Planning for Weakly-Coupled Partially Observable Stochastic Games

AnYuan Guo **Victor Lesser**
University of Massachusetts Amherst
Department of Computer Science
140 Governor's Drive, Amherst, MA 01003
{anyuan, lesser}@cs.umass.edu

1 Introduction

Partially observable stochastic games (POSGs) provide a powerful framework for modeling multi-agent interactions. While elegant and expressive, the framework has been shown to be computationally intractable [Bernstein *et al.*, 2002]. An exact dynamic programming algorithm for POSGs has been developed recently, but due to high computational demands, it has only been demonstrated to work on extremely small problems. Several approximate approaches have been developed [Emery-Montemerlo *et al.*, 2004; Nair *et al.*, 2003], but they lack strong theoretical guarantees. In light of these theoretical and practical limitations, there is a need to identify special classes of POSGs that can be solved tractably.

One dimension along which computational gain can be leveraged is by exploiting the independence present in the problem dynamics. In this paper, we examine a class of POSGs where the agents only interact loosely by restricting one another's available actions. Specifically, rather than having fixed action sets, each agent's action set is a function of the global state. The agents are independent from one another otherwise, i.e. they have separate transition and reward functions that do not interact. This class of problems arises frequently in practice. Many real world domains are inhabited by self-interested agents that act to achieve their individual goals. They may not affect each other in any way, except for occasionally putting restrictions on what each other can do.

The best-known solution concepts in game theory are that of computing Nash equilibrium and performing strategy elimination. Our work addresses both of these solution concepts. First, we show that finding a Nash equilibrium where each agent achieves reward at least k is NP-hard. Another contribution of this work is that we present an exact algorithm for performing iterated elimination of dominated strategies. It is able to solve much larger problems than exact algorithms for the general class by exploiting problem structure.

2 POSGs with State-Dependent Action Sets

In this section, we will present a formal definition of our model. An n-agent POSG with state-dependent action sets can be defined as $\langle\{S_i\}, \{s_i^0\}, \{A_i\}, \{B_i\}, \{P_i\}, \{R_i\}\rangle$, where,

- S_i is the state space of agent i. $S = S_1 \times S_2 \times \ldots \times S_n$ denotes the joint state set.
- $s_i^0 \in \Delta(S_i)$ is the initial state distribution of agent i.
- A_i is the action set of agent i. $A = A_1 \times A_2 \times \ldots \times A_n$ denotes the joint action set.
- $B_i : S \rightarrow 2^{A_i}$ is the available action function that maps a joint state to the set of available actions for agent i. $B_i(s) \subset A_i$ for all $s \in S_i$.
- $P_i : S_i \times A_i \times S_i \rightarrow [0, 1]$ is the transition function of agent i. The joint transition function is completely factored, where $P((s'_1 \ldots s'_n) \mid (s_1 \ldots s_n), (a_1 \ldots a_n)) = \prod_{i=1}^{n} P_i(s'_i|s_i, a_i)$.
- $R_i : S_i \times A_i \rightarrow \Re$ is the reward function for agent i. This states that the rewards of the agents depend only on the local state and the action taken by agent i.

In our model, each agent has a complete view of its own local state. A local policy for each agent is a mapping from the local state and the available action set to an action in that set. A joint policy is a tuple of local policies, one for each agent.

Definition 1 *A* local policy π_i*, for agent i of an n-agent POSG with state-dependent action sets, is a mapping from pairs of local states and local action sets $\langle s_i, A_i(s)\rangle$ to actions in the current local action set $A_i(s)$, where $s = \langle s_1, \ldots, s_i, \ldots, s_n\rangle$.*

Here is an example of a POSG with state-dependent action sets. Two autonomous rovers are exploring an unknown terrain and collecting rock samples. There is a river with a narrow bridge on it that only allows one rover to pass at a time. To simplify the illustration, we will assume the rovers have two states {*on land, on bridge*}, and three actions each {*move, get on bridge, pick up rock*}. The rovers receive reward for the number of rocks picked up. The state-dependent action set is used to constrain the rover's actions such that only one is allowed on the bridge at a time. For example, the available actions for rover 1 are specified as follows, $B_1(\langle s_1 =$ *on land*$, s_2 =$ *on land*$\rangle) =$ {*move, get on bridge, pick up rock*}, $B_1(\langle s_1 =$ *on land*$, s_2 =$ *on bridge*$\rangle) =$ {*move, pick up rock*}, and analogously for rover 2.

3 Complexity

We state our decision problem, denoted as POSG-NE, as follows: given a POSG with state-dependent action sets,

$G = \langle \{S_i\}, \{s_i^0\}, \{A_i\}, \{B_i\}, \{P_i\}, \{R_i\} \rangle$, does there exist a Nash equilibrium where all agents have expected utility at least k?

Theorem 1 *POSG-NE is NP-hard even in problems involving 2 agents and a horizon of 2.*

4 Iterated Action Elimination Algorithm

We present an algorithm that performs iterated elimination of dominated strategies. The algorithm is able to work directly with the compact representation of a POSG without first converting it to the double exponentially large normal form representation. In this algorithm, first, we first fix the action sets of each agent at each state to the optimistic and pessimistic action sets. The idea behind this is that although an agent cannot predict exactly what actions will be available at each state, it can find out what actions would be available in the best and worst case scenario. In the best case, none of the actions that can be restricted by the state of the other agents are, and in the worst case, all of the actions that can be restricted are in fact unavailable.

For each agent i:

1. For each state s_k, compute the optimistic and pessimistic action sets. Here $s = \langle s^1, s^2, \ldots, s^n \rangle$ and $s^i = s^i_k$.
$$A_{opt}(s^i_k) = \bigcup_s B_i(s)$$
$$A_{pes}(s^i_k) = \bigcap_s B_i(s)$$
2. Compute the value functions of the 2 MDPs that correspond to alternately fixing the available action sets to the A_{opt} and A_{pes}.
$$U(s) = \max_{a \in A_{opt}(s)} \left[R(s,a) + \sum_{s'} P(s'|s,a) U(s') \right]$$
$$L(s) = \max_{a \in A_{pes}(s)} \left[R(s,a) + \sum_{s'} P(s'|s,a) L(s') \right]$$
3. For each action in each state, derive upper and lower bounds on the action value from the value function bounds $U(s)$ and $L(s)$.
$$Q_U(s,a) = R(s,a) + \sum_{s'} P(s'|s,a) U(s)$$
$$Q_L(s,a) = R(s,a) + \sum_{s'} P(s'|s,a) L(s)$$
4. At each state, eliminate all actions a' whose action value is dominated by another action a, $Q_L(s,a) \geq Q_U(s,a')$.
5. Repeat steps 1 to 4 until no more actions can be eliminated at any state.

Table 1: The iterated action elimination algorithm.

# states	before pruning	after pruning
6	972,000	13
8	34,992,000	168
9	2.1×10^8	69
12	4.5×10^{10}	774
16	5.9×10^{13}	2,198

Table 2: Average size of the policy space before and after action elimination for problems with 3 constrained states.

Once we fix the available action sets of each agent at all the states, the agents no longer depend on each other in any way. Our model decomposes to a set of MDPs. For each agent, we solve two MDPs, one using the optimistic action sets and the other using the pessimistic action sets. The solutions will provide us with an upper and a lower bound on the actual value function of each agent. With these upper and lower bounds on the expected values of each state, we can now derive bounds on the expected action values. At each state, dominated actions are pruned. We iterate the action elimination procedure until no more actions can be pruned. Since the number of actions at each state is finite, the procedure will eventually converge.

Theorem 2 *The iterated action elimination algorithm corresponds to the iterated elimination of very weakly dominated strategies.*

5 Experiments

We tested the algorithm on a simplified 2-agent autonomous rover exploration scenario. The size of each agent's local state space varies from 6 to 16. We introduced up to three constrained states in each agent's state space. 100 trials were run for each size of state space and number of constraints. Table 2 shows detailed results for problems with three constrained states. For problems with one and two constrained states, the final policy space ranges from 3 to 106 and 8 to 252 respectively. In all three cases, the iterated action elimination algorithm is able to reduce the size of the policy space by several orders of magnitude.

References

[Bernstein *et al.*, 2002] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27:4, November 2002.

[Emery-Montemerlo *et al.*, 2004] R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *Proceedings of the Third Joint Conference on Autonomous Agents and Multiagent Systems*, 2004.

[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 Eighteenth International Joint Conference on Artificial Intelligence*, 2003.