Single and Multi-agent Exploration of a Markov Decision Process

Timothy C. Y. Chan
Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139
tcychan@mit.edu

Eric Feron
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139
feron@mit.edu

Abstract

In this paper, we investigate the problem of efficiently visiting every state of a Birth-Death Markov Decision Process (BDP). In the single-agent case, we consider two different BDPs and derive their corresponding optimal policies - one being a pure greedy policy, and the other being a threshold policy based on greedy actions. We then generalize to a multi-agent setting and prove the optimality of a greedy policy in a related problem.

1 Introduction

This paper aims to understand the concept of "efficiently" visiting the states of a Markov Decision Process. In particular, we study Birth-Death Processes (BDPs) with the goal of characterizing policies which minimize the expected time to visit every state in the state space (at least) once. We will refer to this time as the expected "covering time" of the process.

As a motivation, consider one or more autonomous vehicles doing search and discovery over an unexplored terrain. Uncertainties may arise from imperfect sensor measurements, the presence of obstacles, or environmental conditions such as weather. The goal is to search the entire region as quickly as possible, while managing the uncertainties which affect planning and mobility. One may also view the single-agent problem as a generalization of a TSP problem. In fact, it is not difficult to show that finding an optimal covering policy in a specially structured MDP is equivalent to finding a minimum cost tour in a related directed graph.

In essence, the problem we are faced with is a "controlled" random walk on a graph. Random walks on graphs have been studied extensively and in that domain, there is a similar notion of a "cover time", which is defined to be the maximum, over all starting vertices, of the time it takes the walk to reach each vertex once [2], [3], [4]. Just as there is an equivalence between a random walk on a graph and a Markov chain, our controlled random walk will be represented by a Markov Decision Process. We use the modified term "covering time" to distinguish our controlled version of a random walk on a graph. In addition, we will study the general case where the probability distribution of transitions out of a state is not uniform, which is generally not considered.

1.1 Notation and Terminology

The Markov Decision Process will be described by a set of states, $\mathcal{S}$, and a set of actions $\mathcal{A}$, both of finite cardinality. For each $a \in \mathcal{A}$, there is an associated probability transition matrix P^a, where $P^a(i, j)$ is the probability of going from state i to state j if action a is used. We will also write this as $p^a_{i,j}$. Throughout, we will refer to the time until all states are visited as the "covering time" for that particular problem. Time will be counted in discrete steps, with one unit of time corresponding to one transition of the process. We will assume transitions occur at integer times, and thus, when we talk about a minimal time, we are referring to a minimal number of transitions.

2 Optimal Single-Agent Covering Policies for Birth-Death Processes

2.1 Problem Setup

Picture a Birth-Death Markov chain with n states numbered 1 to n from left to right. These are the "nominal states" as they refer to the underlying Markov chain. We make this distinction here to avoid any confusion with the "state" in the Dynamic Programming formulation to follow. States 1 and n are the *terminal states*, and if the process starts at a terminal state, we assume it is state 1, without loss of generality. Also, for processes which start in state 1, we assume that state n is absorbing for all actions (since reaching n means that all states have been covered). When referring to the process as going "left" or "right", we are talking about a transition from state i to $i-1$, or from state i to $i+1$, respectively. As for the evolution of the process, the agent will choose an action out of m available choices at each time step and will transition out of its current state according to the probability distribution governed by that action (self-transitions are possible). The process terminates once each state has been visited once.

Let $V^*(\sigma)$ denote the optimal expected covering time given that the current state of the system is σ, while for a general policy π, V_π is the corresponding value function vector. $\pi(i)$ denotes the action to be used at state i, which will also be written $a(i)$. We denote by $a^*(i)$ the optimal action at state i.

Finally, we assume the existence of a *proper policy*[1], and that every *improper policy* has infinite cost (which is satisfied here since a process which never reaches the termination state will have infinite expected covering time). Specifically, for a process which starts in state 1, this assumption amounts to the existence of actions which at each state i have positive probability of going to state $i+1$. Similarly, for a process which starts in a non-terminal state, there must either be positive probability of going right to n and then left to 1, or left to 1 and then right to n.

2.2 A Dynamic Programming formulation

The problem as described falls in the class of Stochastic Shortest Path (SSP) problems [1], and the problem of minimizing the expected covering time can be formulated as

$$\begin{aligned} V^*(\sigma) &= \min_a \{1 + \sum_\tau p^a_{\sigma,\tau} V^*(\tau)\} \quad \forall\, \sigma \\ V^*(\upsilon) &= 0, \end{aligned}$$

[1]A proper policy ensures a positive probability of reaching the termination state after some finite time (see [1]).

where v is the termination state. In general, to keep track of the unvisited nominal states, the state space of the DP will grow exponentially - the “curse of dimensionality”. However, for Birth-Death MDPs, the special structure of the chain allows for a compact representation of the state space.

It turns out that the presence/absence of self-transitions in the non-terminal states, and the starting state of the process are the main factors which influence the different optimal policies. This leads to four different processes and the table below illustrates their relationships to each other.

	No self-transitions	Self-transitions
Start in state 1	Type 1	Type 2
Start in non-terminal state	Type 3	Type 4

Table 1: Four types of Birth-Death MDPs

The subsequent analysis will focus mainly on the cases where there are no self-transitions in the non-terminal states. Type 2 BDPs will be discussed briefly, but the main results will come from Type 1 and Type 3 BDPs. Type 4 will be omitted.

2.3 Type 1: no self-transitions; start in state 1

Assume that for states $i = 2, \ldots, n-1$, $p^a_{i,i} = 0\ \forall a \in \mathcal{A}(i)$. Since the process starts in state 1, the state of the system is captured by the nominal state $i \in \mathcal{S}$. Let $V^*(i)$ be the optimal expected time to go from state i to n. First we prove a simple, but useful lemma, which is true for any proper policy.

Lemma 2.1 *$V^*(i+1) < V^*(i)$ for $i = 1, \ldots, n-1$.*

Proof: Given that the current state is i, the process must pass through state $i+1$ to get to n. So $V^*(i) = \tilde{V}^*(i, i+1) + V^*(i+1)$, where $\tilde{V}^*(i, i+1)$ is the optimal expected time to reach $i+1$ from i. $\tilde{V}^*(i, i+1) \geq 1$, thus, $V^*(i+1) < V^*(i)$.

■

Using this lemma, we now show that the locally “greedy policy” is globally optimal.

Theorem 2.1 *For a Type 1 BDP, the optimal covering policy consists of actions, which at each state i, maximize the probability of going to state $i+1$, for $i = 1, \ldots, n-1$.*

Proof: For $i = 2, \ldots, n-1$,

$$\begin{aligned} V^*(i) &= \min_a \{1 + p^a_{i,i+1} V^*(i+1) + p^a_{i,i-1} V^*(i-1)\} \\ &= 1 + V^*(i-1) + \min_a \{p^a_{i,i+1}(V^*(i+1) - V^*(i-1))\} \\ &= 1 + V^*(i-1) + (\max_a p^a_{i,i+1})(V^*(i+1) - V^*(i-1)) \qquad (1) \end{aligned}$$

where (1) comes from Lemma 2.1. The case for $i = 1$ is similar.

■

The intuition behind this should be very clear. If we want to reach state n as quickly as possible, we gain nothing by choosing actions with lower probabilities of going to the right. This is true because at state i, any probability of not going to state $i+1$ is *entirely* deflected to state $i-1$. So in this case, it is always best to choose the actions in a greedy manner.

2.4 Type 2: self-transitions; start in state 1

Again, since we start in state 1, the objective is to reach state n as quickly as possible. It may be tempting to think that the same greedy policy from before would be optimal here, but in fact, this is not the case. The existence of self-transitions makes the problem more complex as they can now "absorb" a lot of the probability that was either thrown to the left or right transition out of i in the previous case. Consider the example shown in Figure 1 with three states and two actions available at each state.

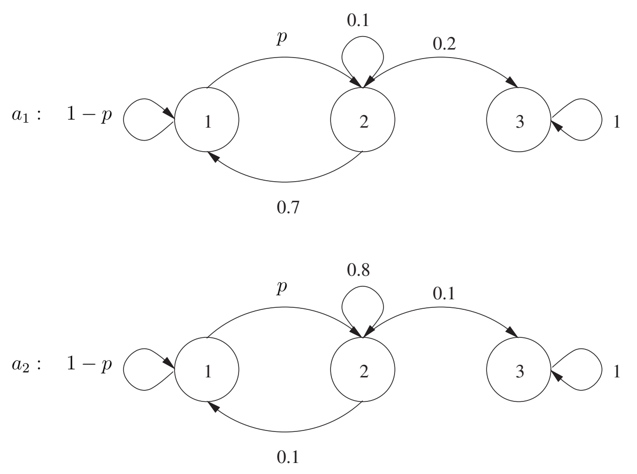

Figure 1: A counterexample to show that the greedy policy is not optimal in Type 2 BDPs.

For this process, there are only two possible policies. Let π_1 correspond to the policy of choosing a_1 at state 2, while π_2 uses a_2 at state 2. It is straightforward to show that

$$V_{\pi_1}(1) = 5 + \frac{9}{2p} \quad \text{and} \quad V_{\pi_1}(2) = 5 + \frac{7}{2p},$$

and

$$V_{\pi_2}(1) = 10 + \frac{2}{p} \quad \text{and} \quad V_{\pi_2}(2) = 10 + \frac{1}{p}.$$

If $p = \frac{1}{2}$, both policies are equivalent in terms of optimal expected covering time. If $p > \frac{1}{2}$, then π_1 is optimal, while if $p < \frac{1}{2}$, then π_2 is optimal. This shows that the greedy policy (π_1) is not always optimal.

Notice that although $p_{2,3}^{a_1} > p_{2,3}^{a_2}$, choosing a_1 at state 2 results in a high probability of going back to state 1, whereas choosing a_2 leads to a low probability of returning to state 1. The intuition is that even if some action has the highest probability of going right at state i, it may also have a high probability of going left, and so it may be more advantageous to pick an action with a smaller probability of going right, but a high probability of staying in i.

In a sense, this illustrates a trade-off between risk and return. The greedy action, a_1, can potentially garner a greater return (go right with a higher probability), but it is more risky (high probability of going left). The degree of risk-averseness of the decision-maker is shaped by other factors such as how quickly one could recover from a temporary loss due to choosing a risky action or investment (eg. the maximum probability of going from $i-1$ to i over $a \in \mathcal{A}(i-1)$). In our example, a larger "recovery" probability ($p > \frac{1}{2}$) will result in a less risk-averse choice (pick action a_1 in state 2).

2.5 Type 3: no self-transitions; start in non-terminal state

In order to cover all the states, we must now consider actions which will quickly take us in both directions. In fact, the challenge in this problem is to characterize the optimal actions at states $2, \ldots, n-1$ *before* one of the terminal states is reached. Once either state 1 or n is reached, the problem is reduced back to a Type 1 BDP and the greedy policy will be optimal to cover the remaining unvisited states. Thus, an optimal policy on the nominal states is non-stationary.

A standard method to solve such a problem would be to augment the original state space with an entity which denotes which terminal state (if any) has been visited yet. With the appropriate transition probabilities for each action, a, the DP recursion can be solved to give us an optimal *stationary* policy in this augmented state space. However, there is a more clever way to solve this problem without the need for a larger state space. Since the greedy policy is optimal once the process reaches a terminal state, if we put "weights" α and β at state 1 and n, respectively, equal to the optimal expected time to reach the other terminal state under the corresponding greedy policy, solving the following DP will give us an optimal policy for this problem.

$$\begin{aligned} V^*(i) &= \min_a \{1 + p^a_{i,i+1} V^*(i+1) + p^a_{i,i-1} V^*(i-1)\} \quad i = 2, \ldots, n-1 \\ V^*(1) &= \alpha \\ V^*(n) &= \beta. \end{aligned}$$

In what follows, we will be working with this "weighted" version of the state space, and as a result, we are only interested in the time up until a terminal state is first visited. With this representation, we will show that an optimal policy is a "threshold policy." That is, if the process is in a state to the right (left) of the threshold, the optimal action will be the action with the highest probability of going right (left). Two lemmas will help us to establish this result.

Before we continue, we make a quick note about our terminology. Earlier, we described what it meant to "go right" or "go left" when referring to movement along the chain. Now we also use these terms to describe actions. An action, $a^*(i)$, "goes right" at i if it satisfies

$$p^{a^*(i)}_{i,i+1} \geq p^a_{i,i+1} \quad \forall a \in \mathcal{A}(i). \tag{2}$$

We will use $a^*(i) = \arg\max_a p^a_{i,i+1}$ as a shorthand (as there may be more than one maximizer) for the more precise expression (2). A similar definition prevails for the action which "goes left" at state i.

Lemma 2.2 *$V^*(i)$ will always be minimized at $a^*(i)$ satisfying*

$$p^{a^*(i)}_{i,i+1} \geq p^a_{i,i+1} \quad \forall a \in \mathcal{A}(i),\ i = 1, \ldots, n-1$$

or

$$p^{a^*(i)}_{i,i-1} \geq p^a_{i,i-1} \quad \forall a \in \mathcal{A}(i),\ i = 2, \ldots, n.$$

Proof: This lemma is a straightforward generalization of Theorem 2.1.

■

Although the optimal action at state i may not be unique, the important point is that one of $\arg\max_a p^a_{i,i+1}$ or $\arg\max_a p^a_{i,i-1}$ will always be optimal. This reduces the "min" in Bellman's equation to essentially a minimum over two actions - possibly a major computational savings.

Lemma 2.3 *If at state i, the optimal action is $a^*(i) = \arg\max_a p^a_{i,i+1}$ $(= \arg\max_a p^a_{i,i-1})$, then at state $i+1$ $(i-1)$, the optimal action is $a^*(i+1) = \arg\max_a p^a_{i+1,i+2}$ $(a^*(i-1) = \arg\max_a p^a_{i-1,i-2})$.*

Proof: Because of symmetry, it is sufficient to show

$$a^*(i) = \arg\max_a p^a_{i,i+1} \quad \Rightarrow \quad a^*(i+1) = \arg\max_a p^a_{i+1,i+2}.$$

Suppose the optimal action at i is $a^*(i) = \arg\max_a p^a_{i,i+1}$. So

$$\begin{aligned} V^*(i) &= \min_a \{1 + p^a_{i,i+1} V^*(i+1) + p^a_{i,i-1} V^*(i-1)\} \\ &= 1 + p^{a^*(i)}_{i,i+1} V^*(i+1) + p^{a^*(i)}_{i,i-1} V^*(i-1). \end{aligned} \tag{3}$$

This implies $V^*(i+1) \le V^*(i-1)$. If we now suppose that $V^*(i) \le V^*(i+2)$, then

$$\begin{aligned} V^*(i+1) &= \min_a \{1 + p^a_{i+1,i+2} V^*(i+2) + p^a_{i+1,i} V^*(i)\} \\ &= 1 + p^{a^*(i+1)}_{i+1,i+2} V^*(i+2) + p^{a^*(i+1)}_{i+1,i} V^*(i), \end{aligned} \tag{4}$$

where

$$a^*(i+1) = \arg\max_a p^a_{i+1,i}.$$

Since we assumed that $V^*(i) \le V^*(i+2)$, (4) is equivalent to

$$V^*(i) \le V^*(i+1) - 1. \tag{5}$$

Using (3) together with (5) and letting $p = p^{a^*(i)}_{i,i+1}$, we have

$$\begin{aligned} & V^*(i+1) - 1 \ge 1 + pV^*(i+1) + (1-p)V^*(i-1) \qquad (6) \\ \Leftrightarrow \quad & V^*(i+1) \ge \frac{2}{1-p} + V^*(i-1) \\ \Rightarrow \quad & V^*(i+1) > V^*(i-1) \end{aligned}$$

contradicting the optimality of $a^*(i)$ (if $p = 1$, the contradiction appears in (6)). Hence, $V^*(i+2) < V^*(i)$ and $a^*(i+1)$ satisfies $p^{a^*(i+1)}_{i+1,i+2} \ge p^a_{i+1,i+2} \quad \forall a \in \mathcal{A}(i+1)$.

■

This lemma makes concrete the intuitive fact that if at state i the optimal action is to go right, then it must be best to try and reach state n first, so at states to the right of i, the optimal action is still to go right. Next, we characterize the optimal policy.

Theorem 2.2 *Given a Type 3 BDP and up to the time at which a terminal state is first reached, there exists a threshold $k \in \{2, \ldots, n-1\}$ such that $a^*(i) = \arg\max_a p^a_{i,i+1}$ for $i = k+1, \ldots, n-1$, $a^*(i) = \arg\max_a p^a_{i,i-1}$ for $i = 2, \ldots, k-1$, and $a^*(k) = \arg\max_a p^a_{k,k+1}$ or $a^*(k) = \arg\max_a p^a_{k,k-1}$ or both will be optimal.*

Proof: From Lemma 2.2, we know that the optimal action at a non-terminal state i either has the highest probability of going to $i+1$ or the highest probability of going to $i-1$. Suppose there exists two distinct states $i_1, i_2 \in \{2, \ldots, n-1\}$ such that at i_1 it is optimal to go left and at i_2 it is optimal to go right (if not, either $k = 2$ or $k = n-1$).

Without loss of generality, let i_1 be the largest (rightmost) state at which it is optimal to go left, and let i_2 be the smallest (leftmost) state at which it is optimal to go right. Then Lemma 2.3 asserts that $i_1 = i_2 - 1$, so $k = i_1$ or i_2.

A simple, symmetric example with n odd should convince the reader that it is possible for there to exist a state k right on the "boundary" of the threshold. That is, at k, both $a^*(k) = \arg\max_a p^a_{k,k+1}$ and $a^*(k) = \arg\max_a p^a_{k,k-1}$ could be optimal, which is equivalent to the expected covering time being equal in both directions. In fact, this state is unique, if it exists. If there are two such states, k_1 and k_2, with $k_1 < k_2$, then

$$V_{\pi_l}(k_1) < V^*(k_2) < V_{\pi_r}(k_1),$$

where π_l is the optimal policy employing $a^*(k_1) = \arg\max_a p^a_{k_1,k_1-1}$, and π_r is the optimal policy employing $a^*(k_1) = \arg\max_a p^a_{k_1,k_1+1}$, which contradicts the assumption that both the "go right" and "go left" action are optimal at k_1. Thus, there is at most one state right on the threshold, and for the other states, their optimal actions are determined by the side of the threshold on which they fall.

■

Finally, we show that policy iteration, given *any* initial policy, will terminate with the optimal policy up to the time a terminal state is first reached, in at most $n+1$ iterations.

Definition 2.1 *Given a set $\mathcal{S} = \{1, \dots, n\}$ and a function $f : S \to \mathbb{R}$, a* local max *is an element $i \in \mathcal{S}$ which satisfies $f(i) \geq f(i+1)$ and $f(i) \geq f(i-1)$ (only $f(i) \geq f(i+1)$ if $i = 1$ and only $f(i) \geq f(i-1)$ if $i = n$). If $f(i) \geq f(j)$ $\forall j \in \mathcal{S}$, then i is a* global max*. If the inequalities are strict, then i is a* strict local max *or* strict global max*, respectively. Similar definitions prevail for a* min *with the inequalities reversed.*

Definition 2.2 *A* maximal plateau *is a sequence of consecutive elements $(i, i+1, \dots, i+k)$ of $\mathcal{S}$ with each element being a local max. An analogous definition prevails for a* minimal plateau*.*

Remark 2.1 *From the definition of a local max (min), the elements $(i, \dots, i+k)$ of a maximal (minimal) plateau satisfy $f(i) = \cdots = f(i+k)$.*

Proposition 2.1 *For any policy, π,*

a) Either $V_\pi(\cdot)$ has a unique, strict global max or $V_\pi(\cdot)$ has a maximal plateau of two elements, both of which are global maxima.

b) In the presence of a global max (maximal plateau), $V_\pi(\cdot)$ is monotonically increasing up to the global max (maximal plateau) and monotonically decreasing after it.

Remark 2.2 *Part b) implies that $V_\pi(\cdot)$ is (the discrete analogue of) quasi-concave.*

Proof: *a)* V_π is a function from the finite set $\{1, \dots, n\}$ to the real line. The image of V_π is clearly a finite set so there exists a maximal element. Suppose there are at least two (local) maximizers of V_π in the set $\{1, \dots, n\}$ which do not form a plateau. Then between two of these maximizers must be a state, j, which is a local min. From the definition of a local min, $V_\pi(j) \leq pV_\pi(j+1) + (1-p)V_\pi(j-1)$ for all $p \in [0, 1]$, which contradicts Bellman's equation at state j. If there are three or more maximizers forming a plateau, then the "interior" maximizers must have the same function value as their neighbors, which again contradicts Bellman's equation. Thus, V_π either has a unique maximum or it has a maximal plateau of two elements.

b) Since there is only one maximum (or maximal plateau of two elements), the value function must be increasing to the left of it and decreasing to the right of it. In fact, V_π must be strictly monotonic on either side of the maximum, otherwise there would be a plateau and the plateau element which is a local min would not satisfy Bellman's equation.

$\blacksquare$

With these properties of the value function, we move to the final result of this section.

Theorem 2.3 *Given any initial policy, policy iteration will terminate with the optimal policy for a Type 3 BDP in at most $n+1$ iterations. If the initial policy is a threshold policy, then policy iteration will take at most n iterations.*

Proof: Let π_k be any policy and let i^* satisfy $V_{\pi_k}(i^*) \geq V_{\pi_k}(i)$ for all i. Then for $i > i^*$, $V_{\pi_k}(i+1) < V_{\pi_k}(i-1)$ and so the next policy in policy iteration, π_{k+1}, will have $\pi_{k+1}(i) = \arg\max_a p^a_{i,i+1}$. Similarly, for $i < i^*$, $\pi_{k+1}(i) = \arg\max_a p^a_{i,i-1}$. At i^*, $\pi_{k+1}(i^*)$ will either be $\arg\max_a p^a_{i,i+1}$ or $\arg\max_a p^a_{i,i-1}$ from Lemma 2.3.

Thus, policy iteration will always generate a threshold policy at each iteration. Since there are n possible threshold policies, if policy iteration is started with a threshold policy, it will terminate in at most n steps ($n+1$ steps if it is started with an arbitrary policy).

$\blacksquare$

3 Multi-agent Covering Policies for Birth-Death Processes

3.1 Problem Setup

We consider a BDP as before, except now there are two communicating agents, denoted I and II, working as a team to cover the states - a state is considered visited when one of the two agents have reached it. We assume these agents are identical and independent. That is, agent I's available actions at any state i are the same as the agent II's at i, and the agents' choice of actions need not depend on each other. The state of the system must now take into account information about both agents.

We define $C^*(\sigma, \tau)$ as the minimum expected covering time given that agent I's state is σ and agent II's state is τ. σ and τ may contain the current nominal state plus a history of visited states for agents I and II, respectively. Here, we consider a simple extension of a single-agent Type 1 BDP to the two-agent case. Consider the scenario when both agents start at state 1 and the goal is to reach state n as quickly as possible. Analogous results hold for the case where I starts in 1 and II starts in n.

3.2 Type 1: no self-transitions, both start in state 1

If we assume that there are no self-transitions at the non-terminal states, then as we saw in Section 2.3, the agents gain nothing by acting in a "non-greedy" manner. Since both agents start in state 1, their current nominal state is the complete state, again, analogous to the single-agent case. Let $C^*(i,j)$ be the minimum expected covering time given that agent I is currently in state i and agent II is currently in state j. Bellman's equation is now

$$\begin{aligned} C^*(i,j) &= \min_a \min_b \{1 + \sum_k \sum_l p^a_{i,k} p^b_{j,l} C^*(k,l)\} && i \neq n, j \neq n \\ C^*(i,j) &= 0 && i = n \text{ or } j = n. \end{aligned}$$

It should be clear that the double sum above will produce exactly four terms. The following is a generalization of Lemma 2.1 to the two-agent case.

Lemma 3.1

a) Given any $j = 1, \ldots, n$, $C^*(i+1,j) \leq C^*(i,j)$, $\forall i = 1, \ldots, n-1$.

b) Given any $i = 1, \ldots, n$, $C^*(i,j+1) \leq C^*(i,j)$, $\forall j = 1, \ldots, n-1$.

Proof: We will only prove a) since b) follows from a symmetrical argument. Let X^i be the (random) time until agent I reaches state n under an optimal policy, given that it is currently in state i. Y^j is defined similarly for agent II. Then, $C^*(i,j) = \mathbb{E}(Z^{i,j})$, where $Z^{i,j} = \min\{X^i, Y^j\}$.

Any sample path of agent I which reaches state n must go through state $i+1$, so

$$X^i = \tilde{X}^{i,i+1} + X^{i+1},$$

where $\tilde{X}^{i,i+1}$ is the random variable which describes the time it takes to go from state i to $i+1$ under agent I's optimal policy. Clearly, $\tilde{X}^{i,i+1} > 0$, so $X^{i+1} < X^i$. Thus,

$$\begin{aligned} Z^{i+1,j} &= \min\{X^{i+1}, Y^j\} \\ &\leq \min\{X^i, Y^j\} \\ &= Z^{i,j}, \end{aligned}$$

and taking expectations, we get $C^*(i+1,j) \leq C^*(i,j)$.

■

Now, we can generalize Theorem 2.1 to two agents. The following theorem asserts that each agent's optimal policy is the greedy policy.

Theorem 3.1 *For a two-agent Type 1 BDP, the optimal covering policies for the agents consists of actions, which at each state i, maximize the probability of going to state $i+1$, for $i = 1, \ldots, n-1$.*

Remark 3.1 *Since the agents are identical, their optimal policies will be the same greedy policy.*

Proof: For $i, j \in \{2, \ldots, n-1\}$, (the case for $i = 1$ or $j = 1$ will be omitted since the proof is the same with the indices appropriately adjusted)

$$\begin{aligned} C^*(i,j) = \min_a \min_b \{1 &+ p^a_{i,i+1} p^b_{j,j+1} C^*(i+1,j+1) + p^a_{i,i+1} p^b_{j,j-1} C^*(i+1,j-1) \\ &+ p^a_{i,i-1} p^b_{j,j+1} C^*(i-1,j+1) + p^a_{i,i-1} p^b_{j,j-1} C^*(i-1,j-1)\}. \end{aligned} \tag{7}$$

Fix agent II's policy to an arbitrary policy, π, and let $p^{\pi(j)}_{j,j+1} = q$. Then the above can be re-written as

$$\begin{aligned} C^*(i,j) = \min_a \{ & 1 + q(p^a_{i,i+1}C^*(i+1,j+1) + p^a_{i,i-1}C^*(i-1,j+1)) \\ & + (1-q)(p^a_{i,i+1}C^*(i+1,j-1) + p^a_{i,i-1}C^*(i-1,j-1))\}. \end{aligned} \tag{8}$$

Since $C^*(i+1,j+1) \leq C^*(i-1,j+1))$ and $C^*(i+1,j-1) \leq C^*(i-1,j-1))$ from Lemma 3.1, the action which achieves the min in (8) must be the one which maximizes the probability of going right at state i. So the greedy policy is optimal for agent I.

A symmetrical argument holds for agent II, and therefore, if both agents follow the greedy policy, both mins will be achieved in (7) and this will be optimal for the system.

■

The upshot of this theorem is that each agent can act *independently*, according to their own optimal, greedy policy, and in fact this will be optimal for the system. Moreover, it can be shown that the above result holds for an agent starting in a terminal state even when the other agent starts in a non-terminal state.

4 Conclusion and Future Directions

In this paper, we have considered different variations of the problem of efficiently visiting the states of a Markov Decision Process. In the single-agent case, we characterized optimal policies for Birth-Death MDPs and showed that they can be calculated efficiently. We then showed that these results can be extended to a multi-agent environment.

Generalizing the problem to different MDPs will likely result in Dynamic Programs which will be too difficult to solve (especially as the number of agents grows), and thus, the goal should be to extract efficient approximation algorithms to cover the states. Furthermore, the multi-agent problem described deserves more attention and a generalization of the single-agent threshold policy should be possible.

References

[1] Bertsekas, D. P. *Dynamic Programming and Optimal Control.* Athena Scientific, 2000.

[2] Cooper, C. and Frieze, A. The cover time of random regular graphs. Pre-print, 2003.

[3] Coppersmith, D., Feige, U., Shearer, J. Random Walks on Regular and Irregular Graphs. *SIAM Journal on Discrete Mathematics* 9(2), pp. 301-308, 1996.

[4] Doyle, P. G. and Snell. J. L. *Random Walks and Electrical Networks.* The Mathematical Association of America, 1984.