http://pubsonline.informs.org/journal/moor/

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 43, No. 3, August 2018, pp. 789–812
ISSN 0364-765X (print), ISSN 1526-5471 (online)

Improvements and Generalizations of Stochastic Knapsack and Markovian Bandits Approximation Algorithms

Will Ma[a]

[a] Operations Research Center, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
Contact: willma@mit.edu, http://orcid.org/0000-0002-2420-4468 (WM)

Received: February 28, 2016
Revised: December 28, 2016
Accepted: June 7, 2017
Published Online in Articles in Advance: February 5, 2018

MSC2010 Subject Classification: Primary: 68W25; secondary: 68W40
OR/MS Subject Classification: Primary: analysis of algorithms: suboptimal algorithms; secondary: production/scheduling: stochastic; programming: stochastic

https://doi.org/10.1287/moor.2017.0884

Abstract. We study the multi-armed bandit problem with arms which are Markov chains with rewards. In the finite-horizon setting, the celebrated Gittins indices do not apply, and the exact solution is intractable. We provide approximation algorithms for the general model of Markov decision processes with nonunit transition times. When preemption is allowed, we provide a $(1/2 - \epsilon)$-approximation, along with an example showing this is tight. When preemption isn't allowed, we provide a 1/12-approximation, which improves to a 4/27-approximation when transition times are unity. Our model captures the Markovian Bandits model of Gupta et al., the Stochastic Knapsack model of Dean et al., and the Budgeted Learning model of Guha and Munagala. Our algorithms improve existing results in all three areas. In our analysis, we encounter and overcome to our knowledge a new obstacle: an algorithm that provably exists via analytical arguments, but cannot be found in polynomial time.

Funding: The author's research was partly supported by a Natural Sciences and Engineering Research Council of Canada (NSERC) Postgraduate Scholarship—Doctoral.

Keywords: approximation algorithms • stochastic knapsack • Markovian multi-armed bandit • stochastic programming

1. Introduction

We are interested in a broad class of stochastic control problems: there are multiple evolving systems competing for the attention of a single operator, who has limited time to extract as much reward as possible. Classical examples include a medical researcher allocating his time between different clinical trials, or a graduate student shifting her efforts between different ongoing projects. Before we describe the model in detail, we introduce three problems in the literature which motivated this work.

1.1. Markovian Bandits

The Markovian multi-armed bandit problem is the following: there are some number of Markov chains (*arms*), each of which only evolves to the next node[1] and return some reward when you play (*pull*) that arm. The controller has to allocate a fixed number of pulls among the arms to maximize expected reward. The reward returned by the next pull of an arm depends on the current node that arm is on. When an arm is pulled, the controller observes the transition taken before having to choose the next arm to pull. Multi-armed bandit (MAB) problems capture the tradeoff between *exploring* arms that could potentially transition to high-reward nodes, versus *exploiting* arms that have the greatest immediate payoff.

The infinite-horizon version of this problem with discounted rewards can be solved by the celebrated index policy of Gittins; see the book Gittins et al. [12] for an in-depth treatment of Gittins indices. However, Gittins indices are dependent on the time horizon being infinite (see Gittins et al. [12, section 3.4.1]). The Gittins index measures the asymptotic performance of an arm, and does not apply when there are a finite number of time steps remaining.

Also, when we refer to multi-armed bandit in this paper, it is not to be confused with the Stochastic Bandits setting, where each arm is an unknown reward distribution, playing that arm collects a random sample from its distribution, and the objective is to learn which arm has the highest mean in a way that minimizes regret. For a comprehensive summary of Stochastic Bandits and other bandit settings, we refer to the survey by Bubeck and Cesa-Bianchi [7]. The main difference between Markovian Bandits and Stochastic Bandits is that in the former, the parameters governing the uncertainty are given as input and the challenge is computational; in the latter, there is ambiguity in the parameters and the challenge is to compete with an adversary who knows the parameters in advance.

The finite-horizon Markovian Bandits problem is intractable even in special cases (see Goel et al. [14], and the introduction of Guha and Munagala [18]), so we turn to approximation algorithms. The state of the art is an LP-relative 1/48-approximation[2] by Gupta et al. [19] (see Gupta et al. [20] for the conference version). We improve this bound by providing an LP-relative 4/27-approximation for a more general model.

1.1.1. Martingale Reward Bandits and Bayesian Bandits.

While Markovian Bandits is a different problem from Stochastic Bandits, it is a generalization of the closely related Bayesian Bandits problem, where each arm is an unknown reward distribution, but we have prior beliefs about what these distributions may be, and we update our beliefs as we collect samples from the arms. The objective is to maximize expected reward under a fixed budget of plays.

For each arm, every potential posterior distribution can be represented by a node in a Markov chain, and the transitions between nodes correspond to the laws of Bayesian inference. However, the resulting Markov chain is forced to satisfy the *martingale condition*, i.e., the expected reward at the next node must equal the expected reward at the current node, by Bayes' law. This condition is not satisfied by Stochastic Knapsack with correlated rewards, as well as certain natural applications of the bandit model. For instance, in the marketing problems studied by Bertsimas and Mersereau [4], the arms represent customers who may require repeated pulls (marketing actions) before they transition to a reward-generating node.

Nonetheless, fruitful research has been done in the Bayesian Bandits setting—Guha and Munagala [18] show that constant-factor approximations can be obtained, also under a variety of side constraints. For the Bayesian Bandits problem with no side constraints, Farias and Madan [11] show that *irrevocable* policies—policies which cannot start an arm, stop pulling it at some point, and resume it later—extract a constant fraction of the optimal (nonirrevocable) reward. Motivated by this, Guha and Munagala [18] obtain a $(\frac{1}{2}-\varepsilon)$-approximation for Bayesian Bandits that is in fact an irrevocable policy.

1.1.2. Irrevocable Bandits.

The above can be contrasted with the work of Gupta et al., who construct a non-martingale instance where irrevocable policies (they refer to these policies as *nonpreempting*) can only extract an arbitrarily small fraction of the optimal reward (Gupta et al. [19, Appendix A.3]). Therefore, without the martingale assumption, one can only hope to compare irrevocable policies against the irrevocable optimum. We provide a $(\frac{1}{2}-\varepsilon)$-approximation for this problem, which we refer to as *Irrevocable Bandits*.

1.2. Stochastic Knapsack

The Stochastic Knapsack (SK) problem was introduced by Dean et al. in 2004 (Dean et al. [10]; see Dean et al. [9] for the journal version). We are to schedule some jobs under a fixed time budget. Each job has a stochastic reward and processing time whose distribution is known beforehand. We sequentially choose which job to perform next, only discovering its length and reward in real-time as it is being processed. The objective is to maximize the expected reward before the time budget is spent. A major focus of this work is on the benefit of *adaptive* policies (which can make dynamic choices based on the instantiated lengths of jobs processed so far) over *nonadaptive* policies (which must fix an ordering of the jobs beforehand), but in our work all policies will be adaptive.

Throughout Dean et al. [9], the authors assume uncorrelated rewards—that is, the reward of a job is independent of its length. The state of the art for this setting is a $(\frac{1}{2}-\varepsilon)$-approximation by Bhalgat [5]; a $(\frac{1}{2}-\varepsilon)$-approximation is also obtained for the variant where jobs can be canceled at any time by Li and Yuan [25]. Gupta et al. [19] provides a $\frac{1}{8}$-approximation for Stochastic Knapsack with potentially correlated rewards, and a 1/16-approximation for the variant with cancellation. We improve these bounds by providing an LP-relative $(\frac{1}{2}-\varepsilon)$-approximation for a problem which generalizes both variants with correlated rewards. Furthermore, we construct an example where the true optimum is as small as $\frac{1}{2}+\varepsilon$ of the optimum of the LP relaxation. Therefore, our bound is tight in the sense that one cannot hope to improve the approximation ratio using the same LP relaxation.

However, it is important to mention that our results, as well as the results of Gupta et al. [19], require the job sizes and budget to be given in unary, since these algorithms use a time-indexed LP. It appears that this LP is necessary whenever correlation is allowed—the nontime-indexed LP can be off by an arbitrarily large factor (see Gupta et al. [19, Appendix A.2]). Techniques for discretizing the time-indexed LP if the job sizes and budget are given in binary are provided in Gupta et al. [19], albeit losing some approximation factor. Nonetheless, in this paper, we always think of processing times as discrete hops on a Markov chain, given in unary. Note that the Stochastic Knapsack problem with correlated rewards and sizes given in binary can be shown to be PSPACE-hard (Dean et al. [10, Theorem 6]).

1.3. Futuristic Bandits and Budgeted Bandits

Guha and Munagala [15, 16] have studied many variants of *budgeted learning* problems—including switching costs, concave utilities, and Lagrangian budget constraints. See Guha and Munagala [17] for an updated article that also subsumes some of their other works. Their basic setting, which we refer to as *Futuristic Bandits*, is identical to Bayesian Bandits (i.e., there are Markov chains satisfying the martingale condition), except no rewards are dispensed during the execution of the algorithm. Instead, once the budget[3] is spent, we choose a single arm we believe to be best, and only earn the (expected) reward for that arm. A $\frac{1}{4}$-approximation is provided in Guha and Munagala [17], and this is improved by the same authors to a $(\frac{1}{3} - \varepsilon)$-approximation in Guha and Munagala [18]. Our algorithm works without the martingale assumption, but the approximation guarantee is only 4/27.

1.4. MAB Superprocess with Multi-Period Actions

Motivated by these examples, we now introduce our generalized model, which we call *MAB superprocess with multiperiod actions*. Consider the Markovian Bandits setting, except we allow for a more general family of inputs, in two ways.

First, we allow transitions on the Markov chains to consume more than one pull worth of budget. We can think of these transitions as having a nonunit *processing time*. The processing times can be stochastic, and correlated with the node transition that takes place. The rewards can be accrued upon pulling the node, or only accrued if the processing time completes before the time budget runs out. The applications of such a generalization to U.S. Air Force jet maintenance have recently been considered in Kessler's thesis (Kessler [23]), where it is referred to as *multiperiod actions*.

The second generalization is that we allow each arm to be a Markov decision process; such a problem is referred to as *MAB superprocess* in Gittins et al. [12]. Now, when the controller pulls an arm, they have a choice of actions, each of which results in a different joint distribution on reward, processing time, and transition taken.

The purpose of the first generalization is to allow MAB to model the jobs from Stochastic Knapsack which have rewards correlated with processing time and can't be canceled. The purpose of the second generalization is to allow MAB to model Futuristic Bandits, where exploiting an arm corresponds to a separate action. The details of our reductions, along with examples, will be presented throughout Section 2, once we have introduced formal notation.

We consider two problem variants for our general model: the case with preemption (i.e., we can start playing an arm, not play it for some time steps, and resume playing it later), and the case without preemption. The variant without preemption is necessary to generalize Stochastic Knapsack and Irrevocable Bandits. The variant with preemption generalizes Markovian Bandits and Futuristic Bandits.

1.5. Outline of Paper

Our paper can be outlined as follows:

- reductions from existing problems to *MAB superprocess with multiperiod actions* [Section 2];
- polynomial-sized LP relaxations for both variants of *MAB superprocess with multiperiod actions*, and analytical proofs that they are indeed relaxations [Section 2.5];
- a $(\frac{1}{2} - \varepsilon)$-approximation for *MAB superprocess with multiperiod actions—no preemption*, with runtime polynomial in the input and $1/\varepsilon$ [Section 3];
- a matching upper bound where it is impossible to obtain more than $\frac{1}{2} + \varepsilon$ of the optimum of the LP relaxation [Section 3.1];
- a 4/27-approximation for *MAB superprocess* (with preemption) [Section 4];
- a 1/12-approximation for *MAB superprocess with multiperiod actions* (and preemption) [Section 4.3].

The ways in which these approximation ratios improve previous results on SK and MAB is summarized in Tables 1 and 2.[4]

1.6. Sketch of Techniques

The sketch we provide here is brief, but a more detailed high-level overview is provided at the beginning of each subsection.

In the variant without preemption, we prove that given any feasible solution to the LP relaxation, there exists a policy which plays every node with half the probability it is played in the LP solution. This would yield a $\frac{1}{2}$-approximation, but the policy cannot be specified in polynomial time, because the previous argument is purely existential. Instead, we show how to approximate the policy via sampling, in a way that doesn't cause error propagation.

Table 1. Comparison of results for SK.

Problem	Previous result	Result as a special case of our problems
Binary SK	$\frac{1}{2}-\varepsilon$ (Bhalgat [5])	—
Binary SK w/Cancellation	$\frac{1}{2}-\varepsilon$ (Li and Yuan [25])	—
Unary correlated SK	$\frac{1}{8}$ (Gupta et al. [19])	$\frac{1}{2}-\varepsilon$ (Theorem 1, Ma [26])
Unary correlated SK w/Cancellation	$\frac{1}{16}$ (Gupta et al. [19])	$\frac{1}{2}-\varepsilon$ (Theorem 1)

Table 2. Comparison of results for MAB.

Problem	Previous result	Result as a special case of our problems	Result with martingale assumption
Markovian Bandits	$\frac{1}{48}$	$\frac{4}{27}$ (Theorem 2, Ma [26])	$\frac{1}{2}-\varepsilon$ (Guha and Munagala [18])
Irrevocable bandits	—	$\frac{1}{2}-\varepsilon$ (Theorem 1)	$\frac{1}{2}-\varepsilon$ (Guha and Munagala [18])
Futuristic bandits	—	$\frac{4}{27}$ (Theorem 2)	$\frac{1}{3}-\varepsilon$ (Guha and Munagala [18])
Budgeted bandits	—	$\frac{1}{12}$ (Theorem 3)	$\frac{1}{4}-\varepsilon$ (Guha and Munagala [17])

In the variant with preemption, we derive an approximation algorithm, which uses priority indices, based on an optimal solution to the LP relaxation, to accomplish the explore-exploit tradeoff. Our priority-based policy is based on the ideas behind the *convex decomposition* and *gap filling* operations from Gupta et al. [19]. We perform a tighter analysis for our algorithm and show how it can be generalized to the model of Markov decision processes with nonunit transition times. Our analysis uses Samuels' conjecture (Samuels [31]) for $n = 3$ (which is proven) to bound the upper tail.

1.7. Related Work

The results on bandits, Stochastic Knapsack, and budgeted learning that are most related to our results have already been introduced in the earlier subsections, but we mention some additional results here. One such result for Stochastic Knapsack is the bi-criteria $(1-\varepsilon)$-approximation of Bhalgat et al. [6] that uses $1+\varepsilon$ as much space; such a result is also obtained via alternate methods by Li and Yuan [25] and generalized to the setting with both correlated rewards and cancellation. Also, Gupta et al. [21] introduce the new *stochastic orienteering* problem, which associates jobs in SK with locations in a metric space. The benefit of adaptive policies for this problem is also addressed by Bansal and Nagarajan [1].

Another example of a stochastic optimization problem where adaptive policies are necessary is the *stochastic matching* problem of Bansal et al. [2]—in fact, we use one of their lemmas in our analysis. Recently, the setting of stochastic matching has been integrated into online matching problems by Mehta and Panigrahi [27].

All of the problems described thus far focus on expected reward. Recently, Ilhan et al. [22] studied the variant of SK where the objective is to maximize the probability of achieving a target reward; older work on this model includes Carraway et al. [8]. Approximation algorithms for minimizing the expected sum of weighted completion times when the processing times are stochastic are provided in Möhring et al. [28], and Skutella and Uetz [33]. SK with chance constraints—maximizing the expected reward subject to the probability of running overtime being at most p—is studied in Goel and Indyk [13], and Kleinberg et al. [24].

Looking at more comprehensive synopses, we point the reader interested in infinite-horizon Markovian Bandits to the book by Gittins et al. [12]. Families of bandit problems other than Markovian, including Stochastic and Adversarial, are surveyed by Bubeck and Cesa-Bianchi [7]. For an encyclopedic treatment of using dynamic programming to solve stochastic control problems, we refer the reader to the book by Bertsekas [3]. For an encyclopedic treatment of stochastic scheduling, we refer the reader to the book by Pinedo [30].

2. Problem Definition

We first define the fully general *MAB superprocess with multiperiod actions* (and preemption) problem described in the introduction. We introduce the variant without preemption later.

Problem 1 (Original Problem)**.** There are $n \in \mathbb{N}$ arms which are Markov decision processes. For each arm i, let $\mathcal{S}_i$ denote its finite set of nodes, with the starting node, or *root node*, being ρ_i. To *play* an arm i that is currently on node $u \in \mathcal{S}_i$, we select an action a from the finite, nonempty action set A_u, after which the arm will transition to a new node $v \in \mathcal{S}_i$ in t time steps, accruing reward over this duration. We will also refer to this process as *playing action a on node u*, since for each pair (u,a), we are given as input the joint distribution of the destination node, transition time, and reward. Specifically, for all $u \in \mathcal{S}_i$, $a \in A_u$, and possible pairs of destination and transition time (v,t), let $p^a_{u,v,t}$ denote the probability of transitioning to node v in exactly t time steps, when action a is played on node u. We will refer to this transition by the quadruple (u,a,v,t), and conditioned on it occurring, let $R^a_{u,v,t,t'}$ denote the reward accrued t' time steps from the present, for all $t' = 0,\dots,t-1$. $R^a_{u,v,t,t'}$ is a random variable with known distribution, taking values in $[0,\infty)$.

Each Markov decision process i starts on its root node, ρ_i. There is a total budget of $B \in \mathbb{N}$ time steps over which we would like to maximize the reward, in expectation. At each time step, if no arm is in the middle of a transition, then we may choose a new arm to play, along with an action.[5] We observe the destination and sequence of rewards over the transition time, realized according to the probabilities defined above. After the transition is over, we may choose a new arm and action to play. After B total time steps pass, our final reward is the sum of rewards collected up to that point, and an arm in the middle of transition is cut off.

2.1. Problem Simplification

We now perform a sequence of transformations to simplify the problem and notation, as well as aid in the analysis throughout the rest of the paper. Assuming that the budget B and transition times t are given in unary, all of the transformations can be performed in polynomial time.

Transformation 1. We use $A = \bigcup_{i=1}^{n} \bigcup_{u \in \mathcal{S}_i} A_u$, a common set of actions for all nodes across all arms, defining incompatible actions to result in no transition.

Transformation 2. We change all transition times t greater than B to equal B, cutting off rewards with $t' \geq B$. Since there are only B time steps, the exact value of any $t \geq B$ does not matter, and only the rewards with $t' = 0,\dots,B-1$ may be obtainable.

Transformation 3. We replace each random reward $R^a_{u,v,t,t'}$ with its deterministic equivalent $r^a_{u,v,t,t'} = \mathbb{E}[R^a_{u,v,t,t'}]$. This does not affect the objective of maximizing expected reward.

Transformation 4. We add self-loops with unit-time and zero reward so that for all nodes u and actions a, $\sum_{v \in \mathcal{S}_i} \sum_{t=1}^{B} p^a_{u,v,t} = 1$. That is, an arm always makes a transition, instead of stopping.

Transformation 5. We expand all transitions with nonunit processing times. For any transition (u,a,v,t) with $t > 1$, we:

1. add *bridge nodes* $w_1,\dots,w_{t-1}$;
2. set transition probability $p^a_{u,w_1,1} = p^a_{u,v,t}$, and change $p^a_{u,v,t}$ to be 0;
3. set transition probabilities $p^b_{w_1,w_2,1} = \cdots = p^b_{w_{t-1},v,1} = 1$ for all $b \in A$;
4. set all other transition probabilities involving $w_1,\dots,w_{t-1}$ to be 0;
5. set rewards $r^a_{u,w_1,1,0} = r^a_{u,v,t,0}$, $r^b_{w_1,w_2,1,0} = r^a_{u,v,t,1},\dots,r^b_{w_{t-1},v,1,0} = r^a_{u,v,t,t-1}$ for all $b \in A$.

As long as we enforce that the bridge nodes must be played as soon as they are reached, the new problem is equivalent to the old problem. Notationally, we will assume that they are played with action α. Also, we will eliminate the subscripts t,t' and just write $p^a_{u,v}$, $r^a_{u,v}$ now that all transitions with $t > 1$ have been eliminated.

Transformation 6. For all $i \in [n]$, $u \in \mathcal{S}_i$, and $a \in A$, define $r^a_u = \sum_{v \in \mathcal{S}_i} p^a_{u,v} \cdot r^a_{u,v}$, and consider the Markov decision process that earns deterministic reward r^a_u every time action a is played on node u, instead of a random reward $r^a_{u,v}$ that depends on the destination node v. Under the objective of maximizing expected reward, the two processes are again equivalent.

Transformation 7. We convert each rooted Markov decision process into a layered acyclic digraph, up to depth B. That is, we assume there exists a function depth mapping nodes to $0,\dots,B$ such that $\mathsf{depth}(\rho_i) = 0$ for all $i \in [n]$, and all transitions (u,a,v) with $p^a_{u,v} > 0$ satisfy $\mathsf{depth}(v) = \mathsf{depth}(u) + 1$. This can be done by expanding each node in the original graph into a time-indexed copy of itself for $t = 1,\dots,B$—we refer to Gupta et al. [19, Appendix E.1] for the standard reduction, which immediately generalizes to the case of Markov decision processes with bridge nodes. We can cut off at depth B since there are only B time steps in total.

We restate the problem after all the transformations, summarizing the notation.

Problem 2 (Transformed Problem)**.** An instance of *MAB superprocess with multiperiod actions* consists of the following:

- A: the global set of actions, indexed by a, containing
  - —α: the default action used to play bridge nodes;
- n: the number of arms, indexed by i, each with
  - —$\mathcal{S}_i$: a finite set of nodes;
  - —$\mathcal{B}_i$: the set of *bridge nodes*, a potentially empty subset of $\mathcal{S}_i$;
  - —ρ_i: a root node in $\mathcal{S}_i \setminus \mathcal{B}_i$;
  - —$p^a_{u,v}$: the probability of transitioning to node v when action a is played on node u, for all $a \in A$ and $u, v \in \mathcal{S}_i$ (with $u = v$ possible);
  - —r^a_u: the reward obtained when action a is played on node u, for all $a \in A$ and $u \in \mathcal{S}_i$;
- B: the number of time steps, indexed by t;
- depth: a function from $\bigcup_{i=1}^n \mathcal{S}_i$ to $\{0, \dots, B\}$ satisfying
  - —$\mathsf{depth}(\rho_i) = 0$ for all i;
  - —$\mathsf{depth}(v) = \mathsf{depth}(u) + 1$ for all (u, a, v) with $p^a_{u,v} > 0$.

The objective is to choose an arm and an action during each time step to maximize expected reward. At a time step, if the arm last played is on a bridge node, then the same arm must be played again.

2.2. Dynamic Programming

Algorithms for this problem are described in the form of an adaptive *policy*, a specification of which arm and action to play for each state the system could potentially be in. A state in this case is determined by the following information: the node each arm is on, and the time step we are at.[6] The optimal policy can be defined by an exponential-sized dynamic program. We write the Bellman state-updating equations as constraints to get a linear program whose feasible region is precisely the set of admissible policies. After adding in the objective function of maximizing expected reward, solving this exponential-sized linear program would be equivalent to solving our problem to optimality.

Definition 1. Define the following notation and related terminology.

1. For any positive integer m, let $[m]$ denote the set $\{1, \dots, m\}$.
2. Let $\mathcal{S} = \bigcup_{i=1}^n \mathcal{S}_i$, the union of all nodes across all arms.
3. For all $u \in \mathcal{S}$, let $\mathsf{Par}(u) = \{(v, a) \in \mathcal{S} \times A : p^a_{v,u} > 0\}$, the set of *parents* of u, i.e., the (node, action) combinations that have a positive probability of transitioning to u.
4. Let $\boldsymbol{\mathcal{S}} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$, the set of *joint nodes*, which are ordered n-tuples indicating the node each arm is on.
5. For all $\pi \in \boldsymbol{\mathcal{S}}$ and $u \in \mathcal{S}_i$, let π^u be the joint node such that $\pi^u_i = u$, and $\pi^u_j = \pi_j$ for all $j \neq i$.

A state in the dynamic program can then be defined as a joint node π along with a time t. Let $y_{\pi,t}$ be the probability of having arms on nodes according to π at the beginning of time t. Let $z^a_{\pi,i,t}$ be the probability we play arm i at time t with action a, when the arms are on nodes according to π. Note that some (π, t) pairs are impossible states,[7] but for notational convenience we still have variables for these states.

Our objective is

$$\max \sum_{\pi \in \boldsymbol{\mathcal{S}}} \sum_{i=1}^n \sum_{a \in A} r^a_{\pi_i} \sum_{t=1}^B z^a_{\pi,i,t} \tag{1}$$

with the following constraints on how we can play the arms:

$$\sum_{i=1}^n \sum_{a \in A} z^a_{\pi,i,t} \le y_{\pi,t}, \qquad \pi \in \boldsymbol{\mathcal{S}},\ t \in [B], \tag{2a}$$

$$z^\alpha_{\pi,i,t} = y_{\pi,t}, \qquad \pi \in \boldsymbol{\mathcal{S}},\ i : \pi_i \in \mathcal{B},\ t \in [B], \tag{2b}$$

$$z^a_{\pi,i,t} \ge 0, \qquad \pi \in \boldsymbol{\mathcal{S}}, i \in [n],\ a \in A,\ t \in [B]. \tag{2c}$$

The novel constraint is (2b), which guarantees that we must play a bridge node upon arrival. The remaining constraints update the $y_{\pi,t}$'s correctly:

$$y_{(\rho_1,\dots,\rho_n),1} = 1, \tag{3a}$$

$$y_{\pi,1} = 0, \qquad \pi \in \boldsymbol{\mathcal{S}} \setminus \{(\rho_1, \dots, \rho_n)\}, \tag{3b}$$

$$y_{\pi,t} = y_{\pi,t-1} - \sum_{i=1}^n \sum_{a \in A} z^a_{\pi,i,t-1} + \sum_{i=1}^n \sum_{(u,a) \in \mathsf{Par}(\pi_i)} z^a_{\pi^u,i,t-1} \cdot p^a_{u,\pi_i}, \qquad t > 1,\ \pi \in \boldsymbol{\mathcal{S}}. \tag{3c}$$

Essentially, the only decision variables are the z-variables; there are as many y-variables as equalities in (3a)–(3c). These constraints guarantee $\sum_{\pi\in\boldsymbol{\mathcal{S}}} y_{\pi,t} = 1$ for all $t \in [B]$, and combined with (2a), we obtain

$$\sum_{\pi\in\boldsymbol{\mathcal{S}}} \sum_{i=1}^{n} \sum_{a\in A} z^a_{\pi,i,t} \le 1 \quad t \in [B] \tag{4}$$

Let (ExpLP) denote the linear program defined by objective (1) and constraints (2a)–(2c), (3a)–(3c) which imply (4). This is the dynamic program for *MAB superprocess with multiperiod actions* (and preemption).

2.3. No Preemption Variant

Now we define *MAB superprocess with multiperiod actions—no preemption*. We follow the same set-up from Problem 1, and assume that the same sequence of transformations from Section 2.1 have been performed to arrive at Problem 2. However, we add the further constraint that for each arm, the set of time steps during which we play it must be contiguous. We enforce this by adding a *terminal node* to each arm, from which it cannot be played; if an arm not on its root node is not played during a time step, then it transitions to its terminal node.

Definition 2. Define the following notation and related terminology.

1. For all $i \in [n]$, let ϕ_i denote the terminal node of arm i, and let $\mathcal{S}'_i = \mathcal{S}_i \cup \{\phi_i\}$.
2. Let $\boldsymbol{\mathcal{S}}' = \mathcal{S}'_1 \times \cdots \times \mathcal{S}'_n \backslash \{\pi\colon \pi_i \notin \{\rho_i, \phi_i\}, \pi_j \notin \{\rho_j, \phi_j\}, i \ne j\}$, where we have excluded the joint nodes with two or more arms in the middle of being processed.
3. For all $\pi \in \boldsymbol{\mathcal{S}}'$, let $I(\pi) = \{i\colon \pi_i \ne \phi_i\}$, the indices of arms that could be played from π.
4. For all $i \in [n]$, let $\boldsymbol{\mathcal{A}}_i = \{\pi \in \boldsymbol{\mathcal{S}}'\colon \pi_i \notin \{\rho_i, \phi_i\}\}$, the joint nodes with arm i *active*, i.e., in the middle of being processed.
5. Let $\boldsymbol{\mathcal{A}} = \bigcup_{i=1}^n \boldsymbol{\mathcal{A}}_i$, the set of joint nodes with an active arm.
6. For all $\pi \in \boldsymbol{\mathcal{S}}'$, let $\boldsymbol{\mathcal{P}}(\pi)$ denote the subset of $\boldsymbol{\mathcal{S}}'$ that would transition to π with *no play* during a time step.

(a) If $\pi \notin \boldsymbol{\mathcal{A}}$, then $\boldsymbol{\mathcal{P}}(\pi) = \{\pi\} \cup (\bigcup_{i\notin I(\pi)} \{\pi^u\colon u \in \mathcal{S}_i \backslash \{\rho_i\}\})$. $\boldsymbol{\mathcal{P}}(\pi)$ contains π because $\pi \notin \boldsymbol{\mathcal{A}}$; hence, π does not contain an active arm which would transition to its terminal node with no play, and hence, the system would remain at the same joint node π. Furthermore, for any i such that $i \notin I(\pi)$ (i.e., $\pi_i = \phi_i$), arm i would transition from any $u \in \mathcal{S}_i \backslash \{\rho_i\}$ to ϕ_i with no play; hence, for such u, $\boldsymbol{\mathcal{P}}(\pi)$ contains joint node π^u.

(b) If $\pi \in \boldsymbol{\mathcal{A}}$, then $\boldsymbol{\mathcal{P}}(\pi) = \varnothing$. This is because if joint node π has an active arm i, then we can only arrive at π by playing i, i.e., we cannot arrive at π with no play.

Now we can write the dynamic program for the variant without preemption. The objective is

$$\max \sum_{\pi\in\boldsymbol{\mathcal{S}}'} \sum_{i\in I(\pi)} \sum_{a\in A} r^a_{\pi_i} \sum_{t=1}^{B} z^a_{\pi,i,t} \tag{5}$$

with very similar constraints on the z-variables:

$$\sum_{i\in I(\pi)} \sum_{a\in A} z^a_{\pi,i,t} \le y_{\pi,t}, \qquad \pi \in \boldsymbol{\mathcal{S}}', \ t \in [B], \tag{6a}$$

$$z^{\alpha}_{\pi,i,t} = y_{\pi,t}, \qquad \pi \in \boldsymbol{\mathcal{S}}', \ i\colon \pi_i \in \mathcal{B}, \ t \in [B], \tag{6b}$$

$$z^a_{\pi,i,t} \ge 0, \qquad \pi \in \boldsymbol{\mathcal{S}}', \ i \in I(\pi), \ a \in A, \ t \in [B]. \tag{6c}$$

The only difference from (2a)–(2c) is that arms on terminal nodes cannot be played. However, moving forward, the state-updating constraints become more complicated, because now an arm can make a transition even while it is not being played; namely, the transition to the terminal node. We update the y-variables as follows:

$$y_{(\rho_1,\ldots,\rho_n),1} = 1, \tag{7a}$$

$$y_{\pi,1} = 0 \qquad \pi \in \boldsymbol{\mathcal{S}}' \backslash \{(\rho_1, \ldots, \rho_n)\}, \tag{7b}$$

$$y_{\pi,t} = \sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi)} \left(y_{\pi',t-1} - \sum_{i\in I(\pi')} \sum_{a\in A} z^a_{\pi',i,t-1} \right), \qquad t > 1, \ \pi \in \boldsymbol{\mathcal{S}}' \backslash \boldsymbol{\mathcal{A}}, \tag{7c}$$

$$y_{\pi,t} = \sum_{a\colon(\rho_i,a)\in \mathsf{Par}(\pi_i)} \left(\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_i})} z^a_{\pi',i,t-1} \right) \cdot p^a_{\rho_i,\pi_i}, \qquad t > 1, \ i \in [n], \ \pi \in \boldsymbol{\mathcal{A}}_i, \ \mathsf{depth}(\pi_i) = 1, \tag{7d}$$

$$y_{\pi,t} = \sum_{(u,a)\in \mathsf{Par}(\pi_i)} z^a_{\pi^u,i,t-1} \cdot p^a_{u,\pi_i}, \qquad t > 1, \ i \in [n], \ \pi \in \boldsymbol{\mathcal{A}}_i, \ \mathsf{depth}(\pi_i) > 1. \tag{7e}$$

Equations (7c) update $y_{\pi,t}$ for $\pi \notin \boldsymbol{\mathcal{A}}$, i.e., joint nodes with no active arms. Such a joint node π can only be arrived upon by making no play from a joint node in $\boldsymbol{\mathcal{P}}(\pi)$.

Equations (7d), (7e) update $y_{\pi,t}$ for $\pi \in \boldsymbol{\mathcal{A}}$. To get to joint node $\pi \in \boldsymbol{\mathcal{A}}_i$, we must have played arm i during the previous time step and transitioned to node π_i. However, the restrictions on the previous joint node depend on whether $\mathsf{depth}(\pi_i) = 1$. If so, then arm i was on ρ_i at time step $t-1$, so it's possible to get to π from any joint node in $\boldsymbol{\mathcal{P}}(\pi^{\rho_i})$. That is, in the previous joint node, there could have been an active arm that is not i. This is reflected in (7d). On the other hand, if $\mathsf{depth}(\pi_i) > 1$, then arm i must have been the active arm at time step $t-1$, as described in (7e).

Like before, these equations guarantee that at each time step, we are at exactly one joint node, i.e., $\sum_{\pi \in \boldsymbol{\mathcal{S}}'} y_{\pi,t} = 1$. Combined with (6a), we obtain

$$\sum_{\pi \in \boldsymbol{\mathcal{S}}'} \sum_{i \in I(\pi)} \sum_{a \in A} z^a_{\pi,i,t} \le 1, \quad t \in [B]. \tag{8}$$

Let (ExpLP') denote the linear program defined by objective (5) and constraints (6a)–(6c), (7a)–(7e) which imply (8). This is the dynamic program for *MAB superprocess with multiperiod actions—no preemption*.

2.4. Reductions from SK and MAB

Before we proceed, we explain why our model captures the problems described in the introduction.

Markovian Bandits can be captured immediately by defining the action set to consist of a single action, and not having any transition times greater than unity. Irrevocable Bandits, the nonpreempting variant, can be captured analogously by using the no preemption variant of our problem defined in Section 2.3.

We show how to reduce the Stochastic Knapsack variants to Problem 1, which can then be considered under the nonpreempting variant in Section 2.3. A job in an instance of correlated Stochastic Knapsack can be given as follows, when processing times are in unary (Gupta et al. [19, section 2.1]). Let B be the total time budget. For each $t \in [B]$, let $\tilde{p}_t$ be the probability of the job finishing after exactly t time steps, and in such a case, let $\tilde{r}_t$ be the reward returned upon completion. There are two problem variants, one where jobs cannot be canceled once started, and another where jobs can be canceled at any time (e.g., after observing that it will not finish before a critical threshold).

Transformation 8 (Correlated SK without Cancellation to Problem 1)**.** There is a single action and we will ignore the superscript a. The set of nodes is $\{\rho, \phi\}$, with the root node being ρ. For each $t \in [B]$, we set $p_{\rho,\phi,t} = \tilde{p}_t$, with $R_{\rho,\phi,t,t-1}$ taking the deterministic value of $\tilde{r}_t$. This transition represents the job finishing after exactly t time steps, returning a reward upon processing the final time step.

Transformation 9 (Correlated SK with Cancellation to Problem 1)**.** There is a single action and we will ignore the superscript a. The set of nodes is $\{S_1, \ldots, S_B, \phi\}$, with the root node being S_1. Node S_t represents the job processing its t'th time step, and node ϕ represents the job finishing. For each $t \in [B]$, we set $p_{S_t,\phi,1} = \tilde{p}_t / \sum_{t' \ge t} \tilde{p}_{t'}$, the probability of the job finishing upon processing time step t conditioned on it not finishing before then. We set $p_{S_t,S_{t+1},1} = 1 - p_{S_t,\phi,1}$. The rewards are on the transitions from S_t to ϕ, with $R_{S_t,\phi,1,0}$ taking the deterministic value of $\tilde{r}_t$.

We show some examples of these transformations in Appendix A. As previously stated, for both variants of SK, we use the nonpreempting adaptation of our problem defined in Section 2.3. However, we should point out that for SK with Cancellation, allowing preemption on the jobs under our model results in a distinct problem. In Appendix B, we construct an example where a policy that both preempts and cancels earns more reward than the best policy possible with only cancellation, even when rewards are uncorrelated with processing times.

Finally, we show how to reduce Futuristic Bandits to Problem 1. In Futuristic Bandits, there are n Markov chains with rewards on nodes. There is a budget of T "exploration" time steps. During these time steps, arms can be played so that they transition to different nodes, but no reward is obtained. At the end of these time steps, each arm is on some final node. Then there is a single "exploit" time step where the greatest reward among the n final nodes is obtained. The objective is to maximize the expected amount exploited.

Transformation 10 (Futuristic Bandits to Problem 1)**.** We can use the Markov chains from Futuristic Bandits directly as the arms in Problem 1. However, we place no rewards on nodes. Instead, we add a separate "exploit" action, which when played on a node, returns the reward of that node. The exploit action has processing time $T+1$ and we set our time budget B to be $2T+1$.

This is equivalent to the Futuristic Bandits problem. First, note that it is impossible to collect exploitation rewards from more than one arm. Also, we can explore for at most T time steps if we are going to earn any reward at all. Note that in our problem it is possible to stop exploring before T time steps pass. However, it is never beneficial to do so when the rewards on nodes satisfy the martingale condition, as assumed in Futuristic Bandits—see Section 1.1.1 and the papers referenced in Section 1.3. The Budgeted Bandits generalization can also be reduced to Problem 1 by having different processing times for exploring different arms.

2.5. Polynomial-Sized LP Relaxations

We now write the polynomial-sized LP relaxations of our earlier problems. We keep track of the probabilities of being on the nodes of each arm individually without considering their joint distribution. Let $s_{u,t}$ be the probability arm i is on node u at the beginning of time t. Let $x^a_{u,t}$ be the probability we play action a on node u at time t.

For both variants of the problem, we have the objective

$$\max \sum_{u\in\mathcal{S}} \sum_{a\in A} r^a_u \sum_{t=1}^{B} x^a_{u,t} \tag{9}$$

and constraints on how we can play each individual arm:

$$\sum_{a\in A} x^a_{u,t} \le s_{u,t}, \qquad u\in\mathcal{S},\ \ t\in[B], \tag{10a}$$

$$x^\alpha_{u,t} = s_{u,t}, \qquad u\in\mathcal{B},\ \ t\in[B], \tag{10b}$$

$$x^a_{u,t} \ge 0, \qquad u\in\mathcal{S},\ \ a\in A,\ \ t\in[B]. \tag{10c}$$

Furthermore, there is a single constraint

$$\sum_{u\in\mathcal{S}} \sum_{a\in A} x^a_{u,t} \le 1, \quad t\in[B] \tag{11}$$

enforcing that the total probabilities of plays across all arms cannot exceed 1 at any time step.

The state-updating constraints differ for the two variants of the problem. If we allow preemption, then they are:

$$s_{\rho_i,1} = 1, \qquad i\in[n], \tag{12a}$$

$$s_{u,1} = 0, \qquad u\in\mathcal{S}\backslash\{\rho_1,\dots,\rho_n\}, \tag{12b}$$

$$s_{u,t} = s_{u,t-1} - \sum_{a\in A} x^a_{u,t-1} + \sum_{(v,a)\in \mathsf{Par}(u)} x^a_{v,t-1}\cdot p^a_{v,u}, \qquad t>1,\ \ u\in\mathcal{S}. \tag{12c}$$

If we disallow preemption, then an arm can only be on a nonroot node if we played the same arm during the previous time step. This is reflected in (13c)–(13d):

$$s_{\rho_i,1} = 1, \qquad i\in[n], \tag{13a}$$

$$s_{u,1} = 0, \qquad u\in\mathcal{S}\backslash\{\rho_1,\dots,\rho_n\}, \tag{13b}$$

$$s_{\rho_i,t} = s_{\rho_i,t-1} - \sum_{a\in A} x^a_{\rho_i,t-1}, \qquad t>1,\ \ i\in[n], \tag{13c}$$

$$s_{u,t} = \sum_{(v,a)\in \mathsf{Par}(u)} x^a_{v,t-1}\cdot p^a_{v,u}, \qquad t>1,\ \ u\in\mathcal{S}\backslash\{\rho_1,\dots,\rho_n\}. \tag{13d}$$

Let `(PolyLP)` denote the linear program defined by objective (9) and constraints (10a)–(10c), (11), (12a)–(12c). Similarly, let `(PolyLP')` denote the linear program defined by objective (9) and constraints (10a)–(10c), (11), (13a)–(13d). We still have to prove the polynomial-sized linear programs are indeed relaxations of the exponential-sized linear programs. For any linear program LP, let $\mathsf{OPT}_{\mathrm{LP}}$ denote its optimal objective value.

Lemma 1. *Given a feasible solution $\{z^a_{\pi,i,t}\},\{y_{\pi,t}\}$ to* `(ExpLP)`*, we can construct a solution to* `(PolyLP)` *with the same objective value by setting $x^a_{u,t} = \sum_{\pi\in\boldsymbol{\mathcal{S}}:\pi_i=u} z^a_{\pi,i,t}$, $s_{u,t} = \sum_{\pi\in\boldsymbol{\mathcal{S}}:\pi_i=u} y_{\pi,t}$ for all $i\in[n]$, $u\in\mathcal{S}_i$, $a\in A$, $t\in[B]$. Thus, the feasible region of* `(PolyLP)` *is a projection of that of* `(ExpLP)` *onto a subspace and $\mathsf{OPT}_{\mathtt{ExpLP}} \le \mathsf{OPT}_{\mathtt{PolyLP}}$.*

Lemma 2. *Given a feasible solution $\{z^a_{\pi,i,t}\},\{y_{\pi,t}\}$ to* `(ExpLP')`*, we can construct a solution to* `(PolyLP')` *with the same objective value by setting $x^a_{u,t} = \sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=u} z^a_{\pi,i,t}$, $s_{u,t} = \sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=u} y_{\pi,t}$ for all $i\in[n]$, $u\in\mathcal{S}_i$, $a\in A$, $t\in[B]$. Thus the feasible region of* `(PolyLP')` *is a projection of that of* `(ExpLP')` *onto a subspace and $\mathsf{OPT}_{\mathtt{ExpLP'}} \le \mathsf{OPT}_{\mathtt{PolyLP'}}$.*

Essentially, Lemma 1 says that PolyLP reduces from ExpLP, and Lemma 2 says that PolyLP' reduces from ExpLP'. Recall that the feasible regions of ExpLP and ExpLP' correspond exactly to the admissible policies in the two variants. These lemmas say that the performance of any adaptive policy can be upper bounded by the polynomial-sized relaxations. Our lemmas are analogous to similar statements from earlier works (e.g., Gupta et al. [19, lemma 2.1]), but put into the context of an exponential-sized linear program. Their proofs are mostly analytical and deferred to Appendix C.

2.6. Main Results

Now that we have established the preliminaries, we are ready to state our main results in the form of theorems.

Theorem 1. *Given a feasible solution* $\{x^a_{u,t}\}, \{s_{u,t}\}$ *to* (PolyLP'), *there exists a solution to* (ExpLP') *with* $\sum_{\pi:\pi_i=u} z^a_{\pi,i,t} = \frac{1}{2}x^a_{u,t}$, $\sum_{\pi:\pi_i=u} y_{\pi,t} = \frac{1}{2}s_{u,t}$ *for all* $i \in [n]$, $u \in \mathcal{S}_i$, $a \in A$, $t \in [B]$, *obtaining reward* $\frac{1}{2}\mathsf{OPT}_{\texttt{PolyLP'}}$. *We can use sampling to turn this into a* $(\frac{1}{2}-\varepsilon)$*-approximation algorithm for MAB superprocess with multiperiod actions—no preemption, with runtime polynomial in the input and* $1/\varepsilon$.

We prove this theorem in Section 3, and also show that it is tight, constructing an instance under the special case of correlated SK where it is impossible to obtain reward greater than $(\frac{1}{2}+\varepsilon)\mathsf{OPT}_{(\texttt{PolyLP'})}$.

Theorem 2. *There is a* (PolyLP)*-relative* 4/27*-approximation algorithm for MAB superprocess* (*with preemption*), *when all processing times are* 1.

Theorem 3. *There is a* (PolyLP)*-relative* 1/12*-approximation algorithm for MAB superprocess with multiperiod actions* (*and preemption*).

We prove these theorems in Section 4.

3. Proof of Theorem 1

In this section, we prove Theorem 1. To build intuition, we will first present the upper bound, showing a family of examples with $\mathsf{OPT}_{\texttt{ExpLP'}}/\mathsf{OPT}_{\texttt{PolyLP'}}$ approaching $\frac{1}{2}$.

3.1. Construction for Upper Bound

Let N be a large integer. We will describe our $n=2$ arms as stochastic jobs. Job 1 takes $N+1$ time with probability $1-1/N$, in which case it returns a reward of 1. It takes 1 time with probability $1/N$, in which case it returns no reward. Job 2 deterministically takes 1 time and returns a reward of 1. The budget is $B=N+1$ time steps.

Any actual policy can never get more than 1 reward, since it cannot get a positive reward from both jobs.

After all the reductions from Section 2, the Markov Chains representing these jobs can be denoted as follows. Let $\mathcal{S}_1 = \{S_0, S_1, \ldots, S_N, \phi_1\}$, with $\rho_1 = S_0$. There is only one action, and we will omit the action superscripts. The only uncertainty is at S_0, with $p_{S_0,S_1} = 1-1/N$, $p_{S_0,\phi_1} = 1/N$. The remaining transitions are $p_{S_1,S_2} = \cdots = p_{S_{N-1},S_N} = p_{S_N,\phi_1} = 1$, and self loop on the terminal node $p_{\phi_1,\phi_1} = 1$. The only reward is a reward of 1 on node S_N. Meanwhile, $\mathcal{S}_2$ consists only of nodes $\{\rho_2, \phi_2\}$, with $p_{\rho_2,\phi_2} = p_{\phi_2,\phi_2} = 1$, $r_{\rho_2} = 1$.

Consider the solution for (PolyLP') with $x_{S_0,1} = 1$, $x_{S_1,2} = \cdots = x_{S_N,N+1} = 1-1/N$, $x = 0$, $x_{\rho_2,2} = \cdots = x_{\rho_2,N+1} = 1/N$, all other x-variables equal to 0, and s-variables determined using (13a)–(13d). It can be checked that this solution is feasible, and that its objective value of $2-1/N$ is optimal, since all of the potential reward is acquired. Hence, as we take $N \to \infty$, we get $\mathsf{OPT}_{\texttt{ExpLP'}}/\mathsf{OPT}_{\texttt{PolyLP'}} = \frac{1}{2}$.

Note that we can put all of $\mathcal{S}_1 \backslash \{\rho_1, \phi_1\}$ in $\mathcal{B}$ if we want; it does not change the example whether job 1 can be canceled once started. It also does not matter whether we allow preemption—both $\mathsf{OPT}_{\texttt{ExpLP}}/\mathsf{OPT}_{\texttt{PolyLP}}$ and $\mathsf{OPT}_{\texttt{ExpLP'}}/\mathsf{OPT}_{\texttt{PolyLP'}}$ are $\frac{1}{2}+\varepsilon$ for this example.

Let us analyze what goes wrong when we attempt to replicate the optimal solution to the LP relaxation in an actual policy. We start job 1 at time 1 with probability $x_{S_0,1} = 1$. If it does not terminate after 1 time step, which occurs with probability $1-1/N$, then we play job 1 through to the end, matching $x_{S_1,2} = \cdots = x_{S_N,N+1} = 1-1/N$. If it does, then we start job 2 at time 2. This occurs with unconditional probability $x_{\rho_2,2} = 1/N$, as planned. However, in this case, we cannot start job 2 again at time 3 (since it has already been processed at time 2), even though $x_{\rho_2,3} = 1/N$ is telling us to do so. The LP relaxation fails to consider that event "job 1 takes time 1" is *directly correlated* with event "job 2 is started at time 2," so the positive values specified by $x_{\rho_2,3}, \ldots, x_{\rho_2,N+1}$ are illegal plays.

Motivated by this example, we observe that if we only try to play u at time t with probability $x_{u,t}/2$, then we can obtain a solution to (ExpLP') (and hence a feasible policy) that is a scaled copy of the solution to (PolyLP').

3.2. Specification of Solution to (ExpLP')

Fix a solution $\{x^a_{u,t}, s_{u,t}\}$ to (PolyLP'). Our objective in this subsection is to construct a solution $\{z^a_{\pi,i,t}, y_{\pi,t}\}$ to (ExpLP') such that

$$\sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=u} z^a_{\pi,i,t} = \frac{x^a_{u,t}}{2}, \quad i\in[n],\ u\in\mathcal{S}_i,\ a\in A \tag{14}$$

obtaining half the objective value of (PolyLP'). We will prove feasibility in Section 3.3.

The intuition for the construction is as follows. For any $u\in\mathcal{S}_i$ and $t\in[B]$, in order to play node u at time t, we must have started playing arm i at time $t-\mathsf{depth}(u)$, since preemption is not allowed. Therefore, it is possible to partition the combinations of u,a,t where $x^a_{u,t}>0$ according to the time at which we must play ρ_i. Having established this, we only have to make decisions on which new arm to start, when the values of $x^a_{u,t}$ prescribe that the current arm should be stopped. To satisfy the global constraint (14), the decision to start arm i at time t (in a specific state) depends on the total probability of *being able to* start arm i at time t (conditioned on the past policy and realizations).

For convenience, define $x_{u,t}=\sum_{a\in A} x^a_{u,t}$ and $z_{\pi,i,t}=\sum_{a\in A} z^a_{\pi,i,t}$. We will complete the specification of $\{z^a_{\pi,i,t}, y_{\pi,t}\}$ over B iterations $t=1,\dots,B$. On iteration t:

1. Compute $y_{\pi,t}$ for all $\pi\in\boldsymbol{\mathcal{S}}'$, using (7a)–(7e).
2. Define $\tilde{y}_{\pi,t}=y_{\pi,t}$ if $\pi\notin\boldsymbol{\mathcal{A}}$, and $\tilde{y}_{\pi,t}=y_{\pi,t}-\sum_{a\in A} z^a_{\pi,i,t}$ if $\pi\in\boldsymbol{\mathcal{A}}_i$ for some $i\in[n]$ (if $\pi\in\boldsymbol{\mathcal{A}}_i$, then $\{z^a_{\pi,i,t}: a\in A\}$ has already been set in a previous iteration).
3. For all $i\in[n]$, define $f_{i,t}=\sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=\rho_i}\tilde{y}_{\pi,t}$.
4. For all $i\in[n]$, $\pi\in\boldsymbol{\mathcal{S}}'$ such that $\pi_i=\rho_i$, and $a\in A$, set $z^a_{\pi,i,t}=\tilde{y}_{\pi,t}\cdot\frac{1}{2}\cdot(x^a_{\rho_i,t}/f_{i,t})$.
5. For all $i\in[n]$, and $\pi\in\boldsymbol{\mathcal{S}}'$ such that $\pi_i=\rho_i$ and $\pi_j\in\{\rho_j,\phi_j\}$ for $j\neq i$, define $g_{\pi,i,t}=\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi)} z_{\pi',i,t}$.
6. For all $i\in[n]$, $u\in\mathcal{S}_i\backslash\{\rho_i\}$, $\pi\in\boldsymbol{\mathcal{S}}'$ such that $\pi_i=u$, and $a\in A$, set $z^a_{\pi,i,t+\mathsf{depth}(u)}=g_{\pi^{\rho_i},i,t}\cdot(x^a_{u,t+\mathsf{depth}(u)}/x_{\rho_i,t})$.

In Step 2, $\tilde{y}_{\pi,t}$ represents the probability that we are at joint node π and looking to start a new arm at time t, abandoning the arm in progress if there is any. In Step 3, $f_{i,t}$ is the total probability of *being able to* start arm i at time t, which we define as arm i being available when we are looking to start a new arm at time t. The normalization in Step 4 ensures that each arm is started with the correct probability at time t. In Step 5, $g_{\pi,i,t}$ is the probability arm i is started at time t, and other arms are on nodes $\{\pi_j: j\neq i\}$ while arm i executes (another arm j could have made a transition to ϕ_j during the first time step t). Step 6 specifies how to continue playing arm i in subsequent time steps if it is started at time t. Note that $g_{\pi^{\rho_i},i,t}$ is guaranteed to be defined in this case, since $\pi_i\notin\{\rho_i,\phi_i\}$ and $\pi\in\boldsymbol{\mathcal{S}}'$ implies $\pi_j\in\{\rho_j,\phi_j\}$ for all $j\neq i$.

This completes the specification of the solution to (ExpLP'). Every $y_{\pi,t}$ is set in Step 1, and every $z^a_{\pi,i,t}$ is set in either Step 4 or Step 6.

Using the definition of $f_{i,t}$, Step 4 guarantees that for $i\in[n]$, $a\in A$,

$$\sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=\rho_i} z^a_{\pi,i,t} = \sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=\rho_i}\tilde{y}_{\pi,t}\cdot\frac{1}{2}\cdot\frac{x^a_{\rho_i,t}}{f_{i,t}} = \frac{x^a_{\rho_i,t}}{2}.$$

Meanwhile, Step 6 guarantees that for $i\in[n]$, $u\in\mathcal{S}_i\backslash\{\rho_i\}$, $a\in A$,

$$\sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=u} z^a_{\pi,i,t+\mathsf{depth}(u)} = \sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=u} g_{\pi^{\rho_i},i,t}\cdot\frac{x^a_{u,t+\mathsf{depth}(u)}}{x_{\rho_i,t}} = \frac{x_{\rho_i,t}}{2}\cdot\frac{x^a_{u,t+\mathsf{depth}(u)}}{x_{\rho_i,t}} = \frac{x^a_{u,t+\mathsf{depth}(u)}}{2}.$$

We explain the second equality. Since $u\neq\rho_i$ implies arm i is the active arm in all of $\{\pi\in\boldsymbol{\mathcal{S}}': \pi_i=u\}$, this set is equal to $\{\rho_1,\phi_1\}\times\cdots\times u\times\cdots\times\{\rho_n,\phi_n\}$. Summing $g_{\pi^{\rho_i},i,t}$ over all the possibilities for $\{\pi_j: j\neq i\}$ yields the total probability arm i is started at time t. This is equal to $\sum_{\pi\in\boldsymbol{\mathcal{S}}':\pi_i=\rho_i}\sum_{a\in A} z^a_{\pi,i,t}$, which by the first calculation is equal to $x_{\rho_i,t}/2$.

The proof of (14) is now complete.

3.3. Proof of Feasibility

At a high level, the main challenge in proving feasibility is verifying (6a), i.e., the total probability of plays scheduled for joint node π and time t does not exceed the probability of being at joint node π and time t. Given

the way the solution was constructed, it is mostly an analytical exercise (Lemma 3) to reduce (6a) to showing that the values of $f_{i,t}$ are sufficiently large. $f_{i,t}$ is the probability of being able to start arm i at time t, which we are not able to do if the current arm has not been prescribed to stop, or if arm i has already been played. Lemma 4 shows that the probability of these events occurring is just small enough when the probabilities of the LP relaxation, $x^a_{u,t}$, are scaled down by a factor of 2.

We will inductively prove feasibility over iterations $t = 1, \dots, B$. Suppose all of the variables $\{z^a_{\pi,i,t'}, y_{\pi,t'}\}$ with $t' < t$ have already been set in a way that satisfies constraints (6a)–(6c), (7a)–(7e). Some of the variables $z^a_{\pi,i,t'}$ with $t' \geq t$ may have also been set in Step 6 of earlier iterations; if so, suppose they have already been proven to satisfy (6c).

On iteration t, we first compute in Step 1 $y_{\pi,t}$ for all $\pi \in \boldsymbol{\mathcal{S}}'$; these are guaranteed to satisfy (7a)–(7e) by definition. To complete the induction, we need to show that (6a)–(6c) hold after setting the z-variables in Step 4, and furthermore, (6c) holds for any $z^a_{\pi,i,t'}$ (with $t' > t$) we set in Step 6.

We first prove the following lemma.

Lemma 3. *Suppose $\pi \in \boldsymbol{\mathcal{A}}_i$ for some $i \in [n]$. Let $u = \pi_i$ (which is neither ρ_i nor ϕ_i). Then $\sum_{a \in A} z^a_{\pi,i,t} \leq y_{\pi,t}$, and furthermore if $u \in \mathcal{B}$, then $z^\alpha_{\pi,i,t} = y_{\pi,t}$.*

Proof. First suppose $\mathsf{depth}(u) = 1$. (7d) says $y_{\pi,t} = \sum_{a:(\rho_i,a)\in\mathsf{Par}(u)} \left(\sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} z^a_{\pi',i,t-1}\right) \cdot p^a_{\rho_i,u}$. Every π' in the sum has $\pi'_i = \rho_i$, so $z^a_{\pi',i,t-1}$ was set in Step 4 of iteration $t-1$ to $\tilde{y}_{\pi',t-1} \cdot \frac{1}{2} \cdot (x^a_{\rho_i,t-1}/f_{i,t-1})$. Substituting into (7d), we get

$$\begin{aligned} y_{\pi,t} &= \sum_{a:(\rho_i,a)\in\mathsf{Par}(u)} \left(\sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} \tilde{y}_{\pi',t-1} \cdot \frac{1}{2} \cdot \frac{x^a_{\rho_i,t-1}}{f_{i,t-1}} \right) \cdot p^a_{\rho_i,u} \\ &= \left(\sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} \tilde{y}_{\pi',t-1} \right) \cdot \frac{1}{2 f_{i,t-1}} \cdot \sum_{a:(\rho_i,a)\in\mathsf{Par}(u)} x^a_{\rho_i,t-1} \cdot p^a_{\rho_i,u}. \end{aligned}$$

Meanwhile, for all $a \in A$, $z^a_{\pi,i,t}$ was set in Step 6 of iteration $t-1$ to $g_{\pi^{\rho_i},i,t-1} \cdot (x^a_{u,t}/x_{\rho_i,t-1})$. Hence,

$$\begin{aligned} z^a_{\pi,i,t} &= g_{\pi^{\rho_i},i,t-1} \cdot \frac{x^a_{u,t}}{x_{\rho_i,t-1}} = \sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} z_{\pi',i,t-1} \cdot \frac{x^a_{u,t}}{x_{\rho_i,t-1}} \\ &= \sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} \left(\sum_{b \in A} \tilde{y}_{\pi',t-1} \cdot \frac{1}{2} \cdot \frac{x^b_{\rho_i,t-1}}{f_{i,t-1}} \right) \cdot \frac{x^a_{u,t}}{x_{\rho_i,t-1}} = \left(\sum_{\pi' \in \boldsymbol{\mathcal{P}}(\pi^{\rho_i})} \tilde{y}_{\pi',t-1} \right) \cdot \frac{1}{2 f_{i,t-1}} \cdot x^a_{u,t}, \end{aligned}$$

where the second equality is by the definition of $g_{\pi^{\rho_i},i,t-1}$, and the third equality uses the fact that $z^b_{\pi',i,t-1}$ was set in Step 4 of iteration $t-1$. To prove $\sum_{a\in A} z^a_{\pi,i,t} \leq y_{\pi,t}$, it suffices to show $\sum_{a\in A} x^a_{u,t} \leq \sum_{a:(\rho_i,a)\in\mathsf{Par}(u)} x^a_{\rho_i,t-1} \cdot p^a_{\rho_i,u}$. This follows immediately from combining constraints (10a) and (13d) of (PolyLP'). Furthermore, if $u \in \mathcal{B}$, then we can use (10b) to get $z^\alpha_{\pi,i,t} = y_{\pi,t}$.

Now suppose $\mathsf{depth}(u) > 1$. (7e) says $y_{\pi,t} = \sum_{(v,a)\in\mathsf{Par}(u)} z^a_{\pi^v,i,t-1} \cdot p^a_{v,u}$. Since $v \neq \rho_i$, $z^a_{\pi^v,i,t-1}$ was set in Step 6 of iteration $t' := t - \mathsf{depth}(u)$ to $g_{\pi^{\rho_i},i,t'} \cdot (x^a_{v,t-1}/x_{\rho_i,t'})$. Substituting into (7e), we get

$$y_{\pi,t} = \sum_{(v,a)\in\mathsf{Par}(u)} g_{\pi^{\rho_i},i,t'} \cdot \frac{x^a_{v,t-1}}{x_{\rho_i,t'}} \cdot p^a_{v,u} = \frac{g_{\pi^{\rho_i},i,t'}}{x_{\rho_i,t'}} \sum_{(v,a)\in\mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u}.$$

Meanwhile, for all $a \in A$, $z^a_{\pi,i,t}$ was set in Step 6 of iteration t' to $g_{\pi^{\rho_i},i,t'} \cdot (x^a_{u,t}/x_{\rho_i,t'})$. To prove $\sum_{a\in A} z^a_{\pi,i,t} \leq y_{\pi,t}$, it suffices to show $\sum_{a\in A} x^a_{u,t} \leq \sum_{(v,a)\in\mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u}$. This is again obtained from (10a) and (13d), and if $u \in \mathcal{B}$, then we can use (10b) to get $z^\alpha_{\pi,i,t} = y_{\pi,t}$. □

By the lemma, $\tilde{y}_{\pi,t} \geq 0$ if $\pi \in \boldsymbol{\mathcal{A}}_i$ for some $i \in [n]$. On the other hand, $\tilde{y}_{\pi,t} \geq 0$ is immediate from definition if $\pi \notin \boldsymbol{\mathcal{A}}$. Therefore, $\tilde{y}_{\pi,t} \geq 0$ for all $\pi \in \boldsymbol{\mathcal{S}}'$, and (6c) is satisfied by all the z-variables set in Step 4 or Step 6. Furthermore, the lemma guarantees (6b) for the $z^a_{\pi,i,t}$ with $\pi_i \in \mathcal{B}$ set in previous iterations.

It remains to prove (6a). If $\pi \in \boldsymbol{\mathcal{A}}_i$, then the LHS of (6a) is

$$\sum_{j \in I(\pi)} z_{\pi,j,t} = z_{\pi,i,t} + \sum_{j \in I(\pi)\setminus\{i\}} \tilde{y}_{\pi,t} \cdot \frac{1}{2} \cdot \frac{x_{\rho_j,t}}{f_{j,t}} = z_{\pi,i,t} + (y_{\pi,t} - z_{\pi,i,t}) \cdot \sum_{j \in I(\pi)\setminus\{i\}} \frac{1}{2} \cdot \frac{x_{\rho_j,t}}{f_{j,t}}.$$

For the first equality, note that $z_{\pi,j,t}$ for $j \neq i$ is set in Step 4 of the current iteration, but $z_{\pi,i,t}$ has already been set in an earlier iteration. The second equality is immediate from the definition of $\tilde{y}_{\pi,t}$. Note that $y_{\pi,t} - z_{\pi,i,t} \geq 0$, by Lemma 3. If we knew $\sum_{j\in I(\pi)\setminus\{i\}} \frac{1}{2} \cdot (x_{\rho_j,t}/f_{j,t}) \leq 1$, then we would have $\sum_{j\in I(\pi)} z_{\pi,j,t} \leq z_{\pi,i,t} + (y_{\pi,t} - z_{\pi,i,t})(1) = y_{\pi,t}$, which is (6a).

On the other hand, if $\pi \notin \boldsymbol{\mathcal{A}}$, then the LHS of (6a) is $y_{\pi,t} \cdot \sum_{j \in I(\pi)} \frac{1}{2} \cdot (x_{\rho_j,t}/f_{j,t})$, where in this case all of the $z_{\pi,j,t}$ are set in Step 4 of the current iteration. Similarly, if we knew $\sum_{j \in I(\pi)} \frac{1}{2} \cdot (x_{\rho_j,t}/f_{j,t}) \le 1$, then we would have (6a).

To complete the proof of feasibility, it suffices to show $\sum_{j=1}^{n} \frac{1}{2} \cdot (x_{\rho_j,t}/f_{j,t}) \le 1$ (note that $f_{j,t}$ is always nonnegative, by its definition and Lemma 3). This is implied by the following lemma, which proves a simpler statement.

Lemma 4. *$f_{i,t} \ge \sum_{j=1}^{n} (x_{\rho_j,t}/2)$ for all $i \in [n]$.*

Proof. Fix some $i \in [n]$. By the definitions in Steps 2 and 3,

$$f_{i,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}' : \pi_i = \rho_i} y_{\pi,t} - \sum_{j \ne i} \sum_{\pi \in \boldsymbol{\mathcal{A}}_j : \pi_i = \rho_i} z_{\pi,j,t}.$$

Let's start by bounding $\sum_{\pi \in \boldsymbol{\mathcal{S}}' : \pi_i = \rho_i} y_{\pi,t}$, the total probability arm i is still on ρ_i at the start of time t. This is equal to $1 - \sum_{t' < t} \sum_{\pi \in \boldsymbol{\mathcal{S}}' : \pi_i = \rho_i} z_{\pi,i,t'}$, where we subtract from 1 the total probability arm i was initiated before time t. By (14), $\sum_{\pi \in \boldsymbol{\mathcal{S}}' : \pi_i = \rho_i} z_{\pi,i,t'} = x_{\rho_i,t'}/2$ for all $t' < t$. Furthermore,

$$\sum_{t' < t} x_{\rho_i,t'} \le 1 \tag{15}$$

from iteratively applying (13c) to (13a), and combining with (10a).[8] Therefore, $\sum_{\pi \in \boldsymbol{\mathcal{S}}' : \pi_i = \rho_i} y_{\pi,t} \ge \frac{1}{2}$.

Now we bound the remaining term in the equation for $f_{i,t}$:

$$\begin{aligned} \sum_{j \ne i} \sum_{\pi \in \boldsymbol{\mathcal{A}}_j : \pi_i = \rho_i} z_{\pi,j,t} &= \sum_{j \ne i} \sum_{v \in \mathcal{S}_j \setminus \{\rho_j\}} \sum_{\pi : \pi_i = \rho_i, \pi_j = v} z_{\pi,j,t} \le \sum_{j \ne i} \sum_{v \in \mathcal{S}_j \setminus \{\rho_j\}} \sum_{\pi : \pi_j = v} z_{\pi,j,t} \\ &= \frac{1}{2} \sum_{j \ne i} \sum_{v \in \mathcal{S}_j \setminus \{\rho_j\}} x_{v,t} \le \frac{1}{2} \sum_{j=1}^{n} \sum_{v \in \mathcal{S}_j \setminus \{\rho_j\}} x_{v,t} \le \frac{1}{2} \left(1 - \sum_{j=1}^{n} x_{\rho_j,t} \right). \end{aligned}$$

The first inequality uses the nonnegativity of $z_{\pi,j,t}$ in the inductively proven (6c), the second equality uses (14), the second inequality uses the nonnegativity of $x_{v,t}$ in (10c), and the third inequality uses (11).

Combining the two terms, we get $f_{i,t} \ge \sum_{j=1}^{n} (x_{\rho_j,t}/2)$, as desired. □

3.4. Approximation Algorithm via Sampling

We would like to infer a polynomial-time policy from this exponential-sized solution $\{z^a_{\pi,i,t}, y_{\pi,t}\}$ of (ExpLP'). It is not possible to merely compute the values of $z^a_{\pi,i,t}$ on the realized sample path, because $z^a_{\pi,i,t}$ depends on $f_{i,t}$, the total probability of being able to start arm i at time t over exponentially many states. To overcome this challenge, at each time step t, we *sample* (run the algorithm up to time t a large number of times, realizing new transitions each time) to estimate the values of $f_{i,t}$, before making a decision. We record the sampling results and the decisions prescribed by such, since future sampling depends on past algorithm decisions.

Hereinafter we will assume that the $\{x^a_{u,t}, s_{u,t}\}$ we are imitating is an *optimal* solution of (PolyLP'). Consider the following algorithm, which takes in as parameters a terminal time step $t \in [B]$, and probabilities $\lambda_{i,t'}$ for each $i \in [n]$, $t' \le t$ (which for now should be considered to be $f_{i,t'}$ to aid in the comprehension of the algorithm).

Policy$(t, \{\lambda_{i,t'} : i \in [n], t' \le t\})$

- Initialize $t' = 1$, current $= 0$.
- While $t' \le t$:

1. If current $= 0$, then

(a) For each arm i that is on ρ_i, set current $= i$ with probability $\frac{1}{2} \cdot (x_{\rho_i,t'}/\lambda_{i,t'})$; if the sum of these probabilities exceeds 1 (i.e., this step is inadmissible), then terminate with no reward.

(b) If current was set in this way, leave t' unchanged and enter the next if block. Otherwise, leave current at 0 but increment t' by 1.

2. If current $\ne 0$, then

(a) Let u denote the node arm current is on. For each $a \in A$, play action a on arm current with probability $x^a_{u,t'}/x_{u,t'}$.

(b) Suppose we transition onto node v as a result of this play. With probability $x_{v,t'+1}/s_{v,t'+1}$, leave current unchanged. Otherwise, set current $= 0$.

(c) Increment t' by 1.

Define the following events and probabilities, which depend on the input passed into Policy:

- For all $i \in [n], t' \le (t+1)$, let $\mathcal{A}_{i,t'}$ be the event that at the beginning of time t', current $= 0$ and arm i is on ρ_i. Let Free$(i,t') = \Pr[\mathcal{A}_{i,t'}]$.

- For all $i \in [n], t' \le t$, let $\mathsf{Started}(i,t')$ be the probability that we play arm i from ρ_i at time t'.
- For all $u \in \mathcal{S}, a \in A, t' \le t$, let $\mathsf{Played}(u,a,t')$ be the probability that we play action a on node u at time t'.

It is easy to see that Policy is an algorithmic specification of feasible solution $\{z^a_{\pi,i,t}, y_{\pi,t}\}$ if we run it on input $(B, \{f_{i,t}: i \in [n], t \in [B]\})$. Indeed, we would iteratively have for $t = 1, \dots, B$:

- $\mathsf{Free}(i,t) = f_{i,t}$ for all $i \in [n]$
- $\mathsf{Started}(i,t) = \mathsf{Free}(i,t) \cdot \frac{1}{2} \cdot (x_{\rho_i,t}/f_{i,t}) = x_{\rho_i,t}/2$ for all $i \in [n]$
- $\mathsf{Played}(u,a,t) = \mathsf{Started}(i, t - \mathsf{depth}(u)) \cdot x^a_{u,t}/x_{\rho_i, t-\mathsf{depth}(u)} = x^a_{u,t}/2$ for all $u \in \mathcal{S}$, $a \in A$.

The final statement can be seen inductively:

$$\begin{aligned}\mathsf{Played}(u,a,t) &= \sum_{(v,b)\in\mathsf{Par}(u)} \mathsf{Played}(v,b,t-1) \cdot p^b_{v,u} \cdot \frac{x^a_{u,t}}{s_{u,t}} \\ &= \sum_{(v,b)\in\mathsf{Par}(u)} \left(\mathsf{Started}(i, t-1-\mathsf{depth}(v)) \cdot \frac{x^b_{v,t-1}}{x_{\rho_i,t-1-\mathsf{depth}(v)}}\right) \cdot p^b_{v,u} \cdot \frac{x^a_{u,t}}{s_{u,t}} \\ &= \mathsf{Started}(i, t-\mathsf{depth}(u)) \cdot \frac{x^a_{u,t}}{x_{\rho_i,t-\mathsf{depth}(u)}}, \end{aligned} \tag{16}$$

where the first equality is by Steps 2a–2b, the second equality is by the induction hypothesis, and the final equality is by (13d).

Therefore, we would have a $\frac{1}{2}$-approximation if we knew $\{f_{i,t}: i \in [n], t \in [B]\}$, but unfortunately computing $f_{i,t}$ requires summing exponentially many terms. We can try to approximate it by sampling, but we cannot even generate a sample from the binary distribution with probability $f_{i,t}$ since that requires knowing the exact values of $f_{i,t'}$ for $t' < t$. So we give up trying to approximate $f_{i,t}$, and instead iteratively approximate the values of $\mathsf{Free}(i,t)$ when Policy is ran on previously approximated $\mathsf{Free}(i,t)$ values.

Fix some small $\varepsilon, \delta > 0$ that will be determined later. Let $\mu_{\varepsilon,\delta} = 3\ln(2\delta^{-1})/\varepsilon^2$. Change Policy so that the probabilities in Step 1a are multiplied by $(1-\varepsilon)^2$ (and change the definitions of $\mathcal{A}_{i,t'}$, Free, Started, Played accordingly).

Sampling Algorithm

- Initialize $\mathsf{Free}^{\mathsf{emp}}(i,1) = 1$ for all $i \in [n]$.
- For $t = 2, \dots, B$:
  1. Run $\mathsf{Policy}(t-1, \{\mathsf{Free}^{\mathsf{emp}}(i,t'): i \in [n], t' < t\})$ a total of $M = ((8|\mathcal{S}|B)/\varepsilon) \cdot \mu_{\varepsilon,\delta}$ times. For all $i \in [n]$, let $C_{i,t}$ count the number of times event $\mathcal{A}_{i,t}$ occurred.
- For each $i \in [n]$, if $C_{i,t} > \mu_{\varepsilon,\delta}$, set $\mathsf{Free}^{\mathsf{emp}}(i,t) = C_{i,t}/M$; otherwise set $\mathsf{Free}^{\mathsf{emp}}(i,t) = \sum_{j=1}^n (x_{\rho_j,t}/2)$.

Consider iteration t of Sampling Algorithm. $\{\mathsf{Free}^{\mathsf{emp}}(i,t'): i \in [n], t' < t\}$ have already been finalized, and we are sampling event $\mathcal{A}_{i,t}$ when (the ε-modified) Policy is ran on those finalized approximations to record values for $\{\mathsf{Free}^{\mathsf{emp}}(i,t): i \in [n]\}$. For all $i \in [n]$, if $C_{i,t} > \mu_{\varepsilon,\delta}$, then the probability of $C_{i,t}/M$ lying in $((1-\varepsilon)\cdot\mathsf{Free}(i,t), (1+\varepsilon)\cdot\mathsf{Free}(i,t))$ is at least[9] $1-\delta$. As far as when we have $C_{i,t} > \mu_{\varepsilon,\delta}$, note that if $\mathsf{Free}(i,t) > \varepsilon/(4|\mathcal{S}|B)$, then $\mathbb{E}[C_{i,t}] > 2\mu_{\varepsilon,\delta}$; so the Chernoff bound says $\Pr[C_{i,t} \le \mu_{\varepsilon,\delta}] = O(\delta^{1/\varepsilon^2}) = O(\delta)$. We have discussed two $O(\delta)$ probability events in this paragraph of sampling/Chernoff yielding an unlikely and undesired result; call these events *failures*.

By the union bound, the probability of having any failure over iterations $t = 2, \dots, B$ is at most $2(B-1)\cdot n(\delta + O(\delta)) = O(Bn\delta)$. Assuming no failures, we will inductively prove

$$\frac{(1-\varepsilon)^2}{1+\varepsilon} \cdot \frac{x_{\rho_i,t}}{2} \le \mathsf{Started}(i,t) \le \max\left\{(1-\varepsilon)\cdot\frac{x_{\rho_i,t}}{2}, \frac{\varepsilon}{4|\mathcal{S}|B}\right\} \tag{17}$$

for all $i \in [n]$. This is clear when $t = 1$ since $\mathsf{Started}(i,1) = x_{\rho_i,1}/2$ exactly for all $i \in [n]$.

Now suppose $t \ge 2$. We will first prove a lemma on the true probabilities $\mathsf{Free}(i,t)$, which is the "approximate" version of Lemma 4.

Lemma 5. *Suppose* Policy *is ran on input* $(t-1, \{\mathsf{Free}^{\mathsf{emp}}(i,t'): i \in [n], t' < t\})$ *and there were no failures while obtaining the sample average approximations* $\mathsf{Free}^{\mathsf{emp}}(i,t')$. *Then, for all* $i \in [n]$, $\mathsf{Free}(i,t) \ge \frac{1}{2}\sum_{j=1}^n x_{\rho_j,t}$.

Proof. We know that event $\mathcal{A}_{i,t}$ will occur if at time t, arm i has not yet been started, and no other arm is active. By the union bound, $1 - \mathsf{Free}(i,t) \le \sum_{t'<t} \mathsf{Started}(i,t') + \sum_{j=1}^n \sum_{u\in\mathcal{S}_j\setminus\{\rho_j\}} \sum_{a\in A} \mathsf{Played}(u,a,t)$. Assuming (17) holds, we can bound

$$\sum_{t'<t} \mathsf{Started}(i,t') \le \sum_{t'<t}\left((1-\varepsilon)\cdot\frac{x_{\rho_i,t'}}{2} + \frac{\varepsilon}{4|\mathcal{S}|B}\right) \le \frac{1-\varepsilon}{2}\sum_{t'<t} x_{\rho_i,t'} + \frac{\varepsilon}{4|\mathcal{S}|} \le \frac{1-\varepsilon}{2} + \frac{\varepsilon}{4}$$

where the final inequality uses (15). Similarly, assuming (17) holds, we can bound

$$\begin{aligned}\sum_{j=1}^{n}\sum_{u\in\mathcal{S}_j\setminus\{\rho_j\}}\sum_{a\in A}\mathsf{Played}(u,a,t) &= \sum_{j=1}^{n}\sum_{u\in\mathcal{S}_j\setminus\{\rho_j\}}\mathsf{Started}(i,t-\mathsf{depth}(u))\cdot\frac{x_{u,t}}{x_{\rho_i,t-\mathsf{depth}(u)}}\\ &\le \sum_{j=1}^{n}\sum_{u\in\mathcal{S}_j\setminus\{\rho_j\}}\left((1-\varepsilon)\cdot\frac{x_{\rho_i,t-\mathsf{depth}(u)}}{2}+\frac{\varepsilon}{4|\mathcal{S}|B}\right)\cdot\frac{x_{u,t}}{x_{\rho_i,t-\mathsf{depth}(u)}}\\ &\le \frac{1-\varepsilon}{2}\sum_{j=1}^{n}\sum_{u\in\mathcal{S}_j\setminus\{\rho_j\}}x_{u,t}+\frac{\varepsilon}{4B}\le\frac{1}{2}\left(1-\sum_{j=1}^{n}x_{\rho_j,t}\right)+\frac{\varepsilon}{4}\end{aligned}$$

where the equality uses (16), the second inequality uses the fact that $x_{u,t}\le x_{\rho_i,t-\mathsf{depth}(u)}$, and the final inequality uses (11). Combining these bounds completes the proof of the lemma. □

By the description in Step 1a of Policy, for all $i\in[n]$, we have

$$\mathsf{Started}(i,t)=\mathsf{Free}(i,t)\cdot\frac{1}{2}\cdot\frac{x_{\rho_i,t}}{\mathsf{Free}^{\mathsf{emp}}(i,t)}\cdot(1-\varepsilon)^2. \tag{18}$$

If $C_{i,t}>\mu_{\varepsilon,\delta}$, then $\mathsf{Free}^{\mathsf{emp}}(i,t)$ will be set to $C_{i,t}/M$, and furthermore no failures implies $(1-\varepsilon)\cdot\mathsf{Free}(i,t)\le C_{i,t}/M\le(1+\varepsilon)\cdot\mathsf{Free}(i,t)$. Substituting into (18), we get $((1-\varepsilon)^2/(1+\varepsilon))\cdot(x_{\rho_i,t}/2)\le\mathsf{Started}(i,t)\le(1-\varepsilon)\cdot(x_{\rho_i,t}/2)$ which implies (17). On the other hand, if $C_{i,t}\le\mu_{\varepsilon,\delta}$, then $\mathsf{Free}^{\mathsf{emp}}(i,t)$ will be set to $\sum_{j=1}^{n}(x_{\rho_j,t}/2)$, and assuming no failures, it must have been the case that $\mathsf{Free}(i,t)\le\varepsilon/(4|\mathcal{S}|B)$. Substituting into (18), we get $\mathsf{Started}(i,t)\le(\varepsilon/(4|\mathcal{S}|B))\cdot\frac{1}{2}\cdot(x_{\rho_i,t}/\sum_{j=1}^{n}x_{\rho_j,t})\cdot(1-\varepsilon)^2\le\varepsilon/(4|\mathcal{S}|B)$ which implies the upper bound in (17). For the lower bound, Lemma 5 says $\mathsf{Free}(i,t)\ge\mathsf{Free}^{\mathsf{emp}}(i,t)$, so $\mathsf{Started}(i,t)\ge(1-\varepsilon)^2\cdot(x_{\rho_i,t}/2)\ge((1-\varepsilon)^2/(1+\varepsilon))\cdot(x_{\rho_i,t}/2)$.

This completes the induction for (17). The final thing to check is that with these new parameters $\{\mathsf{Free}^{\mathsf{emp}}(i,t)\colon i\in[n]\}$, the sum of the probabilities in Step 1a of Policy does not exceed 1. $\mathsf{Free}^{\mathsf{emp}}(i,t)$ will either get set to $\sum_{j=1}^{n}(x_{\rho_j,t}/2)$, or be at least $(1-\varepsilon)\cdot\mathsf{Free}(i,t)$, which is at least $(1-\varepsilon)\cdot\sum_{j=1}^{n}(x_{\rho_j,t}/2)$ by Lemma 5. In either case, $\mathsf{Free}^{\mathsf{emp}}(i,t)\ge(1-\varepsilon)\cdot\sum_{j=1}^{n}(x_{\rho_j,t}/2)$ for all $i\in[n]$, so the desired sum in Step 1a is at most $(1/(1-\varepsilon)\cdot(1-\varepsilon)^2\le 1$.

We have an algorithm that fails with probability $O(Bn\delta)$, and when it doesn't fail, $\mathsf{Started}(i,t)\ge((1-\varepsilon)^2/(1+\varepsilon))\cdot(x_{\rho_i,t}/2)$ for all $i\in[n],t\in[B]$, which in conjunction with (16) shows that we obtain expected reward at least $((1-\varepsilon)^2/(1+\varepsilon))\cdot\frac{1}{2}\cdot\mathsf{OPT}_{\mathtt{PolyLP'}}$. Recall from Lemma 2 that $\mathsf{OPT}_{\mathtt{PolyLP'}}\ge\mathsf{OPT}_{\mathtt{ExpLP'}}$. Treating a failed run as a run with 0 reward, we can set $\delta=\Theta(\varepsilon/Bn)$ to get a $(\frac{1}{2}-\varepsilon)$-approximation. Finally, note that the runtime of this approximation algorithm is polynomial in the input, $1/\varepsilon$, and $\ln(1/\delta)$, completing the proof of Theorem 1.

4. Proof of Theorem 2

In this section, we prove Theorem 2, and also show how to modify the proof to prove Theorem 3.

4.1. Description of Algorithm

The high-level description of the algorithm is as follows. A priority index, which is a time step t in $[B]$, is maintained for each arm. To start, the algorithm plays the arm, say i, with the lowest priority index.[10] Arm i will make a transition and its index will evolve. Arm i is played until it reaches a point where its index is much greater than the depth of the node it is on. Once this occurs, the algorithm switches to the arm that now has the smallest index, and repeats this process. Recall that switching back and forth between arms, i.e., preemption, is necessary for an algorithm to be within a constant factor of optimality. On the other hand, the constraint based on depth ensures that the algorithm does not switch away after an arm has been played a large number of times to reach a high-reward node. Our priority-based policy is motivated by the ideas from Gupta et al. [19, sections 4–5].

Fix an optimal solution $\{x^a_{u,t},s_{u,t}\}$ to (PolyLP). The priority indices are maintained based on this solution. For an arm on node u, it will always have some *status* (u,a,t), which says that the next play of the arm should be with action a, and that this play has priority t. We allow $t=\infty$ to indicate that the algorithm will never try to play the arm again; in this case we omit the action argument.

We initialize each arm i to status (ρ_i,a,t) with probability $x^a_{\rho_i,t}/C$, for all $a\in A$ and $t\in[B]$, where $C>0$ is some constant to be optimized later. With probability $1-\sum_{a\in A}\sum_{t=1}^{B}(x^a_{\rho_i,t}/C)$, the arm is initialized to status (ρ_i,∞) and never touched; note that this probability is at least $1-1/C$.

If we play an arm and it transitions to node u, we need to decide what status (u,a,t) to put that arm in. For all $i\in[n]$, $u\in\mathcal{S}_i\setminus\{\rho_i\}$, $a\in A$, $t\in[B]$, $(v,b)\in\mathsf{Par}(u)$, and $t'<t$, we prescribe a probability $q_{v,b,t',u,a,t}$ with

which we will put arm i into status (u,a,t), conditioned on arriving at node u after playing action b on node v at priority t'. The evolution of statuses is independent of other arms. The following lemma shows that it is possible to solve for values of $q_{v,b,t',u,a,t}$ which are feasible, and respect the values of $x^a_{u,t}$:

Lemma 6. *Suppose we are given the x's of a feasible solution to* (PolyLP). *Then we can find* $\{q_{v,b,t',u,a,t}: u \in \mathcal{S}\backslash\{\rho_1,\dots,\rho_n\}, a \in A, t \in [B], (v,b) \in \mathsf{Par}(u), t' < t\}$ *in polynomial time such that*

$$\sum_{a\in A}\sum_{t:t>t'} q_{v,b,t',u,a,t} \le 1, \qquad u \in \mathcal{S}\backslash\{\rho_1,\dots,\rho_n\},\ (v,b)\in\mathsf{Par}(u),\ t' \in [B-1], \tag{19a}$$

$$\sum_{(v,b)\in\mathsf{Par}(u)}\sum_{t':t'<t} x^b_{v,t'}\cdot p^b_{v,u}\cdot q_{v,b,t',u,a,t} = x^a_{u,t}, \qquad u \in \mathcal{S}\backslash\{\rho_1,\dots,\rho_n\},\ a\in A,\ t\in\{2,\dots,B\}. \tag{19b}$$

Furthermore, if $u \in \mathcal{B}$, *then we can strengthen* (19a) *to* $q_{v,b,t',u,\alpha,t'+1} = 1$.

Inequalities (19a) ensures that the probabilities telling us what to do, when we arrive at node u after playing action b on node v at time t', are well-defined; the case where u is a bridge node will be needed to prove Theorem 3. For all $i \in [n]$, $u \in \mathcal{S}_i\backslash\{\rho_i\}$, $(v,b) \in \mathsf{Par}(u)$, and $t' \in [B-1]$, define $q_{v,b,t',u,\infty} = 1 - \sum_{a\in A}\sum_{t>t'} q_{v,b,t',u,a,t}$, the probability we abandon arm i after making the transition to u.

At a high level, Lemma 6 is a flow decomposition result and our analogue to the *convex decomposition* from Gupta et al. [19]. It says that for each arm i, $\{x^a_{u,t} : u \in \mathcal{S}_i, t \in [B], a \in A\}$ is a flow satisfying precedence constraints on both nodes u and times t, and can be decomposed into "local instructions on each transition" (i.e., what to do upon arriving at u after playing b on v at priority t') to reconstruct the original flow. Its proof is deferred to Appendix D.

Having defined the values of $q_{v,b,t',u,a,t}$, the overall algorithm can now be described in two steps:

1. While there exists an arm with priority not ∞, play an arm with the smallest index (breaking ties arbitrarily) until it arrives at a status (u,a,t) such that $t \ge 2\cdot\mathsf{depth}(u)$ ($t=\infty$ would suffice).
2. Repeat until all arms have priority ∞.

To clarify Step 1, once we start playing a lowest-index arm, its index will evolve and increase. However, we do not pause playing it once its index is no longer the lowest; instead, we pause playing it once it reaches a combination of index t and node u such that $t \ge 2\cdot\mathsf{depth}(u)$.

We are also constrained by a budget of B time steps, but it will simplify the analysis to assume our algorithm finishes all the arms and collects reward only for plays up to time B. Under this assumption, the statuses an arm goes through is independent of the outcomes on all other arms; the inter-dependence only affects the order in which arms are played (and thus which nodes obtain reward).

Also, note that this is only a valid algorithm because Theorem 2 assumes all processing times are 1, so there are no bridge nodes. If there were bridge nodes, then we may not be allowed to switch to an arm with lowest index, being forced to play the arm on a bridge node.

4.2. Analysis of Algorithm

For all $i \in [n]$, $u \in \mathcal{S}_i$, $a \in A$, $t \in [B]$, let $\mathsf{time}(u,a,t)$ be the random variable for the time step at which our algorithm plays arm i from status (u,a,t), with $\mathsf{time}(u,a,t) = \infty$ if arm i never gets in status (u,a,t). Then $\Pr[\mathsf{time}(\rho_i,a,t) < \infty] = x^a_{\rho_i,t}/C$ for all $i \in [n]$, $a \in A$, $t \in [B]$. If u is a nonroot node, then we can induct on $\mathsf{depth}(u)$ to prove for all $a \in A$, $t \in [B]$ that

$$\begin{aligned}\Pr[\mathsf{time}(u,a,t) < \infty] &= \sum_{(v,b)\in\mathsf{Par}(u)}\sum_{t'<t}\Pr[\mathsf{time}(v,b,t') < \infty]\cdot p^b_{v,u}\cdot q_{v,b,t',u,a,t}\\ &= \sum_{(v,b)\in\mathsf{Par}(u)}\sum_{t'<t}\frac{x^b_{v,t'}}{C}\cdot p^b_{v,u}\cdot q_{v,b,t',u,a,t} = \frac{x^a_{u,t}}{C}\end{aligned} \tag{20}$$

where the final equality follows from Lemma 6.

For an event $\mathcal{A}$, let $\mathbb{1}_{\mathcal{A}}$ be the indicator random variable for $\mathcal{A}$. The expected reward obtained by our algorithm is

$$\begin{aligned}\mathbb{E}\left[\sum_{u\in\mathcal{S}}\sum_{a\in A} r^a_u \sum_{t=1}^{B}\mathbb{1}_{\{\mathsf{time}(u,a,t)\le B\}}\right] &= \sum_{u\in\mathcal{S}}\sum_{a\in A} r^a_u\sum_{t=1}^{B}\mathbb{E}[\mathbb{1}_{\{\mathsf{time}(u,a,t)\le B\}} \mid \mathsf{time}(u,a,t)<\infty]\cdot\Pr[\mathsf{time}(u,a,t)<\infty]\\ &= \sum_{u\in\mathcal{S}}\sum_{a\in A} r^a_u\sum_{t=1}^{B}\Pr[\mathsf{time}(u,a,t)\le B \mid \mathsf{time}(u,a,t)<\infty]\cdot\frac{x^a_{u,t}}{C}.\end{aligned}$$

For the remainder of this subsection, we will set $C = 3$ and prove for an arbitrary $i \in [n]$, $u \in \mathcal{S}_i$, $a \in A$, $t \in [B]$ that $\Pr[\mathsf{time}(u,a,t) \le B \mid \mathsf{time}(u,a,t) < \infty] \ge \frac{4}{9}$. It suffices to prove that $\Pr[\mathsf{time}(u,a,t) \le t \mid \mathsf{time}(u,a,t) < \infty] \ge \frac{4}{9}$, since $t \le B$.

Case 1. Suppose $t \ge 2 \cdot \mathsf{depth}(u)$. We prove that conditioned on the event $\{\mathsf{time}(u,a,t) < \infty\}$, $\{\mathsf{time}(u,a,t) > t\}$ occurs with probability at most $\frac{5}{9}$.

Note that every node v can have at most one b, t' such that $\mathsf{time}(v,b,t') < \infty$; let $\mathsf{time}(v)$ denote this quantity (and be ∞ if $\mathsf{time}(v,b,t') = \infty$ for all $b \in A$, $t' \in [B]$). The nodes v that are played before u are those with $\mathsf{time}(v) < \mathsf{time}(u,a,t)$. Since our algorithm plays a node at every time step, $\mathsf{time}(u,a,t) > t$ if and only if there are t or more nodes $v \ne u$ such that $\mathsf{time}(v) < \mathsf{time}(u,a,t)$. But this is equivalent to there being exactly t nodes $v \ne u$ such that $\mathsf{time}(v) < \mathsf{time}(u,a,t)$ and $\mathsf{time}(v) \le t$. The $\mathsf{depth}(u)$ ancestors of u are guaranteed to satisfy this.

Hence, the event $\{\mathsf{time}(u,a,t) > t\}$ is equivalent to $\{\mathsf{depth}(u) + \sum_{v \in \mathcal{S} \setminus \mathcal{S}_i} \mathbb{1}_{\{\mathsf{time}(v) < \mathsf{time}(u,a,t)\}} \cdot \mathbb{1}_{\{\mathsf{time}(v) \le t\}} = t\}$. But $t \ge 2 \cdot \mathsf{depth}(u)$, so this implies $\{\sum_{v \in \mathcal{S} \setminus \mathcal{S}_i} \mathbb{1}_{\{\mathsf{time}(v) < \mathsf{time}(u,a,t)\}} \cdot \mathbb{1}_{\{\mathsf{time}(v) \le t\}} \ge t/2\} \implies \{\sum_{v \in \mathcal{S} \setminus \mathcal{S}_i} \mathbb{1}_{\{\mathsf{time}(v) < \mathsf{time}(u,a,t)\}} \ge t/2\}$. Now, whether the sum is at least $t/2$ is unchanged if we exclude all v such that $\mathsf{depth}(v) \ge t/2$. Indeed, if any such v satisfies $\mathsf{time}(v) < \mathsf{time}(u,a,t)$, then all of its ancestors also do, and its first $\lceil t/2 \rceil$ ancestors ensure that the sum, without any nodes of depth at least $t/2$, is at least $t/2$. Thus, the last event is equivalent to

$$\left\{ \sum_{v \in \mathcal{S} \setminus \mathcal{S}_i : \mathsf{depth}(v) < t/2} \mathbb{1}_{\{\mathsf{time}(v) < \mathsf{time}(u,a,t)\}} \ge \frac{t}{2} \right\}. \tag{21}$$

Suppose $\mathsf{time}(v) = \mathsf{time}(v,b,t')$ for some $b \in A$ and $t' \in [B]$. We would like to argue that in order for both $\mathsf{time}(v) < \mathsf{time}(u,a,t)$ and $\mathsf{depth}(v) < t/2$ to hold, it must be the case that $t' \le t$. Suppose to the contrary that $t' > t$. If $t' \ge 2 \cdot \mathsf{depth}(v)$, then the algorithm can only play (v,b,t') once t' becomes the lowest index, which must happen after (u,a,t) becomes the lowest index, hence $\mathsf{time}(v,b,t') < \mathsf{time}(u,a,t)$ is impossible. Otherwise, if $t' < 2 \cdot \mathsf{depth}(v)$, then $\mathsf{depth}(v) > t'/2 > t/2$, violating $\mathsf{depth}(v) < t/2$. Thus indeed $t' \le t$ and

$$\begin{aligned} (21) &\iff \left\{ \sum_{v \in \mathcal{S} \setminus \mathcal{S}_i : \mathsf{depth}(v) < t/2} \sum_{b \in A} \sum_{t'=1}^{t} \mathbb{1}_{\{\mathsf{time}(v,b,t') < \mathsf{time}(u,a,t)\}} \ge \frac{t}{2} \right\} \\ &\implies \left\{ \sum_{v \in \mathcal{S} \setminus \mathcal{S}_i} \sum_{b \in A} \sum_{t'=1}^{t} \mathbb{1}_{\{\mathsf{time}(v,b,t') < \infty\}} \ge \frac{t}{2} \right\}. \end{aligned}$$

We establish that the probability of interest $\Pr[\mathsf{time}(u,a,t) > t \mid \mathsf{time}(u,a,t) < \infty]$ is at most

$$\Pr\left[\sum_{v \in \mathcal{S} \setminus \mathcal{S}_i} \sum_{b \in A} \sum_{t'=1}^{t} \mathbb{1}_{\{\mathsf{time}(v,b,t') < \infty\}} \ge \frac{t}{2} \,\middle|\, \mathsf{time}(u,a,t) < \infty \right] = \Pr\left[\sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{b \in A} \sum_{t'=1}^{t} \mathbb{1}_{\{\mathsf{time}(v,b,t') < \infty\}} \ge \frac{t}{2} \right],$$

where we remove the conditioning due to independence between arms. Now, let

$$Y_j = \min\left\{ \sum_{v \in \mathcal{S}_j} \sum_{b \in A} \sum_{t'=1}^{t} \mathbb{1}_{\{\mathsf{time}(v,b,t') < \infty\}}, \frac{t}{2} \right\},$$

for all $j \ne i$. The previous probability is equal to $\Pr[\sum_{j \ne i} Y_j \ge t/2]$. Note that

$$\mathbb{E}\left[\sum_{j \ne i} Y_j \right] \le \sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{b \in A} \sum_{t'=1}^{t} \Pr[\mathsf{time}(v,b,t') < \infty] \le \sum_{t'=1}^{t} \sum_{v \in \mathcal{S}} \sum_{b \in A} \frac{x_{v,t'}^{b}}{3}, \le \frac{t}{3}$$

where the second inequality uses (20), and the final inequality uses (11). We can do better than the Markov bound on $\Pr[\sum_{j \ne i} Y_j \ge t/2]$ because the random variables $\{Y_j\}_{j \ne i}$ are independent. Furthermore, each Y_j is nonzero with probability of at most $\frac{1}{3}$ (arm j is never touched with probability of at least $\frac{2}{3}$), so since it is at most $t/2$ when it is nonzero, $\mathbb{E}[Y_j] \le t/6$ for all $j \ne i$. We now invoke the following lemma:

Lemma 7. *Let $t > 0$ be arbitrary and $Y_1, \ldots, Y_m$ be independent nonnegative random variables with individual expectations of at most $t/6$ and sum of expectations of at most $t/3$. Then $\Pr[\sum_{j=1}^{m} Y_j \ge t/2]$ is maximized when only two random variables are nonzero, each taking value $t/2$ with probability $\frac{1}{3}$ (and value 0 otherwise). Therefore, $\Pr[\sum_{j=1}^{m} Y_j \ge t/2] \le 1 - (1 - \frac{1}{3})^2 = \frac{5}{9}$.*

This lemma would complete the proof that $\Pr[\mathsf{time}(u,a,t) > t \mid \mathsf{time}(u,a,t) < \infty] \le \frac{5}{9}$ under Case 1, where $t \ge 2 \cdot \mathsf{depth}(u)$. We defer the proof of Lemma 7 to Appendix D. It uses the conjecture of Samuels from Samuels [31] for $n = 3$; the conjecture has been proven for $n \le 4$ in Samuels [32]. The proof also uses a technical lemma from Bansal et al. [2].

Case 2. Suppose $t < 2 \cdot \mathsf{depth}(u)$. Then $\mathsf{depth}(u)$ must be at least 1, so conditioned on $\mathsf{time}(u,a,t) < \infty$, there must be some $(v,b) \in \mathsf{Par}(u)$ and $t' < t$ such that $\mathsf{time}(v,b,t') < \infty$. Furthermore, the algorithm will play status (u,a,t) at time step $\mathsf{time}(v,b,t') + 1$, so $\mathsf{time}(u,a,t) \le t$ will hold so long as $\mathsf{time}(v,b,t') \le t'$, since $t' < t$. Thus $\Pr[\mathsf{time}(u,a,t) \le t \mid \mathsf{time}(u,a,t) < \infty] \ge \Pr[\mathsf{time}(v,b,t') \le t \mid \mathsf{time}(v,b,t') < \infty]$. We can iterate this argument until the problem reduces to Case 1. This completes the proof that $\Pr[\mathsf{time}(u,a,t) \le t \mid \mathsf{time}(u,a,t) < \infty] \ge \frac{4}{9}$ under Case 2.

Therefore, the expected reward obtained by our algorithm is at least $\sum_{u \in \mathcal{S}} \sum_{a \in A} r_u^a \sum_{t=1}^{B} (1 - \frac{5}{9})(x_{u,t}^a/3)$, which is the same as $(4/27)\mathsf{OPT}_{\mathtt{PolyLP}}$, completing the proof of Theorem 2.

4.3. Proof of Theorem 3

In this subsection we show how to modify the algorithm and analysis when there are multiperiod actions, to prove Theorem 3. As mentioned in Section 4.1, we must modify Step 1 of the algorithm when there are bridge nodes. If we arrive at a status (u,a,t) such that $t \ge 2 \cdot \mathsf{depth}(u)$ but $u \in \mathcal{B}$ (and $a = \alpha$), we are forced to immediately play the same arm again, instead of switching to another arm with a lower index.

The overall framework of the analysis still holds, except now the bound is optimized when we set $C = 6$. Our goal is to prove for an arbitrary $i \in [n]$, $u \in \mathcal{S}_i$, $a \in A$, $t \in [B]$ that $\Pr[\mathsf{time}(u,a,t) > t \mid \mathsf{time}(u,a,t) < \infty] \le \frac{1}{2}$. We still have that event $\{\mathsf{time}(u,a,t) > t\}$ implies (21). Suppose for an arbitrary $v \in \mathcal{S} \backslash \mathcal{S}_i$ that $\mathsf{depth}(v) < t/2$ and $\mathsf{time}(v) < \mathsf{time}(u,a,t)$, where $\mathsf{time}(v) = \mathsf{time}(v,b,t')$. We can no longer argue that $t' \le t$, but we would like to argue that $t' \le 3t/2$. Suppose to the contrary that $t' > 3t/2$.

Then $t' > 3 \cdot \mathsf{depth}(v)$, so $t' \ge 2 \cdot \mathsf{depth}(v)$ i.e., we would check priorities before playing (v,b,t'). However, if $v \in \mathcal{B}$, then it could be the case that $\mathsf{time}(v,b,t') < \mathsf{time}(u,a,t)$ even though $t' > t$. If so, consider w, the youngest (largest depth) ancestor of v that isn't a bridge node. Suppose $\mathsf{time}(w) = \mathsf{time}(w,b',t'')$; it must be the case that $t'' \le t$. By the final statement of Lemma 6, the $\mathsf{depth}(v) - \mathsf{depth}(w)$ immediate descendents of w, which are bridge nodes, must have priority indices $t''+1, \ldots, t''+\mathsf{depth}(v)-\mathsf{depth}(w)$, respectively. The youngest of these descendents is v, hence $t' = t'' + \mathsf{depth}(v) - \mathsf{depth}(w)$. But $t'' \le t$ and $\mathsf{depth}(v) < t/2$, so $t' < 3t/2$, causing a contradiction.

Therefore $t' \le 3t/2$. The bound on $\mathbb{E}[\sum_{j \ne i} Y_j]$ changes to

$$\mathbb{E}\left[\sum_{j \ne i} Y_j\right] \le \sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{b \in A} \sum_{t'=1}^{3t/2} \Pr[\mathsf{time}(v,b,t') < \infty] \le \sum_{t'=1}^{3t/2} \sum_{v \in \mathcal{S}} \sum_{b \in A} \frac{x_{v,t'}^b}{6} \le \frac{t}{4}.$$

Thus $\Pr[\mathsf{time}(u,a,t) > t \mid \mathsf{time}(u,a,t) < \infty] \le \Pr[\sum_{j \ne i} Y_j \ge t/2] \le \frac{1}{2}$ where the final inequality is Markov's inequality. Note that we cannot use the stronger Samuels' conjecture here because we would need it for $n = 5$, which is unproven; if we could, then we could get a better approximation factor (and we would re-optimize C).

The rest of the analysis, including Case 2, is the same as before. Therefore, the expected reward obtained by our algorithm is at least $\sum_{u \in \mathcal{S}} \sum_{a \in A} r_u^a \sum_{t=1}^{B} (1 - \frac{1}{2})(x_{u,t}^a/6)$, which is the same as $(1/12)\mathsf{OPT}_{\mathtt{PolyLP}}$, completing the proof of Theorem 3.

5. Conclusion and Open Questions

In this paper, we presented a $(\frac{1}{2} - \varepsilon)$-approximation for the fully general *MAB superprocess with multiperiod actions—no preemption* problem, by following a scaled copy of an optimal solution to the LP relaxation, and this is tight. However, when preemption is allowed, we were only able to obtain a 1/12-approximation, using the solution to the LP relaxation mainly for generating priorities, and resorting to weak Markov-type bounds in the analysis. It seems difficult to follow a scaled copy of a solution to the LP relaxation when preemption is allowed, because arms can be paused and restarted. We do conjecture that our separation of $(2-\varepsilon)$ between the LP and the optimal algorithm is correct in this case, and that it is possible to obtain a $(\frac{1}{2}-\varepsilon)$-approximation, but this remains an open problem. Also, we have not explored how our techniques apply to certain extensions of the multi-armed bandit problem (switching costs, simultaneous plays, delayed feedback, contextual information, etc.).

Acknowledgments

The author would like to thank Michel Goemans, Andreas Schulz, and David Simchi-Levi for helpful discussions and suggestions on the presentation of the results. A preliminary version of this paper appeared in the ACM-SIAM Symposium on Discrete Algorithms (SODA), 2014. The author would like to thank the anonymous reviewers for SODA, as well as the anonymous reviewers for Mathematics of Operations Research, for insightful remarks which improved the paper.

Figure A.1. A Markov chain representing a SK job with correlated rewards.

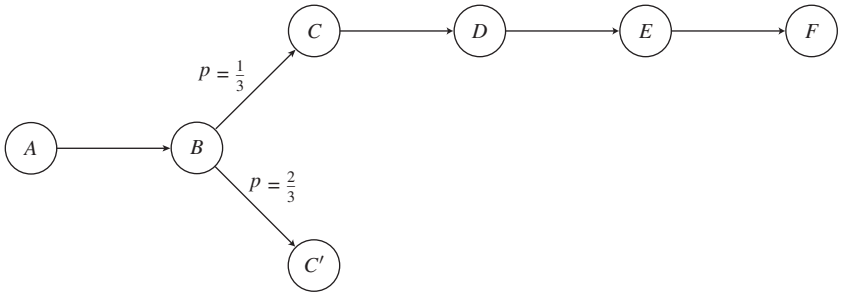
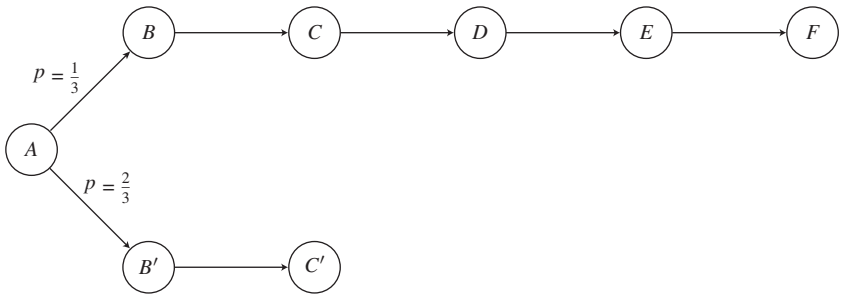


Figure A.2. Another Markov chain representing a SK job with correlated rewards.



Appendix A. Examples of Transforming SK Jobs to Markov Chains

1. Consider a job that takes time 5 with probability $\frac{1}{3}$, and time 2 with probability $\frac{2}{3}$ (and cannot be canceled once started). If it finishes, the reward returned is 2, independent of processing time. This can be modeled by Figure A.1 where $r_B = \frac{4}{3}$, $r_E = 2$, and $\mathcal{B} = \{B, C, D, E\}$. Note that instead of placing reward 2 on arc (B, C'), we have equivalently placed reward $\frac{4}{3}$ on node B. A corollary of this reduction is that the following reward structure is equivalent to the original for the objective of maximizing expected reward: a guaranteed reward of $\frac{4}{3}$ after 2 time steps, after which the job may run for another 3 time steps to produce an additional 2 reward.

2. Consider the same job as the previous one, except the reward is 4 if the processing time was 5, while the reward is 1 if the processing time was 2 (the expected reward for finishing is still 2). All we have to change in the reduction is setting $r_B = \frac{2}{3}$ and $r_E = 4$ instead.

3. Consider either of the two jobs above, except cancellation is permitted (presumably on node C, after observing the transition from node B). All we have to change in the reduction is setting $\mathcal{B} = \emptyset$ instead.

4. Consider the job from the second bullet that can be canceled, and furthermore, we find out after 1 time step whether it will realize to the long, high-reward job or the short, low-reward job. This can be modeled by Figure A.2 where $r_{B'} = 1$, $r_E = 4$, and $\mathcal{B} = \emptyset$.

Appendix B. Example Showing Preemption Is Necessary for Uncorrelated SK

Consider the following example: there are $n = 3$ items, I_1, I_2, I_3. I_1 instantiates to size 6 with probability $\frac{1}{2}$, and size 1 with probability $\frac{1}{2}$. I_2 deterministically instantiates to size 9. I_3 instantiates to size 8 with probability $\frac{1}{2}$, and size 4 with probability $\frac{1}{2}$. I_1, I_2, I_3, if successfully inserted, return rewards of 4, 9, 8, respectively. We have a knapsack of size 10.

We describe the optimal preempting policy. First we insert I_1. After 1 unit of time, if I_1 completes, we go on to insert I_2, which will deterministically fit. If I_1 doesn't complete, we set it aside and insert I_3 to completion. If it instantiates to size 8, then we cannot get any more reward from other items. If it instantiates to size 4, then we can go back and finish inserting the remaining 5 units of I_1. The expected reward of this policy is $\frac{1}{2}(4+9) + \frac{1}{2}(\frac{1}{2}8 + \frac{1}{2}(8+4)) = 11.5$.

Now we enumerate the policies that can only cancel but not preempt. If we first insert I_2, then the best we can do is try to insert I_1 afterward, getting a total expected reward of 11. Note that any policy never fitting I_2 can obtain reward at most 11, since with probability $\frac{1}{4}$ we cannot fit both I_1 and I_3. This rules out policies that start with I_3, which has no chance of fitting alongside I_2. Remaining are the policies that first insert I_1. If it doesn't complete after 1 unit of time, then we can either settle for the 9 reward of I_2, or finish processing I_1 with the hope of finishing I_3 afterward. However, in this case, I_3 only finishes half the time, so we earn more expected reward by settling for I_2. Therefore, the best we can do after first inserting I_1 is to stop processing it after time 1 (regardless of whether it completes), and process I_2, earning a total expected reward of 11.

We have shown that indeed, for uncorrelated SK, there is a gap between policies that can preempt (which includes canceling) versus policies that can only cancel. It appears that this gap is bounded by a constant, contrary to the gap between policies that can cancel versus policies that cannot cancel (see Gupta et al. [19, appendix A.1]).

Appendix C. Proofs from Section 2

C.1. Proof of Lemma 1

Suppose we are given $\{z^a_{\pi,i,t}\}, \{y_{\pi,t}\}$ satisfying (2a)–(2c), (3a)–(3c) which imply (4). For all $i \in [n]$, $u \in \mathcal{S}_i$, $t \in [B]$, let $s_{u,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}: \pi_i = u} y_{\pi,t}$, and let $x^a_{u,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}: \pi_i = u} z^a_{\pi,i,t}$ for each $a \in A$. We aim to show $\{x^a_{u,t}\}, \{s_{u,t}\}$ satisfies (10a)–(10c), (11), (12a)–(12c) and makes (9) the same objective function as (1). For convenience, we adopt the notation that $x_{u,t} = \sum_{a \in A} x^a_{u,t}$ and $z_{\pi,i,t} = \sum_{a \in A} z^a_{\pi,i,t}$.

(11): $\sum_{u \in \mathcal{S}} x_{u,t} = \sum_{i=1}^n \sum_{u \in \mathcal{S}_i} \sum_{\pi: \pi_i = u} z_{\pi,i,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}} \sum_{i=1}^n \sum_{u \in \mathcal{S}_i: u = \pi_i} z_{\pi,i,t}$. But there is a unique $u \in \mathcal{S}_i$ such that $u = \pi_i$, so the sum equals $\sum_{\pi \in \boldsymbol{\mathcal{S}}} \sum_{i=1}^n z_{\pi,i,t}$, which is at most 1 by (4).

(10a): For $u \in \mathcal{S}_i$, $x_{u,t} = \sum_{\pi: \pi_i = u} z_{\pi,i,t}$, and each term in the sum is at most $y_{\pi,t}$ by (2a) and (2c), hence $x_{u,t} \le \sum_{\pi: \pi_i = u} y_{\pi,t} = s_{u,t}$.

(10b): For $u \in \mathcal{B}$, $x^\alpha_{u,t} = \sum_{\pi: \pi_i = u} z^\alpha_{\pi,i,t}$, and each term in the sum is equal to $y_{\pi,t}$ by (2b), hence $x^\alpha_{u,t} = \sum_{\pi: \pi_i = u} y_{\pi,t} = s_{u,t}$.

(10c), (12a), and (12b) are immediate from (2c), (3a), and (3b), respectively. For (12c), fix $t > 1$, $i \in [n]$, and $u \in \mathcal{S}_i$. Sum (3c) over $\{\pi\colon \pi_i = u\}$ to get

$$\begin{aligned}
\sum_{\pi: \pi_i = u} y_{\pi,t} &= \sum_{\pi: \pi_i = u} y_{\pi,t-1} - \sum_{\pi: \pi_i = u} \sum_{j=1}^n z_{\pi,j,t-1} + \sum_{\pi: \pi_i = u} \sum_{j=1}^n \sum_{(v,a) \in \mathsf{Par}(\pi_j)} z^a_{\pi^v,j,t-1} \cdot p^a_{v,\pi_j}, \\
s_{u,t} &= s_{u,t-1} - \sum_{\pi: \pi_i = u} z_{\pi,i,t-1} - \sum_{\pi: \pi_i = u} \sum_{j \ne i} z_{\pi,j,t-1} \\
&\quad + \sum_{\pi: \pi_i = u} \sum_{(v,a) \in \mathsf{Par}(u)} z^a_{\pi^v,i,t-1} \cdot p^a_{v,u} + \sum_{\pi: \pi_i = u} \sum_{j \ne i} \sum_{(v,a) \in \mathsf{Par}(\pi_j)} z^a_{\pi^v,j,t-1} \cdot p^a_{v,\pi_j}, \\
s_{u,t} &= s_{u,t-1} - x_{u,t-1} - \sum_{\pi: \pi_i = u} \sum_{j \ne i} z_{\pi,j,t-1} \\
&\quad + \sum_{(v,a) \in \mathsf{Par}(u)} \left(\sum_{\pi: \pi_i = u} z^a_{\pi^v,i,t-1} \right) \cdot p^a_{v,u} + \sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{a \in A} \sum_{\{\pi: \pi_i = u, \mathsf{Par}(\pi_j) \ni (v,a)\}} z^a_{\pi^v,j,t-1} \cdot p^a_{v,\pi_j}, \\
s_{u,t} &= s_{u,t-1} - x_{u,t-1} - \sum_{\pi: \pi_i = u} \sum_{j \ne i} z_{\pi,j,t-1} \\
&\quad + \sum_{(v,a) \in \mathsf{Par}(u)} \left(\sum_{\pi: \pi_i = v} z^a_{\pi,i,t-1} \right) \cdot p^a_{v,u} + \sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{a \in A} \sum_{\pi: \pi_i = u, \pi_j = v} z^a_{\pi,j,t-1} \cdot \left(\sum_{w: p^a_{v,w} > 0} p^a_{v,w} \right), \\
s_{u,t} &= s_{u,t-1} - x_{u,t-1} - \sum_{\pi: \pi_i = u} \sum_{j \ne i} z_{\pi,j,t-1} \\
&\quad + \sum_{(v,a) \in \mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u} + \sum_{j \ne i} \sum_{v \in \mathcal{S}_j} \sum_{a \in A} \sum_{\pi: \pi_i = u, \pi_j = v} z^a_{\pi,j,t-1} \cdot (1), \\
s_{u,t} &= s_{u,t-1} - x_{u,t-1} - \sum_{j \ne i} \sum_{\pi: \pi_i = u} z_{\pi,j,t-1} + \sum_{(v,a) \in \mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u} + \sum_{j \ne i} \sum_{\pi: \pi_i = u} z_{\pi,j,t-1} \\
s_{u,t} &= s_{u,t-1} - x_{u,t-1} + \sum_{(v,a) \in \mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u},
\end{aligned}$$

which is exactly (12c).

(9): $\sum_{u \in \mathcal{S}} \sum_{a \in A} r^a_u \sum_{t=1}^B x^a_{u,t} = \sum_{i=1}^n \sum_{u \in \mathcal{S}_i} \sum_{a \in A} r^a_u \sum_{t=1}^B \sum_{\pi: \pi_i = u} z^a_{\pi,i,t}$, and by the same manipulation we made for (11), this is equal to $\sum_{\pi \in \boldsymbol{\mathcal{S}}} \sum_{i=1}^n \sum_{a \in A} r^a_{\pi_i} \sum_{t=1}^B z^a_{\pi,i,t}$. Thus (9) is the same as (1), completing the proof of Lemma 1.

C.2. Proof of Lemma 2

Suppose we are given $\{z^a_{\pi,i,t}\}, \{y_{\pi,t}\}$ satisfying (6a)–(6c), (7a)–(7e) which imply (8). For all $i \in [n]$, $u \in \mathcal{S}_i$, $t \in [B]$, let $s_{u,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}': \pi_i = u} y_{\pi,t}$, and let $x^a_{u,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}': \pi_i = u} z^a_{\pi,i,t}$ for each $a \in A$. We aim to show $\{x^a_{u,t}\}, \{s_{u,t}\}$ satisfies (10a)–(10c), (11), (13a)–(13d) and makes (9) the same objective function as (5). For convenience, we adopt the notation that $x_{u,t} = \sum_{a \in A} x^a_{u,t}$ and $z_{\pi,i,t} = \sum_{a \in A} z^a_{\pi,i,t}$.

(11): $\sum_{u \in \mathcal{S}} x_{u,t} = \sum_{i=1}^n \sum_{u \in \mathcal{S}_i} \sum_{\pi: \pi_i = u} z_{\pi,i,t} = \sum_{\pi \in \boldsymbol{\mathcal{S}}'} \sum_{i=1}^n \sum_{u \in \mathcal{S}_i: u = \pi_i} z_{\pi,i,t}$. The difference from the previous derivation of (11) is that there is only a unique $u \in \mathcal{S}_i$ such that $u = \pi_i$, if $\pi_i \ne \phi_i$. So the sum equals $\sum_{\pi \in \boldsymbol{\mathcal{S}}'} \sum_{i \in I(\pi)} z_{\pi,i,t}$, which is at most 1 by (8).

Using this same manipulation, the equivalence of (9) and (5) follows the same derivation as before. (10a) and (10b) also follow the same derivations as before; (10c), (13a), and (13b) are immediate. It remains to prove (13c) and (13d).

(13d): Fix $t > 1$, $i \in [n]$, and $u \in \mathcal{S}_i \backslash \{\rho_i\}$. First consider the case where $\mathsf{depth}(u) > 1$. All $\pi \in \boldsymbol{\mathcal{S}}'$ such that $\pi_i = u$ fall under (7e), so we can sum over these π to get

$$\sum_{\pi: \pi_i = u} y_{\pi,t} = \sum_{\pi: \pi_i = u} \sum_{(v,a) \in \mathsf{Par}(u)} z^a_{\pi^v,i,t-1} \cdot p^a_{v,u}, \qquad s_{u,t} = \sum_{(v,a) \in \mathsf{Par}(u)} \left(\sum_{\pi: \pi_i = u} z^a_{\pi^v,i,t-1} \right) \cdot p^a_{v,u}$$

Since $\mathsf{depth}(u) > 1$, $v \ne \rho_i$, so $\{\pi^v\colon \pi \in \boldsymbol{\mathcal{S}}', \pi_i = u\} = \{\pi\colon \pi \in \boldsymbol{\mathcal{S}}', \pi_i = v\}$. Hence the RHS of the above equals $\sum_{(v,a) \in \mathsf{Par}(u)} x^a_{v,t-1} \cdot p^a_{v,u}$ which is exactly (13d).

For the other case where $\mathsf{depth}(u)=1$, all $\pi\in\boldsymbol{\mathcal{S}}'$ such that $\pi_i=u$ fall under (7d), so we can sum over these π to get

$$\begin{aligned}\sum_{\pi:\,\pi_i=u} y_{\pi,t} &= \sum_{\pi:\,\pi_i=u}\sum_{a:\,(\rho_i,a)\in\mathsf{Par}(u)}\left(\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_i})} z^a_{\pi',i,t-1}\right)\cdot p^a_{\rho_i,u},\\ s_{u,t} &= \sum_{a:\,(\rho_i,a)\in\mathsf{Par}(u)}\left(\sum_{\pi:\,\pi_i=u}\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_i})} z^a_{\pi',i,t-1}\right)\cdot p^a_{\rho_i,u},\\ s_{u,t} &= \sum_{a:\,(\rho_i,a)\in\mathsf{Par}(u)}\left(\sum_{\pi:\,\pi_i=\rho_i} z^a_{\pi,i,t-1}\right)\cdot p^a_{\rho_i,u},\\ s_{u,t} &= \sum_{a:\,(\rho_i,a)\in\mathsf{Par}(u)} x^a_{\rho_i,t-1}\cdot p^a_{\rho_i,u}.\end{aligned}$$

We explain the third equality. Since $u\neq\rho_i$ implies arm i is the active arm in all of $\{\pi\in\boldsymbol{\mathcal{S}}'\colon \pi_i=u\}$, this set is equal to $\{\rho_1,\phi_1\}\times\cdots\times u\times\cdots\times\{\rho_n,\phi_n\}$. Thus $\{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_i})\colon \pi\in\boldsymbol{\mathcal{S}}',\pi_i=u\}=\{\pi'\in\boldsymbol{\mathcal{P}}(\pi)\colon \pi\in\{\rho_1,\phi_1\}\times\cdots\times\rho_i\times\cdots\times\{\rho_n,\phi_n\}\}$. Recall that $\boldsymbol{\mathcal{P}}(\pi)$ is the set of joint nodes that would transition to π with no play. Therefore, this set is equal to $\{\pi'\in\boldsymbol{\mathcal{S}}'\colon \pi'_i=\rho_i\}$, as desired.

(13c): Fix $t>1$ and $i\in[n]$. Unfortunately, $\pi\in\boldsymbol{\mathcal{S}}'$ such that $\pi_i=\rho_i$ can fall under (7c), (7d), or (7e). First let us sum over the π falling under (7c):

$$\begin{aligned}\sum_{\pi\notin\boldsymbol{\mathcal{A}}:\,\pi_i=\rho_i} y_{\pi,t} &= \sum_{\pi\notin\boldsymbol{\mathcal{A}}:\,\pi_i=\rho_i}\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi)}\left(y_{\pi',t-1}-\sum_{j\in I(\pi')} z_{\pi',j,t-1}\right)\\ &= \sum_{\pi\in\boldsymbol{\mathcal{S}}':\,\pi_i=\rho_i}\left(y_{\pi,t-1}-z_{\pi,i,t-1}-\sum_{j\in I(\pi)\setminus\{i\}} z_{\pi,j,t-1}\right) = s_{\rho_i,t-1}-x_{\rho_i,t-1}-\sum_{\pi\in\boldsymbol{\mathcal{S}}':\,\pi_i=\rho_i}\sum_{j\in I(\pi)\setminus\{i\}} z_{\pi,j,t-1},\end{aligned}$$

where the second equality requires the same set bijection explained above. Furthermore,

$$\begin{aligned}\sum_{\pi\in\boldsymbol{\mathcal{S}}':\,\pi_i=\rho_i}\sum_{j\in I(\pi)\setminus\{i\}} z_{\pi,j,t-1} &= \sum_{k\neq i}\sum_{\pi\in\boldsymbol{\mathcal{A}}_k:\,\pi_i=\rho_i}\left(z_{\pi,k,t-1}+\sum_{j\in I(\pi)\setminus\{i,k\}} z_{\pi,j,t-1}\right)+\sum_{\pi\notin\boldsymbol{\mathcal{A}}:\,\pi_i=\rho_i}\sum_{j\in I(\pi)\setminus\{i\}} z_{\pi,j,t-1}\\ &= \sum_{k\neq i}\sum_{\pi\in\boldsymbol{\mathcal{A}}_k:\,\pi_i=\rho_i}\left(z_{\pi,k,t-1}+\sum_{j:\,\pi_j=\rho_j,\,j\neq i} z_{\pi,j,t-1}\right)+\sum_{\pi\notin\boldsymbol{\mathcal{A}}:\,\pi_i=\rho_i}\sum_{j:\,\pi_j=\rho_j,\,j\neq i} z_{\pi,j,t-1}\\ &= \sum_{k\neq i}\sum_{\pi\in\boldsymbol{\mathcal{A}}_k:\,\pi_i=\rho_i} z_{\pi,k,t-1}+\sum_{\pi\in\boldsymbol{\mathcal{S}}':\,\pi_i=\rho_i}\sum_{j:\,\pi_j=\rho_j,\,j\neq i} z_{\pi,j,t-1}.\end{aligned}$$

Now let us sum over the π falling under (7e):

$$\begin{aligned}\sum_{j\neq i}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{depth}(\pi_j)>1\}} y_{\pi,t} &= \sum_{j\neq i}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{depth}(\pi_j)>1\}}\sum_{(v,a)\in\mathsf{Par}(\pi_j)} z^a_{\pi^v,j,t-1}\cdot p^a_{v,\pi_j}\\ &= \sum_{j\neq i}\sum_{v\in\mathcal{S}_j\setminus\{\rho_j\}}\sum_{a\in A}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{Par}(\pi_j)\ni(v,a)\}} z^a_{\pi^v,j,t-1}\cdot p^a_{v,\pi_j}\\ &= \sum_{j\neq i}\sum_{v\in\mathcal{S}_j\setminus\{\rho_j\}}\sum_{a\in A}\sum_{\pi:\,\pi_i=\rho_i,\,\pi_j=v} z^a_{\pi,j,t-1}\cdot\left(\sum_{w:\,p^a_{v,w}>0} p^a_{v,w}\right)\\ &= \sum_{j\neq i}\sum_{v\in\mathcal{S}_j\setminus\{\rho_j\}}\sum_{a\in A}\sum_{\pi:\,\pi_i=\rho_i,\,\pi_j=v} z^a_{\pi,j,t-1}\cdot(1) = \sum_{j\neq i}\sum_{\pi\in\boldsymbol{\mathcal{A}}_j:\,\pi_i=\rho_i} z_{\pi,j,t-1},\end{aligned}$$

where the third equality uses the fact that $v\neq\rho_j$ to convert π^v to π. Finally, let us sum over the π falling under (7d):

$$\begin{aligned}\sum_{j\neq i}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{depth}(\pi_j)=1\}} y_{\pi,t} &= \sum_{j\neq i}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{depth}(\pi_j)=1\}}\sum_{a:\,(\rho_j,a)\in\mathsf{Par}(\pi_j)}\left(\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_j})} z^a_{\pi',j,t-1}\right)\cdot p^a_{\rho_j,\pi_j}\\ &= \sum_{j\neq i}\sum_{a\in A}\sum_{\{\pi:\,\pi_i=\rho_i,\,\mathsf{Par}(\pi_j)\ni(\rho_j,a)\}}\left(\sum_{\pi'\in\boldsymbol{\mathcal{P}}(\pi^{\rho_j})} z^a_{\pi',j,t-1}\right)\cdot p^a_{\rho_j,\pi_j}\\ &= \sum_{j\neq i}\sum_{a\in A}\sum_{\pi:\,\pi_i=\rho_i,\,\pi_j=\rho_j} z^a_{\pi,j,t-1}\cdot\left(\sum_{w:\,p^a_{\rho_j,w}>0} p^a_{\rho_j,w}\right)\\ &= \sum_{j\neq i}\sum_{a\in A}\sum_{\pi:\,\pi_i=\rho_i,\,\pi_j=\rho_j} z^a_{\pi,j,t-1}\cdot(1) = \sum_{\pi:\,\pi_i=\rho_i}\sum_{j:\,\pi_j=\rho_j,\,j\neq i} z_{\pi,j,t-1},\end{aligned}$$

where the third equality requires the same set bijection again. Combining the last four blocks of equations, we get $s_{\rho_i,t}=s_{\rho_i,t-1}-x_{\rho_i,t-1}$ which is exactly (13c), completing the proof of Lemma 2.

Appendix D. Proofs from Section 4

D.1. Proof of Lemma 6

Finding the q's is a separate problem for each arm, so we can fix $i \in [n]$. Furthermore, we can fix $u \in \mathcal{S}_i \backslash \{\rho_i\}$; we will specify an algorithm that defines $\{q_{v,b,t',u,a,t} : a \in A, t \in [B], (v,b) \in \mathsf{Par}(u), t' < t\}$ satisfying (19a) and (19b).

Observe that by substituting (10a) into (12c), we get $s_{u,t'} \le \sum_{(v,b)\in\mathsf{Par}(u)} x^b_{v,t'-1} \cdot p^b_{v,u}$ for all $t' > 1$. Summing over $t' = 2, \dots, t$ for an arbitrary $t \in [B]$, and using (10a) again on the LHS, we get $\sum_{t'=2}^{t} \sum_{a\in A} x^a_{u,t'} \le \sum_{t'=1}^{t-1} \sum_{(v,b)\in\mathsf{Par}(u)} x^b_{v,t'} \cdot p^b_{v,u}$.

Now, for all $t' = 2, \dots, B$ and $a \in A$, initialize $\tilde{x}^a_{t'} := x^a_{u,t'}$. For all $t' = 1, \dots, B-1$ and $(v,b) \in \mathsf{Par}(u)$, initialize $\tilde{x}^b_{v,t'} := x^b_{v,t'} \cdot p^b_{v,u}$. We are omitting the subscript u because u is fixed. The following $B-1$ inequalities hold:

$$\sum_{t'=2}^{t''} \sum_{a\in A} \tilde{x}^a_{t'} \le \sum_{t'=1}^{t''-1} \sum_{(v,b)\in\mathsf{Par}(u)} \tilde{x}^b_{v,t'}, \quad t'' = 2, \dots, B. \tag{D.1}$$

The algorithm updates the variables $\tilde{x}^a_{t'}$ and $\tilde{x}^b_{v,t'}$ over iterations $t = 2, \dots, B$, but we will inductively show that inequality t'' of (D.1) holds until the end of iteration t''. The algorithm can be described as follows:

Decomposition Algorithm

- Initialize all $q_{v,b,t',u,a,t} := 0$.
- For $t = 2, \dots, B$:
  - —While there exists some $a \in A$ such that $\tilde{x}^a_t > 0$:
    1. Choose any nonzero $\tilde{x}^b_{v,t'}$, where $(v,b) \in \mathsf{Par}(u)$, with $t' < t$.
    2. Let $Q = \min\{\tilde{x}^a_t, \tilde{x}^b_{v,t'}\}$.
    3. Set $q_{v,b,t',u,a,t} := Q/(x^b_{v,t'} \cdot p^b_{v,u})$.
    4. Subtract Q from both $\tilde{x}^a_t$ and $\tilde{x}^b_{v,t'}$.

Let us consider iteration t of the algorithm. The inequality of (D.1) with $t'' = t$ guarantees that there always exists such a nonzero $\tilde{x}^b_{v,t'}$ in Step 1. In Step 4, Q is subtracted from both the LHS and RHS of all inequalities of (D.1) with $t'' \ge t$, so these inequalities continue to hold. (Q is also subtracted from the RHS of inequalities of (D.1) with $t' < t'' < t$, so these inequalities might cease to hold.) This inductively establishes that all inequalities of (D.1) with $t'' \ge t$ hold during iteration t, and thus Step 1 of the algorithm is well-defined.

Now we show that (19b) is satisfied. Suppose on iteration t of the algorithm, we have some $\tilde{x}^a_t > 0$ and $\tilde{x}^b_{v,t'} > 0$ on Step 1. Note that $q_{v,b,t',u,a,t}$ must currently be 0, since if it was already set, then either $\tilde{x}^a_t$ or $\tilde{x}^b_{v,t'}$ would have been reduced to 0. Therefore, in Step 3 we are incrementing the LHS of (19b) by $x^b_{v,t'} \cdot p^b_{v,u} \cdot (Q/(x^b_{v,t'} \cdot p^b_{v,u})) = Q$, after which we are subtracting Q from $\tilde{x}^a_t$ in Step 4. Since over iterations $t = 2, \dots, B$, for every $a \in A$, $\tilde{x}^a_t$ gets reduced from $x^a_{u,t}$ to 0, it must be the case that every equation in (19b) holds by the end of the algorithm.

For (19a), we use a similar argument. Fix some $(v,b) \in \mathsf{Par}(u)$ and $t' \in [B-1]$. Whenever we add $Q/(x^b_{v,t'} \cdot p^b_{v,u})$ to the LHS of (19a), we are reducing $\tilde{x}^b_{v,t'}$ by Q. Since $\tilde{x}^b_{v,t'}$ starts at $x^b_{v,t'} \cdot p^b_{v,u}$ and cannot be reduced below 0, the biggest we can make the LHS of (19a) is $x^b_{v,t'} \cdot p^b_{v,u}/(x^b_{v,t'} \cdot p^b_{v,u}) = 1$.

Finally, it is clear that the algorithm takes polynomial time, since every time we loop through Steps 1 to 4 either $\tilde{x}^a_t$ or $\tilde{x}^b_{v,t'}$ goes from nonzero to zero, and there were only a polynomial number of such variables to begin with. Other than the statement for $u \in \mathcal{B}$, this completes the proof of Lemma 6.

Now, if $u \in \mathcal{B}$, then we can strengthen (D.1). Indeed, substituting (10b) (instead of (10a)) into (12c), we get that all inequalities of (D.1) hold as equality. At the start of iteration $t = 2$, it is the case that $\tilde{x}^a_t = \sum_{(v,b)\in\mathsf{Par}(u)} \tilde{x}^b_{v,t-1}$, and by the end of the iteration, it will be the case that $\tilde{x}^\alpha_t = 0$, and $\tilde{x}^b_{v,t-1} = 0$, $q_{v,b,t-1,u,\alpha,t} = 1$ for all $(v,b) \in \mathsf{Par}(u)$. As a result, the equalities of (D.1) with $t'' > t$ will continue to hold as equality. We can inductively apply this argument to establish that $q_{v,b,t',u,\alpha,t'+1} = 1$ for all $(v,b) \in \mathsf{Par}(u)$ and $t' \in [B-1]$, as desired.

D.2. Proof of Lemma 7

We make use of the following conjecture of Samuels, which is proven for $n \le 4$ (see Samuels [31, 32]):

Conjecture 1. *Let $X_1, \dots, X_n$ be independent nonnegative random variables with respective expectations $\mu_1 \ge \cdots \ge \mu_n$, and let $\lambda > \sum_{i=1}^n \mu_i$. Then $\Pr[\sum_{i=1}^n X_i \ge \lambda]$ is maximized when the X_i's are distributed as follows, for some index $k \in [n]$:*

- *For $i > k$, $X_i = \mu_i$ with probability 1.*
- *For $i \le k$, $X_i = \lambda - \sum_{\ell=k+1}^n \mu_\ell$ with probability $\mu_i/(\lambda - \sum_{\ell=k+1}^n \mu_\ell)$, and $X_i = 0$ otherwise.*

If we have $\mathbb{E}[Y_i] + \mathbb{E}[Y_j] \le t/6$ for $i \ne j$, then we can treat $Y_i + Y_j$ as a single random variable satisfying $\mathbb{E}[Y_i + Y_j] \le t/6$. By the pigeonhole principle, we can repeat this process until $n \le 3$, since $\sum_{j=1}^m \mathbb{E}[Y_j] \le t/3$. In fact, we assume n is exactly 3 (we can add random variables that take constant value 0 if necessary), so that we can apply Conjecture 1 for $n = 3$, which has been proven to be true. We get that $\Pr[Y_1 + Y_2 + Y_3 \ge t/2]$ cannot exceed the maximum of the following (corresponding to the cases $k = 3, 2, 1$, respectively):

- $1 - (1 - \mu_1/(t/2))(1 - \mu_2/(t/2))(1 - \mu_3/(t/2))$
- $1 - (1 - \mu_1/(t/2 - \mu_3))(1 - \mu_2/(t/2 - \mu_3))$
- $1 - (1 - \mu_1/(t/2 - \mu_2 - \mu_3))$

Now we employ Lemma 4 from Bansal et al. [2] to bound these quantities.

Lemma 8. *Let r and $p_{\max}$ be positive real values. Consider the problem of maximizing $1-\prod_{i=1}^{t}(1-p_i)$ subject to the constraints $\sum_{i=1}^{t} p_i \le r$, and $0 \le p_i \le p_{\max}$ for all i. Denote the maximum value by $\beta(r, p_{\max})$. Then*

$$\beta(r,p_{\max}) = 1-(1-p_{\max})^{\lfloor r/p_{\max}\rfloor}\left(1-\left(r-\left\lfloor\frac{r}{p_{\max}}\right\rfloor\cdot p_{\max}\right)\right) \le 1-(1-p_{\max})^{r/p_{\max}}.$$

Recall that $\mu_1, \mu_2, \mu_3 \le t/6$ and $\mu_1+\mu_2+\mu_3 \le t/3$.

- In the first case $k=3$, we get $p_{\max}=\frac{1}{3}$ and $r=\frac{2}{3}$, so the quantity is at most $\beta(\frac{2}{3},\frac{1}{3}) \le 1-(1-\frac{1}{3})^2 = \frac{5}{9}$, as desired.
- In the second case $k=2$, for an arbitrary $\mu_3 \in [0, t/6]$, we get that the quantity is at most $\beta((t/3-\mu_3)/(t/2-\mu_3), (t/6)/(t/2-\mu_3)) \le 1-(1-(t/6)/(t/2-\mu_3))^{(t/3-\mu_3)/(t/6)}$. It can be checked that the maximum occurs at $\mu_3=0$, so the quantity is at most $\frac{5}{9}$ for any value of $\mu_3 \in [0, t/6]$, as desired.
- In the third case $k=1$, we get that the quantity is at most $\mu_1/(t/2-(t/3-\mu_1)) = \mu_1/(t/6+\mu_1)$, which at most $\frac{1}{2}$ over $\mu_1 \in [0, t/6]$, as desired.

Therefore, Conjecture 1 tells us that the maximum value of $\Pr[\sum_{j=1}^{m} Y_j \ge t/2]$ is $\frac{5}{9}$, completing the proof of Lemma 7.

Endnotes

[1] We use the word *node* instead of *state* to avoid confusion with the notion of a state in dynamic programming.

[2] All of the problems we discuss will be maximization problems, for which an α-approximation refers to an algorithm that attains at least α of the optimum.

[3] In some variants, there is a cost budget instead of a time budget, and exploring each arm incurs a different cost. We explain in Section 2 why our model also generalizes this setting, which they refer to as *Budgeted Bandits*.

[4] Some of these results, for the simpler settings of Markovian Bandits and Correlated SK without Cancellation, have appeared in a preliminary conference version of this article Ma [26].

[5] For convenience, we allow ourselves to not play an arm at a time step, even though playing is always beneficial, in that all rewards are nonnegative.

[6] Even though we have converted all Markov decision processes into layered acyclic digraphs, we cannot deduce the time elapsed from the nodes each arm is on, since we allow ourselves to not play any arm at a time step. Therefore, the time step must be included separately in the state information.

[7] For example, we could never be at a joint node with two or more arms on bridge nodes, and we could not get an arm on a node of depth 5 at the beginning of time 5.

[8] Intuitively, we are arguing that the solution to the LP relaxation still satisfies the total probability of arm i being played from its root node not exceeding unity.

[9] This is because $\mathsf{Free}^{\mathsf{emp}}(i,t)$ is an average over $M \ge (8/\varepsilon)\cdot\mu_{\varepsilon,\delta}$ runs, which is enough samples to guarantee this probability; see Motwani and Raghavan [29].

[10] Please note that this is not to be confused with the colloquialism of "higher priorities going first."

References

[1] Bansal N, Nagarajan V (2014) On the adaptivity gap of stochastic orienteering. *Integer Programming and Combinatorial Optimization* (Springer, New York), 114–125.
[2] Bansal N, Gupta A, Li J, Mestre J, Nagarajan V, Rudra A (2012) When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica* 63(4):733–762.
[3] Bertsekas DP (1995) *Dynamic Programming and Optimal Control* (Athena Scientific, Belmont, MA).
[4] Bertsimas D, Mersereau AJ (2007) A learning approach for interactive marketing to a customer segment. *Oper. Res.* 55(6):1120–1135.
[5] Bhalgat A (2011) A (2+eps)-approximation algorithm for the stochastic knapsack problem. Unpublished manuscript.
[6] Bhalgat A, Goel A, Khanna S (2011) Improved approximation results for stochastic knapsack problems. Randall D, ed. *Proc. Twenty-Second Annual ACM-SIAM Sympos. Discrete Algorithms, SODA '11* (SIAM, Philadelphia), 1647–1665.
[7] Bubeck S, Cesa-Bianchi N (2012) Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning* 5(1):1–122.
[8] Carraway RL, Schmidt RL, Weatherford LR (1993) An algorithm for maximizing target achievement in the stochastic knapsack problem with normal returns. *Naval Res. Logist. (NRL)* 40(2):161–173.
[9] Dean BC, Goemans MX, Vondrák J (2008) Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.* 33(4):945–964.
[10] Dean BC, Goemans MX, Vondrdk J (2004) Approximating the stochastic knapsack problem: The benefit of adaptivity. *Proc. 45th Annual IEEE Sympos.Foundations Comput. Sci., FOCS '04* (IEEE Computer Society, Washington, DC), 208–217.
[11] Farias VF, Madan R (2011) The irrevocable multiarmed bandit problem. *Oper. Res.* 59(2):383–399.
[12] Gittins J, Glazebrook K, Weber R (2011) *Multi-Armed Bandit Allocation Indices* (John Wiley & Sons, Hoboken, NJ).
[13] Goel A, Indyk P (1999) Stochastic load balancing and related problems. *Proc. 40th Annual Sympos. Foundations Comput. Sci., '99* (IEEE Computer Society, Washington, DC), 579–586.
[14] Goel A, Guha S, Munagala K (2006) Asking the right questions: Model-driven optimization using probes. *Proc. Twenty-Fifth ACM SIGMOD-SIGACT-SIGART Sympos. Principles of Database Systems* (ACM, New York), 203–212.
[15] Guha S, Munagala K (2007) Approximation algorithms for budgeted learning problems. Johnson DS, Feige U, eds. *Proc. Thirty-Ninth Annual ACM Sympos. Theory of Comput., STOC '07* (ACM, New York), 104–113.
[16] Guha S, Munagala K (2007) Model-driven optimization using adaptive probes. Bansal N, Pruhs K, Stein C, eds. *Proc. Eighteenth Annual ACM-SIAM Sympos. Discrete Algorithms, SODA '07* (SIAM, Philadelphia), 308–317.

[17] Guha S, Munagala K (2008) Sequential design of experiments via linear programming. Preprint arXiv:0805.2630.
[18] Guha S, Munagala K (2013) Approximation algorithms for Bayesian multiarmed bandit problems. Preprint arXiv:1306.3525.
[19] Gupta A, Krishnaswamy R, Molinaro M, Ravi R (2011) Approximation algorithms for correlated knapsacks and non-martingale bandits. Preprint arXiv:1102.3749.
[20] Gupta A, Krishnaswamy R, Molinaro M, Ravi R (2011) Approximation algorithms for correlated knapsacks and non-martingale bandits. Ostrovsky R, ed. *Proc. IEEE 52nd Annual Sympos. Foundations Comput. Sci. FOCS '11* (IEEE Computer Society, Washington, DC), 827–836.
[21] Gupta A, Krishnaswamy R, Nagarajan V, Ravi R (2014) Running errands in time: Approximation algorithms for stochastic orienteering. *Math. Oper. Res.* 40(1):56–79.
[22] Ilhan T, Iravani SMR, Daskin MS (2011) Technical note—The adaptive knapsack problem with stochastic rewards. *Oper. Res.* 59(1): 242–248.
[23] Kessler JM (2013) United States Air Force fighter jet maintenance models: Effectiveness of index policies. Ph.D. thesis, Massachusetts Institute of Technology.
[24] Kleinberg J, Rabani Y, Tardos É (2000) Allocating bandwidth for bursty connections. *SIAM J. Comput.* 30(1):191–217.
[25] Li J, Yuan W (2013) Stochastic combinatorial optimization via poisson approximation. Boneh D, Roughgarden T, Feigenbaum J, eds. *Proc. Forty-Fifth Annual ACM Sympos. Theory Comput., STOC '13* (ACM, New York), 971–980.
[26] Ma W (2014) Improvements and generalizations of stochastic knapsack and multiarmed bandit approximation algorithms. Chekuri C, ed. *Proc. Twenty-Fifth Annual ACM-SIAM Sympos. Discrete Algorithms, SODA '14* (SIAM, Philadelphia), 1154–1163.
[27] Mehta A, Panigrahi D (2012) Online matching with stochastic rewards. *Proc. 53rd IEEE Annual Sympos. Foundations Comput. Sci., FOCS '12* (IEEE Computer Society, Washington, DC), 728–737.
[28] Möhring RH, Schulz AS, Uetz M (1999) Approximation in stochastic scheduling: The power of LP-based priority policies. *J. ACM* (*JACM*) 46(6):924–942.
[29] Motwani R, Raghavan P (2010) *Randomized Algorithms* (Chapman and Hall/CRC, Boca Raton, FL).
[30] Pinedo ML (2012) *Scheduling: Theory, Algorithms, and Systems* (Springer, New York).
[31] Samuels SM (1966) On a Chebyshev-type inequality for sums of independent random variables. *Ann. Math. Stat.* 37(1):248–259.
[32] Samuels SM (1968) *More on a Chebyshev-Type Inequality for Sums of Independent Random Variables* (No. Mimeograph Ser-155). Department of Statistics, Purdue University, Lafayette, Indiana.
[33] Skutella M, Uetz M (2001) Scheduling precedence-constrained jobs with stochastic processing times on parallel machines. Kosaraju SR, eds. *Proc. Twelfth Annual ACM-SIAM Sympos. Discrete Algorithms, SODA '01* (SIAM, Philadelphia), 589–590.