The Complexity of Decision-Theoretic Planning for Distributed Agents

Daniel S. Bernstein and Shlomo Zilberstein
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003
{bern, shlomo}@cs.umass.edu

Abstract

Planning for distributed agents with partial state information is considered from a decision-theoretic perspective. We describe *decentralized Markov decision processes (DEC-MDPs)* and *decentralized partially observable Markov decision processes (DEC-POMDPs)*, which are generalizations of MDPs and POMDPs, respectively, in which the process is controlled by multiple distributed agents. The finite-horizon version of a DEC-POMDP with at least two agents is shown to be NEXP-complete. In addition, the finite-horizon DEC-MDP with at least three agents is shown to be NEXP-complete. These complexity results illustrate a fundamental difference between centralized and decentralized control of a Markov process. We briefly discuss the connection between the finite-horizon case and the infinite-horizon case with "synchronization" states, and we suggest a way of reducing this problem to a type of centralized planning problem.

Introduction

Among researchers in artificial intelligence, there has been growing interest in problems with multiple distributed agents working to achieve a common goal (Lesser 1998; desJardins *et al.* 1999). Sometimes, the agents must go for long periods of time without communicating. One problem of this type is that of having multiple spacecraft complete a mission together (Estlin *et al.* 1999). Another problem of this nature, which has been studied by control theorists, is that of maximizing the throughput of a multiple access broadcast channel (Ooi & Wornell 1996). Solutions to these types of problems consist of sets of local plans, one for each agent.

Although several researchers in distributed AI have discussed these decentralized planning problems, a decision-theoretic formalism for them seems to be lacking. By relating these problems to the abundance of work on Markov decision processes, we may be able to gain more insight into them. The insight gained can be used to guide the search for useful algorithms. There has been some related work in AI. For example, Boutilier (1999) talks about multi-agent Markov decision processes (MMDPs). However, in these problems, the agents all have access to the same information; whereas in our model we do not make this assumption. Models very similar to ours do exist in the control theory literature (Ooi *et al.* 1997; Aicardi, Franco, & Minciardi 1987). However, to our knowledge, the computational complexity of the problems involving these models has not been studied. The most closely related piece of work is that of Tsitsiklis and Athans (1985), in which the complexity of non-sequential decentralized decision problems was studied.

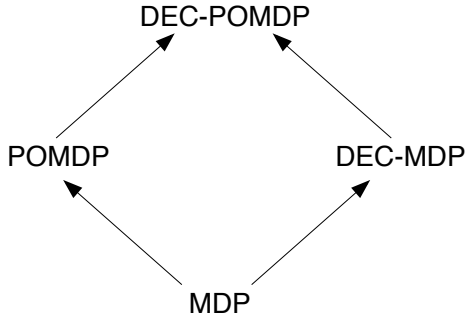

Figure 1: The relationships between the problems.

The Markov decision process (MDP) framework will be used as a basis for our model of planning for distributed agents. Recall that a partially observable Markov decision process (POMDP) is an MDP along with a distribution of observations for each state and action. We will extend the POMDP model to allow for multiple distributed agents to each receive a local observation and choose an action. The state transitions and expected rewards will now depend on the actions of all of the agents. We call this a decentralized partially observable Markov decision process (DEC-POMDP). An interesting special case of a DEC-POMDP satisfies the assumption that at any time step the state can be completely determined just from the current set of observations of the agents. This is denoted a decentralized Markov decision process (DEC-MDP). The obvious relationships between these four problems are shown in Figure 1.

We will discuss the computational complexity of find-

ing optimal policies for the finite-horizon versions of the problems. It is known that solving an MDP is P-complete and that solving a POMDP is PSPACE-complete. We will show that solving a DEC-POMDP with a constant number, $m \geq 2$, of agents is complete for the complexity class nondeterministic exponential time (NEXP). Furthermore, solving a DEC-MDP with a constant number, $m \geq 3$, of agents is NEXP-complete. One consequence of this is that any algorithm for solving either problem exactly will most likely take doubly exponential time in the worst case. These results shed some light on the fundamental differences between centralized and decentralized control of a Markov process.

It should be noted that the case of a DEC-MDP with exactly two agents has so far eluded a precise categorization. The extent of our current knowledge is that the problem is PSPACE-hard and is in NEXP.

Centralized problems

A *Markov decision process (MDP)* models an agent acting in a stochastic environment to maximize its long-term reward. The type of MDP that we will consider contains a finite set S of states, with $s_0 \in S$ as the start state. For each state $s \in S$, A_s is a finite set of actions available to the agent. P is the table of transition probabilities, where $P(s'|s,a)$ is the probability of a transition to state s' given that the agent performed action a in state s. R is the reward function, where $R(s,a)$ is the expected reward received by the agent given that it chose action a in state s.

There are several different ways to define "long-term reward" and thus several different measures of optimality. In this paper, we will focus on *finite-horizon* optimality, for which the aim is to maximize the expected sum of rewards received over T time steps, where T is the horizon. Formally, the agent should maximize

$$E\left[\sum_{t=0}^{T} r(s_t, a_t)\right].$$

A *policy* δ for a finite-horizon MDP is a mapping from each state s and time t to an action $\delta(s,t)$. This is called a *nonstationary* policy. The decision problem corresponding to a finite-horizon MDP is as follows: Given an MDP M, a horizon T, and an integer K, is there a policy that yields total reward at least K?

An MDP can be generalized so that the agent does not necessarily observe the complete state of the environment at each time step. This is called a *partially observable Markov decision process (POMDP)*. The set of states S, start state $s_0 \in S$, table of transition probabilities P, and reward function R are the same as in an MDP. Additionally, a POMDP contains a finite set Ω of observations, and a table O of observation probabilities, where $O(o|a,s')$ is the probability that o is observed, given that action a was taken and led to state s'. There is a finite action set A_o for each observation $o \in \Omega$.

A policy δ is now a mapping from histories of observations $o_1, \ldots, o_t$ to actions in A_{o_t}. The decision problem for a POMDP is stated in exactly the same way as for an MDP.

Decentralized problems

A *decentralized partially observable Markov decision process (DEC-POMDP)* is a generalization of a POMDP to allow for distributed control by m agents that may not be able to observe the entire state. A DEC-POMDP contains a finite set S of states, with $s_0 \in S$ as the start state. The transition probabilities $P(s'|s,a^1,\ldots,a^m)$ and expected rewards $R(s,a^1,\ldots,a^m)$ now depend on the actions of all agents. Ω^i is a finite set of observations for agent i, and O^i is the agent's table of observation probabilities, where $O^i(o^i|a^1,\ldots,a^m,s')$ is the probability that o^i is observed, given that the action tuple $\langle a^1,\ldots,a^m\rangle$ was taken and led to state s'. Each agent i has a set of actions A^i_o for every observation $o^i \in \Omega^i$. Notice that if a centralized agent were able to view all of the observations at each time step and choose all of the actions, this problem would reduce to a POMDP.

For each $a^1,\ldots,a^m,s'$ and agent i, let $\omega^i(a^1,\ldots,a^m,s')$ denote the set of observations that have a nonzero chance of occurring given that the action tuple $\langle a^1,\ldots,a^m\rangle$ was taken and led to state s'. To form a *decentralized Markov decision process (DEC-MDP)*, we add the requirement that for each $a^1,\ldots,a^m,s'$, and each $o^1 \in \omega^1(a^1,\ldots,a^m,s')$, ..., $o^m \in \omega^m(a^1,\ldots,a^m,s')$, the state can be determined from the tuple $\langle o^1,\ldots,o^m\rangle$. Thus, if a centralized agent were able to view all of the observations at each time step and choose all of the actions, this problem would reduce to an MDP.

We define a *local policy*, δ^i, to be a mapping from a *local history* $o^i_1,\ldots,o^i_t$ to an action $a^i \in A^i_{o_t}$. A *joint policy*, $\delta = \langle\delta^1,\ldots,\delta^m\rangle$ is defined to be a tuple of local policies. We wish to find a joint policy that maximizes the total expected return over the finite horizon. The corresponding decision problem is stated in the same way as for an MDP or a POMDP.

Complexity results

To simplify our proofs, we will assume that the observations $O^1(s'),\ldots,O^m(s')$ are deterministic functions of the current state. It is possible to convert the general case to this one with only a polynomial increase in the size of the state set. The states in the new problem are $\langle s', o^1,\ldots,o^m\rangle$ tuples, and the transition probabilities and rewards are modified appropriately.

It will be necessary to consider only problems for which $T < |S|$. If we place no restrictions on T, then the upper bounds don't necessarily hold. With this restriction, it was shown in (Papadimitriou & Tsitsiklis 1987) that the decision problem for an MDP is P-complete. In the same paper, the authors showed that the decision problem for a POMDP is PSPACE-complete, and thus

probably does not admit a polynomial-time algorithm. We will prove that both the decision problem for a two agent DEC-POMDP and for a three agent DEC-MDP are NEXP-complete, where NEXP = NTIME(2^{n^c}) (Papadimitriou 1994). Since P $\neq$ NEXP, we can be certain that there does not exist a polynomial-time algorithm for either problem. Moreover, there probably isn't even an *exponential*-time algorithm that solves either problem.

In order to prove our completeness results, we will need a type of logical formula that characterizes NEXP just as Quantified Boolean Formula (QBF) characterizes PSPACE. Peterson and Reif (1979) proved that determining the truth of a formula called a *Dependency Quantifier Boolean Formula (DQBF)* is NEXP-complete. To understand a DQBF, first consider the following QBF formula:

$$\forall X_1\ \exists Y_1\ \forall X_2\ \exists Y_2\ F(X_1, X_2, Y_1, Y_2),$$

where each of X_1, X_2, Y_1, Y_2 is a tuple of Boolean variables, and $F(X_1, X_2, Y_1, Y_2)$ is a function over all the variables.

The aforementioned formula can be rewritten to show dependencies:

$$\forall X_1\ \forall X_2\ \exists Y_1(X_1)\ \exists Y_2(X_1, X_2)\ F(X_1, X_2, Y_1, Y_2).$$

A formula of this form is a DQBF. A DQBF consists of universal variables, existential variables with dependencies, and a Boolean function. The following is an example of a DQBF that probably cannot be written succinctly as a QBF:

$$\forall X_1\ \forall X_2\ \forall X_3\ \exists Y_1(X_1, X_2)\ \exists Y_2(X_2, X_3)\ \exists Y_3(X_1, X_3)\ F(X_1, X_2, X_3, Y_1, Y_2, Y_3).$$

For our reduction, we will use a restricted version of a DQBF that we call an RDBQF. An RDQBF has the form:

$$\forall X_1\ \forall X_2\ \exists Y_1(X_1)\ \exists Y_2(X_2)\ F(X_1, X_2, Y_1, Y_2).$$

There are exactly two universal tuples and two existential tuples. We require that each existential tuple depends on a different universal tuple. Moreover, the number of variables in X_1 and X_2 must be the same, and the number of variables in Y_1 and Y_2 must be the same.

Lemma 1 *Determining the truth of an RDQBF is NEXP-hard.*

Proof. To prove NEXP-hardness we can use the exact same proof that Peterson and Reif used to prove the NEXP-hardness of the DQBF problem. This is because in the proof, the formula to which they reduce an arbitrary nondeterministic exponential-time Turing machine computation is actually just an RDQBF. We refer the reader to their paper for the details of the reduction. □

Theorem 1 *The decision problem for a finite-horizon DEC-POMDP with $m \geq 2$ agents is NEXP-complete.*

Proof. First, we will show that the problem is in NEXP. We can guess a joint policy δ and write it down in exponential time. This is because a joint policy consists of a finite set of mappings from local histories to actions, and since $T < |S|$, all histories have length less than $|S|$. A DEC-POMDP together with a joint policy is the same as a POMDP together with a policy. This can be converted to an exponentially large belief-state MDP with a policy and evaluated, all in exponential time. From this evaluation, we can find out whether the policy yields expected reward at least K.

Now we will show that the problem is NEXP-hard. For simplicity, we consider only the two agent case. Clearly, the problem with more agents can be no easier. This proof is similar to Papadimitriou and Tsitsiklis' proof of the PSPACE-hardness of a finite-horizon POMDP. We are given an arbitrary RDQBF

$$\forall X_1\ \forall X_2\ \exists Y_1(X_1)\ \exists Y_2(X_2)\ F(X_1, X_2, Y_1, Y_2),$$

where $X_1 = x_1^1, \ldots, x_p^1$, $X_2 = x_1^2, \ldots, x_p^2$, $Y_1 = y_1^1, \ldots, y_q^1$, and $Y_2 = y_1^2, \ldots, y_q^2$. Suppose the formula is in conjunctive normal form (w.l.o.g.) and has d clauses $C_1, \ldots, C_d$. We construct a corresponding DEC-POMDP such that $T < |S|$ and an optimal policy yields a reward of at least zero if and only if the RDQBF is true.

Before presenting the formal details, it will be instructive to expose the intuition behind the construction of the DEC-POMDP. The universal quantifiers in the formula are represented by the stochastic "environment" in the sequential decision problem. Each of the two existential quantifiers corresponds to a different agent choosing actions. The dependencies in the RDQBF are encoded in the observation functions of the agents. Each agent can only "see" the its own universal and existential variables and their values. A legal way of assigning values to variables in the RDQBF has an associated joint policy in the DEC-POMDP.

The process can be thought of as having four "phases", one for each quantifier. Before the start of the first phase, the environment chooses a clause, each with equal probability. This clause remains fixed for the remainder of the process. During the first phase, agent one's universal variables are randomly assigned values by the environment, with only agent one observing the assignments. In the second phase, agent two's variables are randomly assigned values by the environment, with only agent two observing the assignments. The third phase corresponds to agent one assigning values to its existential variables, with agent two observing nothing. In phase four, this process is repeated for agent two. There is a component of the state that denotes whether it is a "satisfied" or "unsatisfied" state. The component is initially set to unsatisfied. If, in the course of the process, a variable is assigned a value such that it causes the current clause to be true, the component is switched to satisfied, and the process remains in satisfied states until the end. If there is a joint policy that guarantees finishing in a satisfied final state regardless

of which clause and which universal values were chosen, then the formula is indeed true. Conversely, a true formula is a guarantee that the corresponding joint policy always causes the process to end in a satisfied state.

Now the construction is expressed more formally. We first describe the state set, S. In order to avoid a mess of subscripts and superscripts, we write the states as tuples. In addition to the start state s_0, S contains six states $\langle X_?, T, i, j, k\rangle$, $\langle X_?, U, i, j, k\rangle$, $\langle X_T, T, i, j, k\rangle$, $\langle X_T, U, i, j, k\rangle$, $\langle X_F, T, i, j, k\rangle$, and $\langle X_F, U, i, j, k\rangle$ for each agent i, each universal variable x_j^i, and each clause C_k. The first component denotes whether the universal variable is about to be assigned a value ($x_?$); has just been assigned true (x_T); or has just been assigned false (x_F). The second component says whether the process is in a satisfied (T) state or an unsatisfied (U) state. The third and fourth components determine the particular variable that is currently being considered, and the last component is the clause.

In addition, S contains six states $\langle Y_?, T, i, j, k\rangle$, $\langle Y_?, U, i, j, k\rangle$, $\langle Y_T, T, i, j, k\rangle$, $\langle Y_T, U, i, j, k\rangle$, $\langle Y_F, T, i, j, k\rangle$, $\langle Y_F, U, i, j, k\rangle$ for each agent i, each existential variable y_j^i, and each clause C_k. The components are the same as described above. Lastly, there are two end states $\langle T, k\rangle$ and $\langle U, k\rangle$ for each clause C_k. The first component says whether this end state is a satisfied state or an unsatisfied state, and the second component is the clause.

Next we describe the observation sets and observation functions. For each agent i and universal variable x_j^i, Ω^i contains $\langle X_?, i, j\rangle$, $\langle X_T, i, j\rangle$, and $\langle X_F, i, j\rangle$. When the state is $\langle X_?, T, i, j, k\rangle$, $\langle X_?, U, i, j, k\rangle$, $\langle X_T, T, i, j, k\rangle$, $\langle X_T, U, i, j, k\rangle$, $\langle X_F, T, i, j, k\rangle$, or $\langle X_F, U, i, j, k\rangle$, agent i observes $\langle X_?, i, j\rangle$, $\langle X_?, i, j\rangle$, $\langle X_T, i, j\rangle$, $\langle X_T, i, j\rangle$, $\langle X_F, i, j\rangle$, $\langle X_F, i, j\rangle$ respectively and the other agent observes D (a dummy observation). In addition, for each agent i and existential variable y_j^i, Ω^i contains $\langle Y_?, i, j\rangle$, $\langle Y_T, i, j\rangle$, and $\langle Y_F, i, j\rangle$. When the state is $\langle Y_?, T, i, j, k\rangle$, $\langle Y_?, U, i, j, k\rangle$, $\langle Y_T, T, i, j, k\rangle$, $\langle Y_T, U, i, j, k\rangle$, $\langle Y_F, T, i, j, k\rangle$, or $\langle Y_F, U, i, j, k\rangle$, agent i observes $\langle Y_?, i, j\rangle$, $\langle Y_?, i, j\rangle$, $\langle Y_T, i, j\rangle$, $\langle Y_T, i, j\rangle$, $\langle Y_F, i, j\rangle$, $\langle Y_F, i, j\rangle$ respectively, and the other agent observes D. For each of the $\langle T, k\rangle$ and $\langle U, k\rangle$, both agents observe D. Thus, each agent "sees" only its own variables and their assignments, and neither agent ever "sees" the clause k or whether the process is in a satisfied or unsatisfied state.

Now we describe the rewards, actions, and transition probabilities. All actions yield zero reward, except for those out of the states $\langle U, k\rangle$, which yield -1. At s_0, each agent has only one action, and the state transitions to each of $\langle X_?, U, 1, 1, k\rangle$, $1 \le k \le d$ with equal probability. This corresponds to the environment choosing a clause and the first phase beginning. There will be two steps for each variable considered. First, the variable is assigned a value by the environment. More precisely, for each state $\langle X_?, U, 1, j, k\rangle$, $1 \le j < p$, each agent only has one action, and the state transitions with equal probability to $\langle X_T, U, 1, j, k\rangle$ and $\langle X_F, U, 1, j, k\rangle$; and for each state $\langle X_?, T, 1, j, k\rangle$, $1 \le j < p$, each agent only has one action, and the state transitions with equal probability to $\langle X_T, T, 1, j, k\rangle$ and $\langle X_F, T, 1, j, k\rangle$. In the second step, it is determined whether the next state will be a satisfied state or an unsatisfied state. Formally, from the $\langle X_T, T, 1, j, k\rangle$, $\langle X_F, T, 1, j, k\rangle$, $\langle X_T, U, 1, j, k\rangle$, $\langle X_F, U, 1, j, k\rangle$, each agent only has one action, and the state transitions deterministically to $\langle X_?, T, 1, j+1, k\rangle$, $\langle X_?, T, 1, j+1, k\rangle$, $\langle X_?, U, 1, j+1, k\rangle$, $\langle X_?, U, 1, j+1, k\rangle$ respectively, with the following two exceptions: If x_j^1 appears positively in C_k, then $\langle X_T, U, 1, j, k\rangle$ transitions to $\langle X_?, T, 1, j+1, k\rangle$ instead of $\langle X_?, U, 1, j+1, k\rangle$; and if x_j^1 appears negatively, then $\langle X_F, U, 1, j, k\rangle$ transitions to $\langle X_?, T, 1, j+1, k\rangle$ instead of $\langle X_?, U, 1, j+1, k\rangle$.

From the states $\langle X_?, T, 1, p, k\rangle$ and $\langle X_?, U, 1, p, k\rangle$, the two steps are essentially the same as above, but now the resulting states after the two steps are $\langle X_?, T, 2, 1, k\rangle$ or $\langle X_?, U, 2, 1, k\rangle$, and the second phase begins. This whole process repeats for the "agent two" states until $\langle X_?, T, 2, p, k\rangle$ or $\langle X_?, U, 2, p, k\rangle$ is reached. From these states, the two steps are again the same as above, but the resulting states are $\langle Y_?, T, 1, 1, k\rangle$ or $\langle Y_?, U, 1, 1, k\rangle$, and the third phase begins.

The third and fourth phases of the process are similar to the first and second, with two steps for each variable, except that now the agents are assigning values to variables rather than the environment assigning values. Out of states of the form $\langle Y_?, T, i, j, k\rangle$, instead of a random transition, agent i has two actions (the other agent has only one) which lead to either $\langle Y_T, T, i, j, k\rangle$ or $\langle Y_F, T, i, j, k\rangle$ with certainty; similarly, out of states of the form $\langle Y_?, U, i, j, k\rangle$, agent i has two actions (the other agent has only one) which lead to either $\langle Y_T, U, i, j, k\rangle$ or $\langle Y_F, U, i, j, k\rangle$ with certainty. Aside from this change, these phases are exactly the same as the first two.

At the end of the four phases, all variables have been considered. The transition out of the last step of phase four takes the process into either $\langle T, k\rangle$ or $\langle U, k\rangle$. Out of the states $\langle T, k\rangle$ and $\langle U, k\rangle$, each agent has only one action, and the transition is to some new state. Recall that the transition from $\langle U, k\rangle$ yields a reward of -1 and is the only transition that results in a nonzero reward. To complete the construction, the horizon, T, is set to $4p + 4q + 1$ (exactly the number of steps it takes the process to reach one of $\langle T, k\rangle$ or $\langle U, k\rangle$).

Now we will show that there exists a joint policy with expected reward zero if and only if the formula is true. Suppose such a policy exists. Notice that the environment chooses a k at the beginning of the process, and the clause component of the state remains k until the end. Thus, the joint policy must have expected reward zero for each choice of k. Hence the joint policy must guarantee that the process ends up in $\langle T, k\rangle$ for all choices of k. Note that this requires that the joint policy cause the process to end in $\langle T, k\rangle$ regardless of what branches are taken out of the $\langle x_?, T, i, j, k\rangle$

and $\langle x_?, U, i, j, k\rangle$ states (i.e., regardless of which values are assigned to the universal variables). Otherwise, the expected reward will be less than zero.

Consider what ending in $\langle T, k\rangle$ entails. It means that at some point in the process, a switch was made from unsatisfied states to satisfied states. In other words, at least one of the following had to happen: For some variable x^i_j in clause C_k, the variable was set by the environment to true and appeared positively in the clause, or it was set to false and appeared negatively in the clause; or, for some variable y^i_j, the joint policy, using only information about $x^i_1, \ldots, x^i_p, y^i_1, \ldots, y^i_{j-1}$ (recall the way we defined the observation function), caused y^i_j to be set to true and it appeared positively in the clause, or caused it to be set to false and it appeared negatively in the clause.

Putting it all together, we know that the joint policy causes the process to end in $\langle T, k\rangle$ for any choice of clause and any assignment of values to the universal variables. Therefore, for any clause and any assignment of values to the universal variables, the joint policy, using only information specified by the dependencies of the formula, can assign values to the existential variables so that the clause evaluates to true. Thus, the joint policy gives a legal assignment that causes the entire formula to evaluate to true.

We should note that because adjacent existential quantifiers can be commuted, the existential variables for a given agent in the RDQBF can actually be assigned values in any order. So, a strategy for assigning values to the y^i_j only needs to "remember" $x^i_1, \ldots, x^i_p$ and the index j of the existential variable currently under consideration. However, since the third and fourth phases of our decision process are deterministic, a policy that uses information about $x^i_1, \ldots, x^i_p, y^i_1, \ldots, y^i_{j-1}$ to determine an action is the same as a policy that uses information about only $x^i_1, \ldots, x^i_p$ and a sequence of dummy observations of length $j-1$ (because the $y^i_1, \ldots, y^i_{j-1}$ are implicitly determined). Thus, the type of policy we describe is no more "powerful" than a legal strategy for assigning values to variables in the RDQBF.

Now for the converse, suppose the formula is true. It follows that there is a way to set the existential variables, using only information about the variables on which they depend, such that the formula is satisfied. This way of assigning values to variables is equivalent to a joint policy for the DEC-POMDP. For every clause and every choice of values for the universal variables, the assignment strategy guarantees that at least one variable in every clause is set to true and appears positively, or is set to false and appears negatively. Thus, in the DEC-POMDP, for any choices the environment can make, the joint policy guarantees that the satisfied states are entered at some point and hence the process always ends in one of the $\langle T, k\rangle$. Therefore, the policy yields expected reward zero. □

Theorem 2 *The decision problem for a finite-horizon DEC-MDP with $m \geq 3$ agents is NEXP-complete.*

Proof. (Sketch) Inclusion in NEXP follows from the fact that a DEC-MDP is a special case of a DEC-POMDP. For NEXP-hardness, we can reduce a DEC-POMDP with two agents to a DEC-MDP with three agents. We simply add a third agent to the DEC-POMDP and impose the following requirement: The state can be determined from the just the third agent's observation, but the third agent always has just one action and cannot affect the state transitions or rewards received. It is clear that the new problem qualifies as a DEC-MDP and is essentially the same as the original DEC-POMDP. □

Discussion

Using the tools of worst-case complexity analysis, we exposed some fundamental differences between centralized finite-horizon control problems and decentralized finite-horizon control problems. In particular, we showed that the decentralized versions of both POMDPs *and* MDPs (except for the aforementioned two agent case) are NEXP-complete, and thus probably take doubly exponential time to solve in the worst case.

A natural next step in the study of these decentralized control problems is to analyze the infinite-horizon cases. It has already been shown that the inifinite-horizon POMDP problem is undecidable (Madani, Hanks, & Condon 1999) under several different optimality criteria. Since a POMDP is a special case of a DEC-POMDP, the corresponding DEC-POMDP problems are also undecidable. Furthermore, it is straightforward to reduce a POMDP to a DEC-MDP by adding a second agent that observes the entire state but can have no effect on the state transitions or rewards obtained. Thus, the DEC-MDP problems are undecidable (even in the two agent case).

To get around these undecidability results, we are currently studying a restricted version of the infinite-horizon problem in which the agents are allowed to request simultaneous access to the underlying state. Thus, the agents can occasionally "synchronize". A similar model was introduced in (Ooi *et al.* 1997). Intuitively, this problem seems like an infinite sequence of finite-horizon problems, one for each of the periods between synchronization states. We are currently working on formalizing this notion and designing an algorithm that will solve this problem exactly under certain circumstances.

However, the only solution we can get in a *reasonable* amount of time is an approximate solution. Fortunately, a principled approach to approximation lies in the discovery that this problem is essentially the same as the problem of centralized planning in a *semi-Markov decision process (SMDP)* with a very large set of temporally extended actions. The states of the SMDP are the synchronization states, and the action set is the set of all joint policies for getting from one synchronization

state to the next. By reducing the number of actions, a simplified version of the SMDP can formed. This easier problem can be solved (possibly in a reasonable amount of time) to optimality using standard techniques. The solution translates back into an optimal way of dynamically switching between heuristics in the decentralized problem.

Acknowledgments

The authors wish to thank Andy Barto, Neil Immerman, Victor Lesser, Ted Perkins, and Ping Xuan for helpful discussions. Support for this work was provided in part by the National Science Foundation under grants IRI-9624992 and IRI-9634938 and an NSF Graduate Fellowship to Daniel Bernstein.

References

Aicardi, M.; Franco, D.; and Minciardi, R. 1987. Decentralized optimal control of Markov chains with a common past information set. *IEEE Transactions on Automatic Control* AC-32(11):1028–1031.

Boutilier, C. 1999. Multiagent systems: Challenges and opportunities for decision-theoretic planning. *AI Magazine* 20(4):35–43.

desJardins, M. E.; Durfee, E. E.; Ortiz, C. L.; and Wolverton, M. J. 1999. A survey of research in distributed, continual planning. *AI Magazine* 20(4):13–22.

Estlin, T.; Gray, A.; Mann, T.; Rabideau, G.; Castaño, R.; Chien, S.; and Mjolsness, E. 1999. An integrated system for mulit-rover scientific exploration. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 541–548.

Lesser, V. R. 1998. Reflections on the nature of multi-agent coordination and its implications for an agent architecture. *Autonomous Agents and Multi-Agent Systems* 1:89–111.

Madani, O.; Hanks, S.; and Condon, A. 1999. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision process problems. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*.

Ooi, J. M., and Wornell, G. W. 1996. Decentralized control of a multiple access broadcast channel: Performance bounds. In *Proceedings of the 35th Conference on Decision and Control*, 293–298.

Ooi, J. M.; Verbout, S. M.; Ludwig, J. T.; and Wornell, G. W. 1997. A separation theorem for periodic sharing information patterns in decentralized control. *IEEE Transactions on Automatic Control* 42(11):1546–1550.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3):441–450.

Papadimitriou, C. H. 1994. *Computational Complexity*. Reading, MA: Addison-Wesley.

Peterson, G. L., and Reif, J. R. 1979. Multiple-person alternation. In *20th Annual Symposium on Foundations of Computer Science*, 348–363.

Tsitsiklis, J. N., and Athans, M. 1985. On the complexity of decentralized decision making and detection problems. *IEEE Transactions on Automatic Control* AC-30(5):440–446.