JMLR: Workshop and Conference Proceedings vol 49:1–15, 2016

An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives*

Shipra Agrawal SHIPRA@IEOR.COLUMBIA.EDU
Columbia University

Nikhil R. Devanur NIKDEV@MICROSOFT.COM
Microsoft Research

Lihong Li LIHONGLI@MICROSOFT.COM
Microsoft Research

Abstract

We consider a contextual version of multi-armed bandit problem with global knapsack constraints. In each round, the outcome of pulling an arm is a scalar reward and a resource consumption vector, both dependent on the context, and the global knapsack constraints require the total consumption for each resource to be below some pre-fixed budget. The learning agent competes with an arbitrary set of context-dependent policies. This problem was introduced by Badanidiyuru et al. (2014), who gave a computationally inefficient algorithm with near-optimal regret bounds for it. We give a *computationally efficient* algorithm for this problem with slightly better regret bounds, by generalizing the approach of Agarwal et al. (2014) for the non-constrained version of the problem. The computational time of our algorithm scales *logarithmically* in the size of the policy space. This answers the main open question of Badanidiyuru et al. (2014). We also extend our results to a variant where there are no knapsack constraints but the objective is an arbitrary Lipschitz concave function of the sum of outcome vectors.

1. Introduction

Multi-armed bandits (*e.g.*, Bubeck and Cesa-Bianchi (2012)) are a classic model for studying the exploration-exploitation tradeoff faced by a decision-making agent, which *learns* to maximize cumulative reward through sequential experimentation in an initially unknown environment. The *contextual bandit* problem (Langford and Zhang, 2008), also known as *associative reinforcement learning* (Barto and Anandan, 1985), generalizes multi-armed bandits by allowing the agent to take actions based on contextual information: in every round, the agent observes the current context, takes an action, and observes a reward that is a random variable with distribution conditioned on the context and the taken action. Despite many recent advances and successful applications of bandits, one of the major limitations of the standard setting is the lack of "global" constraints that are common in many important real-world applications. For example, actions taken by a robot arm may have different levels of power consumption, and the total power consumed by the arm is limited by the capacity of its battery. In online advertising, each advertiser has her own budget, so that her advertisement cannot be shown more than a certain number of times. In dynamic pricing, there are a certain number of objects for sale and the seller offers prices to a sequence of buyers with the goal of maximizing revenue, but the number of sales is limited by the supply.

* Extended abstract. Full version appears as [CoRR, abs/1506.03374].

Recently, a few papers started to address this limitation by considering very special cases such as a single resource with a budget constraint (Ding et al., 2013; Guha and Munagala, 2007; György et al., 2007; Madani et al., 2004; Tran-Thanh et al., 2010; Tran-Thanh et al., 2012), and application-specific bandit problems such as the ones motivated by online advertising (Chakrabarti and Vee, 2012; Pandey and Olston, 2006), dynamic pricing (Babaioff et al., 2015; Besbes and Zeevi, 2009) and crowdsourcing (Badanidiyuru et al., 2012; Singla and Krause, 2013; Slivkins and Vaughan, 2013). Subsequently, Badanidiyuru et al. (2013) introduced a general problem capturing most previous formulations. In this problem, which they called Bandits with Knapsacks (BwK), there are d different resources, each with a pre-specified budget. Each action taken by the agent results in a d-dimensional resource consumption vector, in addition to the regular (scalar) reward. The goal of the agent is to maximize the total reward, while keeping the cumulative resource consumption below the budget. The BwK model was further generalized to the BwCR (Bandits with convex Constraints and concave Rewards) model by Agrawal and Devanur (2014), which allows for arbitrary concave objective and convex constraints on the *sum* of the resource consumption vectors in all rounds. Both papers adapted the popular Upper Confidence Bound (UCB) technique to obtain near-optimal regret guarantees. However, the focus was on the *non-contextual* setting.

There has been significant recent progress (Agarwal et al., 2014; Dudík et al., 2011) in algorithms for *general* (instead of linear (Abbasi-yadkori et al., 2012; Chu et al., 2011)) contextual bandits where the context and reward can have arbitrary correlation, and the algorithm competes with some arbitrary set of context-dependent policies. Dudík et al. (2011) achieved the optimal regret bound for this remarkably general contextual bandits problem, assuming access to the policy set only through a linear optimization oracle, instead of explicit enumeration of all policies as in previous work (Auer et al., 2002; Beygelzimer et al., 2011). However, the algorithm presented in Dudík et al. (2011) was not tractable in practice, as it makes too many calls to the optimization oracle. Agarwal et al. (2014) presented a simpler and computationally efficient algorithm, with a running time that scales as the square-root of the *logarithm* of the policy space size, and achieves an optimal regret bound.

Combining contexts and resource constraints, Agrawal and Devanur (2014) also considered a *static linear* contextual version of BwCR where the expected reward was linear in the context.[1] Wu et al. (2015) considered the special case of random *linear* contextual bandits with a single budget constraint, and gave near-optimal regret guarantees for it. Badanidiyuru et al. (2014) extended the general contextual version of bandits with arbitrary policy sets to allow budget constraints, thus obtaining a contextual version of BwK, a problem they called Resourceful Contextual Bandits (RCB). We will refer to this problem as CBwK (Contextual Bandits with Knapsacks), to be consistent with the naming of related problems defined in the paper. They gave a computationally *inefficient* algorithm, based on Dudík et al. (2011), with a regret that was optimal in most regimes. Their algorithm was defined as a mapping from the history and the context to an action, but the computational issue of finding this mapping was not addressed. They posed an open question of achieving computational efficiency while maintaining a similar or even a sub-optimal regret.

Main Contributions. In this paper, we present a simple and *computationally efficient* algorithm for CBwK/RCB, based on the algorithm of Agarwal et al. (2014). Similar to Agarwal et al. (2014), the running time of our algorithm scales as the square-root of the *logarithm* of the size of the policy

1. In particular, each arm is associated with a fixed vector and the resulting outcomes for this arm have expected value linear in this vector.

set,[2] *thus resolving the main open question posed by Badanidiyuru et al. (2014).* Our algorithm even improves the regret bound of Badanidiyuru et al. (2014) by a factor of $\sqrt{d}$. Another improvement over Badanidiyuru et al. (2014) is that while they need to know the marginal distribution of contexts, our algorithm does not. A key feature of our techniques is that we need to modify the algorithm in Agarwal et al. (2014) in a very minimal way — in an almost blackbox fashion — thus retaining the structural simplicity of the algorithm while obtaining substantially more general results.

We extend our algorithm to a variant of the problem, which we call Contextual Bandits with concave Rewards (CBwR): in every round, the agent observes a context, takes one of K actions and then observes a d-dimensional outcome vector, and the goal is to maximize an arbitrary Lipschitz concave function of the average of the outcome vectors; there are no constraints. This allows for many more interesting applications, some of which were discussed in Agrawal and Devanur (2014). This setting is also substantially more general than the contextual version considered in Agrawal and Devanur (2014), where the context was fixed and the dependence was assumed to be linear.

Organization. In Section 2, we define the CBwK problem, and state our regret bound as Theorem 2. The algorithm is detailed in Section 3, and an overview of the regret analysis is in Section 4. In Section 5, we present CBwR, the problem with concave rewards, state the guaranteed regret bounds, and outline the differences in the algorithm and the analysis. Complete proofs and other details are provided in the full version of the paper (Agrawal et al., 2016).

2. Preliminaries and Main Results

CBwK. The CBwK problem was introduced by Badanidiyuru et al. (2014), under the name of Resourceful Contextual Bandits (RCB). We now define this problem.

Let A be a finite set of K actions and X be a space of possible contexts (the analogue of a feature space in supervised learning). To begin with, the algorithm is given a budget $B \in \Re_+$. We then proceed in rounds: in every round $t \in [T]$, the algorithm observes context $x_t \in X$, chooses an action $a_t \in A$, and observes a reward $r_t(a_t) \in [0,1]$ and a d-dimensional consumption vector $\mathbf{v}_t(a_t) \in [0,1]^d$. The objective is to take actions that maximize the total reward, $\sum_{t=1}^{T} r_t(a_t)$, while making sure that the consumption does not exceed the budget, i.e., $\sum_{t=1}^{T} \mathbf{v}_t(a_t) \leq B\mathbf{1}$.[3] The algorithm stops either after T rounds or when the budget is exceeded in one of the dimensions, whichever occurs first. We assume that one of the actions is a "no-op" action, i.e., it always gives a reward of 0 and a consumption vector of all 0s. Furthermore, we make a stochastic assumption that the context, the reward, and the consumption vectors $(x_t, \{r_t(a), \mathbf{v}_t(a) : a \in A\})$ for $t = 1, 2, \ldots, T$ are drawn *i.i.d.* (independent and identically distributed) from a distribution $\mathcal{D}$ over $X \times [0,1]^A \times [0,1]^{d \times A}$. The distribution $\mathcal{D}$ is unknown to the algorithm.

Policy Set. Following previous work (Agarwal et al., 2014; Badanidiyuru et al., 2014; Dudík et al., 2011), our algorithms compete with an arbitrary set of policies. Let $\Pi \subseteq A^X$ be a finite set of policies[4] that map contexts $x \in X$ to actions $a \in A$. We assume that the policy set contains a "no-op" policy that always selects the no-op action regardless of the context. With global constraints,

2. Access to the policy set is via an "arg max oracle", as in Agarwal et al. (2014).
3. More generally, different dimensions could have different budgets, but this formulation is without loss of generality: scale the units of all dimensions so that all the budgets are equal to the smallest one. This preserves the requirement that the vectors are in $[0,1]^d$.
4. The policies may be randomized in general, but for our results, we may assume without loss of generality that they are deterministic. As observed by Badanidiyuru et al. (2014), we may replace randomized policies with deterministic

distributions over policies in Π could be strictly more powerful than any policy in Π itself.[5] Our algorithms compete with this more powerful set, which is a stronger guarantee than simply competing with fixed policies in Π. For this purpose, define $\mathcal{C}(\Pi) := \{P \in [0,1]^{\Pi} : \sum_{\pi\in\Pi} P(\pi) = 1\}$ as the set of all convex combinations of policies in Π. For a context $x \in X$, choosing actions with $P \in \mathcal{C}(\Pi)$ is equivalent to following a randomized policy that selects action $a \in A$ with probability $P(a|x) = \sum_{\pi\in\Pi:\pi(x)=a} P(\pi)$; we therefore also refer to P as a (mixed) policy. Similarly, define $\mathcal{C}_0(\Pi) := \{P \in [0,1]^{\Pi} : \sum_{\pi\in\Pi} P(\pi) \leq 1\}$ as the set of all non-negative weights over Π, which sum to *at most* 1. Clearly, $\mathcal{C}(\Pi) \subset \mathcal{C}_0(\Pi)$.

Benchmark and Regret. The benchmark for this problem is an optimal static mixed policy, where the budgets are required to be satisfied in expectation only. Let $R(P) := \mathbb{E}_{(x,r,\mathbf{v})\sim\mathcal{D}}[\mathbb{E}_{\pi\sim P}[r(\pi(x))]]$ and $\mathbf{V}(P) := \mathbb{E}_{(x,r,\mathbf{v})\sim\mathcal{D}}[\mathbb{E}_{\pi\sim P}[\mathbf{v}(\pi(x))]]$ denote respectively the expected reward and consumption vector for policy $P \in \mathcal{C}(\Pi)$. We call a policy $P \in \mathcal{C}(\Pi)$ a *feasible* policy if $T\mathbf{V}(P) \leq B\mathbf{1}$. Note that there always exists a feasible policy in $\mathcal{C}(\Pi)$, because of the no-op policy. Define an optimal policy $P^* \in \mathcal{C}(\Pi)$ as a feasible policy that maximizes the expected reward:

$$P^* = \arg\max_{P\in\mathcal{C}(\Pi)} TR(P) \quad \text{s.t.} \quad T\mathbf{V}(P) \leq B\mathbf{1}. \tag{1}$$

The reward of this optimal policy is denoted by $\text{OPT} := TR(P^*)$. We are interested in minimizing the *regret*, defined as

$$\text{regret}(T) := \text{OPT} - \textstyle\sum_{t=1}^{T} r_t(a_t). \tag{2}$$

AMO. Since the policy set Π is extremely large in most interesting applications, accessing it by explicit enumeration is impractical. For the purpose of efficient implementation, we instead only access Π via a maximization oracle. Employing such an oracle is common when considering contextual bandits with an arbitrary set of policies (Agarwal et al., 2014; Dudík et al., 2011; Langford and Zhang, 2008). Following previous work, we call this oracle an "arg max oracle", or AMO.

Definition 1 *For a set of policies Π, the arg max oracle (AMO) is an algorithm, which for any sequence of contexts and rewards, $(x_1, r_1), \ldots, (x_t, r_t) \in X \times [0,1]^A$, returns*

$$\arg\max_{\pi\in\Pi} \textstyle\sum_{\tau=1}^{t} r_\tau(\pi(x_\tau)) \tag{3}$$

Main Results. Our main result is a *computationally efficient low-regret algorithm for CBwK*. Furthermore, we improve the regret bound of Badanidiyuru et al. (2014) by a $\sqrt{d}$ factor; they present a detailed discussion on the optimality of the dependence on K and T in this bound.

Theorem 2 *For the CBwK problem, $\forall \delta > 0$, there is a polynomial-time algorithm that makes $\tilde{O}(d\sqrt{KT\ln(|\Pi|)})$ calls to AMO, and with probability at least $1-\delta$ has regret*

$$\textit{regret}(T) = O\left(\tfrac{OPT}{B} + 1\right)\sqrt{KT\ln(dT|\Pi|/\delta)}.$$

Note that the above regret bound is meaningful only for $B > \Omega(\sqrt{KT\ln(dT|\Pi|/\delta)})$, therefore in the rest of the paper we assume that $B > c'\sqrt{KT\ln(dT|\Pi|/\delta)})$ for some large enough constant c'. We also extend our results to a version with a concave reward function, as outlined in Section 5. For the rest of the paper, we treat $\delta > 0$ as fixed, and define quantities that depend on δ.

policies by appending a random seed to the context. This blows up the size of the context space which does not appear in our regret bounds.

5. E.g., consider two policies that both give reward 1, but each consume 1 unit of a different resource. The optimum solution is to mix uniformly between the two, which does twice as well as using any single policy.

3. Algorithm for the CBwK problem

From previous work on multi-armed bandits, we know that the key challenges in finding the "right" policy are that (1) it should concentrate fast enough on the empirically best policy (based on data observed so far), (2) the probability of choosing an action must be large enough to enable sufficient exploration, and (3) it should be efficiently computable. Agarwal et al. (2014) show that all these can be addressed by solving a properly defined optimization problem, with help of an AMO. We have the additional technical challenge of dealing with global constraints. As mentioned earlier, one complication that arises right away is that due to the knapsack constraints, the algorithm has to compete against the best *mixed* policy in Π, rather than the best pure policy. In the following, we will highlight the main technical difficulties we encounter, and our solution to these difficulties.

Some definitions are in place before we describe the algorithm. Let H_t denote the history of chosen actions and observations before time t, consisting of records of the form $(x_\tau, a_\tau, r_\tau(a_\tau), \mathbf{v}_\tau(a_\tau), p_\tau(a_\tau))$, where $x_\tau, a_\tau, r_\tau(a_\tau), \mathbf{v}_\tau(a_\tau)$ denote, respectively, the context, action taken, reward and consumption vector observed at time τ, and $p_\tau(a_\tau)$ denotes the probability at which action a_τ was taken. (Recall that our algorithm selects actions in a randomized way using a mixed policy.) Although H_t contains observation vectors only for *chosen* actions, it can be "completed" using the trick of importance sampling: for every $(x_\tau, a_\tau, r_\tau(a_\tau), \mathbf{v}_\tau(a_\tau), p_\tau(a_\tau)) \in H_t$, define the fictitious observation vectors $\hat{r}_\tau \in [0,1]^A, \hat{\mathbf{v}}_\tau \in [0,1]^{d\times A}$ by:

$$\hat{r}_\tau(a) := \frac{r_\tau(a_\tau)}{p_\tau(a_\tau)} \mathbb{I}\{a_\tau = a\} ,$$
$$\hat{\mathbf{v}}_\tau(a) := \frac{\mathbf{v}_\tau(a_\tau)}{p_\tau(a_\tau)} \mathbb{I}\{a_\tau = a\} .$$

Clearly, $\hat{r}_\tau, \hat{\mathbf{v}}_\tau$ are *unbiased* estimator of $r_\tau, \mathbf{v}_\tau$: for every a, $\mathbb{E}_{a_\tau}[\hat{r}_\tau(a)] = r_\tau(a), \mathbb{E}_{a_\tau}[\hat{\mathbf{v}}_\tau(a)] = \mathbf{v}_\tau(a)$, where the expectations are over randomization in selecting a_τ.

With the "completed" history, it is straightforward to obtain an unbiased estimate of expected reward vector and expected consumption vector for every policy $P \in \mathcal{C}(\Pi)$:

$$\hat{R}_t(P) := \mathbb{E}_{\tau\sim[t],\pi\sim P}\left[\hat{r}_\tau(\pi(x_\tau))\right] ,$$
$$\hat{\mathbf{V}}_t(P) := \mathbb{E}_{\tau\sim[t],\pi\sim P}\left[\hat{\mathbf{v}}_\tau(\pi(x_\tau))\right] .$$

The convenient notation $\tau \sim [t]$ above, indicating that τ is drawn uniformly at random from the set of integers $\{1, 2, \ldots, t\}$, simply means averaging over time up to step t. It is easy to verify that $\mathbb{E}[\hat{R}_t(P)] = R(P)$, and $\mathbb{E}[\hat{\mathbf{V}}_t(P)] = \mathbf{V}(P)$.

Given these estimates, we construct an optimization problem (OP) which aims to find a mixed policy that has a small "empirical regret", and at the same time provides sufficient exploration over "good" policies. The optimization problem uses a quantity $\widehat{\mathrm{Reg}}_t(P)$, "the empirical regret of policy P", to characterize good policies. Agarwal et al. (2014) define $\widehat{\mathrm{Reg}}_t(P)$ as simply the difference between the empirical reward estimate of policy P and that of the policy with the highest empirical reward. Thus, good policies were characterized as those with high reward. For our problem, however, a policy could have a high reward while its consumption violates the knapsack constraints by a large margin. Such a policy should *not* be considered a good policy. A key challenge in this problem is therefore to define a single quantity that captures the "goodness" of a policy by appropriately combining rewards and consumption vectors.

We define quantities $\mathrm{Reg}(P)$ (and the corresponding empirical estimate $\widehat{\mathrm{Reg}}_t(P)$ up to round t) of $P \in \mathcal{C}(\Pi)$ by combining the regret in reward and constraint violation using a multiplier "Z". The multiplier captures the sensitivity of the problem to violation in knapsack constraints. It is easy to observe from (1) that increasing the knapsack size from B to $(1+\epsilon)B$ can increase the optimal to atmost $(1+\epsilon)$OPT. It follows that if a policy violates any knapsack constraint by γ, it can achieve at most $\frac{\mathrm{OPT}}{B}\gamma$ more reward than OPT. More precisely,

Lemma 3 *For any b, let $OPT(b)$ denote the value of an optimal solution of (1) when the budget is set as b. Then, for any $b \geq 0$, $\gamma \geq 0$,*

$$OPT(b+\gamma) \leq OPT(b) + \tfrac{OPT(b)}{b}\gamma\,. \tag{4}$$

We use this observation to set Z as an estimate of $\frac{\mathrm{OPT}}{B}$. We do this by using the outcomes of the first

$$T_0 := \tfrac{12KT}{B} \ln \tfrac{d|\Pi|}{\delta}$$

rounds, during which we do *pure exploration* (*i.e.*, play an action in A uniformly at random). For notational convenience, in our algorithm description we will index these initial T_0 exploration rounds as $t = -(T_0-1), -(T_0-2), \ldots, 0$, so that the major component of the algorithm can be started from $t = 1$ and runs until $t = T - T_0$. The following lemma provides a bound on the Z that we estimate. Its proof appears in the full version of the paper (Agrawal et al., 2016).

Lemma 4 *For any B, using the first $T_0 = \frac{12KT}{B} \ln \frac{d|\Pi|}{\delta}$ rounds of pure exploration, one can compute a quantity Z such that with probability at least $1-\delta$,*

$$\max\{\tfrac{4OPT}{B}, 1\} \leq Z \leq \tfrac{24OPT}{B} + 8.$$

Now, to define $\mathrm{Reg}(P)$ and $\widehat{\mathrm{Reg}}_t(P)$, we combine regret in reward and constraint violation using the constant Z as computed above. In these definitions, we use a smaller budget amount

$$B' := B - T_0 - c\sqrt{KT\ln(T|\Pi|/\delta)},$$

for a large enough constant c to be specified later. Here, the budget needed to be decreased by T_0 to account for budget consumed in the first T_0 exploration rounds. We use a further smaller budget amount to ensure that with high probability $(1-\delta)$ our algorithm will not abort before the end of time horizon $(T - T_0)$, due to budget violation. For any vector $\mathbf{v} \in \mathbb{R}^d$, let $\phi(\mathbf{v}, B')$ denote the amount by which the vector $\mathbf{v}$ violates the budget B', *i.e.*,

$$\phi(\mathbf{v}, B') := \max_{j=1,\ldots,d} \left(v_j - \tfrac{B'}{T}\right)^+.$$

Let P' denote the optimal policy when budget amount is B', *i.e.*,

$$P' := \arg\max_{P \in \mathcal{C}(\Pi)} TR(P) \quad \text{s.t.} \quad T\mathbf{V}(P) \leq B'\mathbf{1}.$$

And, let P_t denote the empirically optimal policy for the combination of reward and budget violation, defined as:

$$P_t := \arg\max_{P \in \mathcal{C}(\Pi)} \hat{R}_t(P) - Z\phi(\hat{\mathbf{V}}_t(P), B'). \tag{5}$$

We define

$$\mathrm{Reg}(P) := \tfrac{1}{Z+1}(R(P') - R(P) + Z\phi(\mathbf{V}(P), B')),$$

$$\widehat{\mathrm{Reg}}_t(P) := \tfrac{1}{(Z+1)}\left[\hat{R}_t(P_t) - Z\phi(\hat{\mathbf{V}}_t(P_t), B') - \left(\hat{R}_t(P) - Z\phi(\hat{\mathbf{V}}_t(P), B')\right)\right].$$

Note that $\mathrm{Reg}(P') = 0$ and $\widehat{\mathrm{Reg}}_t(P_t) = 0$ by definition.

We are now ready to describe the optimization problem, (OP). This is essentially the same as the optimization problem solved in Agarwal et al. (2014), except for the new definition of $\widehat{\mathrm{Reg}}_t(P)$, which was described above. It aims to find a mixed policy $Q \in \mathcal{C}_0(\Pi)$. This is equivalent to finding a $Q' \in \mathcal{C}(\Pi)$ and $\alpha \in [0,1]$, and returning $Q = \alpha Q'$. Let Q^μ denote a smoothed projection of Q, assigning minimum probability μ to every action: $Q^\mu(a|x) := (1 - K\mu)Q(a|x) + \mu$. (OP) depends on the history up to some time t, and a parameter μ_m that will be set by the algorithm. In the rest of the paper, for convenience, we define a constant $\psi := 100$.

Optimization Problem (OP)

Given: H_t, μ_m, and ψ.
Let $b_P := \frac{\widehat{\mathrm{Reg}}_t(P)}{\psi\mu_m}, \forall P \in \mathcal{C}(\Pi)$.
Find a $Q' \in \mathcal{C}(\Pi)$, and an $\alpha \in [0,1]$, such that the following inequalities hold. Let $Q = \alpha Q'$.

$$\alpha \cdot b_{Q'} \leq 2K,$$

$$\forall P \in \mathcal{C}(\Pi) : \mathbb{E}_{\tau\sim[t]}\mathbb{E}_{\pi\sim P}\left[\frac{1}{Q^{\mu_m}(\pi(x_\tau)|x_\tau)}\right] \leq b_P + 2K.$$

The first constraint in (OP) is to ensure that, under Q, $\widehat{\mathrm{Reg}}_t$ is "small". In the second constraint, the left-hand side, as shown in the analysis, is an upper bound on the variance of estimates $\hat{R}_t(P), \hat{\mathbf{V}}_t(P)$. These two constraints are critical for deriving the regret bound in Section 4. We give an algorithm that efficiently finds a feasible solution to (OP) (and also shows that a feasible solution always exists).

We are now ready to describe the full algorithm, which is summarized in Algorithm 1. The main body of the algorithm shares the same structure as the ILOVETOCONBANDITS algorithm for contextual bandits (Agarwal et al., 2014), with important changes necessary to deal with the knapsack constraints. We use the first T_0 rounds to do pure exploration and calculate Z as given by Lemma 4. (These time steps are indexed from $-(T_0 - 1)$ to 0.) The algorithm then proceeds in epochs with pre-defined lengths; epoch m consists of time steps indexed from $\tau_{m-1} + 1$ to τ_m, inclusively. The algorithm can work with any epoch schedule that satisfies $\tau_m < \tau_{m+1} \leq 2\tau_m$. Our results hold for the schedule where $\tau_m = 2^m$. However, the algorithm can choose to solve (OP) more frequently than what we use here to get a lower regret (but still within constant factors), at the cost of higher computational time. At the end of an epoch m, it computes a mixed policy in $Q_m \in \mathcal{C}_0(\Pi)$ by solving an instance of OP, which is then used for the entire next epoch. Additionally, at the end of every epoch m, the algorithm computes the empirically best policy P_{τ_m} as defined in Equation (5), which the algorithm uses as the default policy in the sampling process defined below. P_0 can be chosen arbitrarily, e.g., as uniform policy.

The sampling process, $\mathsf{Sample}(x, Q, P, \mu)$ in Step 8, samples an action from the computed mixed policy. It takes the following as input: x (context), $Q \in \mathcal{C}_0(\Pi)$ (mixed policy returned by the

optimization problem (OP) for the current epoch), P (default mixed policy), and $\mu > 0$ (a scalar for minimum action-selection probability). Since Q may not be a proper distribution (as its weights may sum to a number less than 1), Sample first computes $\tilde{Q} \in \mathcal{C}(\Pi)$, by assigning any remaining mass (from Q) to the default policy P. Then, it picks an action from the smoothed projection $\tilde{Q}^\mu$ of this distribution defined as: $\tilde{Q}^\mu(a|x) = (1 - K\mu)\tilde{Q}(a|x) + \mu, \forall a \in A$.

The algorithm aborts (in Step 10) if the budget B is consumed for any resource.

Algorithm 1 (Adapted from ILOVETOCONBANDITS of Agarwal et al. (2014))

Input Epoch schedule $0 = \tau_0 < \tau_1 < \tau_2 < \cdots$ such that $\tau_m < \tau_{m+1} \leq 2\tau_m$, allowed failure probability $\delta \in (0, 1)$.

1: Initialize weights $Q_0 := \mathbf{0} \in \mathcal{C}_0(\Pi)$, $P_0 \in \mathcal{C}(\Pi)$ and epoch $m := 1$.
Define $\mu_m := \min\{\frac{1}{2K}, \sqrt{\ln(16\tau_m^2(d+1)|\Pi|/\delta)/(K\tau_m)}\}$ for all $m \geq 0$.
2: **for round** $t = -(T_0 - 1), \ldots, 0$ **do**
3: Select action a_t uniformly at random from the set of all arms.
4: **end for**
5: Compute Z as in Lemma 4.
6: **for round** $t = 1, 2, \ldots$ **do**
7: Observe context $x_t \in X$.
8: $(a_t, p_t(a_t)) := \mathsf{Sample}(x_t, Q_{m-1}, P_{\tau_{m-1}}, \mu_{m-1})$.
9: Select action a_t and observe reward $r_t(a_t) \in [0, 1]$ and consumption $\mathbf{v}_t(a_t)$.
10: Abort unless $\sum_{\tau=-(T_0-1)}^{t} \mathbf{v}_\tau(a_\tau) < B\mathbf{1}$.
11: **if** $t = \tau_m$ **then**
12: Let Q_m be a solution to (OP) with history H_t and minimum probability μ_m.
13: $m := m + 1$.
14: **end if**
15: **end for**

3.1. Computation complexity: Solving (OP) using AMO

Algorithm 1 requires solving (OP) at the end of every epoch. Agarwal et al. (2014) gave an algorithm that solves (OP) using access to the AMO. We use a similar algorithm, except that calls to the AMO are now replaced by calls to a knapsack constrained optimization problem over the empirical distribution. This optimization problem is identical in structure to the optimization problem defining P_t in (5), which we need to solve also. We can solve both of these problems using AMO, as outlined below.

We rewrite (5) as a linear optimization problem where the domain is the intersection of two polytopes. The domain is $[0, 1]^{d+2}$; we represent a point in this domain as $(x, \mathbf{y}, \lambda)$, where x and λ are scalars and $\mathbf{y}$ is a vector in d dimensions. Let

$$K_1 := \{(x, \mathbf{y}, \lambda) : x = \hat{R}_t(P), \mathbf{y} = \hat{\mathbf{V}}_t(P) \text{ for some } P \in \mathcal{C}(\Pi), \lambda \in [0, 1]\},$$

be the set of all reward, consumption vectors achievable on the empirical outcomes upto time t, through some policy in $\mathcal{C}(\Pi)$. Let

$$K_2 := \{(x, \mathbf{y}, \lambda) : \mathbf{y} \leq (B'/T + \lambda)\mathbf{1}\} \cap [0, 1]^{d+2},$$

be the constraint set, given by relaxaing the knapsack constraints by λ. Now (5) is equivalent to

$$\max x - Z\lambda \text{ such that } (x, \mathbf{y}, \lambda) \in K_1 \cap K_2. \tag{6}$$

Recently, Lee et al. (2015, Theorem 49) gave a fast algorithm to solve problems of the kind above, given access to oracles that solve linear optimization problems over K_1 and K_2.[6] The algorithm makes $\tilde{O}(d)$ calls to these oracles, and takes an additional $\tilde{O}(d^3)$ running time.[7] A linear optimization problem over K_1 is equivalent to the AMO; the linear function defines the "rewards" that the AMO optimizes for.[8] A linear optimization problem over K_2 is trivial to solve. As an aside, a solution $Q \in \mathcal{C}_0(\Pi)$ output by this algorithm has support equal to the policies output by the AMO during the run of the algorithm, and hence has size $\tilde{O}(d)$.

Using this, (OP) can be solved using $O(d\sqrt{KT\ln(|\Pi|)})$ calls to the AMO at the end of every epoch, and (5) can be solved using $O(d)$ calls, giving a total of $\tilde{O}(d\sqrt{KT\ln(|\Pi|)})$ calls to AMO. The complete algorithm to solve (OP) is in full version of the paper Agrawal et al. (2016).

4. Regret Analysis

This section provides an outline of the proof of Theorem 2, which provides a bound on the regret of Algorithm 1. (A complete proof is given in full version of the paper (Agrawal et al., 2016).) The proof structure is similar to the proof of Agarwal et al. (2014, Theorem 2), with major differences coming from the changes necessary to deal with mixed policies and constraint violations. We defined the algorithm to minimize $\widehat{\text{Reg}}$ (through the first constraint in the optimization problem (OP)), and the first step is to show that this implies a bound on Reg as well. The alternate definitions of Reg and $\widehat{\text{Reg}}$ require a different analysis than what was in Agarwal et al. (2014), and this difference is highlighted in the proof outline of Lemma 6 below. Once we have a bound on Reg, we show that this implies a bound on the actual reward R, as well as the probability of violating the knapsack constraints.

We start by proving that the empirical average reward $\hat{R}_t(P)$ and consumption vector $\hat{\mathbf{V}}_t(P)$ for any mixed policy P are close to the true averages $R(P)$ and $\mathbf{V}(P)$ respectively. We define m_0 such that for initial epochs $m < m_0$, $\mu_m = \frac{1}{2K}$. Recall that μ_m is the minimum probability of playing any action in epoch $m+1$, defined in Step 1 of Algorithm 1. Therefore, for these initial epochs the variance of importance sampling estimates is small, and we can obtain a stronger bound on estimation error. For subsequent epochs, μ_m decreases, and we get error bounds in terms of max variance of the estimates for policy P across all epochs before time t, defined as $\mathcal{V}_t(P)$. In fact, the second constraint in the optimization problem (OP) seeks to bound this variance.

The precise definitions of above-mentioned quantities are provided in Appendix D of (Agrawal et al., 2016).

Lemma 5 *With probability* $1 - \frac{\delta}{2}$, *for all policies* $P \in \mathcal{C}(\Pi)$,

$$\max\{|\hat{R}_t(P) - R_t(P)|, \|\hat{\mathbf{V}}_t(P) - \mathbf{V}(P)\|_\infty\} \leq \begin{cases} \sqrt{\frac{8Kd_t}{t}} & t \in \textit{epoch } m_0, t \geq t_0 \\ \mathcal{V}_t(P)\mu_{m-1} + \frac{d_t}{t\mu_{m-1}}, & t \in \textit{epoch } m, m > m_0 \end{cases}$$

6. Alternately, one could use the algorithms of Vaidya (1989a,b) to solve the same problem, with a slightly weaker polynomial running time.
7. Here, $\tilde{O}$ hides terms of the order $\log^{O(1)}(d/\epsilon)$, where ϵ is the accuracy needed of the solution.
8. These rewards may not lie in $[0,1]$ but an affine transformation of the rewards can bring them into $[0,1]$ without changing the solution.

Here, $d_t = \ln(16t^2|\Pi|(d+1)/\delta), t_0 := \min\{t \in \mathbb{N} : \frac{d_t}{t} \leq \frac{1}{4K}\}$, $m_0 := \min\{m \in \mathbb{N} : \frac{d_{\tau_m}}{\tau_m} \leq \frac{1}{4K}\}$.

Now suppose the error bounds in above lemma hold. A major step is to show that, for every $P \in \mathcal{C}(\Pi)$, the empirical regret $\widehat{\text{Reg}}_t(P)$ and the actual regret $\text{Reg}(P)$ are close in a particular sense.

Lemma 6 *Assume that the events in Lemma 5 hold. Then, for all epochs $m \geq m_0$, all rounds $t \geq t_0$ in epoch m, and all policies $P \in \mathcal{C}(\Pi)$,*

$$\textit{Reg}(P) \leq 2\widehat{\textit{Reg}}_t(P) + c_0 K\mu_m, \quad \textit{and} \quad \widehat{\textit{Reg}}_t(P) \leq 2\textit{Reg}_t(P) + c_0 K\mu_m,$$

for $\textit{Reg}(P), \widehat{\textit{Reg}}_t(P)$ as defined in Section 3, and c_0 being a constant smaller than 150.

Proof [Proof Outline] The proof of above lemma is by induction, using the second constraint in (OP) to bound the variance $\mathcal{V}_t(P)$. Below, we prove the base case. This proof demonstrates the importance of appropriately chosing Z. Consider $m = m_0$, and $t \geq t_0$ in epoch m. For all $P \in \mathcal{C}(\Pi)$,

$$\begin{aligned}(Z+1)(\widehat{\text{Reg}}_t(P) - \text{Reg}(P)) \quad &= \quad \hat{R}_t(P_t) - \hat{R}_t(P) - R(P') + R(P) \\ &\quad -Z[\phi(\hat{\mathbf{V}}_t(P_t), B') - \phi(\hat{\mathbf{V}}_t(P), B') + \phi(\mathbf{V}(P), B')].\end{aligned} \tag{7}$$

We can assume that $B \geq c'\sqrt{KT\ln(dT|\Pi|/\delta)}$ for any constant c' (otherwise the regret guarantees in Theorem 2 are meaningless). Then, we have that $B \geq 2T_0 + 2c\sqrt{KT\ln(T|\Pi|/\delta)} = 2(B - B')$ implying $B' \geq \frac{B}{2}$. Also, observe that since $B \geq B'$, $\text{OPT}(B) \geq \text{OPT}(B')$. Then, by Lemma 3 and choice of Z as specified by Lemma 4, we have that for any $\gamma \geq 0$

$$\text{OPT}(B' + \gamma) \leq \text{OPT}(B') + \tfrac{Z}{2}\gamma. \tag{8}$$

Now, since P' is defined as the optimal policy for budget B', we obtain that $R(P') = \text{OPT}(B')$. Also, by definition of $\phi(\mathbf{V}(P_t), B')$, we have that $R(P_t) \leq \text{OPT}(B' + \phi(\mathbf{V}(P_t), B'))$, and therefore,

$$R(P') \geq R(P_t)) - \tfrac{Z}{2}\phi(\mathbf{V}(P_t), B') \geq R(P_t)) - Z\phi(\mathbf{V}(P_t), B').$$

Substituting in (7), we can upper bound $(Z+1)(\widehat{\text{Reg}}_t(P) - \text{Reg}(P))$ by

$$\begin{aligned}&\hat{R}_t(P_t) - \hat{R}_t(P) - R(P_t) + Z\phi(\mathbf{V}(P_t), B') + R(P) \\ &\quad -Z[\phi(\hat{\mathbf{V}}_t(P_t), B') - \phi(\hat{\mathbf{V}}_t(P), B') + \phi(\mathbf{V}(P), B')] \\ \leq \quad &|\hat{R}_t(P_t) - R(P_t)| + |\hat{R}_t(P) - R(P)| + Z\|\hat{\mathbf{V}}_t(P_t) - \mathbf{V}(P_t)\|_\infty + Z\|\hat{\mathbf{V}}_t(P) - \mathbf{V}(P)\|_\infty\end{aligned}$$

For the other side, by definition of P_t, we have that $\hat{R}(P_t)) - Z\phi(\hat{\mathbf{V}}(P_t), B') \geq \hat{R}(P) - Z\phi(\hat{\mathbf{V}}(P), B')$. Substituting in (7) as above, and using that $\phi(\mathbf{V}(P'), B') = 0$, we get a similar upper bound on $(Z+1)(\text{Reg}(P) - \widehat{\text{Reg}}_t(P))$. Now substituting bounds from Lemma 5, we obtain,

$$|\widehat{\text{Reg}}_t(P) - \text{Reg}(P)| \leq 4\sqrt{\tfrac{8Kd_t}{t}} \leq c_0 K\mu_m.$$

This completes the base case. The remaining proof is by induction, using the bounds provided by Lemma 5 for epochs $m > m_0$ in terms of variance $\mathcal{V}_t(\cdot)$, and bound on variance provided by the second constraint in (OP). The second constraint in (OP) provides a bound on the variance of any policy P in any past epoch, in terms of $\widehat{\text{Reg}}_\tau(P)$ for τ in *that* epoch; the inductive hypothesis

is used in the proof to obtain those bounds in terms of $\text{Reg}(P)$. ■

Given the above lemma, the first constraint in (OP) which bounds the estimated regret $\widehat{\text{Reg}}_t(Q)$ for the chosen mixed policy Q, directly implies an upper bound on $\text{Reg}(Q)$ for this mixed policy. Specifically, we get that for every epoch m, for mixed policy Q_m that solves (OP),

$$\text{Reg}(Q_m) \leq (c_0 + 2)K\psi\mu_m.$$

Next, we bound the regret in epoch m using above bound on $\text{Reg}(Q_{m-1})$. For simplicity of discussion, here we outline the steps for bounding regret for rewards sampled from policy Q_{m-1} in epoch m. Note that this is not precise in following ways. First, $Q_{m-1} \in \mathcal{C}_0(\Pi)$ may not be in $\mathcal{C}(\Pi)$ and therefore may not be a proper distribution (the actual sampling process puts the remaining probability on default policy P_t to obtain $\tilde{Q}_t$ at time t in epoch m). Second, the actual sampling process picks an action from smoothed projection $\tilde{Q}_t^{\mu_{m-1}}$ of $\tilde{Q}_t$. However, we ignore these technicalities here in order to get across the intuition behind the proof; these technicalities are dealt with rigorously in the complete proof provided in (Agrawal et al., 2016).

The first step is to use the above bound on $\text{Reg}(Q_{m-1})$ to show that expected reward $R(Q_{m-1})$ in epoch m is close to optimal reward $R(P^*)$. Since $\phi(\cdot, B')$ is always non-negative, by definition of $\text{Reg}(Q)$, for any Q

$$(Z+1)\text{Reg}(Q) \geq R(P') - R(Q) \geq R(P^*) - R(Q) - \tfrac{\text{OPT}}{B}\tfrac{(B-B')}{T},$$

where we used Lemma 3 to get the last inequality. If the algorithm *never aborted* due to constraint violation in Step 10, the above observation would bound the regret of the algorithm by

$$\sum_m (R(P^*) - R(Q_{m-1}))(\tau_m - \tau_{m-1}) \leq \sum_m (Z+1)(c_0+2)K\psi\mu_{m-1}(\tau_m - \tau_{m-1}) + \frac{\text{OPT}}{B}(B - B').$$

Then, using that $Z \leq O(\frac{\text{OPT}}{B})$, $B - B' = O(\sqrt{KT\ln(dT|\Pi|/\delta)})$, and properly chosen scaling factors (ψ and μ_m) result in the desired bound of $O(\frac{\text{OPT}}{B}\sqrt{KT\ln(dT|\Pi|/\delta)})$ for expected regret. An application of Azuma-Hoeffding inequality obtains the high probability regret bound as stated in Theorem 2.

To complete the proof, we show that in fact, with probability $1 - \frac{\delta}{2}$, the algorithm *is not aborted* in Step 10 due to constraint violation. This involves showing that with high probability, the algorithm's consumption (in steps $t = 1, \ldots, T_0$) above B' is bounded above by $c\sqrt{KT\ln(|\Pi|/\delta)}$, and since $B' + c\sqrt{KT\ln(|\Pi|/\delta)} + T_0 = B$, we obtain that the algorithm will satisfy the knapsack constraint with high probability. This also explains why we started with a smaller budget. More precisely, we show that for every m,

$$\phi(\mathbf{V}(Q_m), B') \leq 4(c_0 + 2)K\psi\mu_m \tag{9}$$

Recall that $\phi(\mathbf{V}(P), B')$ was defined as the maximum violation of budget $\frac{B'}{T}$ by vector $\mathbf{V}(P)$. To prove the above, we observe that due to our choice of Z, $\phi(\mathbf{V}(P), B')$ is bounded by $\text{Reg}(P)$ as follows. By Equation (8), for all $P \in \mathcal{C}(\Pi)$, $R(P') \geq R(P) - \frac{Z}{2}\phi(\mathbf{V}(P), B')$, so that

$$(Z+1)\ \text{Reg}(P) = R(P') - R(P) + Z\phi(\mathbf{V}(P), B') \geq \tfrac{Z}{2}\phi(\mathbf{V}(P), S).$$

Then, using the bound of $\text{Reg}(Q_m) \leq (c_0+2)K\psi\mu_m$, we obtain the bound in Equation (9). Summing this bound over all epochs m, and using Jensen's inequality and convexity of $\phi(\cdot, B')$, we obtain a bound on the max violation of budget constraint $\frac{B'}{T}$ by the algorithm's expected consumption vector $\frac{1}{T}\sum_m \mathbf{V}(Q_{m-1})(\tau_m - \tau_{m-1})$. This is converted to a high probability bound using Azuma-Hoeffding inequality.

5. The CBwR problem

In this section, we consider a version of the problem with a concave objective function, and show how to get an efficient algorithm for it. The **CBwR problem** is identical to the CBwK problem, except for the following. The outcome in a round is simply the vector $\mathbf{v}$, and the goal of the algorithm is to maximize $f(\frac{1}{T}\sum_{t=1}\mathbf{v}_t(a_t))$, for some concave function f defined on the domain $[0,1]^d$, and given to the algorithm ahead of time. The optimum mixed policy is now defined as

$$P^* = \arg\max_{P\in\mathcal{C}(\Pi)} f(\mathbf{V}(P)). \tag{10}$$

The optimum value is $\text{OPT} = f(\mathbf{V}(P^*))$ and we bound the average regret, which is

$$\text{avg-regret} := \text{OPT} - f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{v}_t(a_t)\right).$$

The main result of this section is an $O(1/\sqrt{T})$ regret bound for this problem. Note that the regret scales as $1/\sqrt{T}$ rather than $\sqrt{T}$ since the problem is defined in terms of the average of the vectors rather than the sum. We assume that f is represented in such a way that we can solve optimization problems of the following form in polynomial time.[9] For any given $a \in \Re^d$,

$$\max f(x) + a\cdot x : x \in [0,1]^d.$$

Theorem 7 *For the CBwR problem, if f is L-Lipschitz w.r.t. norm $\|\cdot\|$, then there is a polynomial time algorithm that makes $\tilde{O}(d\sqrt{KT\ln(|\Pi|)})$ calls to AMO, and with probability at least $1-\delta$ has regret*

$$\textit{avg-regret}(T) = O\left(\frac{\|\mathbf{1}_d\|L}{\sqrt{T}}\left(\sqrt{K\ln(T|\Pi|/\delta)} + \sqrt{\ln(d/\delta)}\right)\right).$$

Remark 8 *A special case of this problem is when there are only constraints, in which case f could be defined as the negative of the distance from the constraint set. Further, one could handle both concave objective function and convex constraints as follows. Suppose that we wish to maximize $h(\frac{1}{T}\sum_{t=1}\mathbf{v}_t(a_t))$, subject to the constraint that $\frac{1}{T}\sum_{t=1}\mathbf{v}_t(a_t) \in S$, for some L-Lipschitz concave function h and a convex set S. Further, suppose that we had a good estimate of the optimum achieved by a static mixed policy, i.e.,*

$$\textit{OPT}' := \max_{P\in\mathcal{C}(\Pi)} h(\mathbf{V}(P)) \quad \textit{s.t.} \quad \mathbf{V}(P) \in S. \tag{11}$$

For some distance function $d(\cdot, S)$ measuring distance of a point from set S, define

$$f(\mathbf{v}) := \min\left\{h(\mathbf{v}) - \textit{OPT}', -Ld(\mathbf{v}, S)\right\}.$$

9. This problem has nothing to do with contexts and policies, and only depends on the function f.

5.1. Algorithm

Since we don't have any hard constraints and don't need to estimate Z as in the case of CBwK, we can drop Steps 2–5 and Step 10 in Algorithm 1, and set $T_0 = 0$. The optimization problem (OP) is also the same, but with new definitions of $\text{Reg}(P)$, P_t and $\widehat{\text{Reg}}_t(P)$ as below. Recall that P^* is the optimal policy as given by Equation (10), and L is the Lipschitz factor for f with respect to norm $\|\cdot\|$. We now define the regret of policy $P \in \mathcal{C}(\Pi)$ as

$$\text{Reg}(P) := \tfrac{1}{\|\mathbf{1}_d\| L} \left(f(\mathbf{V}(P^*)) - f(\mathbf{V}(P)) \right).$$

The best empirical policy is now given by

$$P_t := \arg\max_{P \in \mathcal{C}(\Pi)} f(\hat{\mathbf{V}}_t(P)), \tag{12}$$

and an estimate of the regret of policy $P \in \mathcal{C}(\Pi)$ at time t is

$$\widehat{\text{Reg}}_t(P) := \tfrac{1}{\|\mathbf{1}_d\| L} (f(\hat{\mathbf{V}}_t(P_t)) - f(\hat{\mathbf{V}}_t(P))).$$

Another difference is that we need to solve a *convex* optimization problem to find P_t (as defined in (12)) once every epoch. A similar convex optimization problem needs to be solved in every iteration of a coordinate descent algorithm for solving (OP). In both cases, the problems can be cast in the form

$$\min g(x) : x \in C,$$

where g is a convex function, C is a convex set, and we are given access to a *linear optimization* oracle, that solves a problem of the form $\min c \cdot x : x \in C$. In (12) for instance, C is the set of all $\hat{\mathbf{V}}_t(P)$ for all $P \in \mathcal{C}(\Pi)$. A linear optimization oracle over this C is just an AMO as in Definition 1. We show how to efficiently solve such a convex optimization problem using cutting plane methods (Vaidya, 1989a; Lee et al., 2015), while making only $\tilde{O}(d)$ calls to the oracle.

The details are provided in the full version of the paper (Agrawal et al., 2016).

5.2. Regret Analysis: Proof of Theorem 7

We prove that Algorithm 1 and (OP) with the above new definition of $\widehat{\text{Reg}}_t(P)$ achieves regret bounds of Theorem 7 for the CBwR problem. A complete proof of this theorem is given in the full version of the paper (Agrawal et al., 2016). Here, we sketch some key steps.

The first step of the proof is to use constraints in (OP) to prove a lemma akin to Lemma 6 showing that the empirical regret $\widehat{\text{Reg}}_t(P)$ and actual regret $\text{Reg}(P)$ are close for every $P \in \mathcal{C}(\Pi)$. Therefore, the first constraint in (OP) that bounds the empirical regret $\widehat{\text{Reg}}_t(Q_m)$ of the computed policy implies a bound on the actual regret $\text{Reg}(Q_m) = \frac{1}{L\|\mathbf{1}_d\|}(f(\mathbf{V}(P^*)) - f(\mathbf{V}(Q_m)))$. Ignoring the technicalities of sampling process (which are dealt with in the complete proof), and assuming that Q_{m-1} is the policy used in epoch m, this provides a bound on regret in every epoch. Regret across epochs can be combined using Jensen's inequality which bounds the regret in expectation. Using Azuma-Hoeffding's inequality to bound deviation of expected reward vector from the actual reward vector, we obtain the high probability regret bound stated in Theorem 7.

References

Yasin Abbasi-yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, 2012.

Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert E. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *ICML*, June 2014. URL http://arxiv.org/abs/1402.0555. Full version on arXiv.

Shipra Agrawal and Nikhil R. Devanur. Bandits with concave rewards and convex knapsacks. In *EC*, 2014.

Shipra Agrawal, Nikhil R. Devanur, and Lihong Li. An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives. *CoRR*, abs/1506.03374, 2016. URL http://arxiv.org/abs/1506.03374.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multi-armed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

Moshe Babaioff, Shaddin Dughmi, Robert D. Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. *ACM Trans. Economics and Comput.*, 3(1):4, 2015. doi: 10.1145/2559152. URL http://doi.acm.org/10.1145/2559152.

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Yaron Singer. Learning on a budget: posted price mechanisms for online procurement. In *EC*, pages 128–145. ACM, 2012.

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *FOCS*, pages 207–216, 2013.

Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful contextual bandits. In *COLT*, pages 1109–1134, 2014.

Andrew G. Barto and P. Anandan. Pattern-recognizing stochastic learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(3):360–375, 1985.

Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420, 2009.

Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandit algorithms with supervised learning guarantees. In *AISTATS*, pages 19–26, 2011.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

Deepayan Chakrabarti and Erik Vee. Traffic shaping to optimize ad delivery. In *EC*, 2012.

Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual Bandits with Linear Payoff Functions. In *AISTATS*, 2011.

Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-armed bandit with budget constraint and variable costs. In *AAAI*, pages 232–238, 2013.

Miroslav Dudík, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. In *UAI*, pages 169–178, 2011.

Sudipto Guha and Kamesh Munagala. Approximation algorithms for budgeted learning problems. In *STOC*, pages 104–113, 2007.

András György, Levente Kocsis, Ivett Szabó, and Csaba Szepesvári. Continuous time associative bandit problems. In *IJCAI*, pages 830–835, 2007.

John Langford and Tong Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In *NIPS*, pages 1096–1103, 2008.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *FOCS*, pages 1049–1065. IEEE, 2015. URL http://arxiv.org/abs/1508.04874v1. Full version on arXiv .

Omid Madani, Daniel J Lizotte, and Russell Greiner. The budgeted multi-armed bandit problem. In *Learning Theory*, pages 643–645. Springer, 2004.

Sandeep Pandey and Christopher Olston. Handling advertisements of unknown quality in search advertising. In *NIPS*, pages 1065–1072, 2006.

Adish Singla and Andreas Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *WWW*, pages 1167–1178, 2013.

Aleksandrs Slivkins and Jennifer Wortman Vaughan. Online decision making in crowdsourcing markets: Theoretical challenges (position paper). *CoRR*, abs/1308.1746, 2013.

Long Tran-Thanh, Archie C. Chapman, Enrique Munoz de Cote, Alex Rogers, and Nicholas R. Jennings. Epsilon-first policies for budget-limited multi-armed bandits. In *AAAI*, 2010.

Long Tran-Thanh, Archie C. Chapman, Alex Rogers, and Nicholas R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. In *AAAI*, 2012.

Pravin M Vaidya. A new algorithm for minimizing convex functions over convex sets. In *FOCS*, pages 338–343. IEEE, 1989a.

Pravin M Vaidya. Speeding-up linear programming using fast matrix multiplication. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 332–337. IEEE, 1989b.

Huasen Wu, R. Srikant, Xin Liu, and Chong Jiang. Algorithms with logarithmic or sublinear regret for constrained contextual bandits. In *NIPS*, pages 433–441, 2015.