Combinatorial Network Optimization with Unknown Variables: Multi-Armed Bandits with Linear Rewards

Yi Gai, Bhaskar Krishnamachari and Rahul Jain
Ming Hsieh Department of Electrical Engineering
University of Southern California
Los Angeles, CA 90089, USA
Email: {ygai, bkrishna, rahul.jain}@usc.edu

***Abstract*—In the classic multi-armed bandits problem, the goal is to have a policy for dynamically operating arms that each yield stochastic rewards with unknown means. The key metric of interest is regret, defined as the gap between the expected total reward accumulated by an omniscient player that knows the reward means for each arm, and the expected total reward accumulated by the given policy. The policies presented in prior work have storage, computation and regret all growing linearly with the number of arms, which is not scalable when the number of arms is large. We consider in this work a broad class of multi-armed bandits with dependent arms that yield rewards as a linear combination of a set of unknown parameters. For this general framework, we present efficient policies that are shown to achieve regret that grows logarithmically with time, and polynomially in the number of unknown parameters (even though the number of dependent arms may grow exponentially). Furthermore, these policies only require storage that grows linearly in the number of unknown parameters. We show that this generalization is broadly applicable and useful for many interesting tasks in networks that can be formulated as tractable combinatorial optimization problems with linear objective functions, such as maximum weight matching, shortest path, and minimum spanning tree computations.**

I. Introduction

The problem of multi-armed bandits (MAB) 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 work 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 work, we formulate and consider the following general multi-armed bandit problem. There is a vector $\mathbf{X}$ of N random variables with unknown mean that are each instantiated in an i.i.d. fashion over time. There is a finite (possibly exponentially large) set of vector actions $\mathbf{a} \in \mathcal{F}$ from which any action can be selected at each time. When action $\mathbf{a}$ is performed, all elements of $\mathbf{X}$ that correspond to non-zero elements of $\mathbf{a}$ are observed, and a linear reward $\mathbf{a}^T\mathbf{X}$ is obtained. This generalization captures a very broad class of combinatorial optimization problems with linear objectives and unknown random coefficients.

A naive application of existing approaches for multi-armed bandits, such as the well-known UCB1 index policy of Auer *et al.* [5], for this problem would yield poor performance scaling in terms of regret, storage, and computation. This is because these approaches are focused on maintaining and computing quantities based on arm-specific observations and do not exploit potential dependencies between them. In this work, we instead propose smarter policies that explicitly take into account the linear form of the dependencies and base all storage and computations on the unknown variables directly, rather than the arms. As we shall show, this saves not only on storage and computation, but also substantially reduces the regret.

Specifically, we first present a novel single-arm selection policy for Learning with Linear Rewards (LLR) 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

[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.

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 which can be solved in polynomial time. For such problems, the policy we propose would thus inherit the property of polynomial computation at each step.

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

Through several concrete examples, we show the applicability of our general formulation of multi-armed bandits with linear rewards to combinatorial network optimization. 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. The examples we present are far from exhausting the possible applications of the formulation and the policies we present in this work — there are many other linear-objective network optimization problems [6], [7]. Our framework, for the first time, allows these problems to be solved in stochastic settings with unknown random coefficients, with provably efficient performance.

We expect that our work will also find practical application in other fields where such 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 survery 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 MAB. 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 values in section VIII. Finally, we conclude with a summary of our contribution and point out avenues for future work in section IX.

II. Related Work

Lai and Robbins [8] wrote one of the earliest papers on the classic non-Bayesian infinite horizon multi-armed bandit problem. Assuming K 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 $O(K \log n)$, 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 $\Omega(K \log n)$. Anantharam *et al.* [9] extend this work to the case when M simultaneous plays are allowed. The work by Agrawal [10] presents easier to compute policies based on the sample mean that also has asymptotically logarithmic regret. However, their policies need not be directly applied to our problem formulation in this paper, which involves combinatorial arms that cannot be characterized by a single parameter.

Our work is influenced by the paper of Auer *et al.* [5] that considers arms with non-negative rewards that are i.i.d. over time with an arbitrary un-parameterized distribution that has the only restriction that it have a finite support. Further they provide a simple policy (referred to as UCB1), which achieves logarithmic regret uniformly over time, rather than only asymptotically. However, their work does not exploit potential dependencies between the arms. As we show in this paper, a direct application of their UCB1 policy therefore performs poorly for our problem formulation.

There are also some recent works to propose decentralized policies for the multi-armed bandit problem. Liu and Zhao [4], and Anandkumar *et al.* [3] have both developed policies for the problem of M distributed players operating N independent arms.

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 provide only binary rewards, and in any case, they do not present any theoretical analysis on the expected regret. Ortner [11] 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.* [12] consider a bandit problem where the expected reward is defined as a linear function of an 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 [13] extend [12] 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. Also, [12] and [13] assume that only

the reward is observed at each time. In our work, we assume that the random variables corresponding to non-zero action components are observed at each time (from which the reward can be inferred).

Both [14] and [15] consider linear reward models that are more general than ours, but also under the assumption that only the reward is observed at each time. Auer [14] presents a randomized policy which requires storage and computation to grow linearly in the number of arms. This algorithm is shown to achieve a regret upper bound of $O(\sqrt{N}\sqrt{n}\log^{\frac{3}{2}}(n|\mathcal{F}|))$. Dani *et al.* [15] 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 $O(\frac{N^2}{\Delta}\log^3 n)$ for sufficiently large n with high probability. To our best knowledge, ours is the first paper to consider linear rewards with observation of the random variables corresponding to non-zero action components. We present a deterministic policy with a deterministic combinatorial linear optimization problem finite time bound of regret which grows $O(N^4 \log n)$, 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 which introduced combinatorial multi-armed bandits [16]. The formulation in [16] 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 reward, where the action vector is from a finite set.

III. Problem Formulation

Now we define the problem of multi-armed bandits with linear rewards that we solve in this paper. 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 $\theta_i = E[X_i]$ 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 $\mathbf{a}(n)$, representing an arm, 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 arm $\mathbf{a}$. We treat each $\mathbf{a}(n) \in \mathcal{F}$ as an arm. 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/arm $a(n)$ is selected, the random variables corresponding to non-zero components of a(n) are revealed[2], 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 arm 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:

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

where $\theta^* = \max_{\mathbf{a}\in\mathcal{F}} \sum_{i=1}^{N} a_i\theta_i$, the expected reward of an optimal arm. For the rest of the paper, we use $*$ as the index indicating that a parameter is for an optimal arm. If there is more than one optimal arm 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 sub-linear 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 arms $|\mathcal{F}|$ can be exponential in the number of unknown random variables N.

IV. Policy Design

A. A Naive Approach

A straightforward, relatively naive approach to solving the multi-armed bandits with linear regret problem that we defined is to use the UCB1 policy given by Auer *et al.* [5]. For UCB1, the arm that maximizes $\hat{Y}_k + \sqrt{\frac{2\ln n}{m_k}}$ will be selected at each time slot, where $\hat{Y}_k$ is the mean observed reward on arm k, and m_k is the number of times that arm k has been played. This approach essentially ignores the dependencies across the different arms, storing observed information about each arm independently, and making decisions based on this information alone.

Auer *et al.* [5] showed the following policy performance for regret upper bound as:

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

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

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

Proof: See [5, Theorem 1]. ■

Note that UCB1 requires storage that is linear in the number of arms and yields regret growing linearly with the number of

[2]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.

arms. In a case where the number of arms 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 which more efficiently stores observations from correlated arms and exploits the correlations to make better decisions.

B. A 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)

// INITIALIZATION
If $\max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$ is known, let $L = \max_{\mathbf{a}} |\mathcal{A}_{\mathbf{a}}|$; else, $L = N$;
for $p = 1$ to N **do**
  $n = p$;
  Play any arm $\mathbf{a}$ such that $p \in \mathcal{A}_{\mathbf{a}}$;
  Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
end for
// MAIN LOOP
while 1 **do**
  $n = n + 1$;
  Play an arm $\mathbf{a}$ which solves the maximization problem

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

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

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 arm as a whole. Since the same random variable can be observed while operating different arms, this allows exploitation of information gained from the operation of one arm to make decisions about a dependent arm.

We use two 1 by N vectors to store the information after we play an arm 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 arms 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 arm $\mathbf{a}(n)$ 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:

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

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

N :	number of random variables.
$\mathbf{a}$:	vectors of coefficients, defined on set $\mathcal{F}$; we map each $\mathbf{a}$ as an arm.
$\mathcal{A}_{\mathbf{a}}$:	$\{i : a_i \neq 0, 1 \leq i \leq N\}$.
$*$:	index indicating that a parameter is for an optimal arm.
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.
$\Delta_{\mathbf{a}}$:	$R^* - R_{\mathbf{a}}$.
$\Delta_{\min}$:	$\min_{\mathbf{a}\neq\mathbf{a}^*} \Delta_{\mathbf{a}}$.
$\Delta_{\max}$:	$\max_{\mathbf{a}\neq\mathbf{a}^*} \Delta_{\mathbf{a}}$.
$T_{\mathbf{a}}(n)$:	number of times arm $\mathbf{a}$ has been played in the first n time slots.
$a_{\max}$:	$\max_{\mathbf{a}\in\mathcal{F}} \max_i a_i$.

TABLE I
NOTATION

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, that there exists 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 non-optimal arm is played, and the summing this expectation over all non-optimal arms. While such an approach will work to analyze the LLR policy too, it turns out that the upper-bound for regret consequently obtained is quite loose, being linear in the number of arms, which may grow faster than polynomials. Instead, we give here a tighter analysis of the LLR policy that provides an upper bound which is instead polynomial in N and logarithmic in time. Like the regret analysis in [5], 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 proof Theorem 2, we use the inequalities as stated in the Chernoff-Hoeffding bound [17].

***Lemma** 1 (Chernoff-Hoeffding bound [17]):* $X_1, \ldots, X_n$ are random variables with range $[0,1]$, and $E[X_t|X_1, \ldots, X_{t-1}] = \mu$, $\forall 1 \le t \le n$. Denote $S_n = \sum X_i$. Then for all $a \ge 0$

$$\begin{aligned} Pr\{S_n \ge n\mu + a\} &\le e^{-2a^2/n} \\ Pr\{S_n \le n\mu - a\} &\le e^{-2a^2/n} \end{aligned} \tag{8}$$

Proof of Theorem 2: Denote C_{t,m_i} as $\sqrt{\frac{(L+1)\ln t}{m_i}}$. 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 arm is played; (2) a non-optimal arm is played. In the first case, $(\widetilde{T}_i(n))_{1\times N}$ won't be updated. When an non-optimal arm $\mathbf{a}(n)$ 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 arm, $\widetilde{T}_i(n)$ is increased by 1. If there are multiple such arms, we arbitrarily pick one, say i', and increment $\widetilde{T}_{i'}$ by 1.

Each time when a non-optimal arm 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 non-optimal arms is equal to the summation of all counters in $(\widetilde{T}_i(n))_{1\times N}$. Therefore, we have:

$$\sum_{\mathbf{a}:\mathbf{a}\neq\mathbf{a}^*} \mathbb{E}[T_{\mathbf{a}}(n)] = \sum_{i=1}^{N} \mathbb{E}[\widetilde{T}_i(n)]. \tag{9}$$

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

$$\widetilde{T}_i(n) \le m_i(n), \forall 1 \le i \le N. \tag{10}$$

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

$$\begin{aligned} \widetilde{T}_i(n) &= \sum_{t=N+1}^{n} \mathbb{1}\{\widetilde{I}_i(t) = 1\} \\ &\le l + \sum_{t=N+1}^{n} \mathbb{1}\{\widetilde{I}_i(t) = 1, \widetilde{T}_i(t-1) \ge l\} \end{aligned} \tag{11}$$

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 non-optimal arm $\mathbf{a}(t)$ has been picked for which $m_i = \min_j\{m_j : \forall j \in \mathcal{A}_{\mathbf{a}(t)}\}$. We denote this arm as $\mathbf{a}(t)$ since at each time that $\widetilde{I}_i(t) = 1$, we could get different arms. Then,

$$\begin{aligned} \widetilde{T}_i(n) &\le l + \sum_{t=N+1}^{n} \mathbb{1}\{ \sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*(\hat{\bar{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}) \\ &\le \sum_{j\in\mathcal{A}_{\mathbf{a}(t)}} a_j(t)(\hat{\bar{\theta}}_{j,m_j(t-1)} + C_{t-1,m_j(t-1)}), \widetilde{T}_i(t-1) \ge l\} \\ &\le l + \sum_{t=N}^{n} \mathbb{1}\{ \sum_{j\in\mathcal{A}_{\mathbf{a}^*}} a_j^*(\hat{\bar{\theta}}_{j,m_j(t)} + C_{t,m_j(t)}) \\ &\le \sum_{j\in\mathcal{A}_{\mathbf{a}(t)}} a_j(t)(\hat{\bar{\theta}}_{j,m_j(t)} + C_{t,m_j(t)}), \widetilde{T}_i(t) \ge l\}. \end{aligned} \tag{12}$$

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

$$l \le \widetilde{T}_i(t) \le m_j(t), \forall j \in \mathcal{A}_{\mathbf{a}(t)}. \tag{13}$$

$$\begin{aligned} \widetilde{T}_i(n) &\le l + \sum_{t=N}^{n} \mathbb{1}\{ \min_{0<m_{h_1},\ldots,m_{h_{|\mathcal{A}_{\mathbf{a}^*}|}}\le t} \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*(\hat{\bar{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \\ &\le \max_{l\le m_{p_1},\ldots,m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}\le t} \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)(\hat{\bar{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}})\} \\ &\le l + \sum_{t=1}^{\infty} \sum_{m_{h_1}=1}^{t} \cdots \sum_{m_{h_{|\mathcal{A}^*|}}=1}^{t} \sum_{m_{p_1}=l}^{t} \cdots \sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{t} \\ &\quad \mathbb{1}\{ \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^*(\hat{\bar{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \\ &\quad \le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)(\hat{\bar{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}})\} \end{aligned} \tag{14}$$

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

$$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* \hat{\bar{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*}|} a_{h_j}^* C_{t,m_{h_j}}, \tag{15}$$

$$\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)\hat{\bar{\theta}}_{p_j,m_{p_j}} \ge R_{\mathbf{a}(t)} + \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t) C_{t,m_{p_j}}, \tag{16}$$

$$R^* < R_{\mathbf{a}(t)} + 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t) C_{t,m_{p_j}}. \tag{17}$$

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

We have:

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

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

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

Thus,

$$
\begin{aligned}
&Pr\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a^*_{h_j}\hat{\bar{\theta}}_{h_j,m_{h_j}} \le R^* - \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}*}|} a^*_{h_j}C_{t,m_{h_j}}\}\\
&\le |\mathcal{A}_{\mathbf{a}*}|t^{-2(L+1)}\\
&\le \quad Lt^{-2(L+1)}.
\end{aligned}
\tag{18}
$$

Similarly, we can get the upper bound of the probability for inequality (16):

$$
\begin{aligned}
&Pr\{\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)\hat{\bar{\theta}}_{p_j,m_{p_j}} \ge R_{\mathbf{a}(t)} + \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)C_{t,m_{p_j}}\}\\
&\le Lt^{-2(L+1)}.
\end{aligned}
\tag{19}
$$

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

$$
\begin{aligned}
&R^* - R_{\mathbf{a}(t)} - 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t)C_{t,m_{p_j}}\\
&= R^* - R_{\mathbf{a}(t)} - 2\sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}\sqrt{\frac{(L+1)\ln t}{m_{p_j}}}\\
&\ge R^* - R_{\mathbf{a}(t)} - La_{\max}\sqrt{\frac{4(L+1)\ln n}{l}}\\
&\ge R^* - R_{\mathbf{a}(t)} - La_{\max}\sqrt{\frac{4(L+1)\ln n}{4(L+1)\ln n}\left(\frac{\Delta_{\mathbf{a}(t)}}{La_{\max}}\right)^2}\\
&\ge R^* - R_{\mathbf{a}(t)} - \Delta_{\mathbf{a}(t)} = 0.
\end{aligned}
\tag{20}
$$

Equation (20) implies that condition (15) is false when $l = \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\mathbf{a}(t)}}{La_{\max}}\right)^2} \right\rceil$. If we let $l = \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{La_{\max}}\right)^2} \right\rceil$, then (15) is false for all $\mathbf{a}(t)$.

Therefore,

$$
\begin{aligned}
\mathbb{E}[\widetilde{T}_i(n)] &\le \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{La_{\max}}\right)^2} \right\rceil\\
&+ \sum_{t=1}^{\infty}\left(\sum_{m_{h_1}=1}^{t}\cdots\sum_{m_{h_{|\mathcal{A}^*|}}=1}^{t}\sum_{m_{p_1}=l}^{t}\cdots\sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{t} 2Lt^{-2(L+1)}\right)\\
&\le \frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + 1 + L\sum_{t=1}^{\infty}2t^{-2}\\
&\le \frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3}L.
\end{aligned}
\tag{21}
$$

So under LLR policy, we have:

$$
\begin{aligned}
\mathfrak{R}^{\pi}_n(\Theta) &= R^*n - \mathbb{E}^{\pi}[\sum_{t=1}^{n} R_{\pi(t)}(t)]\\
&= \sum_{\mathbf{a}:R_{\mathbf{a}}<R^*}\Delta_{\mathbf{a}}\mathbb{E}[T_{\mathbf{a}}(n)]\\
&\le \Delta_{\max}\sum_{\mathbf{a}:R_{\mathbf{a}}<R^*}\mathbb{E}[T_{\mathbf{a}}(n)]\\
&= \Delta_{\max}\sum_{i=1}^{N}\mathbb{E}[\widetilde{T}_i(n)]\\
&\le \left[\sum_{i=1}^{N}\frac{4a^2_{\max}L^2(L+1)\ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LN\right]\Delta_{\max}\\
&\le \left[\frac{4a^2_{\max}L^2(L+1)N\ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3}LN\right]\Delta_{\max}.
\end{aligned}
\tag{22}
$$

■

***Remark** 1:* Note that when the set of action vectors consists of binary vectors with a single "1", the problem formulation reduces to an multi-armed bandit problem with N independent arms. In this special case, the LLR algorithm is equivalent to UCB1 in [5]. 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:* Theorem 2 also holds for random variables $X_i, 1 \leq i \leq N$ that are not i.i.d. over time, but with the only weaker assumption that $E[X_i(t)|X_i(1), \ldots, X_i(t-1)] = \theta_i, \forall 1 \leq i \leq N$. This is because the Chernoff-Hoeffding bound only needs a weak assumption $E[X_i(t)|X_i(1), \ldots, X_i(t-1)] = \theta_i, \forall 1 \leq i \leq N$.

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 [18], [19]. 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

$$\begin{aligned} \max \quad & R_{\mathbf{a}}^{MWM} = \sum_{i=1}^{|E|} a_i W_i \\ s.t. \quad & \mathbf{a} \text{ is a matching} \end{aligned} \tag{23}$$

where W_i is the weight associated with each edge i.

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 arm. Our proposed LLR policy is a solution with linear storage, and the regret polynomial in the number of edges, and logarithmic in time.

Since 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 [20], Edmonds's matching algorithm [21] 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.

Figure 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.

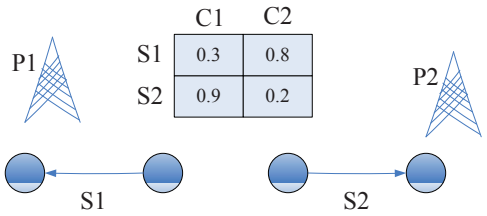

Fig. 1. An illustrative scenario

Note that this problem can be formulated as a multi-armed bandits with linear regret, in which each arm corresponds to a matching of the users to channels, and the reward corresponds to the sum-throughput. In this channel allocation problem, there is $M \times Q$ unknown random variables, and the number of arms are $P(Q, M)$, 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 a M by Q matrix. So the reward as each time slot by choosing a permutation $\mathbf{a}$ is expressed as:

$$R_{\mathbf{a}} = \sum_{i=1}^{M} \sum_{j=1}^{Q} a_{i,j} S_{i,j} \tag{24}$$

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

as:

$$\mathcal{F} = \{\mathbf{a} : a_{i,j} \in \{0,1\}, \forall i,j \wedge \sum_{i=1}^{Q} a_{i,j} = 1 \wedge \sum_{j=1}^{Q} a_{i,j} = 1\}. \tag{25}$$

We use two M by Q matrices to store the information after we play an arm 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 arms 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

$$\arg\max_{\mathbf{a}\in\mathcal{F}} \sum_{(i,j)\in\mathcal{A}_{\mathbf{a}}} \left(\hat{\theta}_{i,j} + \sqrt{\frac{(L+1)\ln n}{m_{i,j}}}\right) \tag{26}$$

on the bipartite graph of users and channels with edge weights $\left(\hat{\theta}_{i,j} + \sqrt{\frac{(L+1)\ln n}{m_{i,j}}}\right)$. It could be solved with polynomial computation time (e.g., using the Hungarian algorithm [20]). 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 find the path from s to d with the minimum sum cost, i.e.,

$$\min \quad C_{\mathbf{a}}^{SP} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{27}$$

$$s.t. \quad a_{i,j} \in \{0,1\}, \forall (i,j) \in E \tag{28}$$

$$\forall i, \sum_j a_{i,j} - \sum_j a_{j,i} = \begin{cases} 1 & : \quad i = s \\ -1 & : \quad i = t \\ 0 & : \quad \text{otherwise} \end{cases} \tag{29}$$

where equation (28) and (29) defines 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 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 below in Algorithm 2, which we refer to as Learning with Linear Costs (LLC).

Algorithm 2 Learning with Linear Cost (LLC)

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

$$\mathbf{a} = \arg\min_{\mathbf{a}\in\mathcal{F}} \sum_{i\in\mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i - \sqrt{\frac{(L+1)\ln n}{m_i}}\right); \tag{30}$$

  Update $(\hat{\theta}_i)_{1\times N}$, $(m_i)_{1\times N}$ accordingly;
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:

$$\mathfrak{R}_n^{\pi}(\Theta) = E^{\pi}[\sum_{t=1}^{n} C_{\pi(t)}(t)] - nC^* \tag{31}$$

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

Using the LLC policy, we map each path between s and t as an arm. The number of unknown variables are $|E|$, while the number of arms could grow exponentially in the worst case. Since there exist polynomial computation time algorithms such as Dijkstra's algorithm [22] and Bellman-Ford algorithm [23], [24] for the shortest path problem, we could apply these algorithms to solve (30) with edge cost $\hat{\theta}_i - \sqrt{\frac{(L+1)\ln n}{m_i}}$. 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 [25], [26]:

$$\min \quad C_{\mathbf{a}}^{SPT} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{32}$$

$$s.t. \quad a_{i,j} \in \{0,1\}, \forall (i,j) \in E \tag{33}$$

$$\sum_{(j,i)\in\mathcal{BS}(i)} a_{j,i} - \sum_{(i,j)\in\mathcal{FS}(i)} a_{i,j} = \begin{cases} -n+1 & : \quad i = s \\ 1 & : \quad i \in V/\{s\} \end{cases} \tag{34}$$

where $\mathcal{BS}(i) = \{(u,v) \in E : v = i\}$, $\mathcal{FS}(i) = \{(u,v) \in E : u = i\}$. (34) and (33) defines 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 (30) 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 [27] and Kruskal's algorithm [28]. The objective for the MST problem can be simply presented as

$$\min_{\mathbf{a}\in\mathcal{F}} C_{\mathbf{a}}^{MST} = \sum_{(i,j)\in E} a_{i,j} D_{i,j} \tag{35}$$

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

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

VII. Numerical Simulation Results

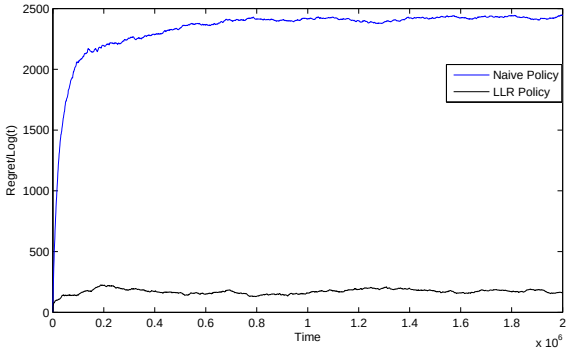

Fig. 2. Simulation Results of a system with 7 orthogonal channels and 4 users.

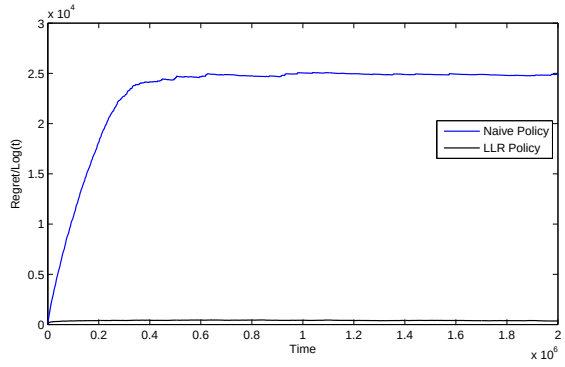

Fig. 3. Simulation Results of a system with 9 orthogonal channels and 5 users.

We present in the 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 with the naive policy in IV-A. We assume that the system consists of $Q = 7$ orthogonal channels in and $M = 4$ 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 below:

$$(\theta_{i,j}) = \begin{pmatrix} 0.3 & 0.5 & \boxed{0.9} & 0.7 & 0.8 & 0.9 & 0.6 \\ 0.2 & 0.2 & 0.3 & 0.4 & \boxed{0.5} & 0.4 & 0.5 \\ \boxed{0.8} & 0.6 & 0.5 & 0.4 & 0.7 & 0.2 & 0.8 \\ 0.9 & 0.2 & 0.2 & 0.8 & 0.3 & \boxed{0.9} & 0.6 \end{pmatrix} \tag{36}$$

where the components in the box are in the optimal arm. 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:

$$(\theta_{i,j}) = \begin{pmatrix} 0.3 & 0.5 & \boxed{0.9} & 0.7 & 0.8 & 0.9 & 0.6 & 0.8 & 0.7 \\ 0.2 & 0.2 & 0.3 & 0.4 & 0.5 & 0.4 & 0.5 & 0.6 & \boxed{0.9} \\ 0.8 & 0.6 & 0.5 & 0.4 & 0.7 & 0.2 & \boxed{0.8} & 0.2 & 0.8 \\ \boxed{0.9} & 0.2 & 0.2 & 0.8 & 0.3 & 0.9 & 0.6 & 0.5 & 0.4 \\ 0.6 & 0.7 & 0.5 & 0.7 & 0.6 & \boxed{0.8} & 0.2 & 0.6 & 0.8 \end{pmatrix}. \tag{37}$$

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 II for both examples when $t = 2 \times 10^6$, we can see that the regret also grows much faster for the naive policy.

TABLE II
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

VIII. K Simultaneous Actions

The reward-maximizing LLR policy presented in Algorithm 1 and the corresponding cost-minimizing LLC policy presented in 2 can also be extended to the setting where K arms are played at each time slot. The goal is to maximize the total rewards (or minimize the total costs) obtained by these K arms. 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 arms are shown in Algorithm 3.

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

Algorithm 3 Learning with Linear Rewards while selecting K arms (LLR-K)

1: // INITIALIZATION PART IS SAME AS IN ALGORITHM 1
2: // MAIN LOOP
3: **while** 1 **do**
4: $n = n + 1$;
5: Play arms $\{\mathbf{a}\}_K \in \mathcal{F}$ with K largest values in (38)

$$\sum_{i \in \mathcal{A}_{\mathbf{a}}} a_i \left(\hat{\theta}_i + \sqrt{\frac{(L+1)\ln n}{m_i}} \right); \tag{38}$$

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

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

$$\left[\frac{4a_{\max}^2 L^2 (L+1) N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} L K^{2L} N \right] \Delta_{\max}. \tag{39}$$

Proof:

The proof is similar to the proof of Theorem 2, but now we have a set of K arms with K largest expected rewards as the optimal arms. We denote this set as $\mathfrak{A}^* = \{\mathbf{a}^{*,k}, 1 \le k \le K\}$ where $\mathbf{a}^{*,k}$ is the arm with k-th largest expected reward. As in the proof of Theorem 2, we define $\widetilde{T}_i(n)$ as a counter when a non-optimal arm is played in the same way. Equation (9), (10), (11) and (13) still hold.

Note that each time when $\widetilde{I}_i(t) = 1$, there exists some arm such that a non-optimal arm is picked for which m_i is the minimum in this arm. We denote this arm as $\mathbf{a}(t)$. Note that $\mathbf{a}(t)$ means there exists m, $1 \le m \le K$, such that the following holds:

$$\begin{aligned} \widetilde{T}_i(n) &\le l + \sum_{t=N}^{n} \{ \sum_{j \in \mathcal{A}_{\mathbf{a}^*,m}} a_j^{*,m} (\hat{\bar{\theta}}_{j,m_j(t)} + C_{t,m_j(t)}) \\ &\le \sum_{j \in \mathcal{A}_{\mathbf{a}(t)}} a_j(t) (\hat{\bar{\theta}}_{j,m_j(t)} + C_{t,m_j(t)}), \widetilde{T}_i(t) \ge l \}. \end{aligned} \tag{40}$$

Since at each time K arms are played, so at time t, an random variable could be observed up to Kt times. Then (14) should be modified as:

$$\begin{aligned} \widetilde{T}_i(n) \le l + \sum_{t=1}^{\infty} \sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|\mathcal{A}^*,m|}}=1}^{Kt} \sum_{m_{p_1}=l}^{Kt} \cdots \sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{Kt} \\ \{ \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}^*,m}|} a_{h_j}^{*,m} (\hat{\bar{\theta}}_{h_j,m_{h_j}} + C_{t,m_{h_j}}) \\ \le \sum_{j=1}^{|\mathcal{A}_{\mathbf{a}(t)}|} a_{p_j}(t) (\hat{\bar{\theta}}_{p_j,m_{p_j}} + C_{t,m_{p_j}}) \}. \end{aligned} \tag{41}$$

Equation (15) to (20) are similar by substituting $\mathbf{a}^*$ with $\mathbf{a}^{*,m}$. So, we have:

$$\begin{aligned} \mathbb{E}[\widetilde{T}_i(n)] &\le \left\lceil \frac{4(L+1)\ln n}{\left(\frac{\Delta_{\min}}{L a_{\max}}\right)^2} \right\rceil \\ &+ \sum_{t=1}^{\infty} \left(\sum_{m_{h_1}=1}^{Kt} \cdots \sum_{m_{h_{|\mathcal{A}^*|}}=1}^{Kt} \sum_{m_{p_1}=l}^{Kt} \cdots \sum_{m_{p_{|\mathcal{A}_{\mathbf{a}(t)}|}}=l}^{Kt} 2Lt^{-2(L+1)} \right) \\ &\le \frac{4a_{\max}^2 L^2 (L+1)\ln n}{(\Delta_{\min})^2} + 1 + \frac{\pi^2}{3} L K^{2L}. \end{aligned} \tag{42}$$

Hence, we get the upper bound for the regret as:

$$\mathfrak{R}_n^{\pi}(\Theta) \le \left[\frac{4a_{\max}^2 L^2 (L+1) N \ln n}{(\Delta_{\min})^2} + N + \frac{\pi^2}{3} L K^{2L} N \right] \Delta_{\max}. \tag{43}$$

■

IX. CONCLUSION

We have considered multi-armed bandit problems that provide for arms with rewards that are a linear function of a smaller set of random variables with unknown means. For such problems, if the number of arms is exponentially large in the number of underlying random variables, existing arm-based index policies such as the well-known UCB1 [5] have poor performance in terms of storage, computation, and regret. The LLR and LLR 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.

While this work 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 non-linear reward functions, at least in structured cases that have proved to be tractable in deterministic settings, such as convex functions.

REFERENCES

[1] S. Pandey, D. Chakrabarti, and D. Agarwal, "Multi-armed Bandit Problems with Dependent Arms", *The 24th Annual International Conference on Machine Learning*, June, 2007.

[2] P. Rusmevichientong and D. P. Williamson, "An Adaptive Algorithm for Selecting Profitable Keywords for Search-Based Advertising Services", *The 7th ACM Conference on Electronic Commerce*, June, 2006.

[3] A. Anandkumar, N. Michael, A. K. Tang, and A. Swami, "Distributed Algorithms for Learning and Cognitive Medium Access with Logarithmic Regret", to appear in *IEEE Journal on Selected Areas on Communications Special Issue on Advances in Cognitive Radio Networking and Communications*.

[4] K. Liu and Q. Zhao, "Decentralized Multi-Armed Bandit with Multiple Distributed Players", *IEEE Transactions on Signal Processing*, November, 2010.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time Analysis of the Multiarmed Bandit Problem", *Machine Learning*, vol. 47, No.2-3, 2002.

[6] R. K. Ahuja, T. L. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[7] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms (4th ed.)*, Springer, 2008

[8] T. Lai and H. Robbins, "Asymptotically Efficient Adaptive Allocation Rules", *Advances in Applied Mathematics*, vol. 6, no. 1, 1985.

[9] V. Anantharam, P. Varaiya, and J. Walrand, "Asymptotically Efficient Allocation Rules for the Multiarmed Bandit Problem with Multiple Plays-Part I: IID Rewards", *IEEE Transactions on Automatic Control*, vol. 32, no. 11, 1987.

[10] R. Agrawal, "Sample Mean Based Index Policies with $O(\log n)$ Regret for the Multi-armed Bandit Problem", *Advances in Applied Probability*, vol. 27, pp, 1054-1078, 1995.

[11] R. Ortner, "Exploiting Similarity Information in Reinforcement Learning", *2nd International Conference on Agents and Artificial Intelligence (ICAART 2010)*, January, 2010.

[12] A. J. Mersereau, P. Rusmevichientong, and J. N. Tsitsiklis, "A Structured Multiarmed Bandit Problem and the Greedy Policy", *IEEE Conference on Decision and Control*, December, 2008.

[13] P. Rusmevichientong and J. N. Tsitsiklis, "Linearly Parameterized Bandits," *Mathematics of Operations Research*, Vol. 35, No. 2, pp. 395-411, 2010

[14] P. Auer, "Using Confidence Bounds for Exploitation-exploration Tradeoffs", *Journal of Machine Learning Research*, 2002.

[15] V. Dani, T. P. Hayes, and S. M. Kakade, "Stochastic Linear Optimization under Bandit Feedback", *The 21st Annual Conference on Learning Theory (COLT)*, July, 2008.

[16] Y. Gai, B. Krishnamachari, and R. Jain, "Learning Multiuser Channel Allocations in Cognitive Radio Networks: A Combinatorial Multi-armed Bandit Formulation", *IEEE Symposium on Dynamic Spectrum Access Networks (DySPAN)*, Singapore, April, 2010.

[17] D. Pollard, *Convergence of Stochastic Processes*. Berlin: Springer, 1984.

[18] 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 Journal on Selected Areas in Communications*, vol. 22, no. 6, August, 2004.

[19] A. Brzezinski, G. Zussman, and E. Modiano, "Enabling Distributed Throughput Maximization in Wireless Mesh Networks C A Partitioning Approach", *ACM International Conference on Mobile Computing and Networking*, September, 2006.

[20] H. W. Kuhn, "The Hungarian Method for the Assignment Problem", *Naval Research Logistics Quarterly*, 1955.

[21] J. Edmonds, "Paths, Trees, and Flowers", *Canadian Journal of Mathematics*, vol. 17, pp. 449-467, 1965.

[22] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.

[23] R. Bellman, "On a Routing Problem", *Quarterly of Applied Mathematics*, vol. 16, pp. 87-90, 1958.

[24] L. R. Ford, Jr, "Network Flow Theory", *Paper P-923*, The RAND Corporation, August, 1956.

[25] J. Krarup and M. N. RØrbech, "LP Formulations of the Shortest Path Tree Problem", *4OR: A Quarterly Journal of Operations Research*, Vol. 2, No. 4, pp. 259-274, 2004.

[26] M. S. Bazaraa, J. J. Jarvis and H. D. Sherali, "Linear Programming and Network Flows,4th Edition", Wiley, December, 2009.

[27] R. C. Prim, "Shortest Connection Networks and Some Generalizations", *Bell System Technical Journal*, vol. 36 pp. 1389-1401, 1957.

[28] J. B. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem", *Proceedings of the American Mathematical Society*, Vol 7, No. 1, pp. 48-50, Febrary, 1956.