On Logarithmic Regret for Bandits with Knapsacks

1st Wenbo Ren
Department of CSE
The Ohio State University
Columbus, USA
ren.453@osu.edu

2nd Jia Liu
Department of ECE
The Ohio State University
Columbus, USA
liu.1736@osu.edu

3rd Ness B. Shroff
Departments of ECE and CSE
The Ohio State University
Columbus, USA
shroff.11@osu.edu

2021 55th Annual Conference on Information Sciences and Systems (CISS) | 978-1-6654-1268-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/CISS50987.2021.9400269

***Abstract*—This paper studies bandits with knapsacks (BwK). In a BwK instance, there are a set of n arms and d types of resources with limited budgets. Each pull of an arm returns a noisy reward and consumes some amount of resources in each type according to some latent distribution of this arm. The decision-maker adaptively pulls arms in order to maximize the accumulated reward gained before some type of resources runs out. We investigate logarithmic distribution-dependent regrets for the BwK problem, specifically for the instances with deterministic costs. We propose a new algorithm with regret in the form of $O(n \log T/\Delta)$ (Δ is the gap of rewards similar to that in standard MAB), which to our knowledge, is of the lowest order till now, and has the same order as the standard MAB problem when $d = 1$. Simulation results also suggest the performance improvements using our algorithms.**

***Index Terms*—multi-armed bandits, bandits with knapsacks, regret minimization**

I. Introduction

A. Background

The multi-armed bandit (MAB) model [10] is a versatile abstraction of sequential decision making under uncertainty and has been widely applied in many areas, such as communication networking [12], advertising [20], clinical trials [9], product testing [26], ranking [1], etc. In a traditional MAB problem, there are a set of arms, each of which is assumed to hold a distribution. Each pull of an arm generates an independent *reward* according to its distribution. The goal of the learner (or decision-maker) is to maximize the accumulated reward.

In this paper, we focus on bandits with knapsacks (BwK) [8]. In a BwK instance, there are n arms and d types of resources. Each arm holds a *latent distribution*. Each pull of an arm generates an independent reward and d independent *costs* according to its latent distribution and consumes each type of resources according to the costs. Each type of resources has a finite budget, and once some type of resources runs out, the learner must stop pulling. The learner wants to maximize the expected cumulated reward before stopping pulling.

BwK abstracts the decision making with one or multiple types of limited resources. When there is only one type of resource and this resource is time (i.e., number of pulls), BwK reduces to the traditional MAB. Thus, BwK can be viewed as a generalized version of MAB. One application of BwK is the dynamic ad allocation, which has been formulated as a BwK problem [27]. An advertiser has n advertisements and a budget B for these ads. There are T incoming new users, and the ad platform dynamically chooses advertisements to present to these T users. For each ad presented to and clicked by a user, the advertiser pays a certain amount of reward to the platform. We can view the advertisements as arms and the payments to the platform as rewards, and the budget of the advertiser and the number of incoming users can be modeled as two types of limited resources. BwK can also be applied to other problems, e.g., bid optimization [31], dynamic pricing [7], and network revenue management [11].

In MAB problems, there are two types of regrets that have been widely adopted: *distribution-dependent* regrets and *distribution-free* regrets. Simply speaking, a distribution-dependent regret depends on an actual instance and can better describe the asymptotic performance of the corresponding algorithm when time or the budgets go to infinity, while the distribution-free regrets usually do not assume any knowledge of the latent distributions. By far, most existing works on BwK have focused on the distribution-free regrets. However, the distribution-dependent regrets are less explored. The focus of this paper is to study the *logarithmic distribution-dependent* regrets of the BwK problem. To be specific, we focus on the instances with deterministic costs.

B. Problem Formulation

Notations. For any positive integer k and vector $\mathbf{a}$ in $\mathbb{R}^k$, we use $a(i)$ to denote the i-th component of $\mathbf{a}$, i.e., $\mathbf{a} = [a(1), a(2), ..., a(k)]^\top$. For vectors $\mathbf{a}$ in $\mathbb{R}^k$, $\mathbf{a} \geq 0$ if and only if $\forall i \in [k]$[1], $a(i) \geq 0$. For numbers a and b in $\mathbb{R}$, define $a \wedge b := \min\{a, b\}$ and $a \vee b := \max\{a, b\}$. For any positive integer k and vector $\mathbf{a} \in \mathbb{R}^k$, the l_1 norm of $\mathbf{a}$ is $||\mathbf{a}||_1 := \sum_{i\in[k]} |a(i)|$. For any $a \in \mathbb{R}$, define $(a)^+ := a \vee 0$.

Arms. There are n arms in total indexed by $1, 2, ..., n$, and d types of resources in total (referred to as resource 1, resource 2,..., resource d). For the t-th pull of arm x, we use $R_{x,t}$ to denote the reward, and use $C_{x,t}(j)$ to denote the consumption of resource j, for any j in $[d]$. We assume that $(R_{x,t}, t \in \mathbb{Z}_+)$ are i.i.d. (independently and identically distributed), $(C_{x,t}(j), t \in \mathbb{Z}_+)$ are also i.i.d., and $(R_{x,t}, x \in [n], t \in \mathbb{Z}_+)$ and $(C_{x,t}(j), x \in [n], t \in \mathbb{Z}_+, j \in [d])$ are independent across time, arms, and resources. We further assume that for any arm x and time t, $(R_{x,t}, C_{x,t}(1), C_{x,t}(2), ..., C_{x,t}(d))$ is within a same bounded support. Without loss of generality, we

[1]For any positive integer k, we define $[k] := \{1, 2, 3, ..., k\}$.

978-1-6654-1268-1/21/$31.00 ©2021 IEEE

rescale the support to $[0,1]^{d+1}$. For any arm x and $j \in [d]$, define $\mu_x := \mathbb{E}[R_{x,1}]$ as the mean reward of arm x, and $\lambda_x(j) := \mathbb{E}[C_{x,1}(j)]$ as the mean cost of resource j of arm x. We write $\vec{\lambda}_x = [\lambda_x(1), \lambda_x(2), ..., \lambda_x(d)]^\top$. The values of $(\mu_x, x \in [n])$ and $(\vec{\lambda}_x, x \in [n])$ are unknown to the learner.

Resources. For any j in $[d]$, let $B(j) > 0$ denote the budget of resource j. We write $\vec{B} = [B(1), B(2), ..., B(d)]^\top$. The learner knows the values of $\vec{B}$. The learner repeats pulling arms, and if for some j, the cumulated consumption of resource j exceeds the budget $B(j)$, the learner must stop pulling. The goal of the learner is to maximize the expected cumulated reward within the limited resources.

Rewards. We use A^t to denote the t-th arm the learner pulls. Let R^t be the reward gained from the t-th pull, and $C^t(j)$ be amount of consumption of resource j incurred by the t-th pull. Let N be the number of pulls before termination, i.e.,

$$N := \inf\{t : \exists j \in [d] \text{ such that } \sum_{k=1}^{t} C^k(j) \geq B(j)\}.$$

The expected reward is defined as

$$\overline{R_w} := \mathbb{E}\Big[\sum_{t=1}^{N} \mathbb{E}[R^t \mid A^t]\Big] = \mathbb{E}\Big[\sum_{t=1}^{N} \mu_{A^t}\Big].$$

Regrets. Define OPT as the expected reward obtained by an oracle algorithm that knows the values of $(\mu_x, x \in [n])$ and $(\vec{\lambda}_x, x \in [n])$. We define the regret as $\overline{R_g} := \text{OPT} - \overline{R_w}$. We note that unlike traditional MAB problems, an optimal algorithm for BwK may not always be sampling the optimal arm, and the notion of optimal arm may not even exist. For instance, let $n = d = 2$, $\mu_1 = \mu_2 = 1$, $\vec{\lambda}_1 = [1, 0]^\top$, $\vec{\lambda}_2 = [0, 1]^\top$, and $\vec{B} = [100, 100]^\top$. The optimal policy is to pull each arm for 100 times. In traditional MAB, it is intuitive to denote the regret of an algorithm by $(\text{OPT} - \overline{R_w})$. However, for BwK, it can be hard to determine the value of OPT. In fact, the problem of finding BwK's OPT value remains open. When the rewards and costs are deterministic, BwK reduces to the bounded knapsack problem, which is already NP-hard [23]. Whether there exists a polynomial-time algorithm that finds the (approximate) value of OPT is unknown.

Since we do not know the value of OPT, we need to find another value to define the regret. In this paper, we adopt the linear programming relaxation, which was also adopted by many previous works [3], [8], [14], [15]. For any $\vec{\beta} = [\beta(1), \beta(2), ..., \beta(d)]^\top \in \mathbb{R}_+^d$, define

$$\begin{aligned} \text{LP}(\vec{\beta}) := \max_{\xi(1),\ldots,\xi(n)\geq 0} \quad & \sum_{x\in[n]} \xi(x)\mu_x, \qquad (1) \\ \text{s.t.} \quad & \sum_{x\in[n]} \xi(x) \cdot \lambda_x(j) \leq \beta(j), \forall j \in [d]. \end{aligned}$$

We then define $\text{OPT}_{\text{LP}} := \text{LP}(\vec{B}) + 1$, where 1 comes from our setting that the last pull of an arm may consume more than the remaining resources for some resource j. It has been shown in [8] that $\text{OPT}_{\text{LP}} \geq \text{OPT}$.

Additional Assumptions. In this paper, we focus on the cases where the costs are deterministic. When there is only one type of resource and this resource is time, the BwK problem reduces to the standard MAB problem. For some applications such as shelf optimization [18], the costs are deterministic. We provide useful insights for these problems in this section by proposing new algorithms.

We further make two weak assumptions. First, the last resource is time, i.e., the number of pulls, and we use T to denote its budget. Second, the last arm (i.e., arm n) is a dummy arm such that $R_n^t = 0$ almost surely for any time t, and $\lambda_x(j) = 0$ for any j in $[d-1]$, that is, arm n does not generate any reward and only costs time. The first assumption has been adopted by many previous works (e.g., [3], [8]). For the second assumption, the dummy arm does not affect the algorithm and is mainly for simplifying the analysis. For the cases where time is not budget-limited, since the budgets of the resources are known and the arms' costs are deterministic, we can find a number T that is an upper bound on the number of pulls and view T as a budget of time.

C. Main Results

We propose a new algorithm with regret upper bound of the form $O(\log T)(\sum_{x\in[n]} \Delta_{\mathbf{s}_x}^{-1})$, where $\Delta_{\mathbf{s}_x}$ is the gap of rewards similar to that in traditional MAB, which does not directly depend on the number of resource types. When d is large, our algorithm outperforms previous works [15], [25] significantly.

D. Related Works

Dating back to 2010, the authors of [28] studied budget-limited MAB problems with a single cost, which can be viewed as a special case of the BwK problem. The BwK problem was first fully formulated in [8], whose authors proposed algorithms with $O(\sqrt{\log T})(\sqrt{n\text{OPT}_{\text{LP}}} + \text{OPT}_{\text{LP}}\sqrt{n/B_{\min}})$ regrets, where $B_{\min} := \min_{j\in[d]} B(j)$. The authors of [3] proposed an algorithm with $O(\sqrt{\log T})(\sqrt{n\text{OPT}_{\text{LP}}} + \text{OPT}_{\text{LP}}\sqrt{n/B_{\min}})$ regret. The Thompson Sampling algorithm proposed in [14] achieved an $O(\sqrt{nT} \cdot \text{ploylog}(T))$ regret. These regrets are of the same order as the $O(\sqrt{nT \log T})$ distribution-free regret of traditional MAB [5]. However, the distribution-dependent regret of BwK is less understood.

The authors of [15], [25] studied the distribution-dependent regrets of BwK. For BwK with deterministic costs, the authors of [15] proposed two algorithms with regret upper bounds $O((d \wedge n)\binom{n+d}{d} \log T/\Delta)$ and $O((d \wedge n)^3 \log T/\Delta^2)$ respectively. Since $\binom{n+d}{d}$ increases exponentially with d, the regret of the first algorithm also increases exponentially with d. For the second algorithm, the factor Δ^{-2} is suboptimal. For best-arm-optimal instances, with $d = 2$ and deterministic costs, the authors of [25] proposed algorithms with regrets in the form of $O(n\Delta^{-2} \log T)$, where "best-arm-optimal" means that there is an optimal policy that only pulls one arm. In contrast, we do not require the best-arm-optimal condition, and propose a new algorithm and improve these regrets to $O(n \log T/\Delta)$, significantly outperforming the results in [15], [25].

Logarithmic regrets of BwK with non-deterministic costs are even less understood. The authors of [15] gave a logarithmic algorithm for $d > 1$, but it requires restrictive assumptions, e.g., there is a constant $\sigma > 0$ such that for all t and j, $R^t \leq \sigma C^t(j)$ almost surely. This condition is not assumed in our paper. Without these restrictive assumptions, the authors of [15] show that it is probably impossible to get a distribution-dependent regret lower than $\Omega(\sqrt{T})$. When there is only one type of resources, i.e., $d = 1$, logarithmic regrets have been obtained by [28] for the cases where costs are deterministic. For non-deterministic costs, [13], [29], [30], [32]–[34] studied the single-resource BwK problems under different settings, and logarithmic regrets were established.

There are also many interesting works on BwK problems under other settings, which are less related to this paper, e.g., the BwK problem in a contextual bandit setting [2], [31], linear submodular bandits with a knapsack [35], combinatorial semi-bandits with knapsacks [24].

II. Preliminaries

In this section, we provide some basic definitions and facts that will be useful in the rest of the paper. Let constant $p > 0$ be given. We define the *confidence radius* [7], [19] as

$$\mathbf{rad}_p(\mu, N, T) := \sqrt{(c_p \mu \log T)/N} + (c_p \log T)/N, \quad (2)$$

where $c_p > 0$ is a constant that only depends on p. We then introduce the following concentration inequality [7], [19].

Lemma 1 ([7], [19])**.** *Let μ be the expectation of a distribution with support $[0, 1]$, and let $\hat{\mu}$ be the empirical mean after N independent sampling of this distribution. Let $T > 0$ be given, $X = \mathbf{rad}_p(\hat{\mu}, N, T)$, and $Y = \mathbf{rad}_p(\mu, N, T)$. Then,*

$$\mathbb{P}\{|\mu - \hat{\mu}| \leq X \leq 3Y\} \geq 1 - T^{-\Omega(1)}.$$

This inequality is similar to Chernoff-Hoeffding inequality [17], but works better when μ is much smaller than 1, as the leading term in the right-hand side of Eq. (2) depends on $\sqrt{\mu}$. This inequality has been shown to be useful for BwK problems [3], [8]. However, to derive better logarithmic regrets for BwK problems, this inequality may not be sufficient. In this paper, we prove a new inequality that can bound the linear combination of empirical means of multiple distributions. The proof (in the appendix[2]) will invoke the works in [19], [22].

Lemma 2. *Assume that there are n distributions with supports being $[0, 1]$. For any i in $[n]$, let μ_i be the expectation of the i-th distribution and $\hat{\mu}_i$ be its empirical mean after N_i independent sampling. For $\mathbf{s} \in \mathbb{R}^n_+$, define $M(\mathbf{s}) := \min_{i\in[n]}[N_i/s(i)]$, $\mu(\mathbf{s}) := \sum_{i\in[n]} s(i)\mu_i$, and $\hat{\mu}(\mathbf{s}) := \sum_{i\in[n]} s(i)\hat{\mu}_i$. Let $c_p \geq 24e^3p/(2e-1)^2$, $X = \mathbf{rad}_p(\hat{\mu}(\mathbf{s}), M(\mathbf{s}), T)$, and $Y = \mathbf{rad}_p(\mu(\mathbf{s}), M(\mathbf{s}), T)$. Then*

$$\mathbb{P}\{|\hat{\mu}(\mathbf{s}) - \mu(\mathbf{s})| \leq X \leq 3Y\} \geq 1 - 2T^{-p}.$$

[2]The appendix of this paper can be found in https://www.dropbox.com/s/bir12z8ydpnvkdt/Appendix.pdf?dl=0

III. Algorithms and Regret Bounds

In this section, we present our BwK algorithms for instances with deterministic costs. Define $b(j) := B(j)/T$ for all $j \in [d]$ and write $\vec{b} := [b(1), b(2), ..., b(d)]^\top$. We transform the linear optimization problem LP($\vec{b}$) into the standard form [21]:

$$\max_{s(x)\geq 0, \eta_j \geq 0, \forall x, j} \quad \sum_{x\in[n]} s(x)\mu_x \quad (3)$$
$$\text{s.t.} \quad \sum_{x\in[n]} s(x)\lambda_x(j) + \eta_j = b(j), \forall j \in [d]$$

where $\eta_1, \eta_2, ..., \eta_d$ are slack variables. Let $\mathcal{D} \subset \mathbb{R}^{n+d}$ be the feasible region of the above optimization problem. A point $\mathbf{s} \in \mathcal{D}$ is said to be an *extreme point* if

$$(\forall \alpha \in (0, 1), \mathbf{u}, \mathbf{v} \in \mathcal{D})[\mathbf{s} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v} \implies \mathbf{u} = \mathbf{v}].$$

It is well known that [21] since all μ_x's are non-negative, Eq. (3) has a maximizer, and therefore, at least one extreme point of $\mathcal{D}$ maximizes Problem (3). We let $\mathcal{B}$ be the set of all extreme points of $\mathcal{D}$. For all extreme points $\mathbf{s}$ in $\mathcal{B}$, there are at most d coordinates of $\mathbf{s}$ that are non-zero [21]; and for each combination of coordinates, there is at most one extreme point. Thus, we have $|\mathcal{B}| \leq \binom{n+d}{d}$.

Notations. Let N^t_x be the number of pulls of arm x till iteration t. Let $\hat{\mu}^t_x$ be the empirical mean reward of arm x till iteration t. For all extreme points $\mathbf{s}$ in $\mathcal{B}$, define

$$M^t(\mathbf{s}) := \min_{x:s(x)>0}[N^t_x/s(x)],$$
$$\mu(\mathbf{s}) := \sum_{x\in[n]} s(x)\mu_x, \text{ and } \hat{\mu}^t(\mathbf{s}) := \sum_{x\in[n]} s(x)\hat{\mu}^t_x.$$

Define the optimal extreme point as

$$\mathbf{s}^* := \arg\max_{\mathbf{s}\in\mathcal{B}} \mu(\mathbf{s}), \text{ and } \mu^* := \mu(\mathbf{s}^*).$$

For every extreme point $\mathbf{s} \in \mathcal{B}$, define the gap $\Delta_{\mathbf{s}} := \mu^* - \mu(\mathbf{s})$. For arm x, define

$$\mathbf{s}_x := \arg\max_{\mathbf{s}\in\mathcal{B}-\{\mathbf{s}^*\}} [s(x)/\Delta^2_{\mathbf{s}}].$$

We note that for some x, $\mathbf{s}_x$ may not exist. In this case, we let $\Delta_{\mathbf{s}_x} = \infty$. We further define

$$G := \sum_{x\in[n]} \mu^*/\Delta_{\mathbf{s}_x} \text{ and } H := \sum_{x\in[n]} \mu^*/\Delta^2_{\mathbf{s}_x}.$$

Basic idea. Since $\mathcal{B}$ is finite and at least one extreme point in $\mathcal{B}$ is optimal, one may solve BwK by treating each extreme point as an arm and using traditional UCB algorithms[3], e.g., [5], [6], [16]. However, since $|\mathcal{B}|$ may scale with $\binom{n+d}{d}$, if we directly treat each extreme point as an arm, the regret may also scale with $\binom{n+d}{d}$, which increases exponentially with d.

In [15], algorithm UCB-Simplex was proposed, which maintains a UCB value for the empirical mean reward $\hat{\mu}^t(\mathbf{s})$ of each extreme point $\mathbf{s}$, and "pulls" the extreme point $\mathbf{s}^t$ with

[3]For instance, at each iteration t, a UCB algorithm may choose an extreme point $\mathbf{s}^t$ with the largest UCB value, and sample arm x with probability $s^t(x)$ or use other strategies to sample arms.

the maximal UCB value for each iteration t. Here, "pulling an extreme point" means increasing the counter of each arm x by $s^t(x)$. When the counter reaches one, the algorithm pulls arm x once and decreases the counter by one. By this method, a logarithmic regret can be obtained. However, since the number of extreme points scales with $\binom{n+d}{d}$ in the worst case, the regret may also scale with $\binom{n+d}{d}$. In fact, its regret upper bound is $O(d\sum_{\mathbf{s}\in\mathcal{B}}\Delta_{\mathbf{s}}^{-1}\log T)$ for $d \le n$, scaling with $|\mathcal{B}|$. Even with $d = 2$, the regret upper bound can be of the form $O(n^2\Delta^{-1}\log T)$, quadratically dependent on n, the number of arms, which is not promising.

The authors of [15] proposed another algorithm UCB-Simplex-v2, which does not directly bound the number of times when an extreme point is chosen, but bounds the number of pulls of each arm, i.e., at each iteration t where $\mathbf{s}^t$ has the maximal UCB value, only the arm that minimizes $N_x^t/s^t(x)$ will be pulled. Through this strategy, the authors obtained regret upper bound $O(d^3\sum_{x\in[n]}\Delta_{\mathbf{s}_x}^{-2}\log T)$ for $d \le n$, which increases linearly with n, better than UCB-Simplex in this aspect. However, this new bound still has suboptimal dependence on $\Delta_{\mathbf{s}_x}^{-2}$ and d^3.

We propose new methods and manage to reduce the regret bounds. First, we improve the dependence of the regret on $\Delta_{\mathbf{s}_x}^{-2}$ to $\Delta_{\mathbf{s}_x}^{-1}$. Roughly speaking, the dependence of the regret on $\Delta_{\mathbf{s}_x}^{-2}$ is because different types of resources are not "evenly" consumed. In UCB-Simplex-v2, due to its specific way of pulling arms, some resources may run out much earlier before others, and the above technique cannot be applied if one type of resources runs out. To solve this problem, we add a new phase to our algorithm. We choose a parameter $\epsilon \in [0, 1/2]$, and then, for the first $(1-\epsilon)$ fraction of resources, we use the similar arm pulling strategy as UCB-Simplex-v2, and for the rest ϵ fraction of resources, we use the BwK algorithm proposed in [3]. The regret upper bound of the algorithm proposed by [3] is $O(\sqrt{n\mathrm{OPT}_{\mathrm{LP}}\log T})$. In each iteration, Algorithm 2 of [3] finds a vector $\mathbf{s}^t$ that solves

$$\begin{aligned}\max_{\mathbf{s}\ge 0:\ ||\mathbf{s}||_1\le 1}\quad & \sum_{x\in[n]} s(x)\Big[\hat{\mu}_x^t + \mathbf{rad}_p(\hat{\mu}_x^t, N_x^t, T)\Big],\\ \text{s.t.}\quad & \sum_{x\in[n]} s(x)\Big[\hat{\lambda}_x^t(j) - \mathbf{rad}_p(\hat{\lambda}_x^t(j), N_x^t, T)\Big]\\ & \qquad \le (1-\epsilon')b(j), \forall j\in[d],\end{aligned}$$

and samples each arm x with probability $s^t(x)$. But in this paper, we make three modifications: (i) We set $\epsilon' = 0$ since the costs are deterministic. (ii) We do not use lower confidence bounds (LCB) on the costs since they are deterministic. (iii) The most importantly, we set up n queues for the arms to ensure that the number of pulls of each arm is close to what it should be. If we sample arms randomly like in [3], some resources may run out before T iterations, making the resources not fully utilized. By separating the pulling to two phases, the dependence on $\Delta_{\mathbf{s}_x}^{-2}$'s can be replaced by $\Delta_{\mathbf{s}_x}^{-1}$'s.

Second, we remove the factor d (d^3) in the regret bound of UCB-Simplex (UCB-Simplex-v2). One reason for these dependences is that the linear combinations of the UCB values of arms' empirical means, i.e., $\sum_{x\in[n]}[s(x)\cdot\mathrm{UCB}(\hat{\mu}_x)]$, are not tight bounds on $\hat{\mu}(\mathbf{s})$, the empirical means of extreme points. To state it clearly, we take the Hoeffding bound as an example. For an arm x with empirical mean reward $\hat{\mu}_x^t$ after N_x^t samples, by Chernoff-Hoeffding inequality [17], we can get bounds

$$\mathbb{P}\{|\hat{\mu}_x^t - \mu_x| \ge \sqrt{\log(2/\delta)/(2N_x^t)}\} \le \delta.$$

However, in some cases (e.g., $s(x) = 1/d$ and $N_x^t = N_y^t$ for all arms x and y with $s(x), s(y) \ne 0$), the linear combination of these bounds for $\hat{\mu}_x^t$'s has $\sum_{x\in[n]}\left[s(x)\sqrt{\frac{1}{2N_x^t}\log\frac{2}{\delta}}\right] = \sqrt{\frac{d}{2M^t(\mathbf{s})}\log\frac{2}{\delta}}$, which depends on $\sqrt{d/M^t(\mathbf{s})}$, while the bound stated in Lemma 2 depends on $\sqrt{1/M^t(\mathbf{s})}$. We believe that this is the reason why the regret upper bound of UCB-Simplex has a dependence on d. In this paper, by the new confidence inequality (i.e., Lemma 2), we can remove this d (or d^3) factor.

Algorithm. We assume that there is only one optimal extreme point $\mathbf{s}^*$. Our algorithm BNPA (Bounding the Number of Pulls of Arms) is described in Algorithm 1, and its regret is stated in Theorem 3. The proof is left to the appendix.

Theorem 3. *The (expected) regret of BNPA is at most*

$$\begin{aligned}&O\big(G\log T + \sqrt{n(\epsilon T + H\log T)\log(\epsilon T + H\log T)}\big)\\ &\qquad + \mu^*(36c_pH\log T - \epsilon T)^+.\end{aligned} \tag{4}$$

Specifically, if $(36c_pH\log T)/T \le \epsilon = O((36c_pH\log T)/T)$, *the (expected) regret is at most*

$$O\big(G\log T + \sqrt{nH\log T\log(H\log T)}\big). \tag{5}$$

Remark. *i)* By Theorem 3, if $(36c_pH\log T)/T \le \epsilon = O((36c_pH\log T)/T)$,

$$\limsup_{\vec{B}=\vec{b}T\to\infty}\ (\mathrm{OPT} - \overline{R_w}(T))/\log T = O(G),$$

and if $\epsilon \le (36c_pH\log T)/T$,

$$\limsup_{\vec{B}=\vec{b}T\to\infty}\ (\mathrm{OPT} - \overline{R_w}(T))/\log T = O(G+H),$$

which shows that our BNPA algorithm has logarithmic regret. We note that when d changes, many values including G, H, and OPT will also change (may increase or decrease, depending on the instances), and thus, for some instances, the regret of BNPA may not increase when d increases. In the algorithm, choosing the value of ϵ may not be easy. However, as shown in Section IV, even when choosing $\epsilon = 0$, our algorithm can still have promising empirical performance.

ii) We note that the function optimized in Line 5 of BNPA is not convex. Besides enumerating all the extreme points in $\mathcal{B}$, we are not aware of a faster approach. Since the size of $\mathcal{B}$ can be up to $\binom{n+d}{d}$, the time complexity of Line 5 can be $O(\binom{n+d}{d})$, which is large when d is large. However, the good news is that for many applications, d is small. For instance, for ad allocation and dynamic pricing formulated by [8], $d = 2$. Even if $d = 2$, our regret upper bound still has improvements compared to previous works [15], [25].

Algorithm 1 Bounding the Number of Pulls of Arms (BNPA).

1: Choose a parameter $\epsilon \in [0, 1/2]$;
2: Sample each arm once;
3: $t \leftarrow n$ and update N_x^t and $\hat{\mu}_x^t$ for all arms x;
4: **repeat**
5: Find an extreme point

$$\mathbf{s}^t \in \arg\max_{\mathbf{s}\in\mathcal{B}} \big\{\hat{\mu}^t(\mathbf{s}) + \mathbf{rad}_p(\hat{\mu}^t(\mathbf{s}), M^t(\mathbf{s}), T)\big\};$$

6: Find an $x^t \in \arg\min_{x\in[n]}\{N_x^t/s^t(x)\}$;
7: Pull arm x^t once;
8: $t \leftarrow t+1$, and update $N_{x^t}^t$ and $\hat{\mu}_{x^t}^t$ coordinately;
9: **until** $\exists j \in [d]$ such that $\sum_{k=1}^{t} C^k(j) \geq (1-\epsilon)B(j)$.
10: For any $j \in [d]$, let $B_1(j)$ be the amount of resource j that has been consumed, and $B'(j) \leftarrow B(j) - B_1(j)$;
11: Let T_1 be the number of pulls till now; $T' \leftarrow T - T_1$;
12: For any arm x, initialize a queue $q_x \leftarrow 0$;
13: **repeat**
14: Find an $\mathbf{s}^t$ that solves

$$\max_{\mathbf{s}\geq 0:||\mathbf{s}||_1\leq 1} \sum_{x\in[n]} s(x)\big[\hat{\mu}_x^t + \mathbf{rad}_p(\hat{\mu}_x^t, N_x^t, T')\big], \quad (6)$$
$$\text{s.t.} \sum_{x\in[n]} s(x)\cdot\lambda_x(j) \leq B'(j)/T', \forall j \in [d],$$

15: For any arm x, update $q_x \leftarrow q_x + s^t(x)$;
16: **for** arm x in $[n]$ such that $q_x \geq 1$ **do**
17: Pull arm x once, and update $q_x \leftarrow q_x - 1$;
18: **end for**
19: Set $t \leftarrow t+1$, and update N_x^t and $\hat{\mu}_x^t$ for all arms;
20: **until** Some resource runs out.

iii) In fact, if we substitute the problem in Line 5 by the following linear programming problem,

$$\mathbf{s}^t \in \arg\max_{\mathbf{s}\in\mathcal{B}} \big\{s(x)\big[\hat{\mu}_x^t + \mathbf{rad}_p(\hat{\mu}_x^t, N_x^t, T)\big]\big\}, \quad (7)$$

the time complexity of Line 5 becomes $\text{poly}(n, d)$, which follows from the time complexity of linear programming [21], and the new regret upper bound will be around d times that of BNPA. We name the new algorithm as BNPA-v2, and state the new regret in Theorem 4. Its proof is provided in the appendix.

Theorem 4. *Replace Line 5 of BNPA with Eq (7), and the expected regret of the new version BNPA-v2 is at most*

$$O\big(dG\log T + \sqrt{n(\epsilon T + dH\log T)\log(\epsilon T + dH\log T)}\big) + \mu^*(36c_p dH\log T - \epsilon T)^+.$$

If $(36c_p dH\log T)/T \leq \epsilon = O((36c_p dH\log T)/T)$*, the regret is at most*

$$O\big(dG\log T + \sqrt{ndH\log T\log(dH\log T)}\big).$$

IV. Numerical Results

In this section, we numerically show the improvements of our algorithm BNPA over previous works.

We compare BNPA and BNPA-v2 with PrimalDualBwK [8], BwCR [3] (Algorithm 2), and UCB-Simplex [15]. PrimalDualBwK and BwCR do not have logarithmic regrets, and their regrets are $O(\sqrt{\log T})(\sqrt{n\text{OPT}_{\text{LP}}} + \text{OPT}_{\text{LP}}\sqrt{n/B_{\min}})$. With deterministic costs, the regret upper bound of UCB-Simplex is $O((d\wedge n)\cdot\sum_{\mathbf{s}\in\mathcal{B}}\Delta_{\mathbf{s}}^{-1}\log T)$, exponentially higher than that of our BNPA algorithm in the worst case.

We choose $\epsilon = 0$ for our algorithm, and the numerical results show that our algorithm still outperforms previous works. For BwCR, we choose $\epsilon = 0$ as the costs are deterministic. Also, in the implementations, we modify all algorithms to versions with known $\lambda_x(j)$-values for fair comparisons, e.g., in BwCR, we let the LCB of $\lambda_x(j)$ equal to $\lambda_x(j)$.

We do not compare the Thompson Sampling algorithm [4] since it knows additional knowledge on the prior distributions of μ_x's and $\lambda_x(j)$'s and the latent distributions of the arms, which is not assumed in our algorithm and may result in unfair comparisons. We do not compare BalancedExploration [8] because it requires to compute a possibly infinite set, and the implementation detail was not given in [8]. Ignoring the difference on the confidence bounds, UCB-Simplex-v2 [15] is the same as BNPA-v2 with $\epsilon = 0$ (but when $\epsilon > 0$, the regret upper bound of BNPA-v2 is lower than UCB-Simplex-v2), and we do not compare it in this paper.

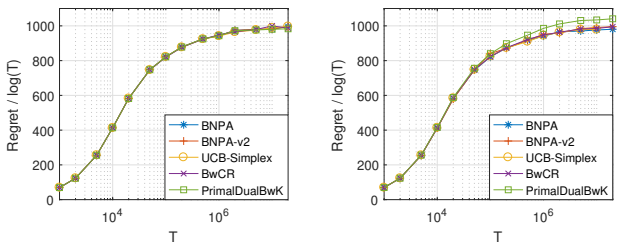
Simulation setup. We adopt Bernoulli rewards for all arms. For fair comparisons, in all algorithms, we use the confidence bounds in Lemma 2, and set $p = 2$ and $c_p = 24e^3p/(2e-1)^2$. We choose $n = 10$ (excluding the dummy arm), $d = \{2, 3, 5, 7\}$. The mean rewards and mean costs are generated as follows: we set $\mu_1 = 0.95$ and $\lambda_1(j) = 0.45$ for all $j \in [d]$; for all arms $x \neq 1$, mean rewards are generated by taking independence samples of Uniform$([0.95-\sigma, 0.95-2\sigma])$ distribution, and mean costs are generated by taking independent samples of Uniform$([0.45+\sigma, 0.45+2\sigma])$ distribution, where $\sigma = 0.2$ is fixed. We vary the value of T, and for all $j \in [d]$, we always set $B(j) = 0.45T$. All algorithms are performed on the same datasets. In every figure, every point is averaged over 100 independent trials.

The numerical results are illustrated in Figure 1 (a)-(d), and we have the following findings. First, as d increases, the regrets of BNPA in four cases do not increase and even decrease when T is large, which indicates that its regret does not directly depend on the number of resources d. The reason of the decreasing regrets may be that the gaps (i.e., $(\Delta_{\mathbf{s}_x}, x \in [n])$) become larger when d increases.

Second, from Figure 1 (a) and (b), we can see that when the value of d is small, the empirical performance of these algorithms are close. However, for large values of d, Figure 1 (c) and (d) indicate that the performance of BNPA becomes better compared with other algorithms, which suggests that our algorithm is more promising when d is large.

Third, in all four subfigures, as T increases, the ratio of BNPA's regret to $\log T$ approaches a constant, which suggests that BNPA has logarithmic regret, consistent with the theory.

Fourth, we can see from Figure 1 that the regret of BNPA-v2 is larger than BNPA, especially when d is large, which



(a) $d = 2$ (b) $d = 3$

(c) $d = 5$ (d) $d = 7$

Fig. 1. Comparisons between BNPA, BNPA-v2, and existing methods in the literature with different values of d.

is consistent with the theoretical results stated in Theorems 3 and 4. Although the time complexity of BNPA-v2 is smaller than BNPA, it suffers from a loss in the regret performance.

V. Conclusion

This paper studied the logarithmic regrets of bandits with knapsacks (BwK) problems. For BwK with deterministic costs, we proposed an algorithm with regret upper bound in the form of $O(n \log T/\Delta)$, which does not directly depend on the number of resources d and outperforms the state of the art. Empirical results also confirmed our theories.

References

[1] Agarwal, A., Agarwal, S., Assadi, S., & Khanna, S. (2017, June). Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In Conference on Learning Theory (pp. 39-75).

[2] Agrawal, S., & Devanur, N. (2016). Linear contextual bandits with knapsacks. Advances in Neural Information Processing Systems, 29, 3450-3458.

[3] Agrawal, S., & Devanur, N. R. (2014, June). Bandits with concave rewards and convex knapsacks. In Proceedings of the fifteenth ACM conference on Economics and computation (pp. 989-1006).

[4] Agrawal, S., & Goyal, N. (2012, June). Analysis of Thompson sampling for the multi-armed bandit problem. In Conference on learning theory (pp. 39-1).

[5] Auer, P., & Ortner, R. (2010). UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. Periodica Mathematica Hungarica, 61(1-2), 55-65.

[6] Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. Machine learning, 47(2-3), 235-256.

[7] Babaioff, M., Dughmi, S., Kleinberg, R., & Slivkins, A. (2015). Dynamic pricing with limited supply. ACM Transactions on Economics and Computation (TEAC), 3(1), 1-26.

[8] Badanidiyuru, A., Kleinberg, R., & Slivkins, A. (2013, October). Bandits with knapsacks. In 2013 IEEE 54th Annual Symposium on Foundations of Computer Science (pp. 207-216). IEEE.

[9] Berry, D. A., & Eick, S. G. (1995). Adaptive assignment versus balanced randomization in clinical trials: A decision analysis. Statistics in medicine, 14(3), 231-246.

[10] Berry, D. A., & Fristedt, B. (1985). Bandit problems: Sequential allocation of experiments (Monographs on statistics and applied probability). London: Chapman and Hall, 5(71-87), 7-7.

[11] Besbes, O., & Zeevi, A. (2012). Blind network revenue management. Operations research, 60(6), 1537-1550.

[12] Bubeck, S., & Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. arXiv preprint arXiv:1204.5721.

[13] Ding, W., Qin, T., Zhang, X. D., & Liu, T. Y. (2013, July). Multi-armed bandit with budget constraint and variable costs. In AAAI (Vol. 13, pp. 232-238).

[14] Ferreira, K. J., Simchi-Levi, D., & Wang, H. (2018). Online network revenue management using Thompson sampling. Operations research, 66(6), 1586-1602.

[15] Flajolet, A., & Jaillet, P. (2015). Logarithmic regret bounds for bandits with knapsacks. arXiv preprint arXiv:1510.01800.

[16] Garivier, A., & Cappé, O. (2011, December). The KL-UCB algorithm for bounded stochastic bandits and beyond. In Proceedings of the 24th annual conference on learning theory (pp. 359-376).

[17] Hoeffding, W. (1994). Probability inequalities for sums of bounded random variables. In The Collected Works of Wassily Hoeffding (pp. 409-426). Springer, New York, NY.

[18] Kleinberg, R., & Leighton, T. (2003, October). The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In 44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings. (pp. 594-605). IEEE.

[19] Kleinberg, R., Slivkins, A., & Upfal, E. (2008, May). Multi-armed bandits in metric spaces. In Proceedings of the fortieth annual ACM symposium on Theory of computing (pp. 681-690).

[20] Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010, April). A contextual-bandit approach to personalized news article recommendation. In Proceedings of the 19th international conference on World wide web (pp. 661-670).

[21] Luenberger, D. G., & Ye, Y. (1984). Linear and nonlinear programming (Vol. 2). Reading, MA: Addison-wesley.

[22] Mitzenmacher, M., & Upfal, E. (2017). Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis. Cambridge university press.

[23] Pisinger, D. (2005). Where are the hard knapsack problems?. Computers & Operations Research, 32(9), 2271-2284.

[24] Sankararaman, K. A., & Slivkins, A. (2018, March). Combinatorial semi-bandits with knapsacks. In International Conference on Artificial Intelligence and Statistics (pp. 1760-1770). PMLR.

[25] Sankararaman, K. A., & Slivkins, A. (2020). Advances in Bandits with Knapsacks. arXiv preprint arXiv:2002.00253.

[26] Scott, S. L. (2010). A modern Bayesian look at the multi-armed bandit. Applied Stochastic Models in Business and Industry, 26(6), 639-658.

[27] Slivkins, A. (2013). Dynamic ad allocation: Bandits with budgets. arXiv preprint arXiv:1306.0155.

[28] Tran-Thanh, L., Chapman, A., Munoz De Cote Flores Luna, J. E., Rogers, A., & Jennings, N. R. (2010). Epsilon–first policies for budget–limited multi-armed bandits.

[29] Tran-Thanh, L., Chapman, A., Rogers, A., & Jennings, N. R. (2012). Knapsack based optimal policies for budget-limited multi-armed bandits. arXiv preprint arXiv:1204.1909.

[30] Watanabe, R., Komiyama, J., Nakamura, A., & Kudo, M. (2018). UCB-SC: A fast variant of KL-UCB-SC for budgeted multi-armed bandit problem. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 101(3), 662-667.

[31] Wu, H., Srikant, R., Liu, X., & Jiang, C. (2015). Algorithms with logarithmic or sublinear regret for constrained contextual bandits. In Advances in Neural Information Processing Systems (pp. 433-441).

[32] Xia, Y., Ding, W., Zhang, X. D., Yu, N., & Qin, T. (2016, February). Budgeted bandit problems with continuous random costs. In Asian conference on machine learning (pp. 317-332).

[33] Xia, Y., Li, H., Qin, T., Yu, N., & Liu, T. Y. (2015). Thompson sampling for budgeted multi-armed bandits. arXiv preprint arXiv:1505.00146.

[34] Xia, Y., Qin, T., Ding, W., Li, H., Zhang, X., Yu, N., & Liu, T. Y. (2017). Finite budget analysis of multi-armed bandit problems. Neurocomputing, 258, 13-29.

[35] Yu, B., Fang, M., & Tao, D. (2016, February). Linear submodular bandits with a knapsack constraint. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (pp. 1380-1386).