Multi-Agent Task Assignment in the Bandit Framework

Jerome Le Ny, Munther Dahleh and Eric Feron

***Abstract*— We consider a task assignment problem for a fleet of UAVs in a surveillance/search mission. We formulate the problem as a restless bandits problem with switching costs and discounted rewards: there are N sites to inspect, each one of them evolving as a Markov chain, with different transition probabilities if the site is inspected or not. The sites evolve independently of each other, there are transition costs c_{ij} for moving between sites i and $j \in \{1,\dots,N\}$, rewards when visiting the sites, and we maximize a mixed objective function of these costs and rewards. This problem is known to be PSPACE-hard. We present a systematic method, inspired from the work of Bertsimas and Niño-Mora [1] on restless bandits, for deriving a linear programming relaxation for such locally decomposable MDPs. The relaxation is computable in polynomial-time offline, provides a bound on the achievable performance, as well as an approximation of the cost-to-go which can be used online in conjunction with standard suboptimal stochastic control methods. In particular, the one-step lookahead policy based on this approximate cost-to-go reduces to computing the optimal value of a linear assignment problem of size N. We present numerical experiments, for which we assess the quality of the heuristics using the performance bound.**

I. INTRODUCTION

In the past decade or so, teams of autonomous collaborating unmanned aerial vehicles (UAVs) have started to be actively used to perform various tasks of surveillance, reconnaissance and information gathering in remote or dangerous environments. Yet, the problem of assigning tasks to these vehicles is in general a complex stochastic control problem, and there is still a need to develop efficient methods aimed at optimizing the performance of these multi-agent systems. The multi-armed bandit (MAB) problem has long been recognized as an important model to help researchers in this task, because of its generality coupled with an efficiently computable solution. Gittins devote a chapter of his book [2] to the application of the MAB model to search theory. Whittle introduced one of the most interesting extension, the restless bandits (RB) problem [3], describing a potential application to a fleet of aircrafts trying to track the positions of enemy submarines. As long as the targets are assumed to evolve independently (which is usually done for tractability), the sensor management problem becomes essentially a multi-armed bandit or restless bandits problem where one controls the information states of the targets [4], [5].

Unlike the basic multi-armed bandit problem however, an optimal solution to the more expressive restless bandits problem is unlikely to be computable efficiently, since the problem is known to be PSPACE-hard [6]. Moreover, in the case of moving sensors onboard UAVs, an important component to take into account in the objective function are switching penalties for changing targets, adding a traveling salesman-like feature to an already difficult problem (the multi-armed bandit problem with switching costs is NP-hard).

In this paper, we extend our previous work on polynomial-time relaxations of the restless bandits problem with switching costs (RBSC) [7], from the single agent to the multi-agent case. In Section II, we formulate the RBSC problem in the framework of Markov decision processes (MDP). In Section III, we show how the local structure of the objective function (decomposable by sites to inspect) and the assumption on the independent evolution of the states of the sites allow us to derive systematically a relaxation for this problem. The performance measures used in a similar way in [1] for the restless bandits problem are always marginals of the state-action frequencies appearing in the exact formulation. The relaxation provides an efficiently computable bound on the achievable performance. Section IV describes how this relaxation can be used to create heuristics to solve the problem in practice, and presents numerical experiments comparing the heuristics to the performance bound. Section V presents a preliminary analysis toward an a priori performance bound for our heuristics.

II. MULTI-AGENT RBSC: EXACT FORMULATION

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: $h_t = (x_1, a_1, x_2, a_2, \dots, x_{t-1}, a_{t-1}, x_t)$. 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 $(u_1, u_2, \dots)$. If the history h_t

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

J. Le Ny and M. Dahleh are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139-4307, USA `jleny@mit.edu, dahleh@mit.edu.`

E. Feron is with the School of Aerospace Engineering, Georgia Tech, Atlanta, GA 30332, USA `eric.feron@aerospace.gatech.edu`

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

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

(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:

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

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

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

We define for any class of policies U_1

$$\mathbf{L}^\alpha_{U_1}(\nu) = \cup_{u \in U_1} f_\alpha(\nu,u),$$

i.e., the set of vectors described by the class of policies considered. Also, let $Q^\alpha(\nu)$ to be the set of vectors $\rho \in \mathbb{R}^{|\mathscr{K}|}$ satisfying

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

$Q^\alpha(\nu)$ 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, i.e., $\mathbf{L}^\alpha_U(\nu) \subseteq Q_\alpha(\nu)$. The following theorem states that $Q_\alpha(\nu)$ describes in fact exactly the set of occupation measures achievable by all policies, and that each policy can be obtained as a randomization over deterministic policies, which represent the extreme points of the polytope $Q_\alpha(\nu)$. See [8] for a proof under a more general form.

Theorem 1: $\mathbf{L}^\alpha_U(\nu) = \mathbf{L}^\alpha_{U_S}(\nu) = \overline{conv}\,\mathbf{L}^\alpha_{U_D}(\nu) = Q^\alpha(\nu)$.

Since deterministic policies represent the extreme points of the polytope of occupation measures, we know from standard LP theory that it is sufficient to look among the deterministic policies for a policy maximizing $R_\alpha(\nu,u)$. Moreover, one can obtain an 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 $M \leq N$ servers can be allocated to M different projects at each time period $t = 1,2,\ldots$ In the following, we use the terms project and site interchangeably; likewise, agent and server have the same meaning (they are the UAVs in our application). At each time period, each server must occupy one site, and different servers must occupy distinct sites. We say that a site is active at time t if it is visited by a server, and is passive otherwise. If a 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 [3]. If in addition 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.

We denote the set $\{1,\ldots,N\}$ by $[N]$. We consider that when no agent is present at a given site, there is a fictitious agent called passive agent at that site. We also call the real agents active agents, since they collect active rewards. The transition of a passive agent between sites does not involve any switching cost, and when a passive agent is present at a site, the passive reward is earned. Therefore, we have a total of N agents including both the real and passive agents, and we can describe the positions of all agents by a vector $\mathbf{s} = (s_1,\ldots,s_N)$, which corresponds to a *permutation* of [N] (due to our constraint that different agents must occupy different sites). We denote the set of these pemutation vectors by $\Pi_{[N]}$. The M first components correspond to the real agents. For example, with $M = 2$ and $N = 4$, the vector $(s_1 = 2, s_2 = 3, s_3 = 1, s_4 = 4) \in \Pi_{[4]}$ means that agent 1 is in site 2, agent 2 in site 3 and sites 1 and 4 are passive.

For an agent $i \in [N]$, we refer to the other agents by $-i$. If we fix $s_i \in [N]$ for some $1 \le i \le N$, then we write $\mathbf{s}_{-i}$ to denote the vector $(s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_N)$, and $\Pi_{[N]-s_i}$ to denote the permutations of the set $[N] - \{s_i\}$. In particular, we write $\sum_{\mathbf{s}_{-i} \in \Pi_{[N]-s_i}}$ to denote the sum over all permutations of the coordinates of the agents $-i$, over the set of sites not occupied by agent i. We also write $\mathscr{S}_{-i}$ to denote the Cartesian product $S_1 \times \ldots S_{i-1} \times S_{i+1} \times \ldots \times S_N$.

The state of the system at time t can be described by the state of each site and the position $\mathbf{s} \in \Pi_{[N]}$ of the servers, including the passive ones (even if more compact state descriptions are possible, our choice is motivated by the formulation of the relaxation in the next section). We denote the complete state by $(x_1, \ldots, x_N; s_1, \ldots, s_N) := (\mathbf{x}; \mathbf{s})$. We can choose which sites are to be visited next, i.e. the action $\mathbf{a}$ belongs to the set $\Pi_{[N]}$ and corresponds to the assignment of the agents, including the passive ones, to the sites for the time period. Once the sites to be visited are chosen, there are costs $c_{s_i a_i}$ for moving the active agent i from site s_i to site a_i, including possibly a nonzero cost for staying at the same site. The reward earned is $\sum_{i=1}^{M} r^1_{a_i}(x_{a_i}) + \sum_{i=M+1}^{N} r^0_{a_j}(x_{a_j})$. We are given a distribution ν on the initial state of the system, and we will assume a product form $\nu(x_1, \ldots, x_N; s_1, \ldots, s_N) = \nu_1(x_1) \ldots \nu_N(x_N) \delta_{d_1}(s_1) \ldots \delta_{d_N}(s_N)$, i.e., the initial states of the sites are independent and server i leaves initially from site d_i, with $\mathbf{d} \in \Pi_{[N]}$.

The transition matrix has a particular structure, since the sites evolve independently and the transitions of the agents are deterministic:

$$\mathscr{P}_{(\mathbf{x}';\mathbf{s}')\mathbf{a}(\mathbf{x};\mathbf{s})} = \prod_{i=1}^{M} p^1_{x'_{a_i} x_{a_i}} \prod_{i=M+1}^{N} p^0_{x'_{a_i} x_{a_i}} \prod_{i=1}^{N} \delta_{s_i}(a_i).$$

With these elements, we can formulate the RBSC problem with multiple agents as follows:

maximize

$$\sum_{\mathbf{s} \in \Pi_{[N]}} \sum_{\mathbf{a} \in \Pi_{[N]}} \sum_{\mathbf{x} \in \mathscr{S}} \left(\sum_{i=1}^{M} (r^1_{a_i}(x_{a_i}) - c_{s_i a_i}) + \sum_{i=M+1}^{N} r^0_{a_i}(x_{a_i}) \right) \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} \qquad (4)$$

subject to

$$\sum_{\mathbf{a} \in \Pi_{[N]}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} - \alpha \sum_{\mathbf{s}' \in \Pi_{[N]}} \sum_{\mathbf{x}' \in \mathscr{S}} \rho_{(\mathbf{x}';\mathbf{s}'),\mathbf{s}} \prod_{i=1}^{M} p^1_{x'_{s_i} x_{s_i}} \prod_{i=M+1}^{N} p^0_{x'_{s_i} x_{s_i}}$$

$$= (1-\alpha) \prod_{i=1}^{N} \nu_i(x_i) \delta_{d_i}(s_i), \quad \forall\, (\mathbf{x}, \mathbf{s}) \in \mathscr{S} \times \Pi_{[N]}$$

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

with the decision variables $\rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}$ corresponding to an occupation measure. Note that the formulation above is of little computational interest since the number of variables and constraints is of the order of $|\mathscr{S}| \times (N!)^2$, that is, exponential in the size of the input.

III. LINEAR PROGRAMMING RELAXATION

The complexity result known for the restless bandits problem justifies the search for efficient methods that approximate the optimal solution of the RBSC problem. An interesting feature of the multi-armed bandit framework is that it leads naturally to a Markov decision process for each site. This was already noticed by Whittle [3] (see also [9]), whose solution was based on relaxing the hard constraints tying the project together, enforcing them only in average.

In this section we will see that in the state-action frequency domain, a relaxation can be easily obtained by considering specific marginals of the occupation measure. Once the relaxation is obtained, we can go back to the value function domain of dynamic programming by taking the dual of the relaxed linear program. Then it becomes clear that the essential features of the problem that allow the method to work are the separable structure of the objective function and the independence assumption of the evolution of the sites. When the coupling between the sites increases (for instance, when we introduce switching costs to the RB problem), the relaxation becomes more complicated and grows in size. In general, the method used to derive a relaxation is the following:

(i) identify the marginals of the state-action frequencies that are sufficient to express the objective function in (4).
(ii) express the constraints on these marginals by partially summing the constraints in (4).
(iii) add the constraints due to the fact that these marginals all derive from the same state-action frequency vector.

The link between this method and the work of Bertsimas and Nino-Mora on restless bandits [1] was also highlighted in our previous paper [7]. As we illustrate now, this method is relatively systematic once the original linear program has been formulated, and in principle can be extended to derive relaxations for other problems with a certain decomposable structure.

We start by rewriting the objective function and we identify the relevant marginals:

$$\sum_{i=1}^{M} \sum_{a_i=1}^{N} \sum_{x_{a_i} \in S_{a_i}} r^1_{a_i}(x_{a_i}) \rho^i_{x_{a_i};a_i} + \sum_{i=M+1}^{N} \sum_{a_i=1}^{N} \sum_{x_{a_i} \in S_{a_i}} r^0_{a_i}(x_{a_i}) \rho^i_{x_{a_i};a_i}$$

$$- \sum_{i=1}^{M} \sum_{a_i=1}^{N} \sum_{s_i=1}^{N} c_{s_i a_i} \tau^i_{s_i;a_i},$$

where the marginals appearing above are obtained as follows:

$$\rho^i_{x_{a_i};a_i} = \sum_{\mathbf{a}_{-i} \in \Pi_{[N]-a_i}} \sum_{\mathbf{s} \in \Pi_{[N]}} \sum_{\mathbf{x}_{-a_i} \in S_{-a_i}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}$$

$$\tau^i_{s_i;a_i} = \sum_{\mathbf{a}_{-i} \in \Pi_{[N]-a_i}} \sum_{\mathbf{s}_{-i} \in \Pi_{[N]-s_i}} \sum_{\mathbf{x} \in \mathscr{S}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}$$

and the superscripts refer to the agents.

Now to express the constraints for these marginals, it turns out that the following variables are sufficient:

$$\rho^i_{(x_j;s_i),a_i} = \sum_{\mathbf{x}_{-j} \in S_{-j}} \sum_{\mathbf{s}_{-i} \in \Pi_{[N]-s_i}} \sum_{\mathbf{a}_{-i} \in \Pi_{[N]-a_i}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} \quad , \qquad (5)$$

$$\forall\, x_j \in S_j,\ \forall\, (i, j, s_i, a_i) \in [N]^4$$

Clearly we have

$$\tau^i_{s_i;a_i} = \sum_{x_j\in S_j} \rho^i_{(x_j;s_i),a_i} = \sum_{x_1\in S_1} \rho^i_{(x_1;s_i),a_i}, \forall\,(i,j,s_i,a_i)\in[N]^4, \tag{6}$$

$$\rho^i_{x_{a_i};a_i} = \sum_{s_i=1}^{N} \rho^i_{(x_{a_i};s_i),a_i}\,, \quad \forall\,(i,a_i,x_{a_i}). \tag{7}$$

Note that we could formulate everything in terms of the variables (5), but we chose to introduce some additional variables for readability. However, the second equality in (6) is important to obtain a sufficiently strong relaxation; it expresses a compatibility condition that is clearly true because the marginals are obtained from the same original distribution.

For agent i, some $\mathbf{x}\in\mathscr{S}$ and $\mathbf{s}$ in $\Pi_{[N]}$, we can sum the constraints in (4) over $x_{s_j}\in S_{s_j}$ for $j\neq i$ to get

$$\sum_{\mathbf{a}\in\Pi_{[N]}} \rho_{(x_{s_i};\mathbf{s}),\mathbf{a}} - \alpha \sum_{\mathbf{s}'\in\Pi_{[N]}} \sum_{x'_{s_i}\in S_{s_i}} \rho_{(\mathbf{x}';\mathbf{s}'),\mathbf{s}}\, p^{1\{i\leq M\}}_{x'_{s_i}x_{s_i}} =$$
$$(1-\alpha)\,\nu_{s_i}(x_{s_i})\delta_{d_1}(s_1)\ldots\delta_{d_N}(s_N), \quad \forall\, x_{s_i}\in S_{s_i},$$

where $1\{i\leq M\}$ is the indicator variable of the set $\{i\leq M\}$. Then we can sum over s_j, $j\neq i$, for a fixed s_i, rewrite $\sum_{\mathbf{a}\in\Pi_{[N]}} = \sum_{a_i=1}^{N}\sum_{\mathbf{a_{-i}}\in\Pi_{[N]-a_i}}$ and $\sum_{\mathbf{s}'\in\Pi_{[N]}} = \sum_{s'_i=1}^{N}\sum_{\mathbf{s}'_{-\mathbf{i}}\in\Pi_{[N]-s'_i}}$, to obtain

$$\sum_{a_i=1}^{N} \rho^i_{(x_{s_i};s_i),a_i} - \alpha \sum_{x'_{s_i}\in S_{s_i}} \sum_{s'_i=1}^{N} \rho^i_{(x'_{s_i};s'_i),s_i} p^{1(i\leq M)}_{x'_{s_i}x_{s_i}} =$$
$$(1-\alpha)\,\nu_{s_i}(x_{s_i})\delta_{d_i}(s_i)\,, \quad \forall\,(i,s_i,x_{s_i})\in[N]^2\times S_{s_i}. \tag{8}$$

Last, there are still some compatibility conditions between the marginals that have not been expressed. To obtain a sufficiently strong relaxation, we want to take into account the fact that no two agents can be at the same time in the same site. This is expressed in terms of marginals as follows:

$$\sum_{s_i\in[N]} \sum_{a_i\in[N]-a} \rho^i_{(x_a;s_i),a_i} = \sum_{k\in[N]-i} \sum_{s_k\in[N]} \rho^k_{(x_a;s_k),a}, \forall i,a,x_a\in S_a, \tag{9}$$

$$\sum_{s_i\in[N]-s} \sum_{a_i\in[N]} \rho^i_{(x_s;s_i),a_i} = \sum_{k\in[N]-i} \sum_{a_k\in[N]} \rho^k_{(x_s;s),a_k}, \forall i,s,x_s\in S_s. \tag{10}$$

Intuitively, on the left hand side we have the probability that agent i does not go to a site a (respectively does not currently occupy a site s), which must equal the probability that some other agent k (passive or not) goes to site a (respectively occupy site s). These relations can be verified by inspection of (5). Finally the relaxation is:

maximize

$$\sum_{i=1}^{M}\sum_{a_i=1}^{N}\sum_{x_{a_i}\in S_{a_i}} r^1_{a_i}(x_{a_i})\rho^i_{x_{a_i};a_i} + \sum_{i=M+1}^{N}\sum_{a_i=1}^{N}\sum_{x_{a_i}\in S_{a_i}} r^0_{a_i}(x_{a_i})\rho^i_{x_{a_i};a_i} \tag{11}$$
$$-\sum_{i=1}^{M}\sum_{a_i=1}^{N}\sum_{s_i=1}^{N} c_{s_i a_i}\tau^i_{s_i;a_i},$$

subject to

$$(8),(9),\,(10),\,(6),\,(7), \quad \rho^i_{(x;s_i),a_i}\geq 0\,, \quad \forall (i,s_i,a_i,x)\in[N]^3\times\bigcup_{k=1}^{N} S_k.$$

There are $O(N^4\times\max_i|S_i|)$ variables $\rho^i_{(x,a,s)}$, which is polynomial in the size of the input. This number if independent of the number of real agents in the instance of our problem. Note that in [7], we obtain in the single agent case a relaxation with $O(N^3\times\max_i|S_i|)$ variables, which is therefore preferable to (11) in the case $M=1$. From a careful comparison of (11) with the relaxation obtained in [7] for the single agent case, we verified that when $M=1$ the two formulation are equivalent, even if (11) involves more variables and constraints. However, (11) has the advantage of being valid for any number of agents, and this number can be given as a parameter to our problem.

IV. Heuristics

A. One-Step Lookahead Policy

Computing the optimum value of the relaxation presented in the previous section provides a bound on the performance achievable by any assignment policy. It is also useful to actually design a policy for the system, via standard suboptimal control techniques.

By taking the dual of the linear program obtained from the state-action frequency formulation, we obtain a linear program whose variables, indexed by the different states of the system, have the interpretation of the value function for these states [8]. Indeed, this dual program can be obtained directly from Bellman's equation. Now we can take the dual of our relaxed formulation (11); in this paper, we do not consider the structure of this dual program in detail, but we note that the dual objective function involves a subset of the dual variables corresponding to the constraints (8) and can be written

$$\text{minimize} \sum_{i=1}^{N}\sum_{s_i=1}^{N}\sum_{x_{s_i}\in S_{s_i}} \lambda^i_{x_{s_i},s_i}\nu_{s_i}(x_{s_i})\delta_{d_i}(s_i)\,,$$

where the dual variables of interest are the $\lambda^i_{x_{s_i},s_i}$. Now consider the system at a given time, for which the state is $(x_1,\ldots,x_N;s_1,\ldots,s_N)$, with $(s_1,\ldots,s_N)$ a permutation of $[N]$. Given the interpretation of the dual variables mentioned above, it is natural to try to form an approximation $\tilde{J}(\mathbf{x};\mathbf{s})$ of the value function in state $(\mathbf{x};\mathbf{s})$ as

$$\tilde{J}(x_1,\ldots,x_N;s_1,\ldots,s_N) = \sum_{i=1}^{N}\bar{\lambda}^i_{x_{s_i},s_i}, \tag{12}$$

where $\bar{\lambda}^i_{x_{s_i},s_i}$ are the optimal values of the dual variables obtained from the LP relaxation. The separable form of this approximate cost function is useful to design an efficiently computable one-step lookahead policy, as follows. At state $(\mathbf{x};\mathbf{s})$, we obtain the assignment of the agents by solving

$$\tilde{u}(\mathbf{x};\mathbf{s})\in\operatorname{argmax}_{\mathbf{a}\in\Pi_{[N]}}\left\{g((\mathbf{x};\mathbf{s}),\mathbf{a}) + \alpha\sum_{\mathbf{x}'\in\mathscr{S}}\mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'}\tilde{J}_{\mathbf{x}',\mathbf{a}}\right\},$$

where $g((\mathbf{x};\mathbf{s}),\mathbf{a}) = \sum_{i=1}^{M}(r^1_{a_i}(x_{a_i}) - c_{s_i a_i}) + \sum_{i=M+1}^{N} r^0_{a_i}(x_{a_i})$, and $\mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'} = \prod_{i=1}^{M} p^1_{x_{a_i}x'_{a_i}}\prod_{i=M+1}^{N} p^0_{x_{a_i}x'_{a_i}}$. We obtain the following maximization problem:

$$\max_{\mathbf{a}\in\Pi[N]} \left\{ \sum_{i=1}^{M} \left(r^1_{a_i}(x_{a_i}) - c_{s_i a_i} + \alpha \sum_{x'_{a_i}\in S_{a_i}} \bar{\lambda}^i_{x'_{a_i},a_i} p^1_{x_{a_i} x'_{a_i}} \right) \right. \tag{13}$$
$$\left. + \sum_{i=M+1}^{N} \left(r^0_{a_i}(x_{a_i}) + \alpha \sum_{x'_{a_i}\in S_{a_i}} \bar{\lambda}^i_{x'_{a_i},a_i} p^0_{x_{a_i} x'_{a_i}} \right) \right\}.$$

Assuming that the optimal dual variables have been stored in memory, the maximization above is actually easy to perform. The evaluation of one parenthesis involves only the data of the problem for the current state of the system, and one summation over the states of one site, i.e., takes a time $O(\max_i |S_i|)$. Let us denote the terms in parenthesis m_{i,a_i}. All these terms can be computed in time $O(N^2 \max_i |S_i|)$, and (13) can be rewritten:

$$\max_{\mathbf{a}\in\Pi_{[N]}} \sum_{i=1}^{N} m_{i,a_i}.$$

This is a linear assignment problem, which can be solved by linear programming or in time $O(N^3)$ by the Hungarian method [10]. Thus, the assignment is computed at each time step in time $O(N^2 \max_i |S_i| + N^3)$ by a centralized controller, which needs to store the optimal dual variables of the relaxation in addition to the parameters of the problem.

B. Computational Considerations

The major bottleneck limiting the use of our method for large-scale problems is the computation of the relaxation (11) which involves a large number of variables. Hence, to solve a problem with 10 sites which can each be in one of five states already requires solving a linear program with 50000 variables (independently of the number of agents used). This relaxation is computed off-line, in order to obtain a bound on the achievable performance and the optimal dual variables. Howevever, the limits of the state-of-the-art linear programming technology are reached for relatively small problems. For these problems, the one-step lookahead policy described in the previous paragraph is easily computed in real-time. If more time is available, then this one-step lookahead policy can sometimes be used as a base policy to obtain a rollout policy [11], whose performance is known to be at least as good as the base policy, and in practice offers interesting improvements.

It is also interesting to consider decentralized algorithms. Assuming that each agent stores only his own optimal dual variables and the parameters of the problems, we expect that existing work on the distributed computation of the value of the assignment problem [12] can be used. We will explore this direction in future research.

C. Numerical Experiments

We now briefly present some numerical experiments with the proposed policy. We can compare the lower bound on the optimal reward obtained through the use of a specific policy to the upper bound obtained from the relaxation. We also compare the one-step lookahead policy to a simple greedy policy, where the approximate cost-to-go $\tilde{J}$ is taken to be 0. It is known that this greedy policy is optimal for the MAB problem (single agent case), when the rewards are deteriorating, i.e., projects become less profitable as they are worked on.

TABLE I
NUMERICAL EXPERIMENTS

Problem	α	N	M	Z_r	Z_{osl}	Z_g
Problem 1	0.9	4	1	372	369	259
Problem 2	0.95	5	2	4527	4382	4488
Problem 3	0.95	5	2	1304	1131	670
Problem 4	0.9	8	4	2788	2397	1933
Problem 5	0.9	10	5	4234	3689	3571

Linear programs are implemented in AMPL and solved using CPLEX. Due to the size of the state space, the expected discounted reward of the various heuristics is computed via 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}. Table I presents results for various RBSC problems. The number of states per site in the scenarios varies between 3 and 5. We adopt the following nomenclature:

- Z_r: optimal value of the relaxation.
- Z_{osl}: estimated expected value of the one-step lookahead policy.
- Z_g: estimated expected value of the greedy policy.

Problem 1 is designed to make the greedy policy underperform: one of the sites has a higher initial reward but it is also very costly to move from this site to the others (the distances are asymmetric here). Note that in this problem the one-step lookahead policy performed remarkably well, and in general it is not as clear how to make it strongly underperform. Problem 2 and 3 have the same number of sites and agents, but in problem 2 the switching costs are small compared to the rewards, whereas in problem 3 the costs and rewards are of the same order of magnitude. These problems confirm that high transition costs have a dramatic importance on the performance of the greedy policy. If they are negligible however, it is not clear that it is beneficial to use the one-step lookahead policy. Problems 4 and 5 have randomly generated data. The LP relaxation for problem 5 was computed in about 30 minutes on a relatively recent desktop running CPLEX.

V. A "PERFORMANCE" BOUND

In this section, we present a result that offers some insight into why we expect the one-step lookahead policy to perform well if the linear programming relaxation of the original problem is sufficiently tight. Essentially, we can follow the type of analysis presented in [13]. First, we note that the vector $\tilde{J}$ in (12) is a feasible vector for the linear program dual of (4). That is, $\tilde{J}$ is a superharmonic vector for the original RBSC problem, and we recall that the exact optimal cost is the smallest superharmonic vector [14]. Writing ν for the vector of initial probability distribution, we see then that

$$\nu^T \tilde{J} \geq \nu^T J^* \tag{14}$$

where J^* is the optimal value function, minimizing the dual of (4).

Proposition 2: Let $f_\alpha(\nu,\tilde{u})$ be the occupation measure vector associated with the one-step lookahead policy $\tilde{u}$, and $J_{\tilde{u}}$ the cost associated to this policy. We have

$$\nu^T(J^* - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} f_\alpha(\nu,\tilde{u})^T(\tilde{J} - J^*) \tag{15}$$

Proof: In the following we denote by $T_{\tilde{u}}$ and T (with $TJ = \max_u T_u J$) the dynamic programming operators. The cost of policy $\tilde{u}$ is given by $J_{\tilde{u}} = T_{\tilde{u}} J_{\tilde{u}}$, i.e.,

$$J_{\tilde{u}}(\mathbf{x};\mathbf{s}) = g((\mathbf{x};\mathbf{s}),\tilde{u}(\mathbf{x};\mathbf{s})) + \alpha \sum_{\mathbf{x}' \in \mathscr{S}} \mathscr{P}_{\mathbf{x}\tilde{u}(\mathbf{x};\mathbf{s})\mathbf{x}'} J_{\tilde{u}}(\mathbf{x}',\tilde{u}(\mathbf{x};\mathbf{s})).$$

Let $g_{\tilde{u}}$ be the vector of size $N! \times |\mathscr{S}|$ with components $g((\mathbf{x};\mathbf{s}),\tilde{u}(\mathbf{x};\mathbf{s}))$, and $P_{\tilde{u}}$ the stochastic matrix with components $\mathscr{P}_{\mathbf{x}\tilde{u}(\mathbf{x};\mathbf{s})\mathbf{x}'}$. We take a compatible ordering of the states for all vectors and matrices, and we have then

$$J_{\tilde{u}} = g_{\tilde{u}} + \alpha P_{\tilde{u}} J_{\tilde{u}}.$$

Now $(I - \alpha P_{\tilde{u}})$ has an inverse since $P_{\tilde{u}}$ is a stochastic matrix and $\alpha < 1$, so we get

$$J_{\tilde{u}} = (I - \alpha P_{\tilde{u}})^{-1} g_{\tilde{u}}.$$

Under policy $\tilde{u}$, the state of the system evolves as a Markov chain, and we have $\mathbb{P}_\nu^u(\mathbf{X}_t = \mathbf{x}, \mathbf{S}_t = \mathbf{s}) = \left[\nu^T P_{\tilde{u}}{}^t\right]_{\mathbf{x},\mathbf{s}}$.

So the occupation measure (state frequency) is

$$f_\alpha(\nu,\tilde{u}) = (1-\alpha)\sum_{t=0}^{\infty} \alpha^t \nu^T P_{\tilde{u}}^t = (1-\alpha)\nu^T(I-\alpha P_{\tilde{u}})^{-1}. \tag{16}$$

Now

$$\tilde{J} - J_{\tilde{u}} = (I-\alpha P_{\tilde{u}})^{-1}\left[(I-\alpha P_{\tilde{u}})\tilde{J} - g_{\tilde{u}}\right] = (I-\alpha P_{\tilde{u}})^{-1}\left[\tilde{J} - (g_{\tilde{u}} + \alpha P_{\tilde{u}}\tilde{J})\right].$$

By the definition of the lookahead policy $g_{\tilde{u}} + \alpha P_{\tilde{u}} \tilde{J} = T\tilde{J}$. So we obtain

$$\tilde{J} - J_{\tilde{u}} = (I - \alpha P_{\tilde{u}})^{-1}\left[\tilde{J} - T\tilde{J}\right].$$

Then starting from (14) and using (16)

$$\nu^T(J^* - J_{\tilde{u}}) \leq \nu^T(\tilde{J} - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} f_\alpha(\nu,\tilde{u})^T(\tilde{J} - T\tilde{J}).$$

Now by Bellman's theorem and the fact that $\tilde{J} \geq T\tilde{J}$, we get $T\tilde{J} \geq T^2\tilde{J} \geq \ldots \geq J^*$. So

$$\nu^T(J^* - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} f_\alpha(\nu,\tilde{u})^T(\tilde{J} - J^*),$$

which is the inequality in the proposition. ■

In words, the proposition says that starting with a distribution ν over the states, the expected difference in reward between the optimal solution and the one-step lookahead policy is bounded by a weighted distance between the estimate $\tilde{J}$ used in the design of the policy and the optimal value function J^*. The weights are given by the occupation measure of the one-step lookahead policy. Note that this is true for every one-step lookahead policy that uses a superharmonic vector as an approximation of the cost-to-go. In future research, we intend to investigate the structure of the relaxation further in order to refine the upper bound (15).

VI. CONCLUSIONS

We have presented a linear programming relaxation for the restless bandits with switching costs problem, which extends our previous work from the single agent to the multi-agent case. This framework, motivated by a multi-UAVs task assignment problem, is general enough to model a wide range of dynamic ressource allocation problems of relatively modest size. An important feature of the method is that it automatically provides a bound on the performance achievable by any policy. The techniques rely on the separable structure of the problem and should be useful for other problems with similar structure. We designed a one-step lookahead policy based on this relaxation, which can be implemented in real-time, and should also be implementable distributively. Future work will focus on trying to obtain a better characterization of the performance of our heuristics.

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, *Multi-armed Bandit Allocation Indices*, ser. Wiley-Interscience series in Systems and Optimization. New York: John Wiley and sons, 1989.

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

[4] R. Washburn, M. Schneider, and J. Fox, "Stochastic dynamic programming based approaches to sensor ressource management," in *Proceedings of the International Conference on Information Fusion*, 2002.

[5] V. Krishnamurthy and R. Evans, "Hidden markov model multiarm bandits: a methodology for beam scheduling in multitarget tracking," *IEEE Transactions on Signal Processing*, vol. 49, no. 12, pp. 2893 – 2908, December 2001.

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

[7] J. Le Ny and E. Feron, "Restless bandits with switching costs: Linear programming relaxations, performance bounds and limited lookahead policies," in *American Control Conference*, Minneapolis, MN, June 2006, to appear.

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

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

[10] A. Schrijver, *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[11] D. Bertsekas and D. Castanon, "Rollout algorithms for stochastic scheduling problems," *Journal of Heuristics*, vol. 5, no. 1, pp. 89–108, 1999.

[12] ——, "Parallel synchronous and asynchronous implementations of the auction algorithm," *Parallel Computing*, vol. 17, pp. 707–732, 1991.

[13] D. de Farias and B. V. Roy, "The linear programming approach to approximate dynamic programming," *Operations Research*, vol. 51, no. 6, pp. 850–865, 2003.

[14] 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.