A Combined Linear/Integer Programming Heuristic to Control a Network of semi-Markov Decision Processes

David W. J. Stein
The MITRE Corporation
202 Burlington Road Bedford, MA 01730
Steven F. Baker
The MITRE Corporation
1155 Academy Park Loop, Colorado Springs, CO 80910

Abstract. For even moderately sized networks and a small number of actions finding an optimal policy, i.e., determining which actions to apply to which nodes of the network given the states of the nodes, may be computationally infeasible. Heuristics have been developed that are applicable to the case of multiple homogeneous actions such that each action affects exactly one entity. For many applications, e.g., remote sensing with localization and identification sensors, multiple disparate actions may be available, an action may affect multiple entities, and actions may require different amounts of time to complete. The present work develops a two-step heuristic to generate action plans under these more general circumstances. Linear programming applied either to the individual nodes of the network or to all nodes collectively but with an average aggregate utilization constraint is used to obtain optimal policies and reduced cost coefficients for each node. Integer programming, using the reduced cost coefficients, is then used at each time-step to assign resources to projects. These methodologies are applied to a simulated remote sensing problem, and the performance of the current methods is compared with an optimal solution where its computation is feasible and with a greedy solution. Results show minimal degradation in comparison with an optimal policy and substantial improvement over the greedy approach. Computation time studies show that the method is practical for large scale real time applications.

Key Words: Markov decision process, dynamic programming, resource allocation, stochastic scheduling

1. Introduction

Markov decision processes (MDPs) have proven themselves very useful across a broad spectrum of activities. They incorporate the richness of stochastic processes with the prescriptive power of classical optimization. Consequently, they enjoy wide employment in manufacturing, medicine, defense, transportation, and communications disciplines, to name a few. Yet problem size for many applications is combinatorially explosive for even moderate instances, which often necessitates some mitigation of exact solution through the use of heuristics. This paper develops

one such heuristic used to assign aircraft surveillance radars to military targets in an efficient and effective manner.

In military surveillance activities, radars and other intelligence collectors are tasked to provide: "systematic observation of aerospace, surface or subsurface areas, places, persons, or things, by visual, aural, electronic, photographic, or other means." [**1**] (p. 2). In the case of airborne collection, an aircraft flies in or near the surveillance region, and observes specific points and larger areas as assigned. An observation typically improves the knowledge of the specified point and area targets; this state change is governed by stochastic process. In the MDP lexicon, observations take the role of actions, targets are labeled as projects, and levels of project knowledge are defined as states. There are many targets (projects) assigned to a collection platform, and target knowledge (state) degrades in absence of observation (action). Thus, the airborne surveillance problem is well characterized as a multiarmed restless bandit MDP [**2**] (pp. 57-61), where the goal is to assign actions to projects over time such that the reward-weighted state space of the projects is maximized.

Practical surveillance problems involve additional complexity. Typically there are several collection platforms to which targets can be assigned. Each platform has multiple sensors with specialized knowledge-gathering capabilities; for example, a traditional radar can locate and identify a target, whereas a moving-target radar can better characterize mobility, or kinematics. In an MDP context, this example indicates that both the action space and the state space are expanded by sensor selection and type of knowledge required, respectively. Furthermore, targets are often geographically clustered in a manner that allows a single radar collection to improve knowledge of multiple targets. Finally, radar collection time may vary due to target size, distance, and other factors, suggesting disparate resource usage by platform location, sensor type, and target cluster.

The literature regarding MDPs, solution methods, and applications to sensor assignment is well developed. Puterman [**2**] offers a broad-ranging work describing theory and solution strategies. Gittins and Jones [**3**] examine the multiarmed bandit problem, and develop the widely regarded optimal index rule for their solution. Whittle [**4**] extends this work to the restless bandit problem and provides several applications. Bertsimas and Nino-Mora [**5**] develop a linear programming relaxation heuristic for this class of problems, which would otherwise become unsolvably huge for even moderately-sized instances. Finally, Hero *et al.* (eds.) [**6**] compile the literature regarding application of MDPs to sensor management.

In section 2, we develop a heuristic that assigns actions to multiarmed restless superprojects for which: 1) multiple simultaneous actions may be performed; 2) the time to complete actions is a function of the action and the project; and 3) a single action may affect multiple projects simultaneously. As described above, this formulation is appropriate to remote sensing applications where disparate sensor types are used, and target spacing is such that several can be simultaneously evaluated by a given sensor. The method reduces to the heuristic developed in [**5**] when applied to a multiarmed restless bandit.

In section 3, we apply the heuristic to a system of airborne radars conducting surveillance over many targets in a military context. In these representative problems, solution time is important due to the rapidly-changing battlefield, and solution quality ensures efficient use of scarce assets. To address the latter, we

compare our results with a greedy approach similar to those used in operational environments. We conclude in section 4 and address ongoing research.

2. Linear/Integer Programming Heuristic

Assume a system of N projects such that each project is modeled as an infinite horizon discounted semi-Markov decision process with finite state and action spaces [**2**] (Chapter 11). The projects are indexed by $n \in \mathcal{N} = \{1, 2, \ldots, N\}$. Projects are defined by: the state space, $\mathcal{S}_n$; the action space, $\mathcal{A}_n$; state probability transition functions, $P_n(i|j,a)$ where $i, j \in \mathcal{S}_n$, $a \in \mathcal{A}_n$; probability distribution functions of the time to state transition, $F_n(t|j,a)$, and probability density $F_n(dt|j,a)$; instantaneous reward functions $k(i,a)$ where $i, \in \mathcal{S}_n$ and $a \in \mathcal{A}_n$; functions $c(i|j,a)$ which give the rate of reward accrual if the system occupies state i and at the previous decision the system was in state j and action a was chosen; a continuous time discount rate α such that one unit of reward received t time units in the future has a present value of $e^{-\alpha t}$. A semi-Markov decision process differs from a Markov decision process in that the decision epochs are random variables. A sample path of a semi-Markov decision process is a sequence $(t_0, s_0, a_0, t_1, s_1, a_1, \ldots)$ which indicates that the system was in state s_i at $\sigma_i = \sum_{j=0}^{i} t_j$ and action a_i is taken at time σ_i. This structure allows the time at which the next decision is made to depend upon the current selected action, as is described in detail below.

$Q_n(t, i|j,a) = F_n(t|j,a)P_n(i|j,a)$ is the probability that the next decision epoch occurs at or before time t and that the system state at that time is i. Let $a_\emptyset$ denote the null-action, i.e., that the project is not being worked on and $\tilde{\mathcal{A}}_n = \mathcal{A}_n \cup \{a_\emptyset\}$. Assume for each bandit a passive, i.e., a null-action, transition matrix $P_n(i|j,a_\emptyset)$ for unit-time and assume that the passive transition over time t is

$$P_n(i|j,a_\emptyset,t) = \exp(t \log(P_n(i|j,a_\emptyset))).$$

2.1. Optimizing the policy of a semi-Markov decision process. A policy is a sequence of functions that identifies the current action. Generally, the current action may be a probability distribution on the action space that depends on the history of states, actions, and decision epochs [**2**] (pp. 22, 534). Thus, letting $\mathcal{P}(\mathcal{A})$ denote the probability distributions on $\mathcal{A}$, $\mathcal{T} = [0, \infty)$, and $(\mathcal{T} \times \mathcal{S} \times \mathcal{A})^m = \mathcal{T} \times \mathcal{S} \times \mathcal{A} \times \cdots \times \mathcal{T} \times \mathcal{S} \times \mathcal{A}$ and suppressing the project index, a policy is a sequence $\pi = (d_1, d_2, \ldots, d_n, \cdots)$, where $d_m : (\mathcal{T} \times \mathcal{S} \times \mathcal{A})^{m-1} \to \mathcal{P}(\mathcal{A})$. The policy is Markovian if it depends only on the current state, i.e. $d_m : \mathcal{S} \to \mathcal{P}(\mathcal{A})$; it is deterministic if the action is specified exactly and not as a distribution, i.e., $d_m : (\mathcal{T} \times \mathcal{S} \times \mathcal{A})^{m-1} \to \mathcal{A}$, and it is stationary if $d_1 = d_2 = \cdots = d_n = \cdots$. Under the current assumption of finite action spaces and infinite horizon discounted reward, the optimal policy is Markovian, stationary and deterministic [**2**] (Theorem 11.3.2). The optimal policy is found by solving the optimality equations.

Puterman [**2**] (pp. 540–546) derives the optimality equations for a finite semi-Markov decision process. Let X_n, Y_n and σ_n be the random variables denoting the state, action and elapsed time from onset until the n^{th} decision, respectively. The value of a policy π, discounted at rate α, is [**2**] (p. 540)

$$\nu_\alpha^\pi = E_s^\pi \left[\sum_{n=0}^{\infty} e^{-\alpha \sigma_n} \left[k\left(X_n, Y_n\right) + \int_{\sigma_n}^{\sigma_n+1} e^{-\alpha(t-\sigma_n)} c(W_t, X_n, Y_n) dt \right] \right], \tag{1}$$

Semi-Markov decision processes can change state between decision epochs, and W_t in (1) denotes the state of the system at time t.

Define

$$\begin{aligned} r(s,a) &= k(s,a) + E_s^a\left(\int_{\sigma_0}^{\sigma_1} e^{-\alpha t} c(W_t,s,a)dt\right) \\ &= k(s,a) + \int_0^\infty \sum_{j\in\mathcal{S}} \left(\int_0^u e^{-\alpha t} c(j,s,a)p(j|t,s,a)dt\right) F(du|s,a). \end{aligned} \tag{2}$$

If $d : \mathcal{S} \to \mathcal{A}$, define $r_d(s) = r(s, d(s))$. Denote by d^∞ the stationary policy $(d, d, \ldots)$. Then

$$v_\alpha^{d^\infty}(s) = r_d(s) + \sum_{j\in\mathcal{S}} \left[\int_0^\infty e^{-\alpha t} p\left(j|t,s,d(s)\right) F\left(dt|s,d(s)\right)\right] v_\alpha^{d^\infty}(j). \tag{3}$$

Let

$$m_d(j|s) = \int_0^\infty e^{-\alpha t} p\left(j|t,s,d(s)\right) F\left(dt|s,d(s)\right), \tag{4}$$

and let M_d be the $|S| \times |S|$ matrix such that $M_d(i,j) = m_d(j|i)$, then 3 may be expressed as v^{d^∞} satisfies the linear equation

$$v_\alpha^{d^\infty} = r_d + M_d v_\alpha^{d^\infty} \tag{5}$$

Puterman [**2**] (pp. 542–543, Theorem 11.3.1) shows that if r is uniformly bounded and $\forall s, a\ \exists \epsilon > 0$ and $\delta > 0$ such that $F(\delta)|s,a) < 1-\epsilon$, then for any $d : \mathcal{S} \to \mathcal{A}$ (3) has the unique solution

$$v_\alpha^{d^\infty} = (I - M_d)^{-1} r_d. \tag{6}$$

Let V be the set of bounded real valued functions on $\mathcal{S}$ and let $D = \{d : \mathcal{S} \to \mathcal{A}\}$. Define the operator:

$$\mathcal{L}(v) = \sup_{d\in D} \left(r_d + M_d v\right). \tag{7}$$

Puterman[**2**] shows that under certain assumptions, including finite action and state spaces considered here, if $\mathcal{L}(v^*) = v^*$ and $d^* = \arg\sup_{d\in D}\left(r_d + M_d v\right)$, then $d^{*\infty}$ is the optimal policy and v^* is the value of the SMDP.

Linear programming may be used to obtain the value function v^*, the optimal policy d^*, and reduced cost coefficients that are subsequently used by the heuristic. Let $\beta : \mathcal{S} \to \mathbf{R}$ such that $\forall\ s \in \mathcal{S}$, $\beta(s) > 0$. The value function is the solution to: [**2**] (pp. 223–231, 546–547)

$$\begin{aligned} \textbf{Minimize } V &= \sum_{j\in\mathcal{S}} \beta(j) v(j) \\ \text{such that } \forall a \in \mathcal{A} \text{ and } s \in \mathcal{S} & \\ v(s) - \sum_{j\in\mathcal{S}} m(j|s,a) v(j) &\geq r(s,a). \end{aligned} \tag{8}$$

The dual of (8)is

$$\begin{aligned} \textbf{Maximize } R(x) &= \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} r(s,a) x(s,a) \\ \text{such that } \forall j \in \mathcal{S} & \end{aligned} \tag{9}$$

$$\begin{aligned} \sum_{a\in\mathcal{A}} x(j,a) - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} m(j|s,a)x(s,a) &= \beta(j) \\ \text{and } \forall j \in \mathcal{S} \text{ and } \forall a \in \mathcal{A} & \\ x(s,a) &\geq 0 \end{aligned}$$

or equivalently

$$\begin{aligned} \textbf{Minimize} - R(x) &= -\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} r(s,a)x(s,a) \\ \text{such that } \forall j \in \mathcal{S} & \\ \sum_{a\in\mathcal{A}} x(j,a) - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} m(j|s,a)x(s,a) &= \beta(j) \\ \text{and } \forall j \in \mathcal{S} \text{ and } \forall a \in \mathcal{A} & \\ x(s,a) &\geq 0 \end{aligned} \tag{10}$$

The Lagrangian of (10) is

$$\begin{aligned} \mathcal{L}(\vec{x},\vec{\lambda},\vec{\gamma}) &= -\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} r(s,a)x(s,a) - \sum_{j\in\mathcal{S}} \lambda_j \Bigg[\sum_{a\in\mathcal{A}} x(j,a) \\ &\quad - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} m(j|s,a)x(s,a) - \beta(j)\Bigg] - \sum_{j\in\mathcal{S}}\sum_{a\in\mathcal{A}} \gamma_{j,a}x(j,a). \end{aligned} \tag{11}$$

At the solution x^*_β of (10) there are Lagrange multipliers λ^*_β and γ^*_β such that the Kuhn-Tucker conditions hold ([**7**], p. 200). Let C be the cost vector and A the constraint matrix in 10, then

$$\begin{aligned} C - (\lambda^*_\beta)^t A - \gamma^*_\beta &= 0 \\ Ax^*_\beta &= \beta \\ x(s,a) &\geq 0 \\ \gamma^*_\beta(s,a) &\geq 0 \\ x_\beta(s,a)\gamma^*_\beta(s,a) &= 0. \end{aligned} \tag{12}$$

By the KT conditions and the Strong Duality Theorem ([**8**] pp. 39–40), λ^*_β is the solution of (8). The reduced cost coefficients are ([**7**], p. 198 eq. 9.1.10)

$$\frac{\partial R}{\partial x(j,a)} = -\gamma_{j,a}. \tag{13}$$

The optimal policy for the SMDP is obtained from the solution$\{x^*_\beta(j,a)\}$ of (10) using the following propositions.

PROPOSITION 1. *Let x^*_β be the solution of* (10)*. If $\forall s \in \mathcal{S}$ $\beta(s) > 0$, then $\forall s \in \mathcal{S}$ $\exists!$ $a \in \mathcal{A}$ such that $x^*_\beta(s,a) > 0$.*

PROOF. Assume that for some s, $x_\beta(s,a) = 0$ for all a. Then, since $m(j|s,a) \geq 0$ for all j, s, a a contradiction is obtained from the constraints in (10):

$$\begin{aligned} 0 &= \sum_{a\in\mathcal{A}} x^*_\beta(s,a) \\ &= \beta(s) + \sum_{j\in\mathcal{S}}\sum_{a\in\mathcal{A}} x(j,a)m(j|s,a) \\ &> 0. \end{aligned}$$

Therefore, for each s, there is at least one a such that $x^*_\beta(s,a) > 0$. The solution to (10) has at most $|S|$ basic variables [**7**] (pp. 152–159), and therefore for each s there is at most one a such that $x^a_\beta(s,a) > 0$. Therefore, for each s there is exactly one a such that $x^a_\beta(s,a) > 0$. □

The optimal stationary policy is then

$$d^*_\beta(s) = arg_{a\in\mathcal{A}} x^*_\beta(s,a) > 0 \tag{14}$$

$\{X^*_\beta(s,d(s))\}$ is an optimal and feasible basis of (10).

THEOREM 1. *Let $\alpha : S \to \mathbf{R}$ satisfy $\alpha(s) \geq 0$. Then 1. $\{X^*_\beta(s,d(s))\}$ is an optimal feasible basis of the linear programming problem:*

$$\textbf{Minimize} - R(x) = -\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} r(s,a)x(s,a) \tag{15}$$

such that $\forall j \in \mathcal{S}$

$$\sum_{a\in\mathcal{A}} x(j,a) - \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} m(j|s,a)x(s,a) = \alpha(j)$$

and $\forall j \in \mathcal{S}$ *and* $\forall a \in \mathcal{A}$

$$x(s,a) \geq 0$$

2. $\lambda^*_\alpha = \lambda^*_\beta$. *3.* $\gamma^*_\alpha = \gamma^*_\beta$.

PROOF. (1). Let M^* be the $|S|\times|S|$ matrix defined by $M(s,j) = m(j|s,d^*(s))$. Let $\sigma(I-M)$ be the largest eigenvalue of $I - M$. Then [**2**] (p. 607) $0 < \sigma(M) \leq \sum_{s\in\mathcal{S}} m(j|s) < 1$. Therefore by Corollary C.4 of [**2**] (p. 608) $(I-M)^{-1}$ exists, and

$$(I-M)^{-1} = \sum_{n=0}^{\infty} M^n. \tag{16}$$

Let

$$X_\alpha = (I-M)^{-1}\alpha \tag{17}$$

$$x_\alpha(s,a) = \begin{cases} 0 & : \quad a \neq d(s) \\ X_\alpha(s) & : \quad s = d(s) \end{cases} \tag{18}$$

is the optimal feasible solution of 15. From (16)–(18) it follows that $x_\alpha(s,a) \geq 0$ for all s, as $m(j|s) \geq 0$ for all j, s, and by construction it follows that $x_\alpha(s,a)$ satisfies the equality constraints in (15) and therefore is a basic feasible solution of the constraint equations. Optimality is then assured, as the optimality condition of a basic feasible solution of a linear program is independent of the constraint vector γ. This can be seen by expressing the constraint matrix in terms of basic and

non-basic variables so that after a permutation of columns, the constraint matrix is $A = [A_B A_N]$ and the cost vector is $c = [c_B c_N]$, where B corresponds to the basic variables and N to the non-basic variables. The optimality condition ([**7**] pp. 154–155) for the basic variables is for every component $c_N \geq -c_N A_B^{-1} A_N$. Thus $x(s, d(s))$ is feasible and optimal for $\alpha = \gamma$.

(2). The solution of the dual problem is $\lambda_\alpha^* = c_B A_B^{-1}$ ([**8**] p. 40 eq 2.52). therefore $\lambda_\alpha^* = \lambda_\beta^*$.

(3). The reduced cost coefficients are ([**8**] p. 28 eq 2.29) $\gamma_\alpha^* = c_N - c_N A_B^{-1} A_N$, and therefore $\gamma_\alpha^* = \gamma_\beta^*$. □

The application of the methods above to determine the optimal policy, state-value function, and the reduced cost coefficients requires the calculation of the integrals in (4) and (2), which is readily achieved if

$$F(du, s, a) = \sum_{\ell=1}^{h_{s,a}} \omega_{s,a}^{\ell} \delta(t - t_{s,a}^{\ell}) \tag{19}$$

$$\sum_{\ell=1}^{h_{s,a}} \omega_{s,a}^{\ell} = 1.$$

and

$$p(j|t, s, a) = \sum_{\ell=1}^{h_{s,a}} \delta(t - t_{s,a}^{\ell}) p(j|\ell, s, a). \tag{20}$$

Assuming that F and p have the form in (19) and (20), respectively, equations (2) and (4) simplify to

$$r(s, a) = k(s, a) + \sum_{\ell=1}^{h_{s,a}} e^{-\alpha t_\ell} c(j, s, a) p(j|\ell, s, a) \omega_{s,a}^{\ell}. \tag{21}$$

and

$$m_d(j|s) = \sum_{\ell=1}^{h_{s,a}} e^{-\alpha t_\ell} \omega_{s,a}^{\ell} p(j|\ell, s, a), \tag{22}$$

respectively.

The SMDPs are restless, i.e., they are allowed to change state when not acted upon or, equivalently, acted upon by the null action ($a_\emptyset$) that does not consume or utilize resources. We denote by $\mathcal{A}$ the action set consisting of the non-null actions, and let $\mathcal{A}^+ = \mathcal{A} \cup \{a_\emptyset\}$.

The size of the linear programming problem required to identify the optimal policy of a network of semi-Markov decision processes is given by the following theorem.

THEOREM 2. *Assume a network of N SMDPs such that each has a state space of size $|\mathcal{S}|$ and the same action set of size $|\mathcal{A}|$, which consists of the non-null actions.*

(1) The combined state space $\mathcal{CS}$ is the Cartesian product of the individual state spaces and has size

$$|\mathcal{CS}| = |\mathcal{S}|^N. \tag{23}$$

(2) Let $L = min(|\mathcal{S}|, |\mathcal{A}|)$ The combined number of actions is

$$|\mathcal{CA}| = \sum_{p=0}^{L} \binom{m}{p}\binom{N}{p} p! \tag{24}$$

(3) The linear programming approach to finding the optimal policy for this network has $|\mathcal{CA}||\mathcal{CS}|$ variables and $|\mathcal{CA}|$ constraints.

PROOF. (1) is immediate. The number of joint actions can be cataloged by choosing up to L actions and then assigning the chosen actions to the SMDPs which can be done in $\binom{N}{p}p!$ different ways, proving (2). (3) follows from Equation (14) and the size of the constraint matrix in (9). □

The constraint matrix for even a small network with $N = 25$, $|\mathcal{S}| = 5$, and $|\mathcal{A}| = 8$ has dimensions $3 \cdot 10^{17} \times 2 \cdot 10^{28}$. A direct solution seems impossible, and therefore, we develop a linear/integer programming heuristic to identify policies for such problems that generalizes the heuristic in [**5**] and can be used for real-time control in which the mean time to determine actions must be less than or equal to the mean action time.

Two versions of the heuristic's linear programming portion are developed. The inputs to the first version are the policies and reduced cost coefficients from Theorem 1, solved using the action set $\mathcal{A}^+$, where the index n is the corresponding SMDP index.

$$\{d^n, \gamma^n_{s_n,a_n} | 1 \le n \le N;\ s_n \in \mathcal{S}_n;\ a \in \mathcal{A}^+_n\}. \tag{25}$$

In the second version a total average resource constraint is included, and the optimal policy and reduced cost coefficients are obtained by solving the following LP. Let A_n and r_n denote the constraint matrix and reward vector, respectively, for the n^{th} SMDP from (9). Assume that the first index of the augmented action set is that of the null action. Arrange the indexing so that $r_n = \vec{r_n}(s, a)$ is the vector

$$r_n = [r_n(1,1), \ldots, r_n(|\mathcal{S}|,1), r_n(1,2), \ldots, r_n(|\mathcal{S}|,2), \ldots, r_n(1,|\mathcal{A}^+|), \ldots, r_n(|\mathcal{S}|,|\mathcal{A}^+|).$$

Let $s_0 = \overbrace{[0 \cdots 0]}^{|\mathcal{S}|}$, $s_1 = \overbrace{[1 \cdots 1]}^{|\mathcal{S}|}$, $w = [s_0 \overbrace{s_1 \cdots s_1}^{|\mathcal{A}|}]$, $W = \overbrace{[w \ldots w]}^{N}$, $r = [r_1 r_2 \ldots r_N]$, $\beta = [\beta_1 \beta_2 \ldots \beta_N]$, and

$$A = \begin{pmatrix} A_1 & 0 & \ldots\ldots & 0 \\ 0 & A_2 & 0 \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots\ldots\ldots & & A_n \end{pmatrix}. \tag{26}$$

Let ψ be the discounted weighted average resource availability, which serves to limit the number of simultaneous actions permitted by (in this case) the number of sensors and their limitations. A complete development of this approximation is given by Whittle [**4**]. Including this optional constraint, the LP portion of the heuristic is given by:

$$\textbf{Maximize } R(x) \quad = \quad r \cdot x \tag{27}$$

such that

$$\begin{aligned} Ax &= \beta \\ Wx &\leq \psi \\ x &\geq 0 \end{aligned} \tag{28}$$

The policies given by solutions to these LPs will generally not be simultaneously implementable as they do not include instantaneous resource constraints. Below we describe a heuristic using mixed integer programming which at each time step imposes the resource constraints. Each MIP is solved assuming that the state of the projects are known (or estimated), and thereby, the number of variables of the MIP is reduced relative to the number of variables of the LP. To simplify notation, the state index is dropped from the reduced cost coefficients in the context of the MIPs, i.e., $\gamma^n_{s_n,a_n}$ is denoted by $\gamma^n_{a_n}$.

2.2. Resource tasking given policy determination. The LPs described in the previous section produce a policy for action tasking. Their associated reduced cost variables facilitate

Define $c \in C$ as the elements within the set of project clusters, $C_a \subset C$ as the subset of clusters upon which action a can be conducted, and $N_c \subset N$ as the projects contained in cluster c. Each resource $b \in B$ can conduct a subset of actions, $A_b \subset A$, by taking action on a cluster (uniquely assigned to each resource). Let τ_{ab} denote the total time required to conduct action a by resource b, and δ_b be the total time available in the planning window for b. Note that τ_{ab} may incorporate multiple visits on the same project; this is warranted when action completion requires periodic revisits to a project. The binary variable y_{ac} indicates that a resource capable of action a is assigned to projects in cluster c. To preclude duplicate objective function rewards for multiple actions on a given project, a second variable, z_{na} indicates the specific assignment of action a on project n.

The binary integer program formulation is thus:

$$\textbf{Maximize} \sum_{n\in N}\sum_{a\in A}(\gamma^n_{a_\emptyset} - \gamma^n_{a_n}) \cdot z_{na} \tag{29}$$

subject to:

$$\sum_{c\in C_a : n\in N_c} y_{ac} \geq z_{na} \quad \forall\, n \in N,\, a \in A \tag{30}$$

$$\sum_{a\in A} z_{na} \leq 1 \quad \forall\, n \in N \tag{31}$$

$$\sum_{a\in A_s : c\in C_a} \tau_{ab} \cdot y_{ac} \leq \delta_b \quad \forall\, b \in B,\, a \in A \tag{32}$$

$$y_{ac} \in \{0,1\} \quad \forall\, a \in A,\, c \in C, \qquad 0 \leq z_{na} \leq 1 \quad \forall\, n \in N,\, a \in A \tag{33}$$

The objective function seeks to maximize the difference between passive and active reduced costs of activated projects, as motivated in the previous section. (See [**5**] for additional development of this rationale.) The first constraint ensures that a project cannot be activated without assignment of an action to a cluster containing that project. The second constraint restricts a project from being activated more

than once (multiple visits required for an action count as a single action). The last functional constraint limits the total resource assignments in a planning window to the length of that window. Finally, the assignment of an action to a cluster, y_{ac}, is restricted to binary; the action-target assignment variable, z_{na}, is not explicitly restricted as binary, but its objective function support combined with the first constraint will so limit it. The application discussed in the next section requires fast solutions, a trait not generally attributable to integer programs. Though this formulation lacks network or other polynomial time structure, the required problem instances are moderate, and the solution times are short. A typical problem instance for this work might have 4 resources, 24 projects, 5 actions, and 24 overlapping clusters of projects. This example yields an integer program with approximately 200 rows, 100 columns, and 500 non-zero elements in the constraint coefficient matrix.

3. Application: Airborne Surveillance of Military Targets

Military surveillance provides the motivation for this work. In the typical modern battlespace, several airborne platforms are tasked to observe dozens of targets arrayed in geographical clusters using a variety of sensors. Although intelligence needs vary, two common classes of required information are: 1) kinematic–where the target is, and 2) identification–what the target is. Good assignments pair sensors and target clusters in a manner that keeps the level of uncertainty for each target low, recognizing that target knowledge undergoes Markov transitions.

Our experiments consider 12 targets that require kinematic (K) intelligence only, and 12 targets that require both kinematic (K) and identification (ID) intelligence. There are three states for kinematic-only targets: good knowledge, medium knowledge, and bad (poor) knowledge with scores 4, 2, and 0, respectively. ID knowledge is characterized as either "good" or "bad." Thus there are six states for kinematic and ID (K/ID) targets, each labeled with kinematic knowledge followed by ID knowledge: Good/Good, Good/Bad, Medium/Good, Medium/Bad, Bad/Good, and Bad/Bad. The scores for these states are 8, 4, 4, 2, 1, and 0, respectively. An action on K/ID targets requires twice the time required for K-only targets. The experiments attempt to maximize the sum of time-weighted knowledge scores for the targets, and parametrically vary: 1) the sensors' inherent intelligence-gathering effectiveness between moderate and high, 2) the number of target clusters between 4 and 24, and 3) the number of sensor platforms between 2 and 4.

For comparative purposes, we also score a simpler assignment strategy that is similar to that in operational use. This greedy strategy assigns sensors to clusters based on a "lowest-knowledge first" methodology, whereby the cluster with the lowest knowledge is assigned the best sensor for its mix of targets, and so forth until all of the available sensor time is exhausted.

The comparison between the MDP and the greedy (G) approaches is presented as a percentage improvement in Table 1. In comparison to the greedy approach, the MDP heuristic improves target knowledge scores by -1% to 77% when only kinematic knowledge is required, and by 83% to 409% when maintaining both kinematic and identification knowledge. The improvement is more pronounced with greater sensor effectiveness, more targets, and more sensors. Thus, problem complexity appears to favor the MDP heuristic.

Detailed results for the high effectiveness, 12 cluster, 4 platform case are given in Figure 1. The MDP Heuristic is able to maintain the average target in the "Good Kinematic, Good ID" state 77% of the time, compared with the greedy method where the average target is in the "Good Kinematic, Bad Identification" state 73% of the time. The remaining target occupancy states using the MDP heuristic are also generally better than those of the greedy method. Other experiments suggest a similar trend: in cases where the two methods' scores diverge, much of the divergence owes to the MDP heuristic's ability to maintain a high fraction of the best state, while the greedy heuristic maintains a similar fraction in the second-best state.

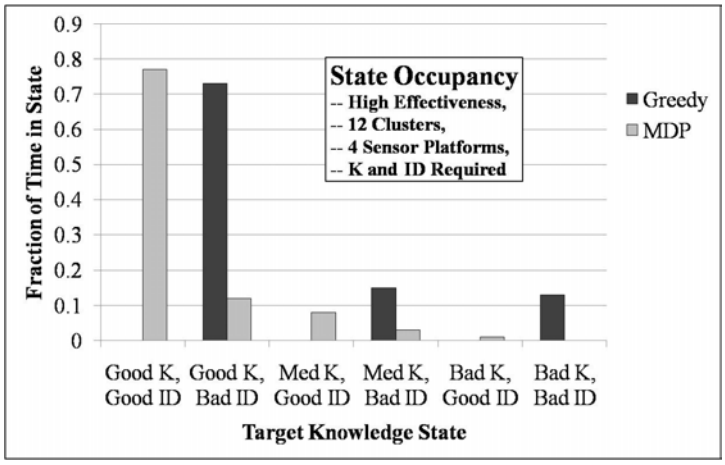

FIGURE 1. The MDP heuristic maintains targets requiring both kinematic and identification knowledge in the highest state 77% of the time in this sample experiment. In contrast, the greedy heuristic maintains the same targets in the second highest state 73% of the time. This result is typical of the experiments conducted.

Sensor assignment and reassignment computations must occur rapidly in order to accommodate a dynamic battlespace. Conceivably, sensor assignments may change as quickly as several times each minute, indicating that the assignment method should require at most a few seconds. Figure 2 gives the average solve times for the integer program; all are well under two hundredths of a second. The linear program portion of the heuristic is much quicker, requiring no more than $6\text{x}10^{-4}$ seconds. Though the solution times increase markedly in the most complex cases, these latter experiments represent the level of complexity for the operational scenarios in which we intend to implement this work. Moreover, we attained these solutions on a single platform two Ghz laptop using MATLAB®, suggesting ample room for speedup if necessary.

Sensor	Target	Sensor	Kinematic		K/ID		Improvement	
Effectiveness	Clusters	Platforms	G	MDP	G	MDP	K	K/ID
Moderate	4	2	.906	.897	.455	.833	-1%	83%
Moderate	4	3	.906	.955	.456	.927	5%	103%
Moderate	4	4	.911	.979	.453	.964	7%	113%
Moderate	6	2	.711	.839	.381	.775	18%	103%
Moderate	6	3	.910	.912	.455	.878	0%	93%
Moderate	6	4	.909	.952	.456	.936	5%	105%
Moderate	12	2	.502	.731	.278	.672	46%	142%
Moderate	12	3	.678	.857	.353	.809	26%	129%
Moderate	12	4	.810	.910	.408	.883	12%	116%
Moderate	24	2	.367	.650	.120	.611	77%	409%
Moderate	24	3	.513	.799	.288	.764	56%	165%
Moderate	24	4	.610	.880	.335	.849	44%	153%
High	4	2	.957	.947	.479	.909	-1%	90%
High	4	3	.957	.989	.478	.974	3%	104%
High	4	4	.954	1.00	.478	1.00	5%	109%
High	6	2	.780	.900	.402	.847	15%	111%
High	6	3	.956	.950	.479	.927	-1%	94%
High	6	4	.957	.980	.478	.972	2%	103%
High	12	2	.534	.808	.290	.758	51%	161%
High	12	3	.651	.898	.391	.858	38%	119%
High	12	4	.829	.936	.432	.911	13%	111%
High	24	2	.434	.702	.235	.690	62%	194%
High	24	3	.514	.843	.299	.807	64%	170%
High	24	4	.624	.900	.366	.874	44%	139%

TABLE 1. Comparative results between the greedy (G) and Markov decision process (MDP) heuristics indicate that significantly improved kinematic (K) and identification (ID) knowledge is maintained by the latter method, particularly in cases of greater complexity (i.e., high sensor effectiveness, more target clusters, and more sensor platforms). The first three columns list the sensor, cluster, and platform parameters; the Kinematic and K/ID columns indicate the fraction of target score (for each heuristic) against "perfect knowledge", where full K and/or ID information is maintained on all targets. The Improvement columns indicate percentage improvement of the MDP method over the greedy method. For instance, in the "High Effectiveness, 24 Cluster, 4 Platform" case (last row), the MDP target knowledge score averages 44% better when only kinematic knowledge is required, and 139% better when both K and ID knowledge is required.

4. Summary

This work develops a method for real-time control of restless bandits using a heuristic based on semi-Markov decision processes. It generalizes previous work[**5**] to allow for diverse actions on clusters of bandits using multiple actors. The

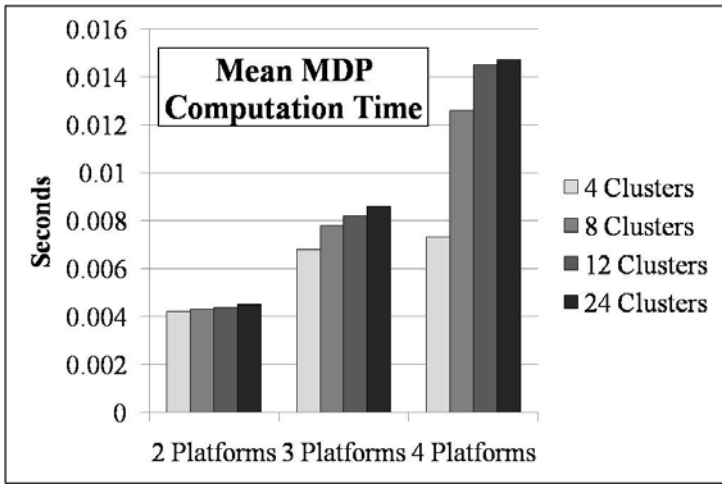

FIGURE 2. The mean time required to perform the integer programming portion of the MDP heuristic varies between 4 and 15 thousandths of a second; the linear programming portion (not depicted) solve times are an order of magnitude shorter. As expected, this time increases with number of sensor platforms and number of target clusters.

variable-length actions are assigned within a specified time window, and can be both periodic and sequential. The method remains tractable by: 1) approximating the Markov decision space by reducing it to one characterized by the states of the individual bandits, and 2) incorporating a compact integer program using dual information from the resulting MDP.

Initial experiments using the combined LP/MIP heuristic indicate that this method is both practical and beneficial for a military sensing application. The problem sizes tested were moderate, involving several sensors and dozens of targets, and solved in fractions of a second on a laptop computer. These tests yielded performance improvements of up to 400% over a greedy method, and reached 60 to 100% of the maximum attainable score. Hence, the technique shows promise for similar problems, as well as larger ones where more more time and computing power are available.

Several challenges remain. Once assigned, individual sensors must be scheduled quickly, and in a manner that preserves action length and target revisit rate. The state rewards used in our work are arbitrary; rigorous reconciliation of these rewards with inherent knowledge value remains key to successful application. Sensor location is also not addressed: motion of the sensor platforms, as well as optimal placement of these platforms remains a promising area to improve target knowledge. Finally, real world applications are subject to communications delays and other phenomena that reduce the interactive ability of the sensors; this clouding of knowledge state could be addressed with partially observable Markov processes.

MDPs provide a prescriptive method for decision-making under uncertainty, but must be applied in a manner that accommodates the myriad probabilistic combinations without precluding timely solution. The development of heuristics backed

by strong theory appears central to this compromise. The combined linear/integer program method described in this paper offers such a strategy, and shows promise for application to a contemporary military problem.

References

[1] Department of the Air Force, *Intelligence, Surveillance, and Reconnaissance Operations, Air Force Doctrine Document 2-5.2*, Air Force Doctrine Center, 21 April 1999.

[2] M. L. Puterman, *Markov Decision Processes*, John Wiley and Sons, Hoboken, New Jersey, 2005.

[3] J. Gittins and D. Jones, *A Dynamic Allocation Index for the Discounted Multiarmed Bandit Problem*, Biometrica **66** (1974), pp. 561–565.

[4] P. Whittle, *Restless Bandits: Activity Allocation in a Changing World*, A Celebration of Applied Probability, Journal of Applied Probability **25A** (1988), pp. 287–298.

[5] D. Bertsimas and J. Nino-Mora, *Restless Bandits, Linear Programming Relaxations, and a Primal-Dual Index Heuristic*, J. Operations Research, **48** (2000), pp. 80–90.

[6] A. Hero, D Castanon, D. Cochran, and K. Kastella (eds.), *Foundations and Applications of Sensor Management*, Springer, New York, 1988.

[7] R. Fletcher, *Practical Methods of Optimization*, Second edition, John Wiley and Sons, New York, 1986.

[8] I. Maros, *Computational Techniques of the Simplex Method*, Kluwer Academic Publishers, Boston, 2003.