Average-Reward Decentralized Markov Decision Processes

Marek Petrik
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
petrik@cs.umass.edu

Shlomo Zilberstein
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
shlomo@cs.umass.edu

Abstract

Formal analysis of decentralized decision making has become a thriving research area in recent years, producing a number of multi-agent extensions of Markov decision processes. While much of the work has focused on optimizing discounted cumulative reward, optimizing average reward is sometimes a more suitable criterion. We formalize a class of such problems and analyze its characteristics, showing that it is NP complete and that optimal policies are deterministic. Our analysis lays the foundation for designing two optimal algorithms. Experimental results with a standard problem from the literature illustrate the applicability of these solution techniques.

1 Introduction

Decentralized decision making under uncertainty is a growing area of artificial intelligence addressing the interaction of multiple decision makers with different goals, capabilities, or information sets. Markov decision processes (MDPs) have been proved useful for centralized decision making in stochastic environments, leading to the development of many effective planning and learning algorithms. More recently, extensions of MDPs to decentralized settings have been developed. Examples include stochastic games (SGs) or competitive Markov decision processes [Filar and Vrieze, 1996], and decentralized partially observable Markov decision processes (DEC-POMDPs) [Bernstein *et al.*, 2000]. SGs concentrate mainly on observable domains, capturing the competitiveness among the players. DEC-POMDPs generalize partially observable MDPs to multi-agent settings, modeling cooperative players who may have different partial knowledge of the overall situation.

This paper addresses the problem of average-reward decentralized MDPs. We analyze a class of DEC-POMDP problems, where the dependency and interaction among the agents is defined by a common reward function. The agents are otherwise independent and they each have full observation of their local, mutually-independent states. A similar model has been previously studied by [Becker *et al.*, 2004] with the objective of optimizing cumulative reward over a finite-horizon. In contrast, we analyze the infinite-horizon *average reward* problem, which has not been previously studied in a decentralized settings.

The need to optimize average reward has been demonstrated in many applications, including ones in reinforcement learning [Mahadevan, 1996], decentralized robot control [Tangamchit *et al.*, 2002], decentralized monitoring, sensor networks, and computer networks [Rosberg, 1983; Hajek, 1984]. In these problems, the system operates over an extended period of time and the main objective is to perform consistently well over the long run. The more common discounted reward criterion usually leads to poor long-term performance in such domains.

One motivation for using discounted reward models–even when the average reward criterion seems more natural–is that the problem is easier to analyze and solve [Puterman, 2005]. But it is not clear if the same argument prevails in decentralized settings. In fact, in some problems the average reward analysis may be simpler because the system may exhibit complex behavior initially, but quickly settle into simple one. One such example is the automaton used to prove that an optimal policy for a POMDP may not be regular [Madani, 2000]. The optimal discounted policy in this simple example is infinite, while the optimal average reward policy is very simple and contains just a single action.

In the case of POMDPs, the analysis of the average reward case is known to be much more involved than the discounted reward criterion. A thorough account of these issues and solutions may be found in [Arapostathis *et al.*, 1993; Puterman, 2005]. Since a DEC-POMDP is a generalization of a POMDP, one would expect to face the same difficulties. However, we demonstrate a way to circumvent these difficulties by focusing on a subclass of the problem–the case of observation and transition independent decentralized MDPs [Becker *et al.*, 2004]. The results we obtain for this class of problems are encouraging and they lay the foundation for studying more complex models and competitive settings.

The main contribution of this paper is the formulation and analysis of the DEC$_2$-MDP problem with the *average reward* criterion. Section 2 defines the problem and provides a motivating example. In Section 3, we show that calculating the *gain* of a fixed policy is easy, but finding an optimal policy is NP complete. In Section 4, we propose two practical algorithms to solve the problem. In addition, these algorithms show a connection with centralized average reward MDPs and thus help to analyze the properties of the optimal policies. Both algorithms are based on mathematical programming formulations. Finally, in Section 5 we demonstrate the algorithms on a standard problem, in which the average-reward objective is the most natural one. Some of the results we provide are easily extensible to the competitive case. In the interest of clarity, we only mention these extensions in this paper.

2 Average Reward DEC-MDPs

This section provides a formal definition of the problem and shows that under certain conditions, the average reward may be expressed compactly, leading to a compact formulation of the problem. The model we propose is similar to the one proposed in [Becker *et al.*, 2004]. We introduce a different definition, however, because the original is not easily extensible to infinite-horizon problems. To simplify the discussion, the problem is defined for two agents. However, the definition as well as *some* of the results can be easily extended to an arbitrary number of decision makers.

Definition 2.1. A DEC$_2$-MDP is an n-tuple (S, A, P, R) such that:

- $A = A_1 \times A_2$ represents the actions of the players;
- $S = S_1 \times S_2$ is the set of states;
- $P = (P_1, P_2)$ is the transition probability. Each P_i is a function: $S_i \times S_i \times A_i \to \mathbb{R}$. The transition probabilities $(s_1, s_2) \to (s_1', s_2')$ given a joint action (a_1, a_2) is:
$P(s_1', s_2') \,|\, (s_1, s_2), (a_1, a_2) = P_1(s_1' \,|\, s_1, a_1) P_2(s_2' \,|\, s_2, a_2)$
- $R(s_1, a_1, s_2, a_2) \in \mathbb{R}$ is the joint reward function; and
- (s_1, s_2) is a given initial state for both agents.

The process, as we defined it, is both transition and observation independent as defined in [Becker *et al.*, 2004]. Transition independence means that the transition to the next state of one agent is independent of the states of all other agents. Observation independence means that the observations that an agent receives are independent of the current states and observations of other agents.

While transition independence is a plausible assumption in many domains, observation independence is somewhat restrictive. It prevents any kind of communication among the agents. This assumption is suitable when communication is prohibitively expensive, risky, or when it does not add much value to the process. When communication is useful, the problem with no observations provides as an easily-computable lower bound that could help to decide when to communicate.

We refer to (S_1, A_1, P_1) and (S_2, A_2, P_2) as the *local* processes. In fact, they represent MDPs with no reward. To simplify the analysis, we assume that the state set S and action set A are finite, and that the Markov chains for both agents, induced by any policy, are *aperiodic*. A Markov chain is aperiodic when it converges to its stationary distribution in the limit [Puterman, 2005]. We discuss how to relax these assumptions in Section 6.

A policy π is a function that maps the history of the *local* states into an action for each agent. To prevent the need for introducing measure-theoretic concepts, we assume that the policy may only depend on a finitely-bounded number of previous states. If the action selection depends only on the last state of both agents, the policy is *stationary*, or *Markov*.

We extend the standard terminology for the structure of average reward MDPs to the decentralized case. A DEC$_2$-MDP is said to be *recurrent* or *unichain* if the local decision processes of all the agents are recurrent or unichain respectively. Otherwise, it is said to be *multichain* [Puterman, 2005].

Given a fixed policy, the history of the process can be represented by a sequence of random variables $\{(X_t, Y_t)\}_{t=0}^{\infty}$. The random variables X_t and Y_t represent the state-action pairs of the two agents at stage t. The corresponding sequence of rewards is defined as $\{R(X_t, Y_t)\}_{t=0}^{\infty}$. We seek to maximize the expected average reward, or *gain*, defined as follows.

Definition 2.2. The *gain* of a Markov policy is

$$g(s_1, s_2) = \lim_{N\to\infty} \frac{1}{N} \mathbf{E}_{s_1, s_2, u_0} \left[\sum_{t=0}^{N-1} R(X_t, Y_t) \right].$$

The expectation subscript denotes the initial value of the random variables, thus $X_0 = (s_1, \pi(s_1))$ and $Y_0 = (s_2, \pi(s_2))$. Furthermore, the *gain* matrix G is defined as $G(s_1, s_2) = g(s_1, s_2)$. The actual gain depends on the agents' initial distributions α_1, α_2 and may be calculated as $g = \alpha_1^T G \alpha_2$.

One problem that fits the average reward criterion is the Multiple Access Broadcast Channel (MABC) [Rosberg, 1983; Ooi and Wornell, 1996]. In this problem, which has been used widely in recent studies of decentralized control, two communication devices share a single channel, and they need to periodically transmit some data. However, the channel can transmit only a single message at the same time. Thus, when several agents send messages at the same time interval, this leads to a collision, and the transmission fails. The memory of the devices is limited, thus they need to send the messages sooner rather than later. We adapt the model from [Rosberg, 1983], which is particularly suitable because it assumes no sharing of local information among the devices.

The MABC problem may be represented as a DEC$_2$-MDP where each device is represented by an agent. The state space of each agent represents the number of messages in the buffer. There are two possible actions, *send* and *do not send*. The arriving data is modeled by random transitions among the buffer states. We provide more details on the formulation and the optimal policy in Section 5.

3 Basic Properties

In this section, we derive some basic properties of DEC$_2$-MDPs, showing how to efficiently evaluate fixed policies and establishing the complexity of finding optimal policies. The analysis is restricted to stationary policies. An extension to non-stationary policies is mentioned in Section 6.

For average reward MDPs, the gain of a policy can be calculated directly by solving a set of linear equations, which resemble value function equations in the discounted case [Puterman, 2005]. These value optimality functions cannot be used in DEC$_2$-MDPs, but as the following theorem states, the gain can be expressed using the invariant or *limiting distribution*. The limiting distribution p represents the average probability of being in each state over the whole infinite execution.

It is defined as:

$$p(s_i) = \lim_{N \to \infty} \frac{1}{N} \sum_{t=0}^{N-1} \mathbf{P}\left[X_t = s_i\right].$$

The limiting distribution can be calculated from a the initial distribution using a *limiting matrix*. Limiting matrix of transition matrix P is defined as

$$P^* = \lim_{N \to \infty} \frac{1}{N} \sum_{t=0}^{N-1} P^t.$$

Then, we have that $p = \alpha P*$, where α is the initial distribution.

Theorem 3.1. *Suppose P_1 and P_2 are state-action transition matrices of the two agents for fixed policies, and R is the reward matrix. Then, the gain matrix can be calculated as follows:*

$$G = P_1^* R (P_2^T)^*.$$

The theorem follows immediately from the fact that both processes are aperiodic and from the definition of gain.

Calculating P^* is not practical, because it typically requires calculating all eigenvectors of the matrix. However, the same idea used for calculating the gain in the single agent case can be used here as follows.

Theorem 3.2. *For any initial distribution α and transition matrix P, the limiting distribution p fulfills:*

$$(I - P^T)p = 0 \tag{3.1}$$

$$p + (I - P^T)q = \alpha. \tag{3.2}$$

Moreover, if p fulfills (3.1) and (3.2) then:

$$p = (P^T)^* \alpha. \tag{3.3}$$

Note that I denotes the identity matrix of the appropriate size and that (3.2) also implies that $\sum_{i=1}^{n} p(i) = 1$. Due to space constraints, we do not include the proof, which is a similar derivation to the one in [Puterman, 2005].

Next, we examine the complexity of finding an optimal policy in this model. The complexity of finite-horizon DEC-MDPs has been shown to be NEXP complete, while infinite-horizon problems are undecidable [Bernstein *et al.*, 2000]. While the model we study here is not tractable, the following theorem shows that it is easier than the general problem. However, we focus only on stationary policies, while the truly optimal policy may depend on a longer history. The complexity further justifies the use of the algorithms we present later, since a polynomial algorithm is unlikely to exist.

Theorem 3.3. *The problem of finding an optimal* Markov *policy for the* aperiodic *DEC$_2$-MDP is NP-complete.*

Proof. To show that finding an optimal policy is in NP, we note that Theorem 4.2 states the optimal Markov policies are deterministic. Since calculating the gain, given fixed policies is possible in polynomial time, and the policy size is polynomial, all policies can be checked in nondeterministic polynomial time.

The NP hardness can be shown by reduction from satisfiability (SAT). The main idea is that the first agent may assign values to variable instances to satisfy the problem, and the second one can assign values to variables. The agents are penalized for assigning a different value to the actual variable and to the variable instance.

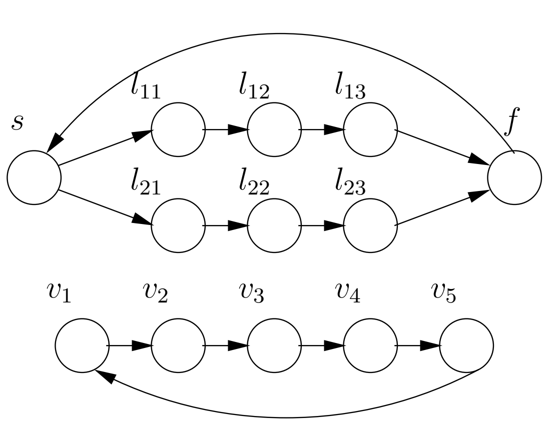

Figure 1: A sketch of the DEC$_2$-MDP created from a 3-SAT problem with 2 clauses and 5 variables.

Let $S = (V, C)$ be a 3-SAT problem in conjunctive normal form, where V is the set of variables and C is the set of clauses. If the literal l is an instantiation of variable v, then we write $l \sim v$. We construct a corresponding DEC$_2$-MDP in the following way. The state and action sets for the agents are:

$$\begin{aligned} S_1 &= \{s, f\} \cup \{l_{ij} \mid i \in C, j = 1, 2, 3\} \\ S_2 &= V \\ A_1 &= \{t, f\} \\ A_2 &= \{t, f\} \end{aligned}$$

Each state $l_{ij} \in S_1$ represents the j-th literal in the i-th clause, which is an instantiation of a single variable. States S_2 represent the variables of the problem. The actions represent whether the variable value is *true* or *false*. The main idea is to make the first agent assign arbitrary values to variable instances to satisfy the formula. The second agent then determines whether the value assigned to multiple variable instances is the same.

For lack of space, we omit the complete definition of transition matrices. A sketch of the reduction is shown in Figure 1. The first agent follows the transitions depicted when the action is chosen such that the literal is not *true*. Otherwise, the agent transitions to state s. For both agents, for each state, there is a small probability (< 1) of remaining the in same state. For the second agent, the transition probabilities are independent of the actions taken. Notice that this process is *aperiodic*.

The rewards are defined such that a negative reward is received when the value assigned by the first agent to an instance of a variable is different from the value assigned to the variable by the second agent. In addition, a negative reward is received when the first agent transitions to state f. Notice that this happens only when the assigned variable instance values do not satisfy at least one clause.

$$R((f, a_1), (s, a_2)) = -1 \quad \forall a_1 \in A_1, a_2 \in A_2, s \in S_2$$
$$R((l_{ij}, a_1), (s, a_2)) = -1 \quad l_{i,j} \sim v \text{ and } a_1 \neq a_2.$$

In all other cases the reward is zero.

It is easy to show that the gain of the above DEC$_2$-MDP is 0 if and only if the SAT problem is satisfiable. □

4 Optimal DEC$_2$-MDP Algorithms

In this section we present two methods for finding optimal policies for DEC$_2$-MDPs. Both methods are based on formulating the problem as a mathematical program. We also discuss additional approaches to solving the problem.

In previous work, a similar problem was solved using an iterative method that fixes the policy of one agent, while computing the optimal policy of the other agent [Nair *et al.*, 2003]. Such methods are unlikely to produce good results in the average-reward case, because of the high number of local minima. In contrast, the following MILP formulation facilitates an efficient search through the set of all local minima.

4.1 Quadratic Programming Solution

In this section, we give a mathematical program that defines the optimal solutions. The reward definition is based on Theorem 3.1. We also discuss how to obtain the optimal policy from the solution of the quadratic program.

$$\begin{aligned}
&\text{maximize} && F = p_1^T R p_2 \\
&\text{subject to} && p_1 \geq 0 \\
& && p_2 \geq 0 \\
&\forall j \in S_1 && \sum_{a \in A_1} p_1(j,a) - \sum_{s \in S_1} \sum_{a \in A_1} p_1(s,a) P_1(j \mid s,a) = 0 \\
&\forall j \in S_1 && \sum_{a \in A_1} p_1(j,a) + \sum_{a \in A_1} q_1(j,a) - \\
& && \sum_{s \in S_1} \sum_{a \in A_1} q_1(s,a) P_1(j \mid s,a) = \alpha_1(j) \\
&\forall j \in S_2 && \sum_{a \in A_2} p_2(j,a) - \sum_{s \in S_2} \sum_{a \in A_2} p_2(s,a) P_2(j \mid s,a) = 0 \\
&\forall j \in S_2 && \sum_{a \in A_2} p_2(j,a) + \sum_{a \in A_2} q_2(j,a) - \\
& && \sum_{s \in S_2} \sum_{a \in A_2} q_2(s,a) P_2(j \mid s,a) = \alpha_2(j) \qquad (4.1)
\end{aligned}$$

The benefit of this formulation is not so much the computational efficiency of the solution technique, but that it helps identify some key properties of optimal policies. Later, we use these properties to develop a more efficient formulation, which is also shown to be optimal.

Theorem 4.1. *An optimal solution of (4.1) has the optimal gain.*

We omit the proof due to lack of space. The correctness of the constraints follows from the dual formulation of optimal average reward, shown for example in [Puterman, 2005], and from Theorem 3.2. The choice of the objective function follows from Theorem 3.1.

The solution of (4.1) may also be used to determine an optimal policy, given an optimal value of the program's variables. The procedure is the same as determining an optimal policy from a solution to the dual of an MDP. That is, the optimal policy is randomized such that for each *recurrent* state:

$$\mathbf{P}\left[\pi_i^*(s) = a'\right] = \frac{p_i(s,a')}{\sum_{a \in A} p_i(s,a)}, \quad i = 1, 2.$$

This is well-defined because the recurrent states are those for which $\sum_{a \in A} p_i(s,a) > 0$. For *transient* states the policy can be determined as

$$\mathbf{P}\left[\pi_i^*(s) = a'\right] = \frac{q_i(s,a')}{\sum_{a \in A} q_i(s,a)}, \quad i = 1, 2.$$

Proposition 4.1. *The policy constructed according to the above description is optimal.*

The proof is only a small variation of the proof of optimal policy construction for average reward MDPs. See, for example, [Puterman, 2005].

It has also been shown that for any average-reward MDP there exists an optimal deterministic policy. Notice, that when p_1 is fixed, the formulation for p_2 is equivalent to that of the dual of average-reward linear program. Thus, for any optimal p_1^* and p_2^*, we can construct a deterministic policy as follows. Fixing p_1^*, we can find $\tilde{p_2}$ with the same gain, by solving the problem as an MDP. Following the same procedure, we can obtain also a deterministic $\tilde{p_1}$. Hence the following theorem.

Theorem 4.2. *There is always a deterministic* Markov *policy for DEC$_2$-MDP with optimal gain.*

4.2 Mixed Integer Linear Program

This formulation offers a method that can be used to calculate the optimal policies quite efficiently. The approach generalizes the approach of [Sandholm *et al.*, 2005], which is based on game-theoretic principles and works only for matrix games. Our approach is based on Lagrange multipliers analysis of the problem and thus may be extended to include any additional linear constraints.

For clarity, we derive the algorithm only for the *unichain* DEC$_2$-MDP. The algorithm for the multi-chain case is similar and is derived in the same way. To streamline the presentation, we reformulate the problem in terms of matrices as follows:

$$\begin{aligned}
&\text{maximize} && F = p_1^T R p_2 \\
&\text{subject to} && T_1 p_1 = 0 \qquad T_2 p_2 = 0 \\
& && e_1^T p_1 = 1 \qquad e_2^T p_2 = 1 \\
& && p_1 \geq 0 \qquad p_2 \geq 0
\end{aligned} \qquad (4.2)$$

where e_1 and e_2 are vectors of ones of length n_1 and n_2 respectively, and T_1 and T_2 are matrix representations of the 3rd and the 5th constraint in (4.1).

The Mixed Integer Linear Program (MILP) formulation is based on application of the Karush-Kuhn-Tucker (KKT) theorem [Bertsekas, 2003]. Thus, the Lagrangian for the formulation (4.2) is

$$\begin{aligned}
&L(p_1, p_2, \lambda, \rho, \mu) = \\
&= \quad p_1^T R p_2 + \lambda_1(e_1^T p_1 - 1) + \lambda_2(e_2^T p_2 - 1) + \\
&+ \quad \rho_1^T T_1 p_1 + \rho_2^T T_2 p_2 + \mu_1^T p_1 + \mu_2^T p_2.
\end{aligned}$$

Let A_1 and A_2 denote the sets of active constraints for inequalities $p_1 \geq 0$ and $p_2 \geq 0$ respectively. Following from the KKT theorem, every local extremum must fulfill the following constraints:

$$\begin{array}{ll} Rp_2 = -\lambda_1 e_1 - T_1^T \rho_1 - \mu_1 & R^T p_1 = -\lambda_2 e_2 - T_2^T \rho_2 - \mu_2 \\ p_1(i) = 0 \quad \forall i \in A_1 & p_2(i) = 0 \quad \forall i \in A_2 \\ \mu_1(i) = 0 \quad \forall i \notin A_1 & \mu_2(i) = 0 \quad \forall i \notin A_2 \\ \mu_1(i) \geq 0 \quad \forall i \in A_1 & \mu_2(i) \geq 0 \quad \forall i \in A_2 \end{array}$$

The gain in a local extremum simplifies to:

$$p_1^T R p_2 = -\lambda_1 p_1^T e_1 - p_1^T T_1^T \rho_1 - p_1^T \mu_1 = -\lambda_1,$$

because $P_1^T T_1^T = 0$ and for all i, we have $p_1(i) = 0 \vee \mu_i = 0$. Though the above constraints are linear, the problem cannot be directly solved since A_1 and A_2 are not known. They can however be represented by binary variables $b_1(i), b_2(i) \in \{0,1\}$, which determine for each constraint whether it is active or not. Therefore, we can get the following MILP:

$$\begin{array}{lll} \text{maximize} & -\lambda_1 & \\ \text{subject to} & T_1 p_1 = 0 & T_2 p_2 = 0 \\ & e_1^T p_1 = 1 & e_2^T p_2 = 1 \\ & Rp_2 = -\lambda_1 e_1 - T_1^T \rho_1 - \mu_1 & \\ & R^T p_1 = -\lambda_2 e_2 - T_2^T \rho_2 - \mu_2 & \\ & \mu_1 \geq 0 & \mu_2 \geq 0 \\ & \mu_1 \leq m_1 b_1 & \mu_2 \leq m_2 b_2 \\ & p_1 \geq 0 & p_2 \geq 0 \\ & p_1 \leq e_1 - b_1 & p_2 \leq e_2 - b_2 \\ & b_1(i) \in \{0,1\} & b_2(i) \in \{0,1\} \end{array} \tag{4.3}$$

In this program, m_1 and m_2 are sufficiently large constants. It can be show that they may be bounded by the maximal *bias* [Puterman, 2005] of any two policies. However, we are not aware of a simple method to calculate these bounds. In practice, it is sufficient to choose a number that is close the largest number representable by the computer. This only eliminates solutions that are not representable using the limited precision of the computer.

The optimal policy can be in this case obtained in the same way as in Subsection 4.1. Notice that $\lambda_1 = \lambda_2$, so unlike a competitive situation, one of them may be eliminated. The preceding analysis leads to the following proposition.

Proposition 4.2. *The gain and optimal policy obtained from (4.2), and (4.3) are equivalent.*

Though we present this formulation only for cooperative cases, it may be easily extended to the competitive variant of the problem.

5 Application

This section presents an application of our average-reward decentralized MDP framework. It is a simple instance of the practical application introduced in Section 2. The experimental component of the paper is designed to illustrate the scope of problems that can be solved quickly using our new algorithm.

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8
a_1	N	N	N	S	S	S	S	S
a_2	N	N	S	S	S			

Figure 2: Sample policy for agents a_1 and a_2. In each state, action S represents sending and N, not sending. The states are $s_1 - s_8$.

As mentioned earlier, the problem we use is an adaptation of the Multiple Access Broadcast Channel problem solved in [Rosberg, 1983]. In order to make the problem more interesting, we assume that the following events may occur: a single or two messages arrive, a single message in the buffer expires, or nothing happens. These events occur independently with some fixed probabilities: $(\alpha, \beta, \gamma, \delta)$.

The communication devices may have buffers of fixed but arbitrary length, unlike the original problem. In addition, the possible actions are *send* and *not send*. For the action *not send*, the probabilities are updated according to the list stated above. For the action *send*, the buffer level is decreased by 1, and then the transition probabilities are the same as for the action *not send*. Though [Rosberg, 1983] derives a simple analytical solution of the problem, it is not applicable to our extended problem.

The reward associated with overflowing either of the buffers of the agents are r_1 and r_2 respectively. If both agents decide to broadcast the message at the same time, a collision occurs and the corresponding reward is r. The above rewards are negative to model the cost of losing messages. Otherwise, the reward is zero.

We ran the experiments with the following message arrival probabilities $(0.1, 0.3, 0.5, 0.1)$. An optimal policy for the problem with rewards $r_1 = -1, r_2 = -1.2, r = -4.5$ and buffer sizes of 8 and 5 is depicted in Figure 2. While the optimal policy has a simple structure in this case, the algorithm does not rely on that. Note that choosing different thresholds for sending messages leads to significantly lower gains.

To asses the computational complexity of the approach and its practicality, we ran experiments using a basic commercially available MILP solver running on a PC. The solver was a standard branch-and-bound algorithm, using linear programming bounds. The linear programs to calculate the bound were solved by the simplex algorithm. For all our runs, we fixed the transition probabilities and experimented with two sets of rewards. The first was $r_1 = -1, r_2 = -1.2, r = -4.5$ and the second was $r_1 = -1, r_2 = -2, r = -1.5$.

The results are summarized in Figure 3. They show that the time required to calculate an optimal solution is below 10 seconds for problems of this size. It is clear from the results that the different reward sets have significant impact on how quickly the problem is solved. This is quite common among branch-and-bound algorithms. For larger problems, it may be beneficial to use one of the branch-and-cut algorithms, which often offer good performance even on large problems.

n_1	n_2	rs	g	t (sec.)
5	5	1	-0.76513	2.4936
5	5	2	-0.31529	0.8312
10	5	1	-0.76465	7.7712
10	5	2	-0.28743	2.5236
10	10	1	-0.72156	8.4922
10	10	2	-0.24128	4.2261
15	5	1	-1e-10	7.9915
15	5	2	0	5.1274

Figure 3: Performance results: n_1 and n_2 represent the number of states of each agent, rs is the reward set, g is the gain, and t is runtime in seconds.

6 Conclusion

We introduced an average-reward model for observation and transition-independent decentralized MDPs. Analysis of the model shows that finding optimal joint policies is easier compared with the general DEC-POMDP framework, but it remains intractable. Using standard mathematical programming, we designed algorithms that can solve problem instances of substantial size for decentralized problems. While we expect performance to vary depending on the actual structure of the problem instance, the efficiency of the techniques is encouraging.

The definitions and policy analysis that we presented generalize in a straightforward manner to problems with an arbitrary number of decision makers. The quadratic program formulation may be generalized as well, but the objective function becomes a high-degree polynomial. This, unfortunately, makes the analysis that leads to the MILP formulation inapplicable.

One issue that we have not addressed here is the use of non-Markov policies. The algorithms we have presented can be extended to find the optimal policies of problems with dependency on a longer history than a single state with increased complexity. However, the question whether the optimal policy for any problem is finite or may be infinitely long remains open.

In future work, we plan to relax some of the limiting assumptions, most importantly, allowing observations in the model. This extension is not trivial, since such observations break the independence among the state-action visitation probabilities.

This work lays the foundation for expanding models of decentralized decision making to address the practical average-reward objective. Moreover, the MILP formulation we introduce could be adapted to DEC-MDPs with other optimality criteria. This opens up several new research avenues that may help tackle more complex domains and competitive settings.

Acknowledgments

This work was supported in part by the Air Force Office of Scientific Research under Award No. FA9550-05-1-0254 and by the National Science Foundation under Grants No. IIS-0328601 and IIS-0535061.

References

[Arapostathis *et al.*, 1993] A. Arapostathis, V. S. Borkar, E. Fernandez-Gaucherand, M. K. Ghosh, and S. Markus. Discrete time controlled Markov processes with average cost criterion - a survey. *SIAM Journal of Control and Optimization*, 31:282–344, 1993.

[Becker *et al.*, 2004] Raphen Becker, Shlomo Zilberstein, Victor Lesser, and Claudia V. Goldman. Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research*, 22:423–455, 2004.

[Bernstein *et al.*, 2000] Daniel S. Bernstein, Shlomo Zilberstein, and Neil Immerman. The complexity of decentralized control of Markov decision processes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 32–37, 2000.

[Bertsekas, 2003] Dimitri P. Bertsekas. *Nonlinear programming*. Athena Scientific, 2003.

[Filar and Vrieze, 1996] Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes*. Springer, 1996.

[Hajek, 1984] Bruce Hajek. Optimal control of two interacting service stations. In *IEEE Transactions on Automatic Control*, volume 29, pages 491–499, 1984.

[Madani, 2000] Omid Madani. *Complexity results for infinite-horizon Markov decision processes*. PhD thesis, University of Washington, 2000.

[Mahadevan, 1996] Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22(1-3):159–195, 1996.

[Nair *et al.*, 2003] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proceedings of the International Joint Conference on Artificial Inteligence*, pages 705–711, 2003.

[Ooi and Wornell, 1996] J. M. Ooi and G. W. Wornell. Decentralized control of a multiple access broadcast channel: performance bounds. In *Proceeding of the IEEE Conference on Decision and Control*, volume 1, pages 293–298, 1996.

[Puterman, 2005] Martin L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 2005.

[Rosberg, 1983] Z. Rosberg. Optimal decentralized control in a multiaccess channel with partial information. *IEEE Transactions on Automatic Control*, 28:187–193, 1983.

[Sandholm *et al.*, 2005] Tuomas Sandholm, Andrew Gilpin, and Vincent Conitzer. Mixed-integer programming methods for finding Nash equilibria. In *Proceedings of the National Conference on Aritifical Intelligence*, pages 495–501, 2005.

[Tangamchit *et al.*, 2002] Poj Tangamchit, John M. Dolan, and Pradeep K. Koshla. The necessity of average rewards in cooperative multirobot learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1296–1301, 2002.