Combinatorial Network Optimization With Unknown Variables: Multi-Armed Bandits With Linear Rewards and Individual Observations

Yi Gai, *Student Member, IEEE, Member, ACM*, Bhaskar Krishnamachari, *Member, IEEE, ACM*, and Rahul Jain, *Member, IEEE, ACM*

***Abstract*—We formulate the following combinatorial multi-armed bandit (MAB) problem: There are N random variables with unknown mean that are each instantiated in an i.i.d. fashion over time. At each time multiple random variables can be selected, subject to an arbitrary constraint on weights associated with the selected variables. All of the selected individual random variables are observed at that time, and a linearly weighted combination of these selected variables is yielded as the reward. The goal is to find a policy that minimizes regret, defined as the difference between the reward obtained by a genie that knows the mean of each random variable, and that obtained by the given policy. This formulation is broadly applicable and useful for stochastic online versions of many interesting tasks in networks that can be formulated as tractable combinatorial optimization problems with linear objective functions, such as maximum weighted matching, shortest path, and minimum spanning tree computations. Prior work on multi-armed bandits with multiple plays cannot be applied to this formulation because of the general nature of the constraint. On the other hand, the mapping of all feasible combinations to arms allows for the use of prior work on MAB with single-play, but results in regret, storage, and computation growing exponentially in the number of unknown variables. We present new efficient policies for this problem that are shown to achieve regret that grows logarithmically with time, and polynomially in the number of unknown variables. Furthermore, these policies only require storage that grows linearly in the number of unknown parameters. For problems where the underlying deterministic problem is tractable, these policies further require only polynomial computation. For computationally intractable problems, we also present results on a different notion of regret that is suitable when a polynomial-time approximation algorithm is used.**

***Index Terms*—Combinatorial network optimization, multi-armed bandits (MABs), online learning.**

Manuscript received November 23, 2010; revised October 22, 2011; accepted November 27, 2011; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor E. Modiano. This work was supported in part by the U.S. Army Research Laboratory under the Network Science Collaborative Technology Alliance Agreement No. W911NF-09-2-0053 and the U.S. National Science Foundation under Award no. CNS-1049541. The work of R. Jain was supported by the AFOSR under Grant FA9550-10-1-0307 and the NSF under CAREER Award CNS-0954116.

The authors are with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089 USA (e-mail: ygai@usc.edu; bkrishna@usc.edu; rahul.jain@usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2011.2181864

I. INTRODUCTION

THE PROBLEM of multi-armed bandits (MABs) is a classic one in learning theory. In its simplest form, there are N arms, each providing stochastic rewards that are independent and identically distributed over time, with unknown means. A policy is desired to pick one arm at each time sequentially to maximize the reward. MAB problems capture a fundamental tradeoff between exploration and exploitation: On the one hand, various arms should be explored in order to learn their parameters, and on the other hand, the prior observations should be exploited to gain the best possible immediate rewards. MABs have been applied in a wide range of domains including Internet advertising [1], [2] and cognitive radio networks [3], [4].

As they are fundamentally about combinatorial optimization in unknown environments, one would indeed expect to find even broader use of multi-armed bandits. However, we argue that a barrier to their wider application in practice has been the limitation of the basic formulation and corresponding policies, which generally treat each arm as an independent entity. They are inadequate to deal with many combinatorial problems of practical interest in which there are large (exponential) numbers of arms. In such settings, it is important to consider and exploit any structure in terms of dependencies between the arms. We show in this paper that when the dependencies take a linear form, they can be handled tractably with policies that have provably good performance in terms of regret as well as storage and computation.

In this paper, we consider the following multi-armed bandit problem. There are N random variables with unknown mean that are each instantiated in an i.i.d. fashion over time. At each time, a particular set of multiple random variables can be selected, subject to a general arbitrary constraint on weights associated with the selected variables. All of the selected individual random variables are observed at that time, and a linearly weighted combination of these selected variables is yielded as the reward.

Our general formulation of multi-armed bandits with linear rewards is applicable to a very broad class of combinatorial network optimization problems with linear objectives. These include maximum weight matching in bipartite graphs (which is useful for user-channel allocations in cognitive radio networks), as well as shortest path, and minimum spanning tree computation. In these examples, there are random variables associated with each edge on a given graph, and the constraints on the set of elements allowed to be selected at each time correspond to sets of edges that form relevant graph structures (such as matchings, paths, or spanning trees).

Because our formulation allows for arbitrary constraints on the multiple elements that are selected at each time, prior work on multi-armed bandits that only allow for a fixed-number of multiple plays along with individual observations at each time [5], [6] cannot be directly used for this more general problem. On the other hand, by treating each feasible weighted combination of elements as a distinct arm, it is possible to handle these constraints using prior approaches for multi-armed bandits with single play (such as the well-known UCB1 index policy of Auer *et al.* [7]). However, this approach turns out to be naive and yields poor performance scaling in terms of regret, storage, and computation. This is because this approach maintains and computes quantities for each possible combination separately and does not exploit potential dependencies between them. In this paper, we instead propose smarter policies to handle the arbitrary constraints that explicitly take into account the linear nature of the dependencies and base all storage and computations on the unknown variables directly. As we shall show, this saves not only on storage and computation, but also substantially reduces the regret compared to the naive approach.

Specifically, we first present a novel policy called Learning with Linear Rewards (LLR) that requires only $O(N)$ storage and yields a regret that grows essentially[1] as $O(N^4 \ln n)$, where n is the time index. We also discuss how this policy can be modified in a straightforward manner while maintaining the same performance guarantees when the problem is one of cost minimization rather than reward maximization. A key step in these policies we propose is the solving of a deterministic combinatorial optimization with a linear objective. While this is NP-hard in general (as it includes 0-1 integer linear programming), there are still many special-case combinatorial problems of practical interest that can be solved in polynomial time. For such problems, the policy we propose would thus inherit the property of polynomial computation at each step. Furthermore, we present in the Appendix suitably relaxed results on the regret that would be obtained for computationally harder problems when an approximation algorithm with a known guarantee is used.

We also present in this paper a more general K-action formulation, in which the policy is allowed to pick $K \geq 1$ different combinations of variables each time. We show how the basic LLR policy can be readily extended to handle this and present the regret analysis for this case as well.

The examples of combinatorial network optimization that we present are far from exhausting the possible applications of the formulation and the policies we present in this paper—there are many other linear-objective network optimization problems [8], [9]. Our framework, for the first time, allows these problems to be solved in stochastic settings with unknown random coefficients with provably efficient performance. Besides communication networks, we expect that our work will also find practical application in other fields where linear combinatorial optimization problems arise naturally, such as algorithmic economics, data mining, finance, operations research, and industrial engineering.

This paper is organized as follows. We first provide a survey of related work in Section II. We then give a formal description of the multi-armed bandits with linear rewards problem we solve in Section III. In Section IV, we present our LLR policy and show that it requires only polynomial storage and polynomial computation per time period. We present the novel analysis of the regret of this policy in Section V and point out how this analysis generalizes known results on MABs. In Section VI, we discuss examples and applications of maximum weight matching, shortest path, and minimum spanning tree computations to show that our policy is widely useful for various interesting applications in networks with the tractable combinatorial optimization formulation with linear objective functions. Section VII shows the numerical simulation results. We show an extension of our policy for choosing K largest actions in Section VIII. We conclude with a summary of our contribution and point out avenues for future work in Section IX. We also present an Appendix where we show results on a suitably relaxed notion of regret that is useful for computationally hard problems where an approximation algorithm with a known guarantee is available.

II. Related Work

Lai and Robbins [10] wrote one of the earliest papers on the classic non-Bayesian infinite horizon multi-armed bandit problem. Assuming N independent arms, each generating rewards that are i.i.d. over time from a given family of distributions with an unknown real-valued parameter, they presented a general policy that provides expected regret that is $O(N \log n)$, i.e., linear in the number of arms and asymptotically logarithmic in n. They also show that this policy is order-optimal in that no policy can do better than $\Omega(N \log n)$. Anantharam *et al.* [5] extend this work to the case when M multiple plays are allowed. Agrawal *et al.* [11] further extend this to the case when there are multiple plays and also switching costs are taken into account.

Our study is influenced by the works by Agrawal [6] and Auer *et al.* [7]. The work by Agrawal [6] first presented easy-to-compute upper confidence bound (UCB) policies based on the sample-mean that also yield asymptotically logarithmic regret. Auer *et al.* [7] build on [6] and present variants of Agrawal's policy, including the so-called UCB1 policy, and prove bounds on the regret that are logarithmic uniformly over time (i.e., for any finite n, not only asymptotically), so long as the arm rewards have a finite support. There are similarities in the proof techniques used in these works [6], [7], which both use known results on large-deviation upper bounds. In our paper, we also make use of this approach, leveraging the same Chernoff–Hoeffding bound utilized in [7]. However, these works do not exploit potential dependencies between the arms.[2] As we show in this paper, a direct application of the UCB1 policy therefore performs poorly for our problem formulation.

Unlike Lai and Robbins [10], Agrawal [6], and Auer *et al.* [7], we consider in this paper a more general combinatorial version of the problem that allows for the selection of a set of multiple variables simultaneously so long as they satisfy *a given*

[1] This is a simplification of our key result in Section V which gives a tighter expression for the bound on regret that applies uniformly over time, not just asymptotically.

[2] Both the papers by Agrawal [6] and Auer *et al.* [7] indicate in the problem formulation that the rewards are independent across arms. However, since their proof technique bounds separately the expected time spent on each nonoptimal arm, in fact the bounds of the expected regret that they get through linearity of expectation applies even when the arm rewards are not independent. Nevertheless, as we indicate, the policies do not exploit any dependencies that may exist between the arm rewards.

arbitrary constraint. The constraint can be specified explicitly in terms of sets of variables that are allowed to be picked together. There is no restriction on how these sets are constructed. They may correspond, for example, to some structural property such as all possible paths or matchings on a graph. While we credit the paper by Anantharam *et al.* [5] for being the first to consider multiple simultaneous plays, we note that our formulation is much more general than that work. Specifically, Anantharam *et al.* [5] consider only a particular kind of constraint: It allows selection of all combinations of a fixed number of arms (i.e., in [5], exactly M arms must be played at each time). For this reason, the algorithm presented in [5] cannot be directly used for the more general combinatorial problem in our formulation. For the same reason, the algorithm presented in [11] also cannot be used directly for our problem.

In our formulation, we assume that the rewards from each individual variable in the selected combination are observed, and the total reward is a linearly weighted combination of these variables. Because we consider a fundamentally different and more general problem formulation, our proof strategy, while sharing structural similarities with [6] and [7], has a nontrivial innovative component as well. In particular, in our setting, it turns out to be difficult to directly bound the number of actual plays of each weighted combination of variables, and therefore we create a carefully defined virtual counter for each individual variable and bound that instead.

There are also some recent works to propose decentralized policies for the multi-armed bandit problem with multiple plays. Liu and Zhao [4], Anandkumar *et al.* [3], and a work by two of the authors of this paper [12] present policies for the problem of M distributed players operating N arms. These papers address decentralized versions of the same fixed M-play problem considered in [5] and are therefore also not applicable to the combinatorial generalization of this paper. The problem of formulating and solving the combinatorial setting that we present here in a decentralized fashion is currently open.

While our focus and those of the above works are primarily on i.i.d. rewards, there have also been some prior and recent works looking at non-Bayesian regret formulations for multi-armed bandits with Markovian rewards [13]–[17]. We have recently obtained some preliminary results on weaker notions of regret (with respect to suboptimal single-action policies) for rested Markovian rewards for the combinatorial problem of maximum-weight matching in bipartite graphs [18]. However, the problem of generalizing our work to obtain stronger regret results for combinatorial multi-armed bandits with rested and restless Markovian rewards remains open.

While these above key papers and many others have focused on independent arms, there have been some works treating dependencies between arms. The paper by Pandey *et al.* [1] divides arms into clusters of dependent arms (in our case, there would be only one such cluster consisting of all the arms). Their model assumes that each arm provides only binary rewards, and in any case, they do not present any theoretical analysis on the expected regret. Ortner [19] proposes to use an additional arm color to utilize the given similarity information of different arms to improve the upper bound of the regret. They assume that the difference of the mean rewards of any two arms with the same color is less than a predefined parameter δ, which is known to the user. This is different from the linear reward model in our paper.

Mersereau *et al.* [20] consider a bandit problem where the expected reward is defined as a linear function of a random variable, and the prior distribution is known. They show the upper bound of the regret is $O(\sqrt{n})$ and the lower bound of the regret is $\Omega(\sqrt{n})$. Rusmevichientong and Tsitsiklis [21] extend [20] to the setting where the reward from each arm is modeled as the sum of a linear combination of a set of unknown static random numbers and a zero-mean random variable that is i.i.d. over time and independent across arms. The upper bound of the regret is shown to be $O(N\sqrt{n})$ on the unit sphere and $O(N\sqrt{n}\log^{3/2} n)$ for a compact set, and the lower bound of regret is $\Omega(N\sqrt{n})$ for both cases. The linear models in these works are different from our paper, in which the reward is expressed as a linear combination as a set of random processes. A key difference, however, is that in [20] and [21], it is assumed that only the total reward is observed at each time, not the individual rewards. In our paper, we assume that all the selected individual random variables are observed at each time (from which the total reward can be inferred). Because of the more limited coarse-grained feedback, the problems tackled in [20] and [21] are indeed much more challenging, perhaps explaining why they result in a higher regret bound order.

Both [22] and [23] consider linear reward models that are more general than ours, but also under the assumption that only the total reward is observed at each time. Auer [22] presents a randomized policy that requires storage and computation to grow linearly in the number of arms. This algorithm is shown to achieve a regret upper bound of $O(\sqrt{N}\sqrt{n}\log^{3/2}(n|\mathcal{F}|))$. Dani *et al.* [23] develop another randomized policy for the case of a compact set of arms and show the regret is upper-bounded by $O(N\sqrt{n}\log^{3/2} n)$ for sufficiently large n with high probability, and lower-bounded by $\Omega(N\sqrt{n})$. They also show that when the difference in costs (denoted as Δ) between the optimal and next-to-optimal decision among the extremal points is greater than zero, the regret is upper-bounded by $O(N^2/\Delta\log^3 n)$ for sufficiently large n with high probability.

Another paper that is related to our work is by Awerbuch and Kleinberg [24]. They consider the problem of shortest path routing in a nonstochastic, adversarial setting, in which only the total cost of the selected path is revealed at each time. For this problem, assuming the edge costs on the graph are chosen by an adaptive adversary that can view the past actions of the policy, they present a policy with regret scaling as $O(n^{2/3}(\log(n))^{1/3})$ over n time-steps. However, although as we discuss our formulation can also be applied to online shortest path routing, our work is different from [24] in that we consider a stochastic, nonadversarial setting and allow for observations of the individual edge costs of the selected path at each time.

To summarize, ours is the first paper on stochastic combinatorial multi-armed bandits to consider linearly weighted rewards along with observation of the selected random variables, allowing for arbitrary constraints on the set of weights. We present a deterministic policy with a finite-time bound of regret that grows as $O(N^4\log n)$, i.e., polynomially in the number of unknown random variables and strictly logarithmically in time.

Our work in this paper is an extension of our recent work that introduced combinatorial multi-armed bandits [25]. The

formulation in [25] has the restriction that the reward is generated from a matching in a bipartite graph of users and channels. Our work in this paper generalizes this to a broader formulation with linear rewards and arbitrary constraints.

III. Problem Formulation

We consider a discrete time system with N unknown random processes $X_i(n), 1 \leq i \leq N$, where time is indexed by n. We assume that $X_i(n)$ evolves as an i.i.d. random process over time, with the only restriction that its distribution have a finite support. Without loss of generality, we normalize $X_i(n) \in [0,1]$. We do not require that $X_i(n)$ be independent across i. This random process is assumed to have a mean $\theta_i = \mathbb{E}[X_i]$ that is unknown to the users. We denote the set of all these means as $\Theta = \{\theta_i\}$.

At each decision period n (also referred to interchangeably as time-slot), an N-dimensional *action* vector $\mathbf{a}(n)$ is selected under a policy $\pi(n)$ from a finite set $\mathcal{F}$. We assume $a_i(n) \geq 0$ for all $1 \leq i \leq N$. When a particular $\mathbf{a}(n)$ is selected, only for those i with $a_i(n) \neq 0$, the value of $X_i(n)$ is observed. We denote $\mathcal{A}_{\mathbf{a}(n)} = \{i : a_i(n) \neq 0, 1 \leq i \leq N\}$, the index set of all $a_i(n) \neq 0$ for an action $\mathbf{a}$. The reward is defined as

$$R_{\mathbf{a}(n)}(n) = \sum_{i=1}^{N} a_i(n) X_i(n). \tag{1}$$

When a particular action $a(n)$ is selected, the random variables corresponding to nonzero components of $a(n)$ are revealed,[3] i.e., the value of $X_i(n)$ is observed for all i such that $\mathbf{a}(n) \neq 0$.

We evaluate policies with respect to *regret*, which is defined as the difference between the expected reward that could be obtained by a genie that can pick an optimal action at each time, and that obtained by the given policy. Note that minimizing the regret is equivalent to maximizing the rewards. Regret can be expressed as

$$\mathfrak{R}_n^{\pi}(\Theta) = n\theta^* - E^{\pi}\left[\sum_{t=1}^{n} R_{\pi(t)}(t)\right] \tag{2}$$

where $\theta^* = \max_{\mathbf{a}\in\mathcal{F}} \sum_{i=1}^{N} a_i\theta_i$, the expected reward of an optimal action. For the rest of the paper, we use $*$ as the index indicating that a parameter is for an optimal action. If there is more than one optimal action exist, $*$ refers to any one of them.

Intuitively, we would like the regret $\mathfrak{R}_n^{\pi}(\Theta)$ to be as small as possible. If it is sublinear with respect to time n, the time-averaged regret will tend to zero and the maximum possible time-averaged reward can be achieved. Note that the number of actions $|\mathcal{F}|$ can be exponential in the number of unknown random variables N.

IV. Policy Design

A. Naive Approach

A unique feature of our problem formulation is that the action selected at each time can be chosen such that the corresponding collection of individual variables satisfies an arbitrary structural constraint. For this reason, as we indicated in our related works discussion, prior work on MABs with fixed number of multiple plays, such as [5], or on linear reward models, such as [23], cannot be applied to this problem. One straightforward, relatively naive approach to solving the combinatorial multi-armed bandits problem that we defined is to treat each arm as an action, which allows us to use the UCB1 policy given by Auer *et al.* [7]. Using UCB1, each action is mapped into an arm, and the action that maximizes $\hat{Y}_k + \sqrt{2\ln n/m_k}$ will be selected at each time-slot, where $\hat{Y}_k$ is the mean observed reward on action k, and m_k is the number of times that action k has been played. This approach essentially ignores the dependencies across the different actions, storing observed information about each action independently, and making decisions based on this information alone.

Auer *et al.* [7] showed the following policy performance for regret upper bound as in Theorem 1.

Theorem 1: The expected regret under UCB1 policy is at most

$$\left[8\sum_{k:\theta_k<\theta^*}\left(\frac{\ln n}{\Delta_k}\right)\right] + \left(1+\frac{\pi^2}{3}\right)\left(\sum_{k:\theta_k<\theta^*}\Delta_k\right) \tag{3}$$

where $\Delta_k = \theta^* - \theta_k$, $\theta_k = \sum_{i\in\mathcal{A}_k} a_i\theta_i$.

Proof: See [7, Theorem 1]. ∎

Note that UCB1 requires storage that is linear in the number of actions and yields regret growing linearly with the number of actions. In a case where the number of actions grow exponentially with the number of unknown variables, both of these are highly unsatisfactory.

Intuitively, UCB1 algorithm performs poorly on this problem because it ignores the underlying dependencies. This motivates us to propose a sophisticated policy that more efficiently stores observations from correlated actions and exploits the correlations to make better decisions.

B. New Policy

Our proposed policy, which we refer to as Learning with Linear Rewards (LLR), is shown in Algorithm 1.

Algorithm 1: Learning with Linear Rewards (LLR)

1: // Initialization
2: If $\max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$ is known, let $L = \max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$; else, $L = N$;
3: **for** $p = 1$ to N **do**
4: $n = p$;
5: Play any action $\mathbf{a}$ such that $p \in \mathcal{A}_{\mathbf{a}}$;
6: Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
7: **end for**
8: // Main loop
9: **while** 1 **do**
10: $n = n + 1$;
11: Play an action $\mathbf{a}$ which solves the maximization problem

$$\mathbf{a} = \arg\max_{\mathbf{a}\in\mathcal{F}} \sum_{i\in\mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i + \sqrt{\frac{(L+1)\ln n}{m_i}}\right); \tag{4}$$

12: Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
13: **end while**

[3] As noted in the related work, this is a key assumption in our work that differentiates it from other prior work on linear dependent-arm bandits [14], [15]. This is a very reasonable assumption in many cases, for instance, in the combinatorial network optimization applications we discuss in Section VI, it corresponds to revealing weights on the set of edges selected at each time.

TABLE I
NOTATION

N :	number of random variables.
$\mathbf{a}$:	vectors of coefficients, defined on set $\mathcal{F}$.
$\mathcal{A}_{\mathbf{a}}$:	$\{i : a_i \neq 0, 1 \leq i \leq N\}$.
$*$:	index indicating that a parameter is for an optimal action.
m_i:	number of times that X_i has been observed up to the current time slot.
$\hat{\theta}_i$:	average (sample mean) of all the observed values of X_i up to the current time slot. Note that $\mathbb{E}[\hat{\theta}_i(n)] = \theta_i$.
$\hat{\bar{\theta}}_{i,m_i}$:	average (sample mean) of all the observed values of X_i when it is observed m_i times.
$\Delta_{\mathbf{a}}$:	$R^* - R_{\mathbf{a}}$.
$\Delta_{\min}$:	$\min_{R_{\mathbf{a}}<R^*} \Delta_{\mathbf{a}}$.
$\Delta_{\max}$:	$\max_{R_{\mathbf{a}}<R^*} \Delta_{\mathbf{a}}$.
$T_{\mathbf{a}}(n)$:	number of times action $\mathbf{a}$ has been played in the first n time slots.
$a_{\max}$:	$\max_{\mathbf{a}\in\mathcal{F}} \max_i a_i$.

Table I summarizes some notation we use in the description and analysis of our algorithm.

The key idea behind this algorithm is to store and use observations for each random variable, rather than for each action as a whole. Since the same random variable can be observed while operating different actions, this allows exploitation of information gained from the operation of one action to make decisions about a dependent action.

We use two $1 \times N$ vectors to store the information after we play an action at each time-slot. One is $(\hat{\theta}_i)_{1\times N}$, in which $\hat{\theta}_i$ is the average (sample mean) of all the observed values of X_i up to the current time-slot (obtained through potentially different sets of actions over time). The other one is $(m_i)_{1\times N}$, in which m_i is the number of times that X_i has been observed up to the current time-slot.

At each time-slot n, after an action $\mathbf{a}(n)$ is played, we get the observation of $X_i(n)$ for all $i \in \mathcal{A}_{\mathbf{a}(n)}$. Then, $(\hat{\theta}_i)_{1\times N}$ and $(m_i)_{1\times N}$ (both initialized to 0 at time 0) are updated as follows:

$$\hat{\theta}_i(n) = \begin{cases} \frac{\hat{\theta}_i(n-1)m_i(n-1)+X_i(n)}{m_i(n-1)+1}, & \text{if } i \in \mathcal{A}_{\mathbf{a}(n)} \\ \hat{\theta}_i(n-1), & \text{else} \end{cases} \quad (5)$$

$$m_i(n) = \begin{cases} m_i(n-1)+1, & \text{if } i \in \mathcal{A}_{\mathbf{a}(n)} \\ m_i(n-1), & \text{else.} \end{cases} \quad (6)$$

Note that while we indicate the time index in the above updates for notational clarity, it is not necessary to store the matrices from previous time steps while running the algorithm.

LLR policy requires storage linear in N. In Section V, we will present the analysis of the upper bound of regret and show that it is polynomial in N and logarithmic in time. Note that the maximization problem (4) needs to be solved as the part of LLR policy. It is a deterministic linear optimal problem with a feasible set $\mathcal{F}$, and the computation time for an arbitrary $\mathcal{F}$ may not be polynomial in N. As we show in Section VI, there exist many practically useful examples with polynomial computation time.

V. Analysis of Regret

Traditionally, the regret of a policy for a multi-armed bandit problem is upper-bounded by analyzing the expected number of times that each nonoptimal action is played and summing this expectation over all nonoptimal actions. While such an approach will work to analyze the LLR policy as well, it turns out that the upper bound for regret consequently obtained is quite loose, being linear in the number of actions, which may grow faster than polynomials. Instead, we give here a tighter analysis of the LLR policy that provides an upper bound that is instead polynomial in N and logarithmic in time. Like the regret analysis in [7], this upper bound is valid for finite n.

Theorem 2: The expected regret under the LLR policy is at most

$$\left[\frac{4a_{\max}^2 L^2(L+1)N\ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LN\right]\Delta_{\max}. \quad (7)$$

To prove Theorem 2, we use the inequalities as stated in the Chernoff–Hoeffding bound [26].

Lemma 1 (Chernoff–Hoeffding Bound [26]): $X_1, \ldots, X_n$ are random variables with range [0,1], and $\mathbb{E}[X_t|X_1, \ldots, X_{t-1}] = \mu, \forall 1 \leq t \leq n$. Denote $S_n = \sum X_i$. Then, for all $a \geq 0$

$$Pr\{S_n \geq n\mu + a\} \leq e^{-2a^2/n}$$
$$Pr\{S_n \leq n\mu - a\} \leq e^{-2a^2/n}. \quad (8)$$

Proof of Theorem 2: Denote C_{t,m_i} as $\sqrt{(L+1)\ln t/m_i}$. We introduce $\widetilde{T}_i(n)$ as a counter after the initialization period. It is updated in the following way.

At each time-slot after the initialization period, one of the two cases must happen: 1) an optimal action is played; 2) a nonoptimal action is played. In the first case, $(\widetilde{T}_i(n))_{1\times N}$ will not be updated. When an nonoptimal action $\mathbf{a}(n)$ is picked at time n, there must be at least one $i \in \mathcal{A}_{\mathbf{a}}$ such that $i = \arg\min_{j\in\mathcal{A}_{\mathbf{a}}} m_j$. If there is only one such action, $\widetilde{T}_i(n)$ is increased by 1. If there are multiple such actions, we arbitrarily pick one, say i', and increment $\widetilde{T}_{i'}$ by 1.

Each time when a nonoptimal action is picked, exactly one element in $(\widetilde{T}_i(n))_{1\times N}$ is incremented by 1. This implies that the total number that we have played the nonoptimal actions is equal to the summation of all counters in $(\widetilde{T}_i(n))_{1\times N}$, i.e., $\sum_{\mathbf{a}:R_{\mathbf{a}}<R^*} T_{\mathbf{a}}(n) = \sum_{i=1}^{N} \widetilde{T}_i(n)$, and hence

$$\mathbb{E}\left[\sum_{\mathbf{a}:R_{\mathbf{a}}<R^*} T_{\mathbf{a}}(n)\right] = \mathbb{E}\left[\sum_{i=1}^{N} \widetilde{T}_i(n)\right]. \quad (9)$$

Therefore, we have

$$\sum_{\mathbf{a}:R_{\mathbf{a}}<R^*} \mathbb{E}\left[T_{\mathbf{a}}(n)\right] = \sum_{i=1}^{N} \mathbb{E}\left[\widetilde{T}_i(n)\right]. \quad (10)$$

Also note, for $\widetilde{T}_i(n)$, the following inequality holds:

$$\widetilde{T}_i(n) \leq m_i(n) \qquad \forall 1 \leq i \leq N. \quad (11)$$

Denote by $\widetilde{I}_i(n)$ the indicator function that is equal to 1 if $\widetilde{T}_i(n)$ is added by one at time n. Let l be an arbitrary positive integer. Then

$$\begin{aligned}\widetilde{T}_i(n) &= \sum_{t=N+1}^{n} \mathbb{1}\left\{\widetilde{I}_i(t)=1\right\} \\ &\le l + \sum_{t=N+1}^{n} \mathbb{1}\left\{\widetilde{I}_i(t)=1, \widetilde{T}_i(t-1)\ge l\right\} \end{aligned} \quad (12)$$

where $\mathbb{1}(x)$ is the indicator function defined to be 1 when the predicate x is true, and 0 when it is false. When $\widetilde{I}_i(t) = 1$, a nonoptimal action $\mathbf{a}(t)$ has been picked for which $m_i = \min_j\{m_j : \forall j \in \mathcal{A}_{\mathbf{a}(t)}\}$. We denote this action as $\mathbf{a}(t)$ since at each time that $\widetilde{I}_i(t) = 1$, we could get different actions. Then

$$\begin{aligned}&\widetilde{T}_i(n) \\ &\le l+\sum_{t=N+1}^{n}\mathbb{1}\Bigg\{\sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*\left(\hat{\overline{\theta}}_{j,m_j(t-1)}+C_{t-1,m_j(t-1)}\right) \\ &\quad \le \sum_{j\in\mathcal{A}_{\mathbf{a}(t)}} a_j(t)\left(\hat{\overline{\theta}}_{j,m_j(t-1)}+C_{t-1,m_j(t-1)}\right), \widetilde{T}_i(t-1)\ge l\Bigg\} \\ &\le l+\sum_{t=N}^{n}\mathbb{1}\Bigg\{\sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*\left(\hat{\overline{\theta}}_{j,m_j(t)}+C_{t,m_j(t)}\right) \\ &\quad \le \sum_{j\in\mathcal{A}_{\mathbf{a}(t+1)}} a_j(t+1)\left(\hat{\overline{\theta}}_{j,m_j(t)}+C_{t,m_j(t)}\right), \widetilde{T}_i(t)\ge l\Bigg\}. \end{aligned} \quad (13)$$

Note that $l \le \widetilde{T}_i(t)$ implies

$$l \le \widetilde{T}_i(t) \le m_j(t),\ \forall j \in \mathcal{A}_{\mathbf{a}(t+1)}. \quad (14)$$

$$\begin{aligned}&\widetilde{T}_i(n) \\ &\le l+\sum_{t=N}^{n}\mathbb{1}\Bigg\{\min_{0<m_{h_1},\ldots,m_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}\le t}\ \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\left(\hat{\overline{\theta}}_{h_j,m_{h_j}}+C_{t,m_{h_j}}\right) \\ &\quad \le \max_{l\le m_{p_1},\ldots,m_{p_{|\mathcal{A}_{\mathbf{a}(t+1)}|}}\le t}\ \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\left(\hat{\overline{\theta}}_{p_j,m_{p_j}}+C_{t,m_{p_j}}\right)\Bigg\} \\ &\le l+\sum_{t=1}^{\infty}\sum_{m_{h_1}=1}^{t}\cdots\sum_{m_{h_{|\mathcal{A}^*|}}=1}^{t}\sum_{m_{p_1}=l}^{t}\cdots\sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t+1)}|}}=l}^{t} \\ &\quad \mathbb{1}\Bigg\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a_{h_j}^*\left(\hat{\overline{\theta}}_{h_j,m_{h_j}}+C_{t,m_{h_j}}\right) \\ &\qquad \le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\left(\hat{\overline{\theta}}_{p_j,m_{p_j}}+C_{t,m_{p_j}}\right)\Bigg\} \end{aligned} \quad (15)$$

where $h_j (1 \le j \le |\mathcal{A}_{\mathbf{a}*}|)$ represents the jth element in $\mathcal{A}_{\mathbf{a}*}$ and $p_j (1 \le j \le |\mathcal{A}_{\mathbf{a}(t+1)}|)$ represents the jth element in $\mathcal{A}_{\mathbf{a}(t+1)}$.

$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*(\hat{\overline{\theta}}_{h_j,m_{h_j}}+C_{t,m_{h_j}}) \le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)(\hat{\overline{\theta}}_{p_j,m_{p_j}}+C_{t,m_{p_j}})$ means that at least one of the following must be true:

$$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a_{h_j}^* C_{t,m_{h_j}} \quad (16)$$

$$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\hat{\overline{\theta}}_{p_j,m_{p_j}} \ge R_{\mathbf{a}(t+1)} + \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)C_{t,m_{p_j}} \quad (17)$$

$$R^* < R_{\mathbf{a}(t+1)} + 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)C_{t,m_{p_j}}. \quad (18)$$

Now we find the upper bound for $Pr\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* C_{t,m_{h_j}}\}$.

We have

$$\begin{aligned}&Pr\left\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* C_{t,m_{h_j}}\right\} \\ &= Pr\left\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\theta_{h_j} - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* C_{t,m_{h_j}}\right\} \\ &\le Pr\Big\{\text{At least one of the following must hold :} \\ &\qquad a_{h_1}^*\hat{\overline{\theta}}_{h_1,m_{h_1}} \le a_{h_1}^*\theta_{h_1} - a_{h_1}^* C_{t,m_{h_1}} \\ &\qquad a_{h_2}^*\hat{\overline{\theta}}_{h_2,m_{h_2}} \le a_{h_2}^*\theta_{h_2} - a_{h_2}^* C_{t,m_{h_2}} \\ &\qquad\qquad \vdots \\ &\qquad a_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}^*\hat{\overline{\theta}}_{h_1,m_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}} \le a_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}^*\theta_{h_{|\mathcal{A}_{\mathbf{a}^*}|}} \\ &\qquad\qquad - a_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}^* C_{t,m_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}}\Big\} \end{aligned} \quad (19)$$

$$\begin{aligned}&\le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} Pr\left\{a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \le a_{h_j}^*\theta_{h_j} - a_{h_j}^* C_{t,m_{h_j}}\right\} \\ &= \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} Pr\left\{\hat{\overline{\theta}}_{h_j,m_{h_j}} \le \theta_{h_j} - C_{t,m_{h_j}}\right\}. \end{aligned} \quad (20)$$

$\forall 1 \le j \le |\mathcal{A}_{\mathbf{a}*}|$, applying the Chernoff–Hoeffding bound stated in Lemma 1, we could find the upper bound of each item in the previous equation as

$$\begin{aligned}&Pr\left\{\hat{\overline{\theta}}_{h_j,m_{h_j}} \le \theta_{h_j} - C_{t,m_{h_j}}\right\} \\ &= Pr\left\{m_{h_j}\hat{\overline{\theta}}_{h_j,m_{h_j}} \le m_{h_j}\theta_{h_j} - m_{h_j}C_{t,m_{h_j}}\right\} \\ &\le e^{-2\cdot\frac{1}{m_{h_i j}}\cdot(m_{h_j})^2\cdot\frac{(L+1)\ln t}{m_{h_j}}} \\ &= e^{-2(L+1)\ln t} \\ &= t^{-2(L+1)}. \end{aligned}$$

Thus

$$\begin{aligned} &Pr\left\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a^*_{h_j}\hat{\overline{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a^*_{h_j} C_{t,m_{h_j}}\right\} \\ &\le |\mathcal{A}_{\mathbf{a}*}| t^{-2(L+1)} \\ &\le \quad Lt^{-2(L+1)}. \end{aligned} \tag{21}$$

Similarly, we can get the upper bound of the probability for inequality (17)

$$\begin{aligned} &Pr\left\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\hat{\overline{\theta}}_{p_j,m_{p_j}} \right. \\ &\qquad \left. \ge R_{\mathbf{a}(t+1)} + \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1) C_{t,m_{p_j}}\right\} \\ &\le Lt^{-2(L+1)}. \end{aligned} \tag{22}$$

Note that for $l \ge \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\mathbf{a}(t+1)}}{La_{\max}}\right)^2} \right\rceil$

$$\begin{aligned} &R^* - R_{\mathbf{a}(t+1)} - 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1) C_{t,m_{p_j}} \\ &= R^* - R_{\mathbf{a}(t+1)} - 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\sqrt{\frac{(L+1)\ln t}{m_{p_j}}} \\ &\ge R^* - R_{\mathbf{a}(t+1)} - La_{\max}\sqrt{\frac{4(L+1)\ln n}{l}} \\ &\ge R^* - R_{\mathbf{a}(t+1)} - La_{\max}\sqrt{\frac{4(L+1)\ln n}{4(L+1)\ln n}\left(\frac{\Delta_{\mathbf{a}(t+1)}}{La_{\max}}\right)^2} \\ &\ge R^* - R_{\mathbf{a}(t+1)} - \Delta_{\mathbf{a}(t+1)} = 0. \end{aligned} \tag{23}$$

Equation (23) implies that condition (18) is false when $l = \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\mathbf{a}(t+1)}}{La_{\max}}\right)^2} \right\rceil$. If we let $l = \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{La_{\max}}\right)^2} \right\rceil$, then (18) is false for all $\mathbf{a}(t+1)$.

Therefore

$$\begin{aligned} &\mathbb{E}\left[\widetilde{T}_i(n)\right] \\ &\le \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{La_{\max}}\right)^2} \right\rceil \\ &\quad + \sum_{t=1}^{\infty}\left(\sum_{m_{h_1}=1}^{t}\cdots\sum_{m_{h_{|\mathcal{A}^*|}}=1}^{t}\sum_{m_{p_1}=l}^{t}\cdots\sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{t} 2Lt^{-2(L+1)}\right) \\ &\le \frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + 1 + L\sum_{t=1}^{\infty}2t^{-2} \\ &\le \frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3}L. \end{aligned} \tag{24}$$

Thus, under LLR policy, we have

$$\begin{aligned} \mathfrak{R}^{\pi}_n(\Theta) &= R^* n - \mathbb{E}^{\pi}\left[\sum_{t=1}^{n} R_{\pi(t)}(t)\right] \\ &= \sum_{\mathbf{a}:R_{\mathbf{a}}<R^*} \Delta_{\mathbf{a}}\mathbb{E}[T_{\mathbf{a}}(n)] \\ &\le \Delta_{\max}\sum_{\mathbf{a}:R_{\mathbf{a}}<R^*}\mathbb{E}[T_{\mathbf{a}}(n)] \\ &= \Delta_{\max}\sum_{i=1}^{N}\mathbb{E}\left[\widetilde{T}_i(n)\right] \\ &\le \left[\sum_{i=1}^{N}\frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LN\right]\Delta_{\max} \\ &\le \left[\frac{4a^2_{\max}L^2(L+1)N\ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LN\right]\Delta_{\max}. \end{aligned} \tag{25}$$

■

Remark 1: Note that when the set of action vectors consists of binary vectors with a single "1," the problem formulation reduces to a multi-armed bandit problem with N independent actions. In this special case, the LLR algorithm is equivalent to UCB1 in [7]. Thus, our results generalize that prior work.

Remark 2: We have presented $\mathcal{F}$ as a finite set in our problem formation. We note that the LLR policy we have described and its analysis actually also work with a more general formulation when $\mathcal{F}$ is an infinite set with the following additional constraints: the maximization problem in (4) always has at least one solution; $\Delta_{\min}$ exists; a_i is bounded. With the above constraints, Algorithm 1 will work the same, and the conclusion and all the details of the proof of Theorem 2 can remain the same.

Remark 3: In fact, Theorem 2 also holds for certain kinds of non-i.i.d. random variables $X_i, 1 \le i \le N$ that satisfy the condition that $E[X_i(t)|X_i(1),\ldots,X_i(t-1)] = \theta_i, \forall 1 \le i \le N$. This is because the Chernoff–Hoeffding bound used in the regret analysis requires only this condition to hold.[4]

VI. Applications

We now describe some applications and extensions of the LLR policy for combinatorial network optimization in graphs where the edge weights are unknown random variables.

A. Maximum Weighted Matching

Maximum weighted matching (MWM) problems are widely used in the many optimization problems in wireless networks such as the prior work in [27], [28]. Given any graph $G = (V, E)$, there is a weight associated with each edge, and the objective is to maximize the sum weights of a matching among all the matchings in a given constraint set, i.e., the general formulation for MWM problem is

$$\begin{aligned} \max \quad & R^{\mathrm{MWM}}_{\mathbf{a}} = \sum_{i=1}^{|E|} a_i W_i \\ \text{s.t.} \quad & \mathbf{a} \text{ is a matching} \end{aligned} \tag{26}$$

where W_i is the weight associated with each edge i.

[4]This does not, however, include Markov chains for which we have recently obtained some weaker regret results [35].

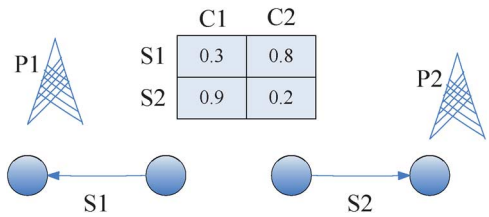

Fig. 1. Illustrative scenario.

In many practical applications, the weights are unknown random variables and we need to learn by selecting different matchings over time. This kind of problem fits the general framework of our proposed policy regarding the reward as the sum weight and a matching as an action. Our proposed LLR policy is a solution with linear storage, and the regret polynomial in the number of edges, and logarithmic in time.

There are various algorithms to solve the different variations in the maximum weighted matching problems, such as the Hungarian algorithm for the maximum weighted bipartite matching [29] and Edmonds's matching algorithm [30] for a general maximum matching. In these cases, the computation time is also polynomial.

Here, we present a general problem of multiuser channel allocations in cognitive radio network. There are M secondary users and Q orthogonal channels. Each secondary user requires a single channel for operation that does not conflict with the channels assigned to the other users. Due to geographic dispersion, each secondary user can potentially see different primary user occupancy behavior on each channel. Time is divided into discrete decision rounds. The throughput obtainable from spectrum opportunities on each user-channel combination over a decision period is denoted as $S_{i,j}$ and modeled as an arbitrarily distributed random variable with bounded support but unknown mean, i.i.d. over time. This random process is assumed to have a mean $\theta_{i,j}$ that is unknown to the users. The objective is to search for an allocation of channels for all users that maximizes the expected sum throughput.

Assuming an interference model whereby at most one secondary user can derive benefit from any channel, if the number of channels is greater than the number of users, an optimal channel allocation employs a one-to-one matching of users to channels, such that the expected sum-throughput is maximized.

Fig. 1 illustrates a simple scenario. There are two secondary users (i.e., links) S1 and S2, that are each assumed to be in interference range of each other. S1 is proximate to primary user P1, who is operating on channel 1. S2 is proximate to primary user P2, who is operating on channel 2. The matrix shows the corresponding Θ, i.e., the throughput each secondary user could derive from being on the corresponding channel. In this simple example, the optimal matching is for secondary user 1 to be allocated channel 2 and user 2 to be allocated channel 1. Note, however, that, in our formulation, the users are not *a priori* aware of the matrix of mean values, and therefore must follow a sequential learning policy.

Note that this problem can be formulated as a multi-armed bandits one with linear regret, in which each action corresponds to a matching of the users to channels, and the reward corresponds to the sum-throughput. In this channel allocation problem, there are $M \times Q$ unknown random variables, and the number of actions is $P(Q, M)$, which can grow exponentially in the number of unknown random variables. Following the convention, instead of denoting the variables as a vector, we refer it as an $M \times Q$ matrix. Therefore, the reward as each time-slot by choosing a permutation $\mathbf{a}$ is expressed as

$$R_{\mathbf{a}} = \sum_{i=1}^{M} \sum_{j=1}^{Q} a_{i,j} S_{i,j} \tag{27}$$

where $\mathbf{a} \in \mathcal{F}$, $\mathcal{F}$ is a set with all permutations, which is defined as

$$\mathcal{F} = \left\{ \mathbf{a} : a_{i,j} \in \{0,1\},\ \forall i,j \wedge \sum_{i=1}^{Q} a_{i,j} = 1 \wedge \sum_{j=1}^{Q} a_{i,j} = 1 \right\}. \tag{28}$$

We use two $M \times Q$ matrices to store the information after we play an action at each time-slot. One is $(\hat{\theta}_{i,j})_{M\times Q}$, in which $\hat{\theta}_{i,j}$ is the average (sample mean) of all the observed values of channel j by user i up to the current time-slot (obtained through potentially different sets of actions over time). The other one is $(m_{i,j})_{M\times Q}$, in which $m_{i,j}$ is the number of times that channel j has been observed by user i up to the current time-slot.

Applying Algorithm 1, we get a linear storage policy for which $(\hat{\theta}_{i,j})_{M\times Q}$ and $(m_{i,j})_{M\times Q}$ are stored and updated at each time-slot. The regret is polynomial in the number of users and channels, and logarithmic in time. Also, the computation time for the policy is also polynomial since (4) in Algorithm 1 now becomes the following deterministic maximum weighted bipartite matching problem:

$$\arg\max_{\mathbf{a}\in\mathcal{F}} \sum_{(i,j)\in\mathcal{A}_{\mathbf{a}}} \left(\hat{\theta}_{i,j} + \sqrt{\frac{(L+1)\ln n}{m_{i,j}}} \right) \tag{29}$$

on the bipartite graph of users and channels with edge weights $(\hat{\theta}_{i,j} + \sqrt{(L+1)\ln n / m_{i,j}})$. It could be solved with polynomial computation time (e.g., using the Hungarian algorithm [29]). Note that $L = \max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}| = \min\{M, Q\}$ for this problem, which is less than $M \times Q$ so that the bound of regret is tighter. The regret is $O(\min\{M,Q\}^3 MQ \log n)$ following Theorem 2.

B. Shortest Path

Shortest path (SP) problem is another example where the underlying deterministic optimization can be done with polynomial computation time. If the given directed graph is denoted as $G = (V, E)$ with the source node s and the destination node d, and the cost (e.g., the transmission delay) associated with edge (i, j) is denoted as $D_{i,j} \geq 0$, the objective is to find the path from s to d with the minimum sum cost, i.e.,

$$\min \quad C^{\mathrm{SP}}_{\mathbf{a}} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{30}$$

$$\text{s.t.} \quad a_{i,j} \in \{0,1\}, \forall (i,j) \in E \tag{31}$$

$$\forall i, \sum_{j} a_{i,j} - \sum_{j} a_{j,i} = \begin{cases} 1, & i = s \\ -1, & i = t \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

where (31) and (32) define a feasible set $\mathcal{F}$, such that $\mathcal{F}$ is the set of all possible pathes from s to d. When (D_{ij}) are random

TABLE II
COMPARISON OF REGRET BOUNDS

	Naive Policy	LLR
Maximum Weighted Matching	$O(\|\mathcal{F}\|\log n)$ where $\|\mathcal{F}\| = P(Q,M)$ (when $Q \geq M$) or $\|\mathcal{F}\| = P(Q,M)$ (when $Q < M$)	$O(\min\{M,Q\}^3 MQ\log n)$
Shortest Path (Complete Graph)	$O(\|\mathcal{F}\|\log n)$ where $\|\mathcal{F}\| = \sum_{k=0}^{\|V\|-2} \frac{(\|V\|-2)!}{k!}$ and $\frac{\|V\|(\|V\|-1)}{2} = \|E\|$	$O(\|E\|^4 \log n)$
Minimum Spanning Tree (Complete Graph)	$O(\|\mathcal{F}\|\log n)$ where $\|\mathcal{F}\| = \|V\|^{\|V\|-2}$ and $\frac{\|V\|(\|V\|-1)}{2} = \|E\|$	$O(\|E\|^4 \log n)$

variables with bounded support but unknown mean, i.i.d. over time, an dynamic learning policy is needed for this multi-armed bandit formulation.

Note that corresponding to the LLR policy with the objective to maximize the rewards, a direct variation of it is to find the minimum linear cost defined on finite constraint set $\mathcal{F}$, by changing the maximization problem in to a minimization problem. For clarity, this straightforward modification of LLR is shown in Algorithm 2, which we refer to as Learning with Linear Costs (LLC).

Algorithm 2: Learning with Linear Cost (LLC)

1: // INITIALIZATION PART IS SAME AS IN ALGORITHM 1
2: // MAIN LOOP
3: **while** 1 **do**
4: $n = n + 1$;
5: Play an action $\mathbf{a}$ which solves the minimization problem

$$\mathbf{a} = \arg\min_{\mathbf{a}\in\mathcal{F}} \sum_{i\in\mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i - \sqrt{\frac{(L+1)\ln n}{m_i}}\right); \tag{33}$$

6: Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
7: **end while**

LLC (Algorithm 2) is a policy for a general multi-armed bandit problem with linear cost defined on any constraint set. It is directly derived from the LLR policy (Algorithm 1), so Theorem 2 also holds for LLC, where the regret is defined as

$$\mathfrak{R}_n^{\pi}(\Theta) = E^{\pi}\left[\sum_{t=1}^{n} C_{\pi(t)}(t)\right] - nC^* \tag{34}$$

where C^* represents the minimum cost, which is cost of the optimal action.

Using the LLC policy, we map each path between s and t as an action. The number of unknown variables is $|E|$, while the number of actions could grow exponentially in the worst case. Since there exist polynomial computation time algorithms such as Dijkstra's algorithm [31] and Bellman–Ford algorithm [32], [33] for the shortest path problem, we could apply these algorithms to solve (33) with edge cost $\hat{\theta}_i - \sqrt{(L+1)\ln n/m_i}$. LLC is thus an efficient policy to solve the multi-armed bandit formulation of the shortest path problem with linear storage, polynomial computation time. Note that $L = \max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}| = |E|$. Regret is $O(|E|^4 \log n)$.

Another related problem is the Shortest Path Tree (SPT), where problem formulation is similar, and the objective is to find a subgraph of the given graph with the minimum total cost between a selected root s node and all other nodes. It is expressed as [34], [35]

$$\min \quad C_{\mathbf{a}}^{\mathrm{SPT}} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{35}$$

$$\text{s.t.} \quad a_{i,j} \in \{0,1\}, \forall (i,j) \in E \tag{36}$$

$$\sum_{(j,i)\in\mathcal{BS}(i)} a_{j,i} - \sum_{(i,j)\in\mathcal{FS}(i)} a_{i,j} = \begin{cases} -n+1, & i = s \\ 1, & i \in V/\{s\} \end{cases} \tag{37}$$

where $\mathcal{BS}(i) = \{(u,v) \in E : v = i\}$, $\mathcal{FS}(i) = \{(u,v) \in E : u = i\}$. Equations (37) and (36) define the constraint set $\mathcal{F}$. We can also use the polynomial computation time algorithms such as Dijkstra's algorithm and Bellman–Ford algorithm to solve (33) for the LLC policy.

C. Minimum Spanning Tree

Minimum Spanning Tree (MST) is another combinatorial optimization with polynomial computation time algorithms, such as Prim's algorithm [36] and Kruskal's algorithm [37]. The objective for the MST problem can be simply presented as

$$\min_{\mathbf{a}\in\mathcal{F}} C_{\mathbf{a}}^{\mathrm{MST}} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{38}$$

where $\mathcal{F}$ is the set of all spanning trees in the graph.

With the LLC policy, each spanning tree is treated as an action, and $L = |E|$. Regret bound also grows as $O(|E|^4 \log n)$.

To summarize, we show in Table II a side-by-side comparison for the bipartite matching, shortest paths, and spanning tree problems. For the matching problem, the graph is already restricted to bipartite graphs. The problem of counting the number of paths on a graph is known to be #-P complete, so there is no known simple formula for a general setting. Similarly, we are not aware of any formulas for counting the number of spanning trees on a general graph. For this reason, for the latter two problems, we present comparative analytical bounds for the special case of the complete graph, where a closed-form expression for number of paths can be readily obtained, and Cayley's formula can be used for the number of spanning trees [38].

VII. NUMERICAL SIMULATION RESULTS

We present in this section the numerical simulation results with the example of multiuser channel allocations in cognitive radio network.

Fig. 2 shows the simulation results of using LLR policy compared to the naive policy in Section IV-A. We assume that the system consists of $Q = 7$ orthogonal channels in and $M =$

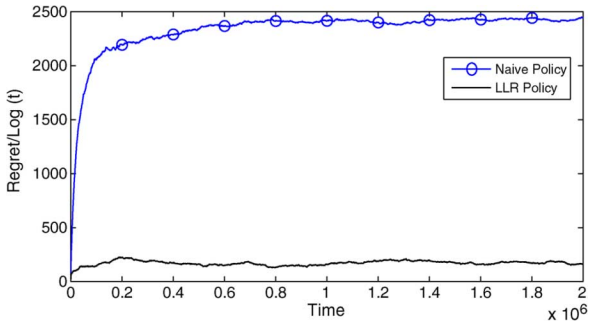

Fig. 2. Simulation results of a system with seven orthogonal channels and four users.

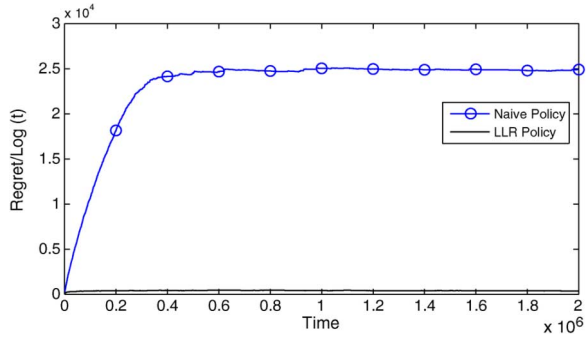

Fig. 3. Simulation results of a system with nine orthogonal channels and five users.

TABLE III
REGRET WHEN $t = 2 \times 10^6$

	Naive Policy	LLR
7 channels, 4 users	2443.6	163.6
9 channels, 5 users	24892.6	345.2

4 secondary users. The throughput $\{S_{i,j}(t)\}_{t\geq 1}$ for the user-channel combination is an i.i.d. Bernoulli process with mean $\theta_{i,j}$ ($(\theta_{i,j})$ is unknown to the players) shown as follows:

$$(\theta_{i,j}) = \begin{pmatrix} 0.3 & 0.5 & \boxed{0.9} & 0.7 & 0.8 & 0.9 & 0.6 \\ 0.2 & 0.2 & 0.3 & 0.4 & \boxed{0.5} & 0.4 & 0.5 \\ \boxed{0.8} & 0.6 & 0.5 & 0.4 & 0.7 & 0.2 & 0.8 \\ 0.9 & 0.2 & 0.2 & 0.8 & 0.3 & \boxed{0.9} & 0.6 \end{pmatrix} \tag{39}$$

where the components in the box are in the optimal action. Note that $P(7,4) = 840$ while $7 \times 4 = 28$, so the storage used for the naive approach is 30 times more than the LLR policy. Fig. 2 shows the regret (normalized with respect to the logarithm of time) over time for the naive policy and the LLR policy. We can see that under both policies the regret grows logarithmically in time. But the regret for the naive policy is a lot higher than that of the LLR policy.

Fig. 3 is another example of the case when $Q = 9$ and $M = 5$. The throughput is also assumed to be an i.i.d. Bernoulli process, with the following mean:

$$(\theta_{i,j}) = \begin{pmatrix} 0.3 & 0.5 & \boxed{0.9} & 0.7 & 0.8 & 0.9 & 0.6 & 0.8 & 0.7 \\ 0.2 & 0.2 & 0.3 & 0.4 & 0.5 & 0.4 & 0.5 & 0.6 & \boxed{0.9} \\ 0.8 & 0.6 & 0.5 & 0.4 & 0.7 & 0.2 & \boxed{0.8} & 0.2 & 0.8 \\ \boxed{0.9} & 0.2 & 0.2 & 0.8 & 0.3 & 0.9 & 0.6 & 0.5 & 0.4 \\ 0.6 & 0.7 & 0.5 & 0.7 & 0.6 & \boxed{0.8} & 0.2 & 0.6 & 0.8 \end{pmatrix}. \tag{40}$$

For this example, $P(9,5) = 15120$, which is much higher than $9 \times 5 = 45$ (about 336 times higher), so the storage used by the naive policy grows much faster than the LLR policy. Comparing with the regrets shown in Table III for both examples when $t = 2 \times 10^6$, we can see that the regret also grows much faster for the naive policy.

VIII. K SIMULTANEOUS ACTIONS

The reward-maximizing LLR policy presented in Algorithm 1 and the corresponding cost-minimizing LLC policy presented in Algorithm 2 can also be extended to the setting where K actions are played at each time-slot. The goal is to maximize the total rewards (or minimize the total costs) obtained by these K actions. For brevity, we only present the policy for the reward-maximization problem; the extension to cost-minimization is straightforward. The modified LLR-K policy for picking the K best actions is shown in Algorithm 3.

Algorithm 3: Learning With Linear Rewards While Selecting K Actions (LLR-K)

1: // INITIALIZATION PART IS SAME AS IN ALGORITHM 1
2: MAIN LOOP
3: **while** 1 **do**
4: $n = n + 1$;
5: Play actions $\{\mathbf{a}\}_K \in \mathcal{F}$ with K largest values in (41)

$$\sum_{i \in \mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i + \sqrt{\frac{(L+1)\ln n}{m_i}} \right); \tag{41}$$

6: Update $(\hat{\theta}_i)_{1\times N}, (m_i)_{1\times N}$ for all actions accordingly;
7: **end while**

Theorem 3 states the upper bound of the regret for the extended LLR-K policy.

Theorem 3: The expected regret under the LLR-K policy with K actions selection is at most

$$\left[\frac{4a_{\max}^2 L^2 (L+1) N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} L K^{2L} N \right] \Delta_{\max}. \tag{42}$$

Proof: The proof is similar to the proof of Theorem 2, but now we have a set of K actions with K largest expected rewards as the optimal actions. We denote this set as $\mathfrak{A}^* = \{\mathbf{a}^{*,k}, 1 \leq k \leq K\}$, where $\mathbf{a}^{*,k}$ is the action with kth largest expected reward. As in the proof of Theorem 2, we define $\widetilde{T}_i(n)$ as a counter when a nonoptimal action is played in the same way. Equations (50), (11), (12), and (14) still hold.

Note that each time when $\widetilde{I}_i(t) = 1$, there exists some action such that a nonoptimal action is picked for which m_i is the minimum in this action. We denote this action as $\mathbf{a}(t)$. Note that

$\mathbf{a}(t)$ means there exists m, $1 \leq m \leq K$, such that the following holds:

$$\widetilde{T}_i(n) \leq l + \sum_{t=N}^{n} \left\{ \sum_{j \in \mathcal{A}_{\mathbf{a}^{*,m}}} a_j^{*,m} \left(\hat{\overline{\theta}}_{j,m_j(t)} + C_{t,m_j(t)} \right) \right. \\ \left. \leq \sum_{j \in \mathcal{A}_{\mathbf{a}(t)}} a_j(t) \left(\hat{\overline{\theta}}_{j,m_j(t)} + C_{t,m_j(t)} \right), \widetilde{T}_i(t) \geq l \right\}. \tag{43}$$

Since at each time K actions are played, at time t, a random variable could be observed up to Kt times. Then, (15) should be modified as

$$\widetilde{T}_i(n) \leq l + \sum_{t=1}^{\infty} \sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|\mathcal{A}^{*,m}|}}=1}^{Kt} \sum_{m_{p_1}=l}^{Kt} \cdots \sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{Kt} \\ \left\{ \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^{*,m}}|} a_{h_j}^{*,m} \left(\hat{\overline{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}} \right) \right. \\ \left. \leq \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t) \left(\hat{\overline{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}} \right) \right\}. \tag{44}$$

Equations (16)–(23) are similar by substituting $\mathbf{a}^*$ with $\mathbf{a}^{*,m}$. Thus, we have

$$\begin{aligned} &\mathbb{E}\left[\widetilde{T}_i(n)\right] \\ &\leq \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{La_{\max}}\right)^2} \right\rceil \\ &\quad + \sum_{t=1}^{\infty} \left(\sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|\mathcal{A}^*|}}=1}^{Kt} \sum_{m_{p_1}=l}^{Kt} \cdots \sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{Kt} 2Lt^{-2(L+1)} \right) \\ &\leq \frac{4a_{\max}^2 L^2 (L+1)\ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3} LK^{2L}. \end{aligned} \tag{45}$$

Hence, we get the upper bound for the regret as

$$\mathfrak{R}_n^{\pi}(\Theta) \leq \left[\frac{4a_{\max}^2 L^2 (L+1) N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} LK^{2L} N \right] \Delta_{\max}. \tag{46}$$

■

IX. Conclusion

We have considered multi-armed bandit problems in which at each time an arbitrarily constrained set of random variables are selected, the selected variables are revealed, and a total reward that is a linear function of the selected variables is yielded. For such problems, existing single-play MAB policies such as the well-known UCB1 [7] can be utilized, but have poor performance in terms of storage, computation, and regret. The LLR and LLR-K policies we have presented are smarter in that they store and make decisions at each time based on the stochastic observations of the underlying unknown-mean random variables alone; they require only linear storage and result in a regret that is bounded by a polynomial function of the number of unknown-mean random variables. If the deterministic version of the corresponding combinatorial optimization problem can be solved in polynomial time, our policy will also require only polynomial computation per step. We have shown a number of problems in the context of networks where this formulation would be useful, including maximum weight matching, shortest path, and spanning tree computations. For the case where the deterministic version is NP-hard, one has often at hand a polynomial-time approximation algorithm. In the Appendix, we show that under a suitably relaxed definition of regret, the LLR algorithm can also employ such an approximation to give provable performance.

While this paper has provided useful insights into real-world linear combinatorial optimization with unknown-mean random coefficients, there are many interesting open problems to be explored in the future. One open question is to derive a lower bound on the regret achievable by any policy for this problem. We conjecture on intuitive grounds that it is not possible to have regret lower than $\Omega(N \log n)$, but this remains to be proved rigorously. It is unclear whether the lower bound can be any higher than this, and hence, it is unclear whether it is possible to prove an upper bound on regret for some policy that is better than the $O(N^4 \log n)$ upper bound shown in our work.

In the context of channel access in cognitive radio networks, other researchers have recently developed distributed policies in which different users each select an arm independently [3], [4]. A closely related problem in this setting would be to have distributed users selecting different elements of the action vector independently. The design and analysis of such distributed policies is an open problem.

Finally, it would be of great interest to see if it is possible to also tackle nonlinear reward functions, at least in structured cases that have proved to be tractable in deterministic settings, such as convex functions.

Appendix
LLR With Approximation Algorithm

One interesting question arises in the context of NP-hard combinatorial optimization problems, where even the deterministic version of the problem cannot be solved in polynomial time with known algorithms. In such cases, if only an approximation algorithm with some known approximation guarantee is available, what can be said about the regret bound?

For such settings, let us consider that a factor-β approximation algorithm (i.e., which for a maximization problem yields a solutions that have reward more than OPT/β) is used to solve the maximization step in (4) in our LLR Algorithm 1. Accordingly, we define an *β-approximate action* to be an action whose expected reward is within a factor β of that of the optimal action, and all other actions as *non-β-approximate*. Now, we define *β-approximation regret* as follows:

$$\mathfrak{R}_n^{\beta,\pi}(\Theta) = \mathbb{E}[\text{total number of times non-}\beta\text{approximate actions are played by strategy } \pi \text{ in } n \text{ time slots}] \tag{47}$$

$$= \mathbb{E}\left[\sum_{\mathbf{a}:\mathbf{a} \text{ is not a } \beta\text{-approximate action}} m_{\mathbf{a}}(n) \right] \tag{48}$$

where $m_{\mathbf{a}}(n)$ is the total number of time that $\mathbf{a}$ has been played up to time n. We define $\Delta^{\beta}_{\min}$ as the minimum distance between an β-approximate action and a non-β-approximate action. We assume $\Delta^{\beta}_{\min} > 0$.

We have the following theorem regarding LLR with a β-approximation algorithm.

Theorem 4: The β-approximation regret under the LLR policy with a β-approximation algorithm is at most

$$\frac{4a_{\max}^2 L^2(L+1)N\ln n}{\left(\Delta^{\beta}_{\min}\right)^2} + N + \frac{\pi^2}{3}LN. \tag{49}$$

Proof: We modify the proof of Theorem 2 to show Theorem 4. We replace "optimal action" with "β-approximate action," and "non-optimal action" with "non-β-approximate action" everywhere shown in the proof of Theorem 2, and we still define a virtual counter $(\widetilde{T}_i(n))_{1\times N}$ in a similar way. We still use $*$ to refer to an optimal action. Thus, (50) becomes

$$\sum_{\mathbf{a}:\mathbf{a}\text{ is a non-}\beta\text{-approximate action}} \mathbb{E}[T_{\mathbf{a}}(n)] = \sum_{i=1}^{N} \mathbb{E}\left[\widetilde{T}_i(n)\right]. \tag{50}$$

Now we note that for LLR with a β-approximation algorithm, when $\widetilde{I}_i(t) = 1$, a non-β-approximate action $\mathbf{a}(t)$ has been picked for which $m_i = \min_j\{m_j : \forall j \in \mathcal{A}_{\mathbf{a}(t)}\}$. Define $s_{\max}(t)$ is the optimal solution for (4) in Algorithm 1. Then, we have

$$\sum_{j\in\mathcal{A}_{\mathbf{a}(t)}} a_j(t)\left(\hat{\overline{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}\right) \geq \frac{1}{\beta}s_{\max}(t) \tag{51}$$

$$\geq \frac{1}{\beta}\sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*\left(\hat{\overline{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}\right). \tag{52}$$

Therefore

$$\begin{aligned}\widetilde{T}_i(n) &\leq l + \sum_{t=N+1}^{n} \mathbb{1}\left\{\frac{1}{\beta}\sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*\left(\hat{\overline{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}\right)\right.\\ &\left. \leq \sum_{j\in\mathcal{A}_{\mathbf{a}(t)}} a_j(t)\left(\hat{\overline{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}\right), \widetilde{T}(t-1) \geq l\right\}.\end{aligned} \tag{53}$$

With a similar analysis, as in (13) to (15), we have

$$\begin{aligned}\widetilde{T}_i(n) \leq l + \sum_{t=1}^{\infty}\sum_{m_{h_1}=1}^{t}\cdots\sum_{m_{h_{|\mathcal{A}^*|}}=1}^{t}\sum_{m_{p_1}=l}^{t}\cdots\sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t+1)}|}}=l}^{t}\\ \mathbb{1}\left\{\frac{1}{\beta}\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\left(\hat{\overline{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}}\right)\right.\\ \left. \leq \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\left(\hat{\overline{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}}\right)\right\}.\end{aligned} \tag{54}$$

Now we note that $1/\beta\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*(\hat{\overline{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \leq \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)(\hat{\overline{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}})$ implies that at least one of the following must be true:

$$\frac{1}{\beta}\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*\hat{\overline{\theta}}_{h_j,m_{h_j}} \leq \frac{1}{\beta}R^* - \frac{1}{\beta}\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* C_{t,m_{h_j}} \tag{55}$$

$$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)\hat{\overline{\theta}}_{p_j,m_{p_j}} \geq R_{\mathbf{a}(t+1)} + \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)C_{t,m_{p_j}} \tag{56}$$

$$\frac{1}{\beta}R^* < R_{\mathbf{a}(t+1)} + 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)C_{t,m_{p_j}}. \tag{57}$$

Equations (55) and (56) are equivalent to (16) and (17), and we note that for (57), when $l \geq \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta^{\beta}_{\min}}{La_{\max}}\right)^2}\right\rceil$

$$\begin{aligned}\frac{1}{\beta}R^* - R_{\mathbf{a}(t+1)} - 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t+1)}|} a_{p_j}(t+1)C_{t,m_{p_j}}\\ \geq \frac{1}{\beta}R^* - R_{\mathbf{a}(t+1)} - \Delta^{\beta}_{\min} = 0.\end{aligned} \tag{58}$$

Therefore, (45) still holds, and we have the upper bound for β-approximation regret as

$$\mathfrak{R}_n^{\beta,\mathrm{LLR}}(\Theta) \leq \frac{4a_{\max}^2 L^2(L+1)N\ln n}{\left(\Delta^{\beta}_{\min}\right)^2} + N + \frac{\pi^2}{3}LN.$$

■

REFERENCES

[1] S. Pandey, D. Chakrabarti, and D. Agarwal, "Multi-armed bandit problems with dependent arms," in *Proc. 24th Annu. Int. Conf. Mach. Learn.*, Jun. 2007, pp. 721–728.

[2] P. Rusmevichientong and D. P. Williamson, "An adaptive algorithm for selecting profitable keywords for search-based advertising services," in *Proc. 7th ACM Conf. Electron. Commerce*, Jun. 2006, pp. 260–269.

[3] A. Anandkumar, N. Michael, A. K. Tang, and A. Swami, "Distributed algorithms for learning and cognitive medium access with logarithmic regret," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 4, pp. 731–745, Apr. 2011.

[4] K. Liu and Q. Zhao, "Distributed learning in multi-armed bandit with multiple players," *IEEE Trans. Signal Process.*, vol. 58, no. 11, pp. 5667–5681, Nov. 2010.

[5] V. Anantharam, P. Varaiya, and J. Walrand, "Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays—Part I: I.I.D. rewards," *IEEE Trans. Autom. Control*, vol. AC-32, no. 11, pp. 968–976, Nov. 1987.

[6] R. Agrawal, "Sample mean based index policies with $O(\log n)$ regret for the multi-armed bandit problem," *Adv. Appl. Probab.*, vol. 27, pp. 1054–1078, 1995.

[7] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2–3, pp. 235–256, 2002.

[8] R. K. Ahuja, T. L. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ: Prentice-Hall, 1993.
[9] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 4th ed. New York: Springer, 2008.
[10] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Adv. Appl. Math.*, vol. 6, no. 1, pp. 4–22, 1985.
[11] R. Agrawal, M. V. Hegde, and D. Teneketzis, "Multi-armed bandits with multiple plays and switching cost," *Stochastics Stochastic Rep.*, vol. 29, pp. 437–459, 1990.
[12] Y. Gai and B. Krishnamachari, "Decentralized online learning algorithms for opportunistic spectrum access," in *Proc. IEEE GLOBECOM*, Dec. 2011, pp. 1–6.
[13] V. Anantharam, P. Varaiya, and J. Walrand, "Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays—Part II: Markovian rewards," *IEEE Trans. Autom. Control*, vol. AC-32, no. 11, pp. 977–982, Nov. 1987.
[14] C. Tekin and M. Liu, "Online learning in opportunistic spectrum access: A restless bandit approach," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2462–2470.
[15] H. Liu, K. Liu, and Q. Zhao, "Logarithmic weak regret of non-Bayesian restless multi-armed bandit," in *Proc. IEEE ICASSP*, May 2011, pp. 1968–1971.
[16] W. Dai, Y. Gai, B. Krishnamachari, and Q. Zhao, "The non-Bayesian restless multi-armed bandit: A case of near-logarithmic regret," in *Proc. IEEE ICASSP*, May 2011, pp. 2940–2943.
[17] N. Nayyar, Y. Gai, and B. Krishnamachari, "On a restless multi-armed bandit problem with non-identical arms," in *Proc. Allerton Conf. Commun., Control, Comput.*, Sep. 2011, pp. 369–376.
[18] Y. Gai, B. Krishnamachari, and M. Liu, "On the combinatorial multi-armed bandit problem with Markovian rewards," in *Proc. IEEE GLOBECOM*, Dec. 2011, pp. 1–6.
[19] R. Ortner, "Exploiting similarity information in reinforcement learning," in *Proc. 2nd ICAART*, Jan. 2010, pp. 203–210.
[20] A. J. Mersereau, P. Rusmevichientong, and J. N. Tsitsiklis, "A structured multiarmed bandit problem and the greedy policy," in *Proc. IEEE Conf. Decision Control*, Dec. 2008, pp. 4945–4950.
[21] P. Rusmevichientong and J. N. Tsitsiklis, "Linearly parameterized bandits," *Math. Oper. Res.*, vol. 35, no. 2, pp. 395–411, 2010.
[22] P. Auer, "Using confidence bounds for exploitation-exploration tradeoffs," *J. Mach. Learn. Res.*, vol. 3, pp. 397–422, 2002.
[23] V. Dani, T. P. Hayes, and S. M. Kakade, "Stochastic linear optimization under bandit feedback," in *Proc. 21st Annu. COLT*, Jul. 2008, pp. 355–366.
[24] B. Awerbuch and R. D. Kleinberg, "Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches," in *Proc. 36th Annu. ACM STOC*, Jun. 2004, pp. 45–53.
[25] Y. Gai, B. Krishnamachari, and R. Jain, "Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation," in *Proc. IEEE DySPAN*, Apr. 2010, pp. 1–9.
[26] D. Pollard, *Convergence of Stochastic Processes*. Berlin, Germany: Springer, 1984.
[27] H. Balakrishnan, C. L. Barrett, V. S. A. Kumar, M. V. Marathe, and S. Thite, "The distance-2 matching problem and its relationship to the MAC-layer capacity of ad hoc wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 6, pp. 1069–1079, Aug. 2004.
[28] A. Brzezinski, G. Zussman, and E. Modiano, "Enabling distributed throughput maximization in wireless mesh networks C A partitioning approach," in *Proc. ACM MobiCom*, Sep. 2006, pp. 26–37.
[29] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, no. 1–2, pp. 83–97, 1955.
[30] J. Edmonds, "Paths, trees, and flowers," *Canadian J. Math.*, vol. 17, pp. 449–467, 1965.
[31] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Num. Math.*, vol. 1, pp. 269–271, 1959.
[32] R. Bellman, "On a routing problem," *Quart. Appl. Math.*, vol. 16, pp. 87–90, 1958.
[33] L. R. Ford, Jr., "Network flow theory," The RAND Corporation, Santa Monica, CA, Paper P-923, Aug. 1956.
[34] J. Krarup and M. N. RØrbech, "LP formulations of the shortest path tree problem," *4OR*, vol. 2, no. 4, pp. 259–274, 2004.
[35] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*, 4th ed. Hoboken, NJ: Wiley, Dec. 2009.
[36] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Syst. Tech. J.*, vol. 36, pp. 1389–1401, 1957.
[37] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, Feb. 1956.
[38] A. Cayley, "A theorem on trees," *Quart. J. Math.*, vol. 23, pp. 376–378, 1889.

Yi Gai (S'10) received the B.E. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2005 and 2007, respectively, and is currently pursuing the Ph.D. degree in electrical engineering at the University of Southern California (USC), Los Angeles.

Her primary research interests include the design and analysis of online learning algorithms for network optimization with unknown variables, network game theory, cognitive radio networking, and wireless sensor networks.

Miss Gai is a recipient of an Annenberg Graduate Fellowship, WiSE Merit Fellowship, and Oakley Fellowship at USC. She received the USC Center for Applied Mathematical Sciences (CAMS) Prize in 2011.

Bhaskar Krishnamachari (M'97) received the B.E. degree from The Cooper Union, New York, NY, in 1998, and the M.S. and Ph.D. degrees from Cornell University, Ithaca, NY, in 1999 and 2002, respectively, all in electrical engineering.

He is currently an Associate Professor and a Ming Hsieh Faculty Fellow with the Department of Electrical Engineering, Viterbi School of Engineering, University of Southern California, Los Angeles. His primary research interest is in the design and analysis of algorithms and protocols for next-generation wireless networks.

Rahul Jain (M'06) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 1997, the M.S. degree in electrical and computer engineering from Rice University, Houston, TX, in 1999, and the M.A. degree in statistics and Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 2002 and 2004, respectively.

He is an Assistant Professor of electrical engineering and industrial and systems engineering with the University of Southern California, Los Angeles. His research interests lie broadly in network science and stochastic systems with current focus on game theory and economics for networks, and stochastic control and learning.

Dr. Jain is the lead Guest Editor of an upcoming special issue of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. He served as the TPC Co-Chair of GameNets 2011 and as a TPC member of several international conferences including IEEE INFOCOM. He received the James H. Zumberg Faculty Research and Innovation Award in 2009 and an IBM Faculty Award and the NSF CAREER Award in 2010. He also won a Best Paper Award at the ValueTools Conference 2009.