, invoking Claim 5 we have that $\{\zeta'_t(a) : a \in \mathcal{A}\}$ conditioned on $\mathcal{G}_{t-1}$ satisfy (8.2). Thus, family $\{\zeta_t(a) : t \in [T], a \in \mathcal{A}\}$ satisfies the assumptions in Theorem 14, and therefore satisfies Eq. (8.14). We obtain:

$$\Pr\left[\ \frac{1}{nT}\Big(\sum_{t\in[T]}\sum_{a\in\mathcal{A}} C_t^-(a)\, x_t(a) - \tilde{\zeta}_t(a)\Big) \geq \eta\ \right] \leq c \cdot e^{-2nT\eta^2}.$$

Combing the two tails we have,

$$\Pr\left[\ \frac{1}{nT}\Big|\sum_{t\in[T]}\sum_{a\in\mathcal{A}} C_t^-(a) Y_t(a) - C_t^-(a)\, x_t(a)\Big| \geq \eta\ \right] \leq 2\, c \cdot e^{-2nT\eta^2}. \tag{8.28}$$

Once again, setting $\eta = \sqrt{\frac{\alpha}{nT}}$, we obtain Eq. (8.27) with probability at least $1 - e^{-\Omega(\alpha)}$.

Proof of Lemma 27. Denote $G = \sqrt{\sum_{t\leq T} \mathbf{C}(\mathbf{j}) \cdot \mathbf{Y_t}}$. From Equation (8.25), (8.26) and (8.27), we have that $G^2 - 2\Omega(\sqrt{\alpha n})G \leq \sum_{t\leq T} \mathbf{C_t^-}(\mathbf{j}) \cdot \mathbf{x_t} + O(\alpha n) + \sqrt{\alpha n T}$. Note that $\sum_{t\leq T} \mathbf{C_t^-}(\mathbf{j}) \cdot \mathbf{x_t} \leq B_\epsilon$. Hence, $G^2 - 2\Omega(\sqrt{\alpha n})G \leq B_\epsilon + O(\alpha n) + \sqrt{\alpha n T}$. Hence,

re-arranging this gives us $G \leq \sqrt{B_\epsilon} + O(\sqrt{\alpha n}) + (\alpha n T)^{1/4}$. Plugging this back in Equations (8.25), (8.26) and (8.27), we get Lemma 27.

□

8.6.5 Putting it all together

Similar to Agrawal and Devanur [11], we will handle the hard constraint on budget, by choosing an appropriate value of ϵ. We then combine the above Lemma on "rewards" clean event to compare the reward of the algorithm with that of the optimal value of LP to obtain the regret bound in Theorem 13. Additionally, we use the Lemma on "consumption" clean event to argue that the algorithm doesn't exhaust the resource budget before round T. Formally, consider the following.

Recall that from Lemma 24, we have $\text{OPT}_{\texttt{ALG},} \geq \frac{1}{T}(1-\epsilon)\,\text{OPT}_{\texttt{DP}}$. Let us define the performance of the algorithm as $\text{ALG} = \sum_{t \leq T} r_t$. From Lemma 26, that with probability at least $1 - ndT\, e^{-\Omega(\alpha)}$

$$\begin{aligned}\text{ALG} &\geq (1-\epsilon)\,\text{OPT}_{\texttt{DP}} - O(\sqrt{\alpha n\,\text{ALG}}) - O(\alpha n) - \sqrt{\alpha n T} \\ &\geq (1-\epsilon)\,\text{OPT}_{\texttt{DP}} - O(\sqrt{\alpha n\,\text{OPT}_{\texttt{DP}}}) - O(\alpha n) - \sqrt{\alpha n T} \quad (\text{since } \text{ALG} \leq \text{OPT}_{\texttt{DP}}).\end{aligned}$$

Choosing $\epsilon = \sqrt{\frac{\alpha n}{B}} + \frac{\alpha n}{B} + \frac{\sqrt{\alpha n T}}{B}$ and using the assumption that $B > 3(\alpha n + \sqrt{\alpha n T})$, we derive Eq. (8.9). For any given δ, we set $\alpha = \Omega(\log(\frac{ndT}{\delta}))$ to obtain a success probability of at least $1 - \delta$.

Now we will argue that the algorithm does not exhaust the resource budget before round T with probability at least $1 - ndT\, e^{-\Omega(\alpha)}$. Note that for every resource

$j \in [d]$,

$$\sum_{t \le T} \mathbf{C_t^-(j)} \cdot \mathbf{x_t} \le (1-\epsilon)B.$$

Hence, combining this with Lemma 27, we have $\sum_{t \le T} \mathbf{C_t(j)}\, \mathbf{Y_t} \le (1-\epsilon)B + \epsilon B \le B$.

8.7 Applications and special cases

Let us discuss some notable examples of `SemiBwK` (which generalize some of the numerous applications listed in Badanidiyuru et al. [33]). Our results for these examples improve exponentially over a naive application of the BwK framework. Compared to what can be derived from Agrawal and Devanur [11, 12], our results feature a substantially better dependence on parameters, a much better per-round running time, and apply to a wider range of parameters. However, we leave open the possibility that the regret bounds can be improved for some special cases.

8.7.1 Dynamic pricing.

The dynamic pricing application is as follows. The algorithm has d products on sale with limited supply: for simplicity, B units of each. Following Besbes and Zeevi [39], we allow supply constraints *across* products, *e.g.,* a "gadget" that goes into multiple products. In each round t, an agent arrives (who can buy any subset of the products), the algorithm chooses a vector of prices $p_t \in [0,1]^d$ to offer the agent, and the agent chooses what to buy at these prices. For simplicity, the agent is interested in buying (or is only allowed to buy) at most one item of each product. The agent has a valuation vector over products, so that the agent buys a given

product if and only if her valuation for this product is at least as high as the offered price. The entire valuation vector is drawn as an independent sample from a fixed and unknown distribution (but valuations may be correlated across products). The algorithm maximizes the total revenue from sales.

To side-step discretization issues, we assume that prices are restricted to a known finite subset $S \subset [0,1]$. Achieving general regret bounds without such restriction appears beyond reach of the current techniques for BwK.[10]

To model it as a `SemiBwK` problem, the set of atoms is all price-product pairs. The combinatorial constraint is that at most one price is chosen for each product. (If an action does not specify a price for some product, the default price is used.) This is a "partition matroid" constraint. Rewards correspond to revenue from sales, and resources correspond to inventory constraints.

We obtain regret $\tilde{O}(d\sqrt{dB|S|} + \sqrt{T|S|})$ using Corollary 5, whenever $B > \tilde{\Omega}(n + \sqrt{nT})$. This is because $\mathrm{OPT}_{\texttt{DP}} \leq dB$, since that is the maximum number of products available, and the number of atoms is $n = d|S|$.

For comparison, results of Agrawal and Devanur [11, 12] apply only when $B > \sqrt{n}\,T^{3/4}$, and yield regret bound of $\tilde{O}(d^3|S|^2\sqrt{T})$.[11] Thus, our regret bounds feature a better dependence on the number of allowed prices $|S|$ (which can be

[10] Prior work on dynamic pricing with limited supply (*e.g.,* Babaioff et al. [28], Badanidiyuru et al. [33], Besbes and Zeevi [38]) achieves regret bounds without restricting itself to a particular finite set of prices, but only for a simple special case of (essentially) a single product.

[11] We obtain this by plugging in $\mathrm{OPT}_{\texttt{DP}} \leq dB$ and $n = d|S|$ into their regret bound. For dynamic pricing the total per-resource consumption is bounded by 1, so we can apply their results without rescaling the consumption.

very large) and the number of products d. Further, our regret bounds hold in a meaningful way for the much larger range of values for budget B.

For a naive application of the BwK framework, arms correspond to every possible realization of prices for the d products. Thus, we have $|S|^d$ arms, with a corresponding exponential blow-up in regret.

8.7.2 Dynamic assortment.

The dynamic assortment problem is similar to dynamic pricing in that the algorithm is selling d products to an agent, with a limited inventory B of each product, and is interested in maximizing the total revenue from sales. As before, agents can have arbitrary valuation vectors, drawn from a fixed but unknown distribution. However, the algorithm chooses which products to offer, whereas all prices are fixed externally. There is a large number of products to choose from, and any subset of $k \ll d$ of them can be offered in any given round.

To model this as `SemiBwK`, atoms correspond to products, and actions correspond to subsets of at most k atoms. The combinatorial constraint forms a partition matroid. Rewards correspond to sales, and resources correspond to products, as in dynamic pricing. Since $\text{OPT}_{\texttt{DP}} \leq \min(dB, kT)$, Corollary 5 yields regret $\tilde{O}(k\sqrt{dT})$ when $B > \Omega(T)$, and regret $\tilde{O}(d\sqrt{dB} + \sqrt{dT})$ in general.

In a naive application of BwK, arms are subsets of k products. Hence, we have $O(d^k)$ arms. The other parameters of the problem remain the same. This leads to regret bound $\tilde{O}(d\sqrt{Bd^k})$, with an exponential dependence on k.

8.7.3 Repeated auctions.

Consider a repeated auction with adjustable parameters, *e.g.,* repeated second-price auction with reserve price that can be adjusted from one round to another. While prior work (Badanidiyuru et al. [33], Cesa-Bianchi et al. [55]) concerned running one repeated auction, we generalize this scenario to multiple repeated auctions with shared inventory (*e.g.,* the same inventory may be sold via multiple channels to different audiences).

More formally, the auctioneer is running r simultaneous repeated auctions to sell a shared inventory of d products, with limited supply B of each product (*e.g.,* different auctions can cater to different audiences). Each auction has a parameter which the algorithm can adjust over time. We assume that this parameter comes from a finite domain $S \subset [0, 1]$. For simplicity, assume the auctions are synchronized with one another. As in prior work, we assume that in every round of each auction a fresh set of participants arrives, sampled independently from a fixed joint distribution, and only a minimal feedback is observed: the products sold and the combined revenue.

Following prior work (Badanidiyuru et al. [33], Cesa-Bianchi et al. [55]), we only assume minimal feedback: for each auction, what were the products sold and what was the combined revenue from this auction. In particular, we do not assume that the algorithm has access to participants' bids. Not using participants' bids is desirable for privacy considerations, and in order to reduce the participants' incentives to game the learning algorithm.

To model this problem as SemiBwK, atoms are all auction-parameter pairs. The combinatorial constraint is that an action must specify at most one parameter value for each auction. This corresponds to partition matroid constraint. There is a "default parameter" for each auction, in case an action does not specify the parameter. We have a resource for each product being auctioned. For simplicity, each product has supply of B. Note that $\text{OPT}_{\texttt{DP}} \leq dB$ and number of atoms is $n = r|S|$. Hence, our main result yields regret $\tilde{O}(d\sqrt{r|S|B} + \sqrt{r|S|T})$.

A naive application of the BwK framework would have arms that correspond to all possible combinations of parameters, for the total of $O(|S|^r)$ arms. Again, we have an exponential blow-up in regret. Alternatively, one may try running r separate instances of BwK, one for each auction, but that may result result in budgets being violated since the items are shared across the auctions and it is unclear a priori how much of each item will be sold in each auction.

One can also consider a "flipped" version of the previous example, where the algorithm is a bidder rather than the auction maker. The bidder participates in r repeated auctions, *e.g.,* ad auctions for different keywords. We assume a stationary environment: bidder's utility from a given bid in a given round of a given auction is an independent sample from a fixed but unknown distribution. The only limited resource here is the bidder's budget B. Bids are constrained to lie in a finite subset S.

To model this as SemiBwK, atoms correspond to the auction-bid pairs. The combinatorial constraint is that each action must specify at most one bid for each auction. (There is a "default bid" for each auction in case an action does not specify

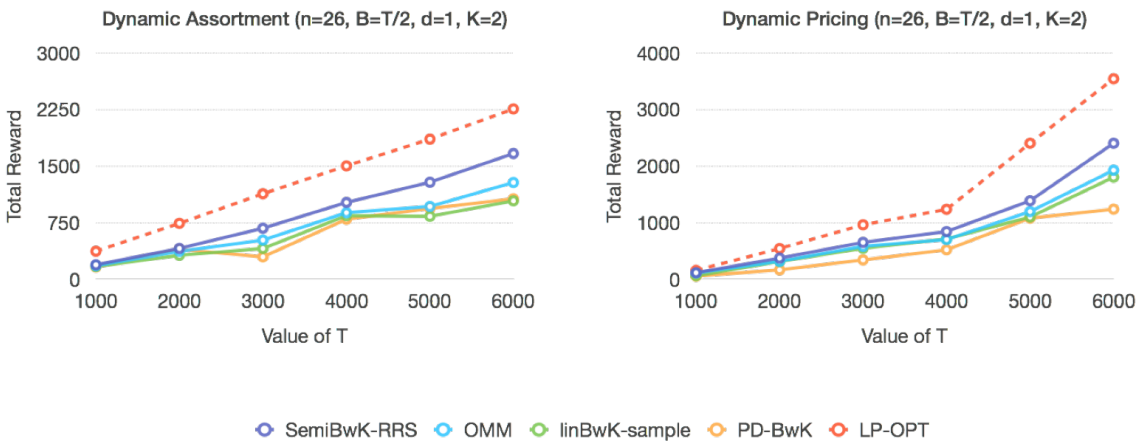

Figure 8.1: Dynamic Assortment (left) and Dynamic Pricing (right) experiments for $n = 26$.

the bid for this auction.) There is exactly one resource, which is money and the total budget is B. Note that the number of atoms is $n = r|S|$. Hence, our main result yields regret $\tilde{O}(\text{OPT}_{\texttt{DP}} \sqrt{r|S|/B} + \sqrt{r|S|T})$.

A naive application of BwK would have arms that correspond to all possible combinations of bids, for the total of $O(|S|^r)$ arms; so we have an exponential blow-up in regret.

8.8 Numerical Simulations

We ran some experiments on simulated datasets in order to compare our algorithm, `SemiBwK-RRS`, with some prior work that can be used to solve `SemiBwK`:

- the primal-dual algorithm for BwK from Badanidiyuru et al. [33], denoted `pdBwK`.
- an algorithm for combinatorial semi-bandits with a matroid constraint: "Optimistic Matroid Maximization" from Kveton et al. [124], denoted `OMM`.

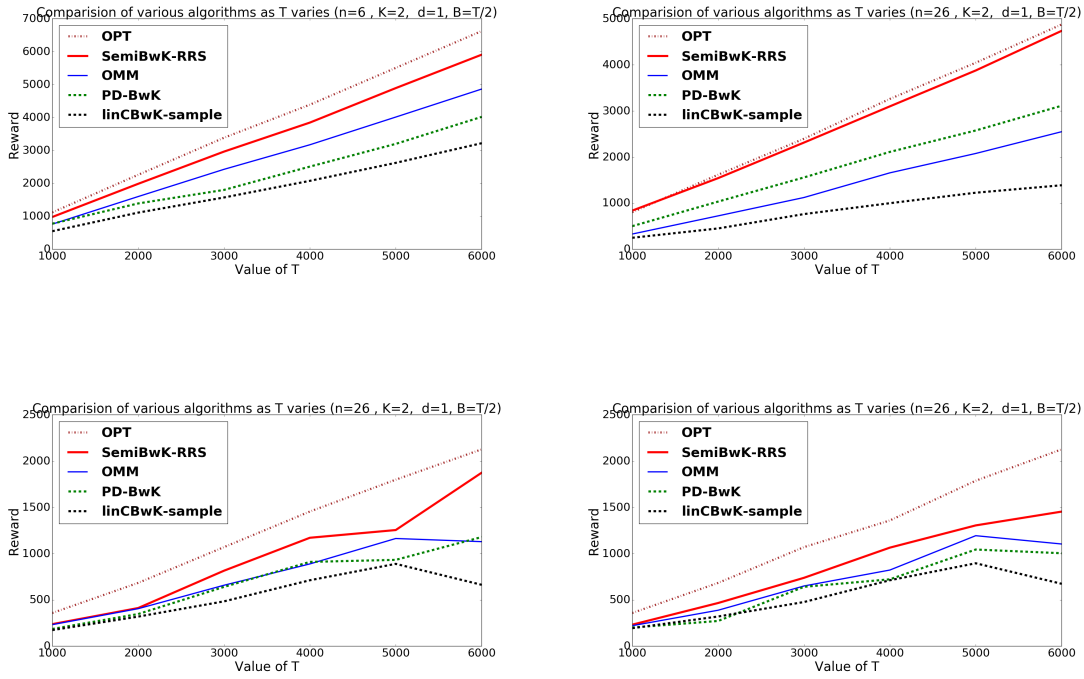

Figure 8.2: Experimental Results for Uniform matroid (left plots) and Partition matroid (right plots) on independent (upper) and correlated (lower) instances for $n = 26$.

- the linear-contextual BwK algorithm from Agrawal and Devanur [12], discussed in the Introduction, denoted `linCBwK`.

To speed up the computation in `linCBwK`, we used a heuristic modification suggested by the authors in a private communication. This modification did not substantially affect average rewards in our preliminary experiments. We also made a heuristic improvement to our algorithm, setting $\epsilon = 0$ and $\alpha = 5$. We use the same value of α for the `pdBwK` algorithm as well.

Problem instances. We did not attempt to comprehensively cover the huge variety of problem instances in `SemiBwK`. Instead, we focus on two representative applications from Section 8.7.

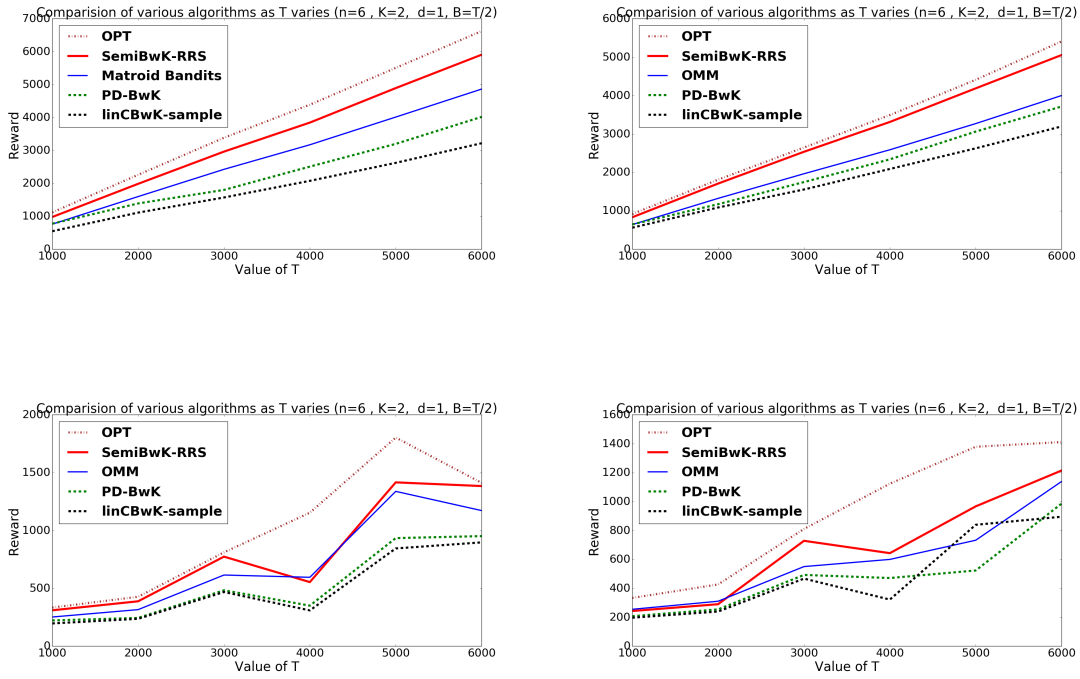

Figure 8.3: Experimental Results for Uniform matroid (left plots) and partition matroid (right plots) on independent (upper) and correlated (lower) instances for $n = 6$.

The first experiment is on dynamic assortment. We have n products, and for each product i there is an atom i and a resource i. The (fixed) price for each product is generated as an independent sample from $U_{[0,1]}$, a uniform distribution on $[0, 1]$. At each round, we sample the buyers's valuation from $U_{[0,1]}$, independently for each product. If the valuation for a given product is greater than the price, one item of this product is sold (and then the reward for atom i is the price, and consumption of resource i is 1). Else, we set reward for atom i and consumption for resource i to be 0.

The second experiment is on dynamic pricing with two products. We have $n/2$ allowed prices, uniformly spaced in the $[0, 1]$ interval. Recall that atoms correspond

to price-product pairs, for the total of n atoms. In each round t, the valuation $v_{t,i}$ for each product i is chosen independently from a normal distribution $\mathcal{N}(v_i^0, 1)$ truncated on $[0, 1]$. The mean valuation v_i^0 is drawn (once for all rounds) from $U_{[0,1]}$. If $v_{t,i}$ is greater than the offered price p, one item of this product is sold. Then reward for the corresponding atom (p, i) is the price p, and consumption of product i is 1. If there is no sale for this product, the reward and consumption for each atom (p, i) is set to 0.

The third experiment is a modification of the dynamic assortment example, in which we ensure that even non-action (e.g., no sale) exhausts resources other than time. As in dynamic assortment, we have n products, and for each product i there is an atom i and a resource i. The (fixed) price for each product is generated as an independent sample from $U_{[0,1]}$, a uniform distribution on $[0, 1]$. At each round, we sample the buyers's valuation from $U_{[0,1]}$, independently for each product. If the valuation for a given product is greater than the price, one item of this product is sold (and then the reward for atom i is the price, and consumption of resource i is 1). Else, we do something different from dynamic assortment: we set reward for atom i and consumption for resource i to be the buyer's valuation.

The fourth experiment is a similar modification of the dynamic pricing example. We have $n/2$ allowed prices, uniformly spaced in the $[0, 1]$ interval. Recall that atoms correspond to price-product pairs, for the total of n atoms. In each round t, the valuation $v_{t,i}$ for each product i is chosen independently from a normal distribution $\mathcal{N}(v_i^0, 1)$ truncated on $[0, 1]$. The mean valuation v_i^0 is drawn (once for all rounds) from $U_{[0,1]}$. If the valuation for a given product i is greater than the offered

price p, one item of this product is sold (and then reward for the corresponding atom (p, i) is the price, and consumption of product i is 1). If there is no sale for this product, we do something different from dynamic pricing. For each atom (p, i), if $p < v_{t,i}$ then the reward for atom (p, i) is drawn independently from $U_{[0,1]}$ and resource consumption is 1; else, reward is 0 and consumption is .3. While dynamic assortment is modeled with a uniform matroid, and dynamic pricing is modeled with a partition matroid, we tried both matroids on each family.

Experimental setup and results. We choose various values of n, B and T and run our algorithms on the above two datasets assuming both a uniform matroid constraint and a partition matroid constraint. We choose $n \in \{6, 26\}$, $T \in \{1000, 2000, 3000, 4000, 5000, 6000\}$ and $B = T/2$. The maximum number of atoms in any action is set to $K = 2$. For a given algorithm, dataset and configuration of n and T, we simulate each algorithm for 20 independent runs and take the average. We calculate the total reward obtained by the algorithm at the end of T time-steps.

Figure 8.1 shows results for the first two experiments. Figures 8.2 and 8.3 show the results on the third and fourth experiments. Our algorithm achieves the best regret among the competitors. As a benchmark, we included the performance of the fractional allocation in $\text{LP}_{\text{OPT}_{\texttt{DP}}}$, denoted $\text{OPT}_{\texttt{DP}}$.

Additional experiment. `linCBwK` and `pdBwK` have running times proportional to the number of actions. We ran an additional experiment which compared per-step running times. We first calculate the average running time for every 10 steps and take the median of 50 such runs. For both Uniform matroid and Partition matroid,

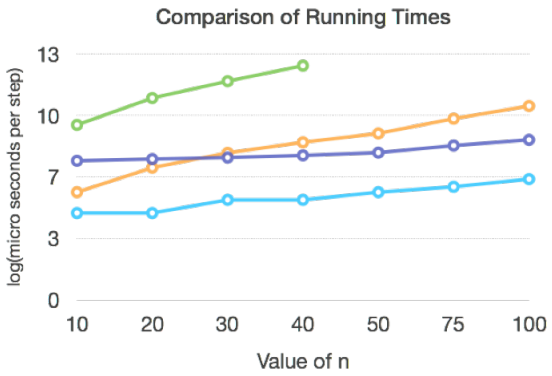

Figure 8.4: Variation of per-step running times as n increases for the various algorithms.

we run the faster RRS due to Gandhi et al. [92]. See Figure 8.4 for results.

Details of heuristic implementation of `linCBwK`. We now briefly describe the heuristic we use to simulate the `linCBwK` algorithm. Note that even though the per-time-step running time of `linCBwK` is reasonable, it takes a significant time when we want to perform simulations for many time-steps. The time-consuming step in the `linCBwK` algorithm is the solution to a convex program for computing the optimistic estimates (namely $\tilde{\mu}_{\mathbf{t}}$ and $\tilde{\mathbf{W}}_{\mathbf{t}}$). Hence, the heuristic gives a faster way to obtain this estimate. We sample multiple times from a multi-variate Gaussian with mean $\hat{\mu}$ and covariance $\mathbf{M_t}$ (to obtain estimate $\tilde{\mu}_{\mathbf{t}}$) and with mean $\hat{\mathbf{w}}_{\mathbf{tj}}$ and covariance $\mathbf{M}_t$ (to obtain estimate $\tilde{\mathbf{w}}_{tj}$ for each resource j). We use these samples to compute the objective to choose the action at time-step t. For each sample, we compute the best action based on the objective in `linCBwK`. We finally choose the action that occurs majority number of times in these samples. The number of samples we choose is set to 30.

Language Details of algorithms. All algorithms except `linCBwK` were imple-

mented in Python. The `linCBwK` algorithm was implemented in MATLAB. This difference is crucial when we compare running times since language construct can speed-up or slow down algorithms in practice. However, it is known that [12] for matrix operations commonly encountered in engineering and statistics, MATLAB implementations runs several orders of magnitude faster than the corresponding python implementation. Since `linCBwK` is the slowest of the four algorithms, our comparison of running times across languages is justified.

8.9 Conclusion

In this chapter we consider an extension of the BwK problem, namely, stochastic combinatorial semi-bandits with knapsacks under matroid constraints and gave a specialized algorithm with optimal bounds. Moreover, we also show applications of `SemiBwK` and run numerical experiments on simulated data to support our theoretical results. The results in this chapter work for a specialized (yet capture all known applications) action set and obtain stronger bounds (essentially optimal) than those in Section 7.9 in Chapter 7.

[12] https://www.mathworks.com/products/matlab/matlab-vs-python.html

Chapter 9: Future Directions

In this chapter we conclude with future research directions that stem out of the problems considered in this thesis. We consider both future directions in the models of SDM considered in this thesis as well as other problems beyond SDM that are motivated by problems in this thesis. We also believe that the work in this thesis can be applied to a wider range of applications and exploring them is a future direction.

Online Matching

The most direct open problem in online matching is to close the gap between the upper and lower-bound for the Online Weighted Matching problem in the *known* distribution model considered in Chapter 3. We provide an algorithm with a competitive ratio of 0.7 while the known lower-bound is 0.823 given by Manshadi et al. [137] which also applies to *unweighted graphs*. Thus, we want to close this gap by either bringing the upper-bound towards the lower-bound or vice versa (or possibly both).

The other direction is to consider stochastic arrival models that relax the notion of independence across time-steps which is often too simplifying. In many practical scenarios, this model of stochastic arrival doesn't exactly capture the reality of

the underlying data. The input often resembles a stochastic process with some predictable correlations across time-steps (*e.g.,* rider request pattern depends on local events). In the context of optimization in matching markets and specifically, online matching, results based on correlated data arrival have not been studied. Thus, it is interesting to formulate and study online matching and its variants under notions of correlation that are both theoretically challenging and practically relevant. Candidate models include notions of limited dependence such as Markov chains. Such models are especially useful in applications where the input is received as a result of *human* decisions. The other direction is to consider corrupted inputs, that are frequently encountered in many applications. Abstractly, this can be modeled as input sequences that are *close* to an i.i.d. sequence with adversarial perturbations. In other words, we want algorithms whose performance smoothly interpolate between the stochastic and the adversarial models. Prior work (Esfandiari et al. [81]) has proposed one interesting model along these lines for a particular variant of the unweighted online matching problem. However, the question of interpolation is not yet fully understood and thus, it will be interesting to study this for the vast landscape of online matching problems.

From a theoretical stand-point one interesting open question is to obtain a competitive ratio bound that holds with *high-probability* (*i.e.,* bound the variance of the algorithm's performance). For the online weighted matching problem the performance guarantees of both the algorithms proposed in Chapter 3 and the one proposed in Haeupler et al. [102] only hold in expectation. Thus, it will be interesting to understand if similar ratios can be proved that hold with high-probability. For

the unweighted version of the problem, such high-probability bounds are known (*e.g.,* Feldman et al. [83]).

Bandits with Knapsacks

The work on adversarial BwK (Chapter 7) directly leads to a few open questions and directions. In particular, it would be illuminating to have a single algorithm that has good performance simultaneously in the stochastic and the adversarial setting *without* knowing a-priori, the environment the algorithm is operating in (so called "best-of-both-worlds algorithm"). Our work almost achieves this, where we develop a common framework to solve both the settings but technical conditions prevent us from using it without knowing the environment. A closely related but distinct question is to obtain *instance-specific* bounds for the stochastic version of the problem. In particular, the goal is to derive bounds that improve on the *easy* instances of the problem while still achieving the worst-case bounds in the *hard* regimes. Another interesting direction is to consider the Stochastic BwK problem with limited dependence, where the environment is stochastic (but not independent) while the rewards/consumptions across times have weak correlation. Limited dependence and multi-armed bandits have been previously studied and is supported by numerous motivating examples. However, in many of these applications we also have limited inventory and thus, the ideal approach would be to use variants of BwK in such applications.

A broader direction stemming out of our work is to investigate the interplay of learning and game-theory in the context of *time-varying* zero-sum games. In a

time-varying zero-sum game, the players play different zero-sum games repeatedly and the goal for each player is to maximize their total payoff. When both players play optimally, the strategy pair is said to be in *equilibrium*. Classical results in online learning show that MAB algorithms played against each other, approximately converge to the equilibrium. However, the results are limited to the settings when the game remains the same across time or the payoff functions across time are sampled independently and identically from the same distribution. Thus, it is interesting to understand the behavior of MAB algorithms when no assumptions about the repeated zero-sum games are made. A very recent work by Cardoso et al. [53] make some progress in this direction.

Beyond SDM and Related Problems

The central thesis in our work was to explore SDM problems with information constraints. However, there are many other problems that are closely related that have similar constraints. Consider the BwK problem. The key challenge for the algorithm was to perform *counterfactual* inference (*i.e.,* answering what if for arms that were not chosen) in the face of limited samples. In sequential experiment design literature, there are multiple approaches to studying this challenge, with bandit-based approach being one of them. An alternate method to do this is to pose this as a problem in *causal inference* and study the effects of *limited* samples on inference algorithms. This question is not well understood; only a handful of prior works (*e.g.,* Ghoshal and Honorio [93], Sankararaman et al. [164, 165]) have considered this question and it would be interesting to understand the limits of finite samples

for counterfactual reasoning. More broadly, understanding the trade-off between limited samples and performance of online algorithms with stochastic input is an interesting research direction (*i.e.,* the algorithm only has access to samples from a distribution as opposed to the *known* distribution assumption).

Part III

Appendix

Appendix: Standard Tools

10.1 Concentration Inequalities

Lemma 28 (Azuma-Hoeffding inequality ([148])). *Let $Y_1, Y_2, \ldots, Y_T$ be a martingale difference sequence (*i.e., *$\mathbb{E}[Y_t \mid Y_1, Y_2, \ldots, Y_{t-1}] = 0$). Suppose $|Y_t| \leq c$ for all $t \in \{1, 2, \ldots, T\}$. Let $R_{0,\delta}(T) := \sqrt{2Tc^2 \ln(1/\delta)}$. Then for every $\delta > 0$,*

$$\Pr\left[\sum_{t\in[T]} Y_t > R_{0,\delta}(T)\right] \leq \delta.$$

Lemma 29 (Chernoff-Hoeffding bounds ([148])). *Let $X_1, X_2, \ldots, X_T$ be a sequence of independent random variables such that $|X_t| \leq c$ for all $t \in \{1, 2, \ldots, T\}$. Let $Z_t := \mathbb{E}[X_t]$. Then for every $\delta > 0$,*

$$\Pr\left[\left|\sum_{t\in[T]} X_t - Z_t\right| > 3\sqrt{\left(\sum_{t\in[T]} Z_t\right) c^2 \ln(1/\delta)}\right] \leq \delta.$$

10.2 Adversarial Online Learning

Let us revisit adversarial online learning, as per Figure 7.2. Denote the benchmark in Eq. (7.4) as

$$\mathrm{OPT}_{\texttt{AOL}}(T) := \max_{a\in A} \sum_{t\in[T]} f_t(a).$$

Recall that $[b_{\min}, b_{\max}]$ is the payoff range, and denote $\sigma = b_{\max} - b_{\min}$.

Lemma 30. *Suppose an algorithm for adversarial online learning satisfies Eq. (7.4) for some $\delta > 0$. Then*

$$\Pr\left[\ \forall \tau \in [T]\ \ \mathrm{OPT}_{\texttt{AOL}}(\tau)\ -\ \textstyle\sum_{t\in[\tau]} \mathbf{f}_t \cdot \mathbf{p}_t \le \sigma \cdot \left(R_{\delta/T}(T) + \sqrt{2T\log(T/\delta)}\right)\ \right] \ge 1 - 2\delta. \tag{10.1}$$

Proof. Let us use the stronger regret bound Eq. (7.5) implied by Eq. (7.4). Note that

$$\mathbb{E}[f_t(a_t) \mid a_1, a_2\,,\ \ldots\ , a_{t-1}] = \mathbf{f}_t \cdot \mathbf{p}_t.$$

Applying the Azuma-Hoeffding inequality for each $\tau \in [T]$, and taking a union bound, we have

$$\Pr\left[\ \forall \tau \in [T] \quad \textstyle\sum_{t\in[\tau]} f_t(a_t) - \sum_{t\in[\tau]} \mathbf{f}_t \cdot \mathbf{p}_t \le \sigma \cdot \sqrt{2T\log(T/\delta)}\ \right] \ge 1 - \delta. \tag{10.2}$$

Taking a union bound over Eq. (10.2) and Eq. (7.5) and adding the equations we get Eq. (10.1). □

Remark 12. *For Hedge algorithm, regret bound Eq. (10.1) is already proved in [88].*

10.3 Lagrangians

10.3.1 Proof of Lemma 12

Assume one of the resources is the dummy resource, and one of the arms is the null arm. Consider the linear program $\mathrm{LP}_{\mathbf{M},B,T}$, for some outcome matrix $\mathbf{M}$. Let $\mathcal{L} = \mathcal{L}_{\mathbf{M},B,T}$ denote the Lagrange function.

Lemma 31 (Lemma 12, restated)**.** *Suppose $(\mathbf{X}^*, \lambda^*)$ is a mixed Nash equilibrium for the Lagrangian game. Then $\mathbf{X}^*$ is an optimal solution for the linear program*

(7.8). *Moreover, the minimax value of the Lagrangian game equals the LP value:* $\mathcal{L}(\mathbf{X}^*, \lambda^*) = \mathrm{OPT}_{LP}$.

In what follows we prove Lemma 31. Writing out the definition of the mixed Nash equilibrium,

$$\mathcal{L}(\mathbf{X}^*, \lambda) \geq \mathcal{L}(\mathbf{X}^*, \lambda^*) \geq \mathcal{L}(\mathbf{X}, \lambda^*) \qquad \forall \mathbf{X} \in \Delta_K, \lambda \in \Delta_d. \tag{10.3}$$

For brevity, denote $r(\mathbf{X}^*) = \sum_{a \in [K]} \mathbf{X}^*(a)\ r(a)$ and $c_i(\mathbf{X}^*) = \sum_{a \in [K]} \mathbf{X}^*(a)\ c_i(a)$.

We first state and prove the complementary slackness condition for the Nash equilibrium.

Claim 6. *For every resource $i \in [d]$ we have,*

(a) $1 - \frac{T}{B}\, c_i(\mathbf{X}^*) \geq 0$,

(b) $\lambda_i^* > 0 \implies 1 - \frac{T}{B}\, c_i(\mathbf{X}^*) = 0$.

Proof. **Part (a).** For contradiction, consider resource i that minimizes the left-hand side in (a), and assume that the said left-hand side is strictly negative. We have two cases: either $\lambda_i^* < 1$ or $\lambda_i^* = 1$. When $\lambda_i^* < 1$, consider another distribution $\tilde{\lambda} \in \Delta_d$ such that $\tilde{\lambda}_i = 1$ and $\tilde{\lambda}_{i'} = 0$ for every $i' \neq i$. Note that we have, $\mathcal{L}(\mathbf{X}^*, \tilde{\lambda}) < \mathcal{L}(\mathbf{X}^*, \lambda^*)$. This contradicts the first inequality in Eq. (10.3).

Consider the second case, when $\lambda_i^* = 1$. Then $\mathcal{L}(\mathbf{X}^*, \lambda^*) = r(\mathbf{X}^*) + 1 - \frac{T}{B} c_i(\mathbf{X}^*)$. Consider any arm $a \in [K]$ such that $X^*(a) \neq 0$. Let $\tilde{\mathbf{X}} \in \Delta_K$ be another distribution such that $\tilde{X}(a) := 0$ and $\tilde{X}(\text{null}) := X^*(\text{null}) + X^*(a)$ and $\tilde{X}(a') = X^*(a')$ for every $a' \notin \{a, \text{null}\}$. Note that $\tilde{X}(\text{null}) \leq 1$. Since $(\mathbf{X}^*, \lambda^*)$ is a Nash equilibrium, we

have that $\mathcal{L}(\tilde{\mathbf{X}}, \lambda^*) \leq \mathcal{L}(\mathbf{X}^*, \lambda^*)$. This implies that $-X^*(a)r(a) + X^*(a)\frac{T}{B}c_i(a) \leq 0$. Re-arranging we obtain, $\frac{T}{B}c_i(a) \leq r(a) \leq 1$. Thus, we have $1 - \frac{T}{B}c_i(a) \geq 0$.

Since this holds for every $a \in [K]$ with $X^*(a) \neq 0$, we obtain a contradiction:

$$1 - \tfrac{T}{B}\, c_i(\mathbf{X}^*) = \textstyle\sum_{a\in[K]} X^*(a) \left(1 - \tfrac{T}{B}c_i(a)\right) \geq 0.$$

Part (b). For contradiction, assume the statement is false for some resource i. Then, by part (a), $\lambda_i^* > 0$ and $1 - \frac{T}{B}c_i(\mathbf{X}^*) > 0$, and consequently $\mathcal{L}(\mathbf{X}^*, \lambda^*) > r(\mathbf{X}^*)$. Now, consider distribution $\tilde{\lambda}$ which puts probability 1 on the dummy resource. We then have $\mathcal{L}(\mathbf{X}^*, \tilde{\lambda}) = r(\mathbf{X}^*) < \mathcal{L}(\mathbf{X}^*, \lambda^*)$, contradicting the first inequality in Eq. (10.3). □

Let $\tilde{\mathbf{X}}$ be some feasible solution for the linear program (7.8). Plugging the feasibility constraints into the definition of the Lagrangian function, $\mathcal{L}(\tilde{\mathbf{X}}, \lambda^*) \geq r(\tilde{\mathbf{X}})$. Claim 6(a) implies that $\mathbf{X}^*$ is a feasible solution to the linear program (7.8). Claim 6(b) implies that $\mathcal{L}(\mathbf{X}^*, \lambda^*) = r(\mathbf{X}^*)$. Thus,

$$r(\mathbf{X}^*) = \mathcal{L}(\mathbf{X}^*, \lambda^*) \geq \mathcal{L}(\tilde{\mathbf{X}}, \lambda^*) \geq r(\tilde{\mathbf{X}}).$$

So, $\mathbf{X}^*$ is an optimal solution to the LP. In particular, $\text{OPT}_{\texttt{LP}} = r(\mathbf{X}^*) = \mathcal{L}(\mathbf{X}^*, \lambda^*)$.

10.3.2 Regret minimization in games

10.3.3 Proof of Lemma 11

Let $W = \sqrt{2T\log(T/\delta)}$ denote the term from Lemma 30 in what follows.

We now prove Lemma 11, similar to the proof in [87] for the deterministic game. Recall that we take averages up to some fixed round $\tau \in [T]$. We prove that

the following two inequalities hold, each with probability at least $1-\delta$.

$$\frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}^{\mathrm{T}}\,\mathbf{G}_t\,\mathbf{p}_{t,2} \geq v^* - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+2W}{\tau}. \tag{10.4}$$

$$\frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}^{\mathrm{T}}\,\mathbf{G}_t\,\mathbf{p}_{t,2} \leq \overline{\mathbf{p}}_1^{\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_2 + \sigma\cdot\frac{R_{2,\,\delta/T}(T)+2W}{\tau} \qquad \forall \mathbf{p}_2\in\Delta_{A_2}. \tag{10.5}$$

Eq. (7.7) in Lemma 11 follows by adding Eq. (10.4) and Eq. (10.5).

First we prove Eq. (10.4). Following the set of inequalities in Section 2.5 of [87] we have,

$$\begin{aligned}
\frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}^{\mathrm{T}}\mathbf{G}_t\mathbf{p}_{t,2} &\geq_{whp} \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_1^{*\mathrm{T}}\,\mathbf{G}_t\,\mathbf{p}_{t,2} - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+W}{\tau} && \text{From Lemma 30}\\
&\geq_{whp} \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_1^{*\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_{t,2} - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+2W}{\tau} && \text{From Lemma 28}\\
&= \max_{\mathbf{p}_1\in\Delta_{A_1}} \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_1{}^{\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_{t,2} - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+2W}{\tau} && \text{From Definition of } \mathbf{p}_1^*.\\
&= \max_{\mathbf{p}_1\in\Delta_{A_1}} \mathbf{p}_1{}^{\mathrm{T}}\,\mathbf{G}\,\overline{\mathbf{p}}_2 - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+2W}{\tau} && \text{From Definition of } \overline{\mathbf{p}}_2.\\
&\geq \min_{\mathbf{p}_2\in\Delta_{A_2}}\ \max_{\mathbf{p}_1\in\Delta_{A_1}} \mathbf{p}_1^{\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_2 - \sigma\cdot\frac{R_{1,\,\delta/T}(T)+2W}{\tau}
\end{aligned}$$

Here $\leq_{whp}$ denotes statements that hold with probability at least $1-\delta$.

Now let us prove Eq. (10.5). Fix distribution $\mathbf{p}_2\in\Delta_{A_2}$. Then:

$$\begin{aligned}
\frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}^{\mathrm{T}}\,\mathbf{G}_t\,\mathbf{p}_{t,2} &\leq_{whp} \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}{}^{\mathrm{T}}\,\mathbf{G}_t\,\mathbf{p}_2 + \sigma\cdot\frac{R_{2,\,\delta/T}(T)+W}{\tau} && \text{From Lemma 30}\\
&\leq_{whp} \frac{1}{\tau}\sum_{t\in[\tau]} \mathbf{p}_{t,1}{}^{\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_2 + \sigma\cdot\frac{R_{2,\,\delta/T}(T)+2W}{\tau} && \text{From Lemma 28}\\
&= \overline{\mathbf{p}_1}^{\mathrm{T}}\,\mathbf{G}\,\mathbf{p}_2 + \sigma\cdot\frac{R_{2,\,\delta/T}(T)+2W}{\tau} && \text{From Definition of } \overline{\mathbf{p}_1}.
\end{aligned}$$

Taking a union bound over all the four high-probability inequalities, we get the lemma.

Bibliography

[1] Jacob D Abernethy and Jun-Kun Wang. On frank-wolfe and equilibrium computation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6584–6593, 2017.

[2] Marek Adamczyk, Fabrizio Grandoni, and Joydeep Mukherjee. Improved approximation algorithms for stochastic matching. In *European Symposium on Algorithms (ESA)*, pages 1–12. Springer, 2015.

[3] Marek Adamczyk, Maxim Sviridenko, and Justin Ward. Submodular stochastic probing on matroids. *Mathematics of Operations Research*, 41(3):1022–1038, 2016.

[4] Alekh Agarwal, Miroslav Dudík, Satyen Kale, John Langford, and Robert E. Schapire. Contextual bandit learning with predictable rewards. In *15th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, pages 19–26, 2012.

[5] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *31st Intl. Conf. on Machine Learning (ICML)*, 2014.

[6] Alekh Agarwal, Sarah Bird, Markus Cozowicz, Miro Dudik, Luong Hoang, John Langford, Lihong Li, Dan Melamed, Gal Oshri, Siddhartha Sen, and Aleksandrs Slivkins. Multiworld testing: A system for experimentation, learning, and decision-making, 2016. A white paper, available at `https://github.com/Microsoft/mwt-ds/raw/master/images/MWT-WhitePaper.pdf`.

[7] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. *Fairness, Accountability, and Transparency in Machine Learning (FATML)*, 2017.

[8] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011.

[9] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. Diversifying search results. In *Proceedings of the second ACM international conference on web search and data mining*, pages 5–14. ACM, 2009.

[10] Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1405–1424. SIAM, 2014.

[11] Shipra Agrawal and Nikhil R. Devanur. Bandits with concave rewards and convex knapsacks. In *15th ACM Conf. on Economics and Computation (ACM EC)*, 2014.

[12] Shipra Agrawal and Nikhil R. Devanur. Linear contextual bandits with knapsacks. In *29th Advances in Neural Information Processing Systems (NIPS)*, 2016.

[13] Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A dynamic near-optimal algorithm for online linear programming. *Operations Research*, 62(4):876–890, 2014.

[14] Shipra Agrawal, Nikhil R. Devanur, and Lihong Li. An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives. In *29th Conf. on Learning Theory (COLT)*, 2016.

[15] Faez Ahmed, John P Dickerson, and Mark Fuge. Diverse weighted bipartite b-matching. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 35–41. AAAI Press, 2017.

[16] Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 18–35. ACM, 2012.

[17] Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. The online stochastic generalized assignment problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 11–25. Springer, 2013.

[18] Noga Alon, Baruch Awerbuch, and Yossi Azar. The online set cover prob-

lem. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 100–105. ACM, 2003.

[19] Venkatachalam Anantharam, Pravin Varaiya, and Jean Walrand. Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-part i: Iid rewards. *IEEE Transactions on Automatic Control*, 32(11): 968–976, 1987.

[20] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, 56(2):5, 2009.

[21] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8 (1):121–164, 2012.

[22] Arash Asadpour, Michel X Goemans, Aleksander Madry, Shayan Oveis Gharan, and Amin Saberi. An o (log n/log log n)-approximation algorithm for the asymmetric traveling salesman problem. In *SODA*, volume 10, pages 379–389. SIAM, 2010.

[23] Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Minimax policies for combinatorial prediction games. In *24th Conf. on Learning Theory (COLT)*, pages 107–132, 2011.

[24] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[25] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002. Preliminary version in *36th IEEE FOCS*, 1995.

[26] Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 542–547. IEEE, 1997.

[27] Yossi Azar, Niv Buchbinder, TH Hubert Chan, Shahar Chen, Ilan Reuven Cohen, Anupam Gupta, Zhiyi Huang, Ning Kang, Viswanath Nagarajan, and Joseph Naor. Online algorithms for covering and packing problems with convex objectives. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 148–157. IEEE, 2016.

[28] Moshe Babaioff, Shaddin Dughmi, Robert D. Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. *ACM Trans. on Economics and Computation*, 3(1):4, 2015. Special issue for *13th ACM EC*, 2012.

[29] Ashwinkumar Badanidiyuru. Buyback problem-approximate matroid intersection with cancellation costs. In *International Colloquium on Automata, Languages, and Programming*, pages 379–390. Springer, 2011.

[30] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Yaron Singer. Learning on a budget: posted price mechanisms for online procurement. In *13th ACM Conf. on Electronic Commerce (EC)*, pages 128–145, 2012.

[31] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Re-

sourceful contextual bandits. In *27th Conf. on Learning Theory (COLT)*, 2014.

[32] Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680. ACM, 2014.

[33] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. *J. of the ACM*, 65(3), 2018. Preliminary version in *FOCS 2013*.

[34] Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph Naor. A polylogarithmic-competitive algorithm for the k-server problem. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 267–276. IEEE, 2011.

[35] Nikhil Bansal, Nitish Korula, Viswanath Nagarajan, and Aravind Srinivasan. Solving packing integer programs via randomized rounding with alterations. *Theory of Computing*, 8(1):533–565, 2012.

[36] Ahron Ben-Tal and Arkadi Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2. Siam, 2001.

[37] Dirk Bergemann and Juuso Välimäki. Bandit Problems. In Steven Durlauf

and Larry Blume, editors, *The New Palgrave Dictionary of Economics, 2nd ed.* Macmillan Press, 2006.

[38] Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57: 1407–1420, 2009.

[39] Omar Besbes and Assaf J. Zeevi. Blind network revenue management. *Operations Research*, 60(6):1537–1550, 2012.

[40] Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandit algorithms with supervised learning guarantees. In *14th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[41] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[42] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. New algorithms, better bounds, and a novel model for online stochastic matching. In *24th Annual European Symposium on Algorithms (ESA 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[43] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning*, 5(1), 2012.

[44] Sébastien Bubeck, Ofer Dekel, Tomer Koren, and Yuval Peres. Bandit convex

optimization: \(\sqrt{T}\) regret in one dimension. In *28th Conf. on Learning Theory (COLT)*, pages 266–278, 2015.

[45] Sébastien Bubeck, Yin Tat Lee, and Ronen Eldan. Kernel-based methods for bandit convex optimization. In *49th ACM Symp. on Theory of Computing (STOC)*, pages 72–85. ACM, 2017.

[46] Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science*, 3(2-3):93–263, 2009.

[47] Niv Buchbinder and Joseph (Seffi) Naor. Online primal-dual algorithms for covering and packing. *Math. Oper. Res.*, 34(2):270–286, May 2009.

[48] Niv Buchbinder and Joseph Seffi Naor. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.

[49] Niv Buchbinder, Kamal Jain, and Joseph Seffi Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *European Symposium on Algorithms*, pages 253–264. Springer, 2007.

[50] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1202–1216. Society for Industrial and Applied Mathematics, 2015.

[51] Niv Buchbinder, Danny Segev, and Yevgeny Tkach. Online algorithms for maximum cardinality matching with edge arrivals. *Algorithmica*, pages 1–19, 2017.

[52] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[53] Adrian Rivera Cardoso, Jacob Abernethy, He Wang, and Huan Xu. Competing against nash equilibria in adversarially changing zero-sum games. In *36th Intl. Conf. on Machine Learning (ICML)*, pages 921–930, 2019.

[54] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge Univ. Press, 2006.

[55] Nicoló Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. Regret minimization for reserve prices in second-price auctions. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2013.

[56] TH Hubert Chan, Fei Chen, Xiaowei Wu, and Zhichao Zhao. Ranking on arbitrary graphs: Rematch via continuous lp with monotone and boundary condition constraints. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1112–1122. SIAM, 2014.

[57] TH Hubert Chan, Zhiyi Huang, Shaofeng H-C Jiang, Ning Kang, and Zhihao Gavin Tang. Online submodular maximization with free disposal: Randomization beats for partition matroids. In *Proceedings of the Twenty-Eighth*

Annual ACM-SIAM Symposium on Discrete Algorithms, pages 1204–1223. SIAM, 2017.

[58] Moses Charikar and Balaji Raghavachari. The finite capacity dial-a-ride problem. In *Proceedings 39th Annual Symposium on Foundations of Computer Science*, pages 458–467. IEEE, 1998.

[59] Chandra Chekuri, Jan Vondrak, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 575–584. IEEE, 2010.

[60] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Multi-budgeted matchings and matroid intersection via dependent rounding. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1080–1097. SIAM, 2011.

[61] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014.

[62] Cheng Chen, Lan Zheng, Venkatesh Srinivasan, Alex Thomo, Kui Wu, and Anthony Sukow. Conflict-aware weighted bipartite b-matching and its application to e-commerce. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1475–1488, 2016.

[63] Tianyi Chen and Georgios B Giannakis. Bandit convex optimization for scalable and dynamic iot management. *IEEE Internet of Things Journal*, 2018.

[64] Tianyi Chen, Qing Ling, and Georgios B Giannakis. An online convex optimization approach to proactive network resource allocation. *IEEE Transactions on Signal Processing*, 65(24):6350–6364, 2017.

[65] Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 151–159. JMLR Workshop and Conference Proceedings, 2013.

[66] Paul Christiano, Jonathan A. Kelner, Aleksander Madry, Daniel A. Spielman, and Shang-Hua Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *43rd ACM Symp. on Theory of Computing (STOC)*, pages 273–282. ACM, 2011.

[67] Richard Combes, Chong Jiang, and Rayadurgam Srikant. Bandits with budgets: Regret lower bounds and optimal algorithms. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):245–257, 2015.

[68] Richard Combes, Mohammad Sadegh Talebi Mazraeh Shahi, Alexandre Proutiere, et al. Combinatorial bandits revisited. In *Advances in Neural Information Processing Systems*, pages 2116–2124, 2015.

[69] Nikhil R Devanur and Thomas P Hayes. The adwords problem: online keyword

matching with budgeted bidders under random permutations. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 71–78. ACM, 2009.

[70] Nikhil R. Devanur and Thomas P. Hayes. The AdWords problem: Online keyword matching with budgeted bidders under random permutations. In *10th ACM Conf. on Electronic Commerce (EC)*, pages 71–78, 2009.

[71] Nikhil R Devanur and Kamal Jain. Online matching with concave returns. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 137–144. ACM, 2012.

[72] Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *12th ACM Conf. on Electronic Commerce (EC)*, pages 29–38, 2011.

[73] Nikhil R Devanur, Balasubramanian Sivan, and Yossi Azar. Asymptotically optimal algorithm for stochastic adwords. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 388–404. ACM, 2012.

[74] Nikhil R Devanur, Kamal Jain, and Robert D Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 101–107. Society for Industrial and Applied Mathematics, 2013.

[75] Nikhil R Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A

Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. *Journal of the ACM (JACM)*, 66(1):7, 2019.

[76] John P Dickerson, Karthik A Sankararaman, Aravind Srinivasan, and Pan Xu. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[77] John P Dickerson, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. Assigning tasks to workers based on historical data: Online task assignment with two-sided arrivals. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 318–326. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[78] John P Dickerson, Karthik A Sankararaman, Aravind Srinivasan, and Pan Xu. Balancing relevance and diversity in online bipartite matching via submodularity. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, AAAI '19, 2019.

[79] Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-armed bandit with budget constraint and variable costs. In *27th AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

[80] Miroslav Dudík, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Lang-

ford, Lev Reyzin, and Tong Zhang. Efficient optimal leanring for contextual bandits. In *27th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2011.

[81] Hossein Esfandiari, Nitish Korula, and Vahab Mirrokni. Online allocation with traffic spikes: Mixing adversarial and stochastic models. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 169–186. ACM, 2015.

[82] Jon Feldman, Nitish Korula, Vahab Mirrokni, S Muthukrishnan, and Martin Pál. Online ad assignment with free disposal. In *Internet and network economics*, pages 374–385. Springer, 2009.

[83] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Foundations of Computer Science (FOCS)*, pages 117–126. IEEE, 2009.

[84] Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online stochastic packing applied to display ad allocation. In *18th Annual European Symp. on Algorithms (ESA)*, pages 182–194, 2010.

[85] Amos Fiat, Richard M Karp, Michael Luby, Lyle A McGeoch, Daniel D Sleator, and Neal E Young. Competitive paging algorithms. *Journal of Algorithms*, 12(4):685–699, 1991.

[86] Abraham Flaxman, Adam Kalai, and H. Brendan McMahan. Online Convex Optimization in the Bandit Setting: Gradient Descent without a Gradient.

In *16th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 385–394, 2005.

[87] Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332. ACM, 1996.

[88] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[89] Yoav Freund and Robert E Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29(1-2):79–103, 1999.

[90] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation. In *New Frontiers in Dynamic Spectrum, 2010 IEEE Symposium on*, pages 1–9. IEEE, 2010.

[91] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations, October 2012.

[92] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM (JACM)*, 53(3):324–360, 2006.

[93] Asish Ghoshal and Jean Honorio. Learning linear structural equation models in polynomial time and sample complexity. In *International Conference on Artificial Intelligence and Statistics*, pages 1466–1475, 2018.

[94] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-Armed Bandit Allocation Indices*. John Wiley & Sons, 2011.

[95] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 982–991, 2008.

[96] Sudipta Guha and Kamesh Munagala. Multi-armed Bandits with Metric Switching Costs. In *36th Intl. Colloquium on Automata, Languages and Programming (ICALP)*, pages 496–507, 2007.

[97] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *52nd IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 827–836, 2011.

[98] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1731–1747. SIAM, 2016.

[99] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Adaptivity gaps for stochastic probing: Submodular and xos functions. In *Proceedings of the*

Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, pages 1688–1702. SIAM, 2017.

[100] András György, Levente Kocsis, Ivett Szabó, and Csaba Szepesvári. Continuous time associative bandit problems. In *20th Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 830–835, 2007.

[101] András György, Tamás Linder, Gábor Lugosi, and György Ottucsák. The online shortest path problem under partial monitoring. *J. of Machine Learning Research (JMLR)*, 8:2369–2403, 2007.

[102] Bernhard Haeupler, Vahab S Mirrokni, and Morteza Zadimoghaddam. Online stochastic weighted matching: Improved approximation algorithms. In *International Workshop on Internet and Network Economics*, pages 170–181. Springer, 2011.

[103] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4): 19, 2016.

[104] Elad Hazan. Introduction to Online Convex Optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2015.

[105] Elad Hazan and Kfir Y. Levy. Bandit convex optimization: Towards tight bounds. In *27th Advances in Neural Information Processing Systems (NIPS)*, pages 784–792, 2014.

[106] Justin Hsu, Zhiyi Huang, Aaron Roth, and Zhiwei Steven Wu. Jointly private convex programming. In *27th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 580–599, 2016.

[107] Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. How to match when all vertices arrive online. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 17–29. ACM, 2018.

[108] Zhiyi Huang, Binghui Peng, Zhihao Gavin Tang, Runzhou Tao, Xiaowei Wu, and Yuhao Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2875–2886. SIAM, 2019.

[109] Nicole Immorlica, Karthik Abinav Sankararaman, Robert Schapire, and Aleksandrs Slivkins. Adversarial bandits with knapsacks. In *Foundations of Computer Science (FOCS)*. IEEE, 2019.

[110] Russell Impagliazzo and Valentine Kabanets. Constructive proofs of concentration bounds. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 617–631. Springer, 2010.

[111] Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 39(3):624–646, 2013.

[112] David S Johnson. Approximation algorithms for combinatorial problems. *Journal of computer and system sciences*, 9(3):256–278, 1974.

[113] Satyen Kale, Lev Reyzin, and Robert E. Schapire. Non-stochastic bandit slate problems. In *24th Advances in Neural Information Processing Systems (NIPS)*, pages 1054–1062, 2010.

[114] Michael Kapralov, Ian Post, and Jan Vondrák. Online submodular welfare maximization: Greedy is optimal. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1216–1225. SIAM, 2013.

[115] Mohammad Karimi, Mario Lucic, Hamed Hassani, and Andreas Krause. Stochastic submodular maximization: The case of coverage functions. In *Advances in Neural Information Processing Systems*, pages 6853–6863, 2017.

[116] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 352–358. ACM, 1990.

[117] Sumeet Katariya, Branislav Kveton, Csaba Szepesvári, and Zheng Wen. DCM bandits: Learning to rank with multiple clicks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1215–1224, 2016.

[118] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *35th Intl. Conf. on Machine Learning (ICML)*, pages 2564–2572, 2018.

[119] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions

to combinatorial auctions. In *European Symposium on Algorithms (ESA)*, pages 589–600. Springer, 2013.

[120] Robert Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *18th Advances in Neural Information Processing Systems (NIPS)*, 2004.

[121] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Bandits and experts in metric spaces. Working paper, published at `http://arxiv.org/abs/1312.1277`, 2018. Merged and revised version of conference papers in *ACM STOC 2008* and *ACM-SIAM SODA 2010*.

[122] Nitish Korula and Martin Pál. Algorithms for secretary problems on graphs and hypergraphs. In *Automata, Languages and Programming*, pages 508–520. Springer, 2009.

[123] Akshay Krishnamurthy, Alekh Agarwal, and Miroslav Dudík. Contextual semibandits via supervised learning oracles. In *29th Advances in Neural Information Processing Systems (NIPS)*, 2016.

[124] Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson. Matroid bandits: Fast combinatorial optimization with learning. In Nevin L. Zhang and Jin Tian, editors, *UAI*, pages 420–429. AUAI Press, 2014.

[125] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine*

Learning (ICML-15), pages 767–776. JMLR Workshop and Conference Proceedings, 2015.

[126] Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvri. Tight regret bounds for stochastic combinatorial semi-bandits. In Guy Lebanon and S. V. N. Vishwanathan, editors, *AISTATS*, JMLR Workshop and Conference Proceedings. JMLR.org, 2015.

[127] John Langford and Tong Zhang. The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits. In *21st Advances in Neural Information Processing Systems (NIPS)*, 2007.

[128] Der-Horng Lee, Hao Wang, Ruey Long Cheu, and Siew Hoon Teo. Taxi dispatch system based on current demands and real-time traffic conditions. *Transportation Research Record*, 1882(1):193–200, 2004.

[129] Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 244–257. Springer, 2009.

[130] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–260, 1994.

[131] László Lovász. On the ratio of optimal integral and fractional covers. *Discrete mathematics*, 13(4):383–390, 1975.

[132] Meghna Lowalekar, Pradeep Varakantham, and Patrick Jaillet. Online spatio-temporal matching in stochastic and dynamic domains. *Artificial Intelligence*, 261:71–112, 2018.

[133] Will Ma. Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1154–1163. Society for Industrial and Applied Mathematics, 2014.

[134] Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. Trading regret for efficiency: online convex optimization with long term constraints. *J. of Machine Learning Research (JMLR)*, 13(Sep):2503–2528, 2012.

[135] Mehrdad Mahdavi, Tianbao Yang, and Rong Jin. Stochastic convex optimization with multiple objectives. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1115–1123, 2013.

[136] Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 597–606. ACM, 2011.

[137] Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012.

[138] Andrew McGregor. Finding graph matchings in data streams. In *Approxima-*

tion, Randomization and Combinatorial Optimization. Algorithms and Techniques, pages 170–181. Springer, 2005.

[139] Aranyak Mehta. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013.

[140] Aranyak Mehta and Debmalya Panigrahi. Online matching with stochastic rewards. In *Foundations of Computer Science (FOCS)*, pages 728–737. IEEE, 2012.

[141] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5):22, 2007. ISSN 0004-5411.

[142] Aranyak Mehta, Bo Waggoner, and Morteza Zadimoghaddam. Online stochastic matching with unequal probabilities. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2015.

[143] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, and Amin Karbasi. Fast constrained submodular maximization: Personalized data summarization. In *International Conference on Machine Learning*, pages 1358–1367, 2016.

[144] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization. *The Journal of Machine Learning Research*, 17(1):8330–8373, 2016.

[145] Baharan Mirzasoleiman, Morteza Zadimoghaddam, and Amin Karbasi. Fast

distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*, pages 3594–3602, 2016.

[146] Baharan Mirzasoleiman, Stefanie Jegelka, and Andreas Krause. Streaming non-monotone submodular maximization: Personalized video summarization on the fly. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[147] Marco Molinaro and R. Ravi. Geometry of online packing linear programs. In *39th Intl. Colloquium on Automata, Languages and Programming (ICALP)*, pages 701–713, 2012.

[148] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Cambridge University Press, Cambridge, 1995.

[149] Michael J Neely and Hao Yu. Online convex optimization with time-varying constraints. *arXiv preprint arXiv:1702.04783*, 2017.

[150] George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

[151] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical programming*, 14(1):265–294, 1978.

[152] Gergely Neu and Gábor Bartók. Importance weighting without importance weights: An efficient algorithm for combinatorial semi-bandits. *J. of Machine Learning Research (JMLR)*, 17(1):5355–5375, 2016.

[153] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1982.

[154] Prabhakar Raghavan and Clark D Tompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7 (4):365–374, 1987.

[155] Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. In *26th Conf. on Learning Theory (COLT)*, pages 993–1019, 2013.

[156] Alexander Rakhlin and Karthik Sridharan. BISTRO: an efficient relaxation-based method for contextual bandits. In *33nd Intl. Conf. on Machine Learning (ICML)*, 2016.

[157] Anshuka Rangi, Massimo Franceschetti, and Long Tran-Thanh. Unifying the stochastic and the adversarial bandits with knapsack. *arXiv preprint arXiv:1811.12253*, 2018.

[158] Adrian Rivera, He Wang, and Huan Xu. Online saddle point problem with applications to constrained online convex optimization. *arXiv preprint arXiv:1806.08301*, 2018.

[159] Herbert Robbins. Some Aspects of the Sequential Design of Experiments. *Bull. Amer. Math. Soc.*, 58:527–535, 1952.

[160] Ryan Rogers, Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. Inducing approximately optimal flow using truthful mediators. In *16th ACM Conf. on Electronic Commerce (EC)*, pages 471–488, 2015.

[161] Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. Watch and learn: Optimizing from revealed preferences feedback. In *48th ACM Symp. on Theory of Computing (STOC)*, pages 949–962, 2016.

[162] Aaron Roth, Aleksandrs Slivkins, Jonathan Ullman, and Zhiwei Steven Wu. Multidimensional dynamic pricing for welfare maximization. In *18th ACM Conf. on Electronic Commerce (EC)*, pages 519–536, 2017.

[163] Karthik Abinav Sankararaman and Aleksandrs Slivkins. Combinatorial semi-bandits with knapsacks. In *Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, pages 1760–1770, 2018.

[164] Karthik Abinav Sankararaman, Anand Louis, and Navin Goyal. Stability of linear structural equation models of causal inference. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, UAI '19, 2019.

[165] Karthik Abinav Sankararaman, Anand Louis, and Navin Goyal. Robust identifiability in the linear structural equation model for causal inference. In *Working Paper*, 2019.

[166] Kanthi K Sarpatwar, Baruch Schieber, and Hadas Shachnai. Constrained submodular maximization via greedy local search. *Operations Research Letters*, 47(1):1–6, 2019.

[167] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. The MIT Press, 2012.

[168] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2002.

[169] Adish Singla and Andreas Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *22nd Intl. World Wide Web Conf. (WWW)*, pages 1167–1178, 2013.

[170] Aleksandrs Slivkins. Dynamic ad allocation: Bandits with budgets. A technical report on `arxiv.org/abs/1306.0155`, June 2013.

[171] Serban Stan, Morteza Zadimoghaddam, Andreas Krause, and Amin Karbasi. Probabilistic submodular maximization in sub-linear time. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3241–3250. JMLR. org, 2017.

[172] Vasilis Syrgkanis, Alekh Agarwal, Haipeng Luo, and Robert E. Schapire. Fast convergence of regularized learning in games. In *28th Advances in Neural Information Processing Systems (NIPS)*, pages 2989–2997, 2015.

[173] Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert E. Schapire. Efficient algorithms for adversarial contextual learning. In *33nd Intl. Conf. on Machine Learning (ICML)*, 2016.

[174] Vasilis Syrgkanis, Haipeng Luo, Akshay Krishnamurthy, and Robert E. Schapire. Improved regret bounds for oracle-based adversarial contextual bandits. In *29th Advances in Neural Information Processing Systems (NIPS)*, 2016.

[175] William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4): 285–294, 1933.

[176] Yongxin Tong, Jieying She, Bolin Ding, Libin Wang, and Lei Chen. Online mobile micro-task allocation in spatial crowdsourcing. In *32nd international conference on data engineering (ICDE)*, pages 49–60. IEEE, 2016.

[177] Long Tran-Thanh, Archie Chapman, Enrique Munoz de Cote, Alex Rogers, and Nicholas R. Jennings. ϵ-first policies for budget-limited multi-armed bandits. In *24th AAAI Conference on Artificial Intelligence (AAAI)*, pages 1211–1216, 2010.

[178] Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. In *26th AAAI Conference on Artificial Intelligence (AAAI)*, pages 1134–1140, 2012.

[179] Van-Anh Truong and Xinshang Wang. Prophet inequality with correlated arrival probabilities, with application to two sided matchings. *arXiv preprint arXiv:1901.02552*, 2019.

[180] Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *Advances in neural information processing systems*, pages 1413–1421, 2014.

[181] Seeun Umboh. Online network design algorithms via hierarchical decompositions. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1373–1387. Society for Industrial and Applied Mathematics, 2015.

[182] Jun-Kun Wang and Jacob Abernethy. Acceleration through optimistic no-regret dynamics. *arXiv preprint arXiv:1807.10455*, 2018.

[183] Yajun Wang and Sam Chiu-wai Wong. Two-sided online bipartite matching and vertex cover: Beating the greedy algorithm. In *International Colloquium on Automata, Languages, and Programming*, pages 1070–1081. Springer, 2015.

[184] Zizhuo Wang, Shiming Deng, and Yinyu Ye. Close the gaps: A learning-while-doing algorithm for single-product revenue management problems. *Operations Research*, 62(2):318–331, 2014.

[185] Chen-Yu Wei and Haipeng Luo. More adaptive algorithms for adversarial bandits. In *31st Conf. on Learning Theory (COLT)*, 2018.

[186] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963, 2015.

[187] Zheng Wen, Branislav Kveton, and Azin Ashkan. Efficient learning in large-scale combinatorial semi-bandits. In Francis R. Bach and David M. Blei, editors, *ICML*, JMLR Workshop and Conference Proceedings, pages 1113–1122. JMLR.org, 2015.

[188] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.

[189] Shi Zong, Hao Ni, Kenny Sung, Nan Rosemary Ke, Zheng Wen, and Branislav Kveton. Cascading bandits for large-scale recommendation problems. 2016.