Restless Bandits with Switching Costs: Linear Programming Relaxations, Performance Bounds and Limited Lookahead Policies

Jerome Le Ny
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139-4307, USA
jleny@mit.edu

Eric Feron
School of Aerospace Engineering
Georgia Tech
Atlanta, GA 30332, USA
eric.feron@aerospace.gatech.edu

Abstract— **The multi-armed bandit problem and one of its most interesting extensions, the restless bandits problem, are frequently encountered in various stochastic control problems. We present a linear programming relaxation for the restless bandits problem with discounted rewards, where only one project can be activated at each period but with additional costs penalizing switching between projects. The relaxation can be efficiently computed and provides a bound on the achievable performance. We describe several heuristic policies; in particular, we show that a policy adapted from the primal-dual heuristic of Bertsimas and Niño-Mora [1] for the classical restless bandits problem is in fact equivalent to a one-step lookahead policy; thus, the linear programming relaxation provides a means to compute an approximation of the cost-to-go. Moreover, the approximate cost-to-go is decomposable by project, and this allows the one-step lookahead policy to take the form of an index policy, which can be computed on-line very efficiently. We present numerical experiments, for which we assess the quality of the heuristics using the performance bound.**

I. Introduction

In the field of stochastic optimization, the multi-armed bandit (MAB) model is of fundamental importance because it is known to be solvable efficiently despite its generality. In the MAB problem, we consider N projects, of which only one can be worked on at any time period. Each project i is characterized at (discrete) time t by its state $x_i(t)$. If project i is worked on at time t, one receives a reward $\alpha^t r(x_i(t))$, where $\alpha \in (0,1)$ is a discount factor. The state $x_i(t)$ then evolves to a new state according to given transition probabilities. The states of all idle projects are unaffected. We assume perfect state information and (in this paper) we consider only a finite number of states for each project. The goal is to find a policy which decides at each time period which project to work on in order to maximize the expected sum of the discounted rewards over an infinite horizon.

The MAB problem was first solved by Gittins [2], [3], who showed that it is possible to attach to each project an index that is a function of the project and of its state alone, and that the optimal policy is simply characterized by operating at each period the project with the greatest current index. Moreover, these indices can be efficiently calculated. Whittle [4] proposed an extension of the model, called the restless bandits problem (RB), in which one can activate several projects at each time period, and the projects that remain passive continue to evolve, possibly using different rules. Finding an optimal policy efficiently for the RB problem is unlikely to be possible however, since the problem was shown to be PSPACE-hard [5], even in the case when only one project is active at each period and deterministic transition rules are in effect.

Another extension of the MAB model concerns the addition of costs for changing the currently active project. This problem is of great interest to various applications where the MAB formulation also applies, in order to model for example set-up and tear-down costs in queuing networks [6], or transition costs in a job search problem (see the survey in [7]). This paper is motivated by an optimal search problem in the context of aerial surveillance.

Compared with the classical MAB problem, relatively few papers are devoted to RB problems and multi-armed bandit problems with switching costs (MABSC). In the same spirit as this paper, [1] presented relaxations providing bounds on the achievable performance for the RB problem. [8] and [9] study the indexability of the RB problem and the MABSC problem respectively. However in the latter case the switching costs must be decomposable as set-up and tear-down costs, which is not valid in general if the costs represent travel distances. Other contributions to the MABSC problem include [10] and [11]. [12] solves a two-armed bandit problem with switching costs analytically, in the case of deteriorating rewards.

In this paper, we consider the restless bandits problem with switching costs (RBSC), when only one project can be set active at each time period. In Section II, we formulate the RBSC problem in the general framework of Markov decision processes (MDP). We also briefly review the state-action frequency approach used here to obtain the linear programming formulation of MDPs. In Section III, we show that even the special case of the MABSC is NP-hard, and we propose a first-order linear relaxation of the RBSC problem, providing an efficiently computable bound on the achievable performance. Section IV describes heuristics to solve the problem in practice and finally, Section V presents numerical experiments comparing these heuristics.

This work was supported by Air Force - DARPA - MURI award 009628-001-03-132 and Navy ONR award N00014-03-1-0171.

II. Exact Formulation of the RBSC

A. The State-Action Frequency Approach to MDPs

In this section, we first review the linear programming approach based on occupation measures to formulate Markov decision processes (MDP). The RBSC problem is then formulated in this framework. A (discrete-time) MDP is defined by a tuple $\{\mathbf{X}, A, \mathscr{P}, c\}$ as follows:

- $\mathbf{X}$ is the finite state space.
- A is the finite set of actions. $A(x) \subset A$ is the subset of actions available at state x. $\mathscr{K} = \{(x,a) : x \in \mathbf{X}, a \in A(x)\}$ is the set of state-action pairs.
- $\mathscr{P}$ are the transition probabilities. $\mathscr{P}_{xay}$ is the probability of moving from state x to state y if action a is chosen.
- $r : \mathscr{K} \to \mathbb{R}$ is an immediate reward.

We define the history at time t to be the sequence of previous states and actions, as well as the current state: $h_t = (x_1, a_1, x_2, a_2, \ldots, x_{t-1}, a_{t-1}, x_t)$. Let $\mathbf{H}_t$ be the set of all possible histories of length t. A policy u in the class of all policies U is a sequence $(u_1, u_2, \ldots)$. If the history h_t is observed at time t, then the controller chooses an action a with probability $u_t(a|h_t)$. A policy is called a Markov policy ($u \in U_M$) if for any t, u_t only depends on the state at time t. A stationary policy ($u \in U_S$) is a Markov policy that does not depend on t. Under a stationary policy, the state process becomes a Markov chain with transition probabilities $P_{xy}[u] = \sum_{a \in A(x)} \mathscr{P}_{xay} u(a|x)$. Finally, a stationary policy is a deterministic policy ($u \in U_D$) if it selects an action with probability one. Then u is identified with a map $u : \mathbf{X} \to A$.

We fix an initial distribution ν over the initial states. In other words, the probability that we are at state x at time 1 is $\nu(x)$. If ν is concentrated on a single state z, we use the Dirac notation $\nu(x) = \delta_z(x)$. Kolmogorov's extension theorem guarantees that the initial distribution ν and any given policy u determine a unique probability measure $\mathbb{P}^u_\nu$ over the space of trajectories of the states X_t and actions A_t. We denote $\mathbb{E}^u_\nu$ the corresponding expectation operation.

For any policy u and initial distribution ν, and for a discount factor $0 < \alpha < 1$, we define

$$
\begin{aligned}
R_\alpha(\nu, u) &= (1-\alpha) \mathbb{E}^u_\nu \sum_{t=1}^{\infty} \alpha^{t-1} r(X_t, A_t) \\
&= (1-\alpha) \sum_{t=1}^{\infty} \alpha^{t-1} \mathbb{E}^u_\nu r(X_t, A_t)
\end{aligned}
$$

(the exchange of limit and expectation is valid in the case of finitely many states and actions using the dominated convergence theorem).

An occupation measure corresponding to a policy u is the total expected discounted time spent in different state-action pairs. More precisely, we define for any initial distribution ν, any policy u and any pair $x \in \mathbf{X}, a \in A(x)$:

$$
f_\alpha(\nu, u; x, a) := (1-\alpha) \sum_{t=1}^{\infty} \alpha^{t-1} \mathbb{P}^u_\nu (X_t = x, A_t = a).
$$

The set $\{f_\alpha(\nu, u; x, a)\}_{x,a}$ defines a probability measure $f_\alpha(\nu, u)$ on the space of state-action pairs that assigns probability $f_\alpha(\nu, u; x, a)$ to the pair (x, a). $f_\alpha(\nu, u)$ is called an occupation measure and is associated to a stationary policy w defined by:

$$
w(a|y) = \frac{f_\alpha(\nu, u; y, a)}{\sum_{a \in A} f_\alpha(\nu, u; y, a)}, \forall y \in \mathbf{X}, a \in A(y), \qquad (1)
$$

whenever the denominator is non-zero, in which case we can choose $w(a|y)$ arbitrarily. We can readily check that

$$
R_\alpha(\nu, u) = \sum_{x \in \mathbf{X}} \sum_{a \in A} f_\alpha(\nu, u; x, a) \, r(x, a). \qquad (2)
$$

Let $Q^\alpha(\nu)$ to be the set of vectors $\rho \in \mathbb{R}^{|\mathscr{K}|}$ satisfying

$$
\begin{cases}
\sum_{y \in \mathbf{X}} \sum_{a \in A(y)} \rho_{y,a} (\delta_x(y) - \alpha \mathscr{P}_{yax}) = (1-\alpha)\nu(x), \forall x \in \mathbf{X} \\
\rho_{y,a} \geq 0, \quad \forall y \in \mathbf{X}, \, a \in A(y).
\end{cases} \qquad (3)
$$

$Q^\alpha(\nu)$ is a closed polyhedron. Note that by summing the first constraints over x we obtain $\sum_{y,a} \rho_{y,a} = 1$, so ρ satisfying the above constraints defines a probability measure. It also follows that $Q^\alpha(\nu)$ is bounded, i.e., is a closed polytope. One can check that the occupation measures $f_\alpha(\nu, u)$ belong to this polytope. In fact $Q^\alpha(\nu)$ describes exactly the set of occupation measures achievable by all policies (See [13] for a proof): the extreme points of the polytope $Q_\alpha(\nu)$ correspond to deterministic policies, and each policy can be obtained as a randomization over these deterministic policies. Thus, one can obtain a (deterministic) optimal occupation measure corresponding to the maximization of (2) as the solution of a linear program over the polytope $Q^\alpha(\nu)$.

B. Exact Formulation of the RBSC Problem

In the RBSC problem, N projects are distributed in space at N sites, and one server can be allocated to a chosen project at each time period $t = 1, 2, \ldots$. In the following, we use the terms project and site interchangeably. At each time period, the server must occupy one site. We say that a site is active at time t if it is visited by the server, and is passive otherwise. If the server travels from site k to site l, we incur a cost c_{kl}. Each site can be in one of a finite number of states $x_n \in S_n$, for $n = 1, \ldots, N$, and we denote the Cartesian product of the individual state spaces $\mathscr{S} = S_1 \times \ldots \times S_N$. If site n in state x_n is visited, a reward $r^1_n(x_n)$ is earned, and its state changes to y_n according to the transition probability $p^1_{x_n y_n}$. If the site is not visited, then a reward (potentially negative) $r^0_n(x_n)$ is earned for that site and its state changes according to the transition probabilities $p^0_{x_n y_n}$. *We assume that all sites change their states independently of each other.* Note that if the transition costs are all 0, we recover the initial formulation of the RB problem [4] for one server. If additionally the passive rewards are 0 and the passive transition matrix is the identity matrix, we obtain the MAB problem. If we just add the switching costs to the basic MAB problem, we call the resulting model MABSC.

The RBSC problem can be cast into the MDP framework. We denote the set $\{1, \ldots, N\}$ by $[N]$. The state of the system at time t is given by the state of each site and the position $s \in [N]$ of the server. We denote the complete state by

$(x_1,\ldots,x_N;s) := (\mathbf{x};s)$. We can choose which site is to be visited next; i.e., the action a belongs to the set $[N]$. Once the next site to be visited is chosen, there is a cost c_{sa} for moving to the new site, including possibly a nonzero cost for staying at the same site. The reward earned is $r_a^1(x_a)+\sum_{j\neq a} r_j^0(x_j)$. We are given a distribution ν on the initial state of the system, for example $\nu(x_1,\ldots,x_N;s) = \nu_1(x_1)\ldots\nu_N(x_N)\delta_1(s)$ if the initial states of the sites are independent and the server leaves initially from site 1. We also define the notation $\sum_{\mathbf{x}\in\mathscr{S}} = \sum_{x_1\in S_1}\cdots\sum_{x_N\in S_N}$. Then the linear program for the resulting MDP can be written:

maximize

$$\sum_{s=1}^{N}\sum_{\mathbf{x}\in\mathscr{S}}\sum_{a=1}^{N}(r_a^1(x_a)+\sum_{j\neq a} r_j^0(x_j)-c_{sa})\rho_{(x_1,\ldots,x_N;s),a} \tag{4}$$

subject to

$$\sum_{a=1}^{N}\rho_{(\mathbf{x};s),a}-\alpha\sum_{s'=1}^{N}\sum_{\mathbf{x}'\in\mathscr{S}}\sum_{a=1}^{N}\rho_{(\mathbf{x}';s'),a}\mathscr{P}_{(\mathbf{x}';s')a(\mathbf{x};s)} = (1-\alpha)\,\nu(\mathbf{x};s), \quad \forall(\mathbf{x},s)\in\mathscr{S}\times[N]$$

$$\rho_{(\mathbf{x};s),a}\geq 0, \quad \forall((\mathbf{x};s),a)\in\mathscr{S}\times[N]^2,$$

with the decision variables $\rho_{(\mathbf{x};s),a}$ corresponding to the occupation measures.

In our case, a non-zero transition probability occurs only if the target site is the same as our selected action because we assume that the movement of the server is deterministic. Using the independence assumption, we obtain the following equality constraints on the variables:

$$\sum_{a=1}^{N}\rho_{(\mathbf{x};s),a}-\alpha\sum_{s'=1}^{N}\sum_{\mathbf{x}'\in\mathscr{S}}\rho_{(\mathbf{x}';s'),s}\,p^0_{x'_1,x_1}\cdots p^1_{x'_s x_s}\cdots p^0_{x'_N,x_N} = (1-\alpha)\,\nu(\mathbf{x},s), \quad \forall(\mathbf{x},s)\in\mathscr{S}\times[N] \tag{5}$$

In this formulation, our decision variables are $\rho_{(x_1,\ldots,x_N;s),a}$. Thus, there are $|S_1|\times\ldots\times|S_N|\times N^2$ variables, i.e., a number exponential in the size of the input data. For example, if we consider a problem with $N=10$ sites and 5 states for each site, we obtain a linear program with more than 976×10^6 variables, and therefore real world instances of the problem cannot be solved by feeding this formulation directly into an LP solver.

III. Linear Programming Relaxation of the RBSC

A. Complexity of the RBSC and MABSC problems.

It could well be that the problem appears difficult to solve only because of our own inability to formulate it in an efficient manner. In the most general formulation above, we know however that the problem is likely to be intractable, since the RB problem is already PSPACE-hard [5], even for the case of deterministic transition rules and one server. Here we show that the special case MABSC is also difficult. In this Section, we also denote (with a slight abuse of notation) MABSC as the recognition version of the optimization problem considered before; that is, given an instance of the MABSC problem and a real number L, is there a policy that achieves a total expected reward greater than or equal to L? This problem is obviously easier than the full version of the optimization problem. The fact that it is NP-hard is deduced from the same result for the HAMILTON CIRCUIT problem.

Theorem 1: MABSC is *NP*-hard.

Proof: Recall that in the HAMILTON CIRCUIT problem, we are given a graph $G=(V,E)$, and we want to know if there is a circuit in G visiting all nodes exactly once. HAMILTON CIRCUIT was actually one of the first combinatorial problems proven to be NP-complete [14]. HAMILTON CIRCUIT is a special case of MABSC. Indeed, given any graph $G=(V,E)$, we construct an instance of MABSC with $N=|V|$ sites, travel costs $c_{ij}=1$ if $\{i,j\}\in E$, and $c_{ij}=2$ otherwise. We choose arbitrarily one of the sites to be the site where the server is initially present. This site has only one state, and the reward for having the server present at this site at any period is 0. To each of the $(N-1)$ other sites, we associate a Markov chain with two states. To the first state is associated a reward of 2 (discounted by α at each time period), and after a visit to a site in this state, the site moves with probability 1 to the second state, associated with a reward of (-1) (discounted by α at each time period). Once in this state, the chain remains there with probability 1.

Now since $\alpha>0$, it is clear that a policy can achieve a reward of $1+\alpha+\ldots+\alpha^{N-2}-\alpha^{N-1}$ if and only if there exists a Hamilton circuit in G. The only possible policy actually just moves the server along the Hamilton circuit without stopping, except when the server is back at the initial site. ■

Note that this easy result is in strong opposition to the case without switching costs, where a greedy policy based on the Gittins indices for each site (computable in polynomial time) is known to be optimal. However, the result is not completely satisfying. It captures the complexity of the combinatorial problem present in MABSC (which looks like a traveling salesman problem) but not the complexity of the whole scheduling problem. Indeed, even in the case of two sites, the problem remains in general difficult (see for example [11]).

B. A Relaxation for the RBSC

The discussion in the previous paragraph serves as a justification for the introduction of a relaxed formulation of the RBSC and MABSC problems. The MAB framework naturally leads to a Markov decision process for each site. This observation was already used by Whittle [4] in the case of restless bandits, together with a relaxation of the constraints tying the projects together. This relaxation allowed him to decompose the formulation by project ([15] uses the same idea). Here we essentially extend the method to RBSC. References for our work include, in particular, the paper by Bertsimas and Niño-Mora on restless bandits [1].

Consider a policy u and a distribution ν on the initial states of the sites; the initial states are assumed independent (i.e. $\nu(x_1,\ldots,x_N;s) = \nu_1(x_1)\ldots\nu_N(x_N)\delta_1(s)$). These generate an occupation measure for each site $f^i_\alpha(\nu,u;(x_i;s),a), i = 1,\dots,N$, where

$$f^i_\alpha(\nu,u;(x_i;s),a) = (1-\alpha)\,\mathbb{E}^u_\nu \sum_{t=1}^{\infty} \alpha^{t-1} 1_{\{X^i_t=x_i,S_t=s,A_t=a\}}.$$

These occupation measures can be thought of as projections of the measure for the complete problem [16], or in terms of probabilities as marginals. Indeed, we have

$$f^i_\alpha(\nu,u;(x_i;s),a) = \\ \sum_{x_1\in S_1} \cdots \sum_{x_{i-1}\in S_{i-1}} \sum_{x_{i+1}\in S_{i+1}} \cdots \sum_{x_N\in S_N} f_\alpha(\nu,u;(x_1,\dots,x_N;s),a).$$

By partial summation over the constraints in (5), one can see that these measures belong to (*and now in general the inclusion is strict*) the polytopes $Q^\alpha_i(\nu_i)$ defined as follows:

$$Q^\alpha_i(\nu_i) = \left\{ \begin{array}{l} \rho^i_{(x_i;s),a} \in \mathbb{R}_+^{|S_i|\times N^2} : \\ \sum_{a=1}^N \rho^i_{(x_i;i),a} - \alpha \sum_{x'_i\in S_i} \sum_{s'=1}^N \rho^i_{(x'_i;s'),i} p^1_{x'_i x_i} = \\ \qquad (1-\alpha)\nu_i(x_i)\delta_1(i) \quad \forall x_i \in S_i \\ \sum_{a=1}^N \rho^i_{(x_i;s),a} - \alpha \sum_{x'_i\in S_i} \sum_{s'=1}^N \rho^i_{(x'_i;s'),s} p^0_{x'_i x_i} = \\ \qquad (1-\alpha)\nu_i(x_i)\delta_1(s) \quad \forall x_i \in S_i,\, s \neq i \end{array} \right\} \tag{6}$$

For all N sites, we have now a total of $O(N^3 \times \max_i(|S_i|))$ variables and constraints, i.e., a number polynomial in the size of the input, and therefore from the discussion in paragraph III-A it is unlikely that these variables will suffice to formulate the RBSC problem exactly. However, we can try to reduce the size of the feasible region spanned by the new decision vectors, by finding additional constraints not present in (6). Indeed, the occupation measures for each site are projections of the same original vector and therefore are tied together.

We can use the intuitive idea of enforcing constraints on average to relax a hard problem. At a given time t, the server is switching from one site to exactly one other site, and all marginal occupation measures should reflect the same change on the information about the server position. That is, at each time period t, we have:

$$\sum_{x_i\in S_i} 1_{\{X^i_t=x_i,S_t=s,A_t=a\}} = \sum_{x_1\in S_1} 1_{\{X^1_t=x_1,S_t=s,A_t=a\}}, \quad \forall i,s,a \in [N].$$

We relax these constraints to enforce them only on average, which implies for the occupation measures

$$\sum_{x_i\in S_i} \rho^i_{(x_i;s),a} = \sum_{x_1\in S_1} \rho^1_{(x_1;s),a}, \quad \forall i,s,a \in [N]. \tag{7}$$

Now note that, in fact, this intuitive interpretation leads to an equation that is clearly true by simply summing the original occupation measure over all states of all sites, but not over s and a. The idea of relaxing a constraint enforced at each time step into a constraint enforced on average was central in the original work of Whittle [4]. We see here that the state-action frequency approach allows us to derive additional constraints between projects automatically.

Our final linear programming relaxation for RBSC is:

maximize

$$\sum_{s=1}^N \sum_{a=1}^N \left[\sum_{x_a\in S_a} (r^1_a(x_a) - c_{sa})\rho^a_{(x_a;s),a} + \sum_{j\neq a} \sum_{x_j\in S_j} r^0_j(x_j)\rho^j_{(x_j;s),a} \right] \tag{8}$$

subject to

$$\boldsymbol{\rho^i} := \{\rho^i_{(x_i;s),a}\}_{\{x_i,s,a\}} \in Q^\alpha_i(\nu_i), \quad \forall i \in [N]$$
$$\sum_{x_i\in S_i} \rho^i_{(x_i;s),a} = \sum_{x_1\in S_1} \rho^1_{(x_1;s),a}, \quad \forall i,s,a \in [N].$$

As noted earlier, the number of variables and constraints in this program is polynomial in the size of the input. Computing the optimal value of this linear program can therefore be done in polynomial time, and provides an upper bound on the performance achievable by any policy for the original problem.

A few remarks can be made about this formulation. First, we could obtain tighter relaxations by considering marginals involving several sites at the same time. In the limit case, we obtain the exact formulation when all sites are considered simultaneously. This idea is followed in [1]. Let's also mention that in the original work on restless bandits, Whittle used Bellman's equation of optimality to formulate the problem. Additional constraints tying the projects together can then be handled using the theory of Lagrange multipliers. If the state-frequency approach is often preferred to deal with constrained MDPs, the dynamic programming and Lagrange multipliers approach has sometimes been used as well (eg. in [15]), since it allows the use of the various algorithms developed for dynamic programming problems. If the linear programming method is used to solve the dynamic program, the corresponding linear program is the dual of the one obtained using occupation measures. We can of course obtain the dual of (8) directly:

$$\text{minimize } (1-\alpha) \sum_{i=1}^N \sum_{s=1}^N \sum_{x_i\in S_i} \nu_i(x_i)\delta_1(s)\lambda^i_{x_i,s} \tag{9}$$

subject to

$$\lambda^1_{x_1,s} - \alpha \sum_{x'_1\in S_1} p^1_{x_1x'_1}\lambda^1_{x'_1,1} + \sum_{i=2}^N \mu^i_{s,1} \geq r^1_1(x_1) - c_{s1}, \quad s\in[N], x_1 \in S_1$$

$$\lambda^1_{x_1,s} - \alpha \sum_{x'_1\in S_1} p^0_{x_1x'_1}\lambda^1_{x'_1,a} + \sum_{i=2}^N \mu^i_{s,a} \geq r^0_1(x_1), \quad s\in[N], x_1 \in S_1, a\neq 1$$

$$\lambda^j_{x_j,s} - \alpha \sum_{x'_j\in S_j} p^1_{x_jx'_j}\lambda^j_{x'_j,j} - \mu^j_{s,j} \geq r^1_j(x_j) - c_{sj}, \quad j\geq 2, s\in[N], x_j \in S_j$$

$$\lambda^j_{x_j,s} - \alpha \sum_{x'_j\in S_j} p^0_{x_jx'_j}\lambda^j_{x'_j,a} - \mu^j_{s,a} \geq r^0_j(x_j), \quad j\geq 2, s\in[N], x_j \in S_j, a\neq j.$$

IV. Heuristics for the RBSC problem

We now present algorithms to solve the RBSC problem in practice. On specific examples, when the optimal solution is too costly to compute, the relaxation presented in section III provides an upper bound on the achievable performance. Based on ideas developed in [1], we present a primal-dual heuristic obtained from the linear programming relaxation. We then show that this heuristic can be viewed as a one-step lookahead policy [17]. Experimentally, this heuristic performed well over a wide range of problems.

A. A Simple Greedy Algorithm

Perhaps the simplest policy for the RBSC problem is the following greedy policy: in state $(x_1,\dots,x_N;s)$, send the server to the site that maximizes the marginal instantaneous reward, i.e., $a_{greedy} \in \text{argmax}_a\{r_a^1(x_a) - r_a^0(x_a) - c_{sa}\}$. This policy is optimal in the case of the MAB problem, with no transition costs, and deteriorating rewards (i.e., projects become less profitable as they are worked on)[3]. It is also a one-step lookahead policy where we approximate the cost-to-go by 0 (see (12)).

B. Reconstructing the Original Occupation Measures

Solving the linear program (8) provides values for the marginals of the original occupation measures. Moreover, each sum in (7) can be interpreted as the probability for the server to be at site s and switch to site a. Then, one can try to reconstruct occupation measures for the original problem, for example by defining

$$\tilde{\rho}_{(x_1,\dots,x_N;s),a} = \frac{1}{(P(s,a))^{N-1}}\left[\rho^1_{(x_1;s),a}\cdots\rho^N_{(x_N;s),a}\right] \qquad (10)$$

with $P(s,a) = \sum_{x_1\in S_1}\rho_{(x_1;s),a}$. Then from (7), we immediately have $\sum_{j\neq i}\sum_{x_j\in S_j}\tilde{\rho}_{(x_1,\dots,x_N;s),a} = \rho^i_{(x_i;s),a}$. Now the algorithm simply chooses, in state $(x_1,\dots,x_N;s)$, action $\tilde{a}$ with probability $\tilde{\rho}_{(x_1,\dots,x_N;s),\tilde{a}}/\sum_{a=1}^N\tilde{\rho}_{(x_1,\dots,x_N;s),a}$ if the denominator is non zero, or leaves the server at its current position otherwise. Of course, these new variables $\tilde{\rho}$ do not satisfy (4) in general, and it is difficult to justify this construction theoretically.

C. A Primal-Dual Index Heuristic

1) Construction from the LP relaxation: For the linear programs (8) and (9), we can obtain optimal primal and dual solutions $\{\bar{\rho}^i_{(x_i;s),a}\}$ and $\{\bar{\lambda}^i_{x_i,s},\bar{\mu}^i_{s,a}\}$. We call $\{\bar{\gamma}^i_{(x_i;s),a}\}$ the corresponding reduced costs, which are given by the difference between the left and right hand side in the constraints of (9). These reduced costs are nonnegative, there is one such coefficient corresponding to each variable of the primal $\bar{\rho}^i_{(x_i;s),a}$, and additionally by complementary slackness $\bar{\gamma}^i_{(x_i;s),a} = 0$ if $\bar{\rho}^i_{(x_i;s),a} > 0$. Bertsimas and Niño-Mora motivated their heuristic for the RB problem using the following interpretation of the reduced costs: *starting from an optimal solution, $\bar{\gamma}^i_{(x_i;s),a}$ is the rate of decrease in the objective value of the primal linear program (8) per unit increase in the value of the variable $\rho^i_{(x_i;s),a}$.*

We use this interpretation and the intuitive idea that taking action a when project i is in state x_i and the server at s implies in some sense increasing the value of $\rho^i_{(x_i;s),a}$. In particular, we would like to keep the quantities $\bar{\rho}^i_{(x_i;s),a}$ found to be 0 in the relaxation as close to 0 as possible in the final solution. Since for these variables we can have $\bar{\gamma}^i_{(x_i;s),a} > 0$, when the system is in state $(x_1,\dots,x_N;s)$, we associate to each action a an index of undesirability

$$I((x_1,\dots,x_N;s),a) = \sum_{i=1}^N \bar{\gamma}^i_{(x_i;s),a},$$

that is, we sum the reduced costs for the N different projects. Then we select action a_{pd} that minimizes these indices:

$$a_{pd}(x_1,\dots,x_N;s) \in \text{argmin}_a\{I((x_1,\dots,x_N;s),a)\}. \qquad (11)$$

Note that this procedure always provides a deterministic policy.

2) Interpretation as a One-Step Lookahead Policy: There is an interesting alternate way of viewing the primal-dual heuristic from a dynamic programming point of view. For state $(x_1,\dots,x_N;s)$, we form an approximation of the (infinite-horizon) cost to go as: $\tilde{J}(x_1,\dots,x_N;s) = \sum_{i=1}^N\bar{\lambda}^i_{x_i,s}$. By summing over the constraints in (9), one can readily see that the vector $\tilde{J}$ is a feasible vector for the linear program obtained from Bellman's equation for the original problem, or alternatively as the dual of (4). That is, $\tilde{J}$ is a superharmonic vector, and we recall that the exact optimal cost is the smallest superharmonic vector [18]. By obtaining a tight relaxation, including the additional constraints on the marginals, we can obtain a vector $\tilde{J}$ which is closer to the optimal cost-to-go vector. Now the one-step lookahead policy is obtained by maximizing

$$\max_{a\in[N]}\left\{(r_a^1(x_a) + \sum_{j\neq a} r_j^0(x_j) - c_{sa}) + \alpha\sum_{\mathbf{x}'\in\mathscr{S}} p^0_{x_1x_1'}\dots p^1_{x_ax_a'}\dots p^0_{x_Nx_N'}\tilde{J}(x_1',\dots,x_N';a)\right\}$$
$$= \max_{a\in[N]}\left\{(r_a^1(x_a) + \sum_{j\neq a} r_j^0(x_j) - c_{sa}) + \alpha(\sum_{i\neq a}\sum_{x_i'\in S_i} p^0_{x_ix_i'}\bar{\lambda}^i_{x_i',a} + \sum_{x_a'\in S_a} p^1_{x_ax_a'}\bar{\lambda}^a_{x_a',a})\right\}$$

It is a straightforward calculation, using the definition of the reduced costs given above, to verify that this maximization is equivalent to the minimization in (11).

3) Computational Aspects and Approximate Indices: The one-step lookahead policy above can be computed in real-time in practice. Note that we can rewrite $\sum_{j\neq a} r_j^0(x_j)$ as $-r_a^0(x_a) + \sum_{j=1}^N r_j^0(x_j)$ and the sum does not depend on a. Therefore the one-step lookahead heuristic is implemented on-line by choosing in state $(x_1,\dots,x_N;s)$ the action that maximizes the approximate indices

$$a_{pd} \in \text{argmax}_a\{m_{\mathbf{x},s}(a)\}, \qquad (12)$$
$$m_{\mathbf{x},s}(a) = r_a^1(x_a) - r_a^0(x_a) - c_{sa} + \alpha\left(\sum_{i\neq a}\sum_{x_i'\in S_i} p^0_{x_ix_i'}\bar{\lambda}^i_{x_i',a} + \sum_{x_a'\in S_a} p^1_{x_ax_a'}\bar{\lambda}^a_{x_a',a}\right).$$

This expression motivates the definition of the simple greedy policy given previously. We can store the $O(N^2\max_i(|S_i|))$ optimal dual variables $\bar{\lambda}^i_{x_i,s}$ and compute these indices in time $O(N^2\max_i(|S_i|))$. Equivalently, as we have seen above, we can store the $O(N^3\max_i(|S_i|))$ optimal reduced costs and compute the indices appearing in (11) in time $O(N^2)$.

V. Numerical Experiments

We now consider problems whose characteristics differently affect the performance of the heuristics presented in section IV. We also compute the optimal performance as the solution of the linear program (4) when allowed by the size of the state space, and an upper bound on this performance using the relaxation (8) in any case. Linear programs are implemented in AMPL and solved using CPLEX. Due to the size of the state space, the expected discounted reward

TABLE I
NUMERICAL EXPERIMENTS

Problem	α	Z^*	Z^1	Z_{greedy}	$\tilde{Z}$	Z_{pd}
Problem 1	0.9	96.4	109.4	96	94	70
Problem 2	0.9	-	281.7	246	186	213
Problem 3	0.9	-	195.3	82	96	107
Problem 4	0.9	-	1799.4	1680	1450	1713
Problem 5	0.9	-	1121.5	985	648	943

of the various heuristics is computed using Monte-Carlo simulations. The computation of each trajectory is terminated after a sufficiently large, but finite horizon: in our case, when α^t times the maximal absolute value of any immediate reward becomes less than 10^{-6}. The server is assumed to start from site 1. To reduce the amount of computation in the evaluation of the policies, we assume that the distribution of the initial states is deterministic.

We adopt the following nomenclature:

- Z^*: Optimal value of the problem (when available).
- Z^1: Optimal value of the relaxation.
- Z_{greedy}: Estimated expected value of the greedy heuristic.
- $\tilde{Z}$: Estimated expected value of the heuristic reconstructing occupation measures from (10).
- Z_{pd}: Estimated expected value of the primal-dual index heuristic (i.e., one-step lookahead).

Problem 1 is a 2-armed bandit (i.e., passive projects are frozen, passive rewards are 0, and there are no switching costs), with 10 states for each project and deteriorating rewards. Therefore the greedy heuristic performs optimally. Problem 2 is an MAB problem with 8 projects, 10 states per project and deteriorating rewards. Z^* was not computed but can be obtained from the simulation of the greedy policy, since we know that it is optimal. Problem 3 is a MABSC problem, with 5 projects, 10 states per project and deteriorating rewards. One of the projects has a high immediate reward at the beginning, but is relatively remote, so we expected the greedy policy to perform poorly. Problems 4 and 5 were RBSC problems with randomly generated data.

The results of the numerical experiments on the different problems are given in table I. Over the range of problems, the primal-dual index heuristic performs reasonably well. Note that since we do not assume the distances to be symmetric, it is very easy to design a problem where the greedy policy performs badly by associating to a site a high immediate reward at the beginning, but also a high switching cost to leave this site. The gap between the original optimal value and the relaxation is seen to be less than 15% in all examples, except possibly in problem 3 where none of the policies approaches the bound. It is not clear at this point which factors make the primal-dual index heuristic underperform.

VI. CONCLUSIONS

In the spirit of existing work on the restless bandits problem, we have presented a linear programming relaxation for the restless bandits with switching costs problem. This set-up is quite powerful to model a wide range of dynamic resource allocation problems. Since the problem is computationally intractable and an optimal solution can in general not be directly computed, the relaxation is useful in providing a bound on the achievable performance. We also presented several heuristics to solve the problem in practice, as well as their performances on specific examples. An interesting part of the analysis showed the link between the linear programming relaxation and a standard suboptimal control method. We also showed that the relaxation can be obtained automatically by first formulating the original problem as a Markov decision process on the whole state space and then considering specific marginals of the occupation measure. Future work will focus on trying to design, for special cases, approximation algorithms with a guaranteed worst-case performance.

REFERENCES

[1] D. Bertsimas and J. Niño-Mora, "Restless bandits, linear programming relaxations, and a primal-dual index heuristic," *Operations Research*, vol. 48, pp. 80–90, 2000.

[2] J. Gittins and D. Jones, "A dynamic allocation index for the sequential design of experiments," in *Progress in Statistics*, J. Gani, Ed. Amsterdam: North-Holland, 1974, pp. 241–266.

[3] J. Gittins, *Multi-armed Bandit Allocation Indices*. New York: John Wiley and sons, 1989.

[4] P. Whittle, "Restless bandits: activity allocation in a changing world," *Journal of Applied Probability*, vol. 25A, pp. 287–298, 1988.

[5] C. Papadimitriou and J. Tsitsiklis, "The complexity of optimal queueing network control," *Mathematics of Operations Research*, vol. 24, no. 2, pp. 293–305, 1999.

[6] M. Van Oyen, D. Pandelis, and D. Teneketzis, "Optimality of index policies for stochastic scheduling with switching penalties," *Journal of Applied Probability*, vol. 29, pp. 957–966, 1992.

[7] T. Jun, "A survey on the bandit problem with switching costs," *De Economist*, vol. 152, no. 4, pp. 513–541, 2004.

[8] J. Niño-Mora, "Restless bandits, partial conservation laws and indexability," *Advances in Applied Probability*, vol. 33, pp. 76–98, 2001.

[9] K. Glazebrook and D. Ruiz-Hernandez, "A restless bandit approach to stochastic scheduling problems with switching costs," March 2005, working paper.

[10] P. Varaiya, J. Walrand, and C. Buyukkoc, "Extensions of the multiarmed bandit problem: the discounted case," *IEEE transactions on automatic control*, vol. 30, no. 5, pp. 426–439, 1985.

[11] M. Asawa and D. Teneketzis, "Multi-armed bandits with switching penalties," *IEEE transactions on automatic control*, vol. 41, no. 3, pp. 328–348, 1996.

[12] F. Dusonchet and M.-O. Hongler, "Optimal hysteresis for a class of deterministic deteriorating two-armed bandit problem with switching costs," *Automatica*, vol. 39, no. 11, pp. 1947–1955, 2003.

[13] E. Altman, *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

[14] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. Miller and J. Thatcher, Eds. New York: Plenum Press, 1972.

[15] D. Castanon, "Approximate dynamic programming for sensor management," in *Proceedings of the 36th Conference on Decision and Control*, December 1997, pp. 1202–1207.

[16] J. Niño-Mora, "Optimal ressources allocation in a dynamic and stochastic environment: A mathematical programming approach," Ph.D. dissertation, Massachusetts Institute of Technology, 1995.

[17] D. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed. Athena Scientific, 2001, vol. 1.

[18] L. Kallenberg, "Survey of linear programming for standard and nonstandard Markovian control problems, Part I: Theory," *ZOR - Methods and Models in Operations Research*, vol. 40, pp. 1–42, 1994.