Non-approximability of Decentralized Control

Zinovi Rabinovich
nomad@cs.huji.ac.il
School of Computer Science
and Engineering
Hebrew University of Jerusalem
Safra Campus, Givat Ram
91904 Jerusalem, Israel

Claudia V. Goldman
clag@cs.umass.edu
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Jeffrey S. Rosenschein
jeff@cs.huji.ac.il
School of Computer Science
and Engineering
Hebrew University of Jerusalem
Safra Campus, Givat Ram
91904 Jerusalem, Israel

Abstract

The complexity of finding an optimal decentralized policy for a cooperative multi-agent system is known to be non-deterministically exponential. Agents in such a system try to maximize a single global utility function, and do not compete; nevertheless, computing the optimal strategies for these agents is intractable. In this paper, we relax the requirement of finding an *optimal* decentralized policy for a cooperative multi-agent system. We consider, instead, the complexity of finding decentralized policy solutions whose values are ϵ-distant from the optimal solution value, i.e., sub-optimal solutions that may have errors. Among the major contributions of this research is showing that the apparently easier problem of finding an approximate solution to decentralized control remains as hard as the optimal version of the problem.

This paper, by formally analyzing the complexity of finding optimal and sub-optimal joint policies for cooperative decentralized systems, sheds new light on the theoretical foundations of multi-agent systems. In particular, we have adopted from complexity theory the concept of multi-prover interactive protocols (MIPs), which can serve as computational models for certain kinds of multi-agent systems. This is a new direction not yet studied within the distributed artificial intelligence community, one that provides new opportunities for the theoretical analysis of multi-agent systems.

1 Introduction

The complexity of finding an optimal decentralized policy for a cooperative multi-agent system is known to be non-deterministically exponential [4]. Agents in such a system try to maximize a single global utility function, and do not compete; nevertheless, computing the optimal strategies for these agents is intractable. We investigate whether allowing agents to reach sub-optimal solutions can result in tractable algorithms. However, as shown in this paper, the complexity bound on finding approximate solutions to a decentralized control problem is equal to that of finding exact solutions. Even though agents may be satisfied with finding a solution that may have errors, algorithms for these agents remain intractable.

Markov Decision Processes (MDPs) capture the dynamics of a stochastically developing system state, and incorporate uncertainty in the consequences of an agent's actions. Therefore, MDPs comprise an appropriate framework for studying dynamic models of agent-environment interactions. The generalized version of MDPs, Partially Observable MDPs (POMDPs), can even model *sensor aliasing*, that is, they can model the inability to completely differentiate among world states.

We chose to focus on the extended POMDP model introduced by Bernstein et al. [4], namely Decentralized POMDPs, denoted Dec_POMDPs. The Dec_POMDP model is a natural extension of POMDPs for multi-agent system modeling — when it is formulated for a single agent, it reduces to a regular POMDP. The latter have been thoroughly studied under numerous limitations of the model's parameters. The research has produced a wide range of complexity results for control problems of POMDPs, from polynomial bounds [5, 9, 17] through undecidability bounds [14]. Other distributed models, similar to Dec_MDPs, have been considered as well. For example, Littman [11, 12] describes multi-agent systems as Markov Games [22, 15, 21], and by applying variations of the Q-learning algorithm[23, 24] an optimal policy in a game theoretic sense is obtained.

Our paper takes a deeper look into the basic Dec_POMDP model [4], as an important representative of multi-agent systems with no communication among agents. We concentrate upon the problem of simultaneously prescribing separate actions to each of the agents in the system (i.e., attaining decentralized control) who are each maximizing the same single global utility function (i.e., the result will be a joint policy of action). The study of Dec_POMDPs showed that finding an exact optimal joint policy of behavior is NEXP-hard. This lower complexity bound, proven in [4], is based on a reduction from a NEXP problem, Wang tiling [16]. Unfortunately, it is problematic to define an *approximation* to the tiling problem. The difficulty follows from the complicated connections between the number of possible tilings, and the amount and the location of the allowed errors in a tiling relation. Thus, the reduction to tiling is unsuitable for providing approximability results for Dec_POMDPs. Instead, we base our results on a different reduction to a general NEXP classifier, thus refining the complexity results for the Dec_POMDP model.

We have formally analyzed the Dec_POMDP model using *Multi-prover Interactive Protocols* (MIPs), which provide us with an insight on the potential sources of computational complexity in multi-agent systems. MIPs, introduced by Ben-Or et al. [3] naturally support the notion of approximability. Their limited form is NEXP equivalent [1, 3, 8]. In this paper, we characterize MIPs as a computational model of multi-agent systems, and reduce MIPs to Dec_POMDPs. The reduction proves our non-approximability results, i.e., finding solutions for Decentralized POMDPs that come closer than some constant to the best possible value, is NEXP-hard.

2 Markov Decision Processes

Markov Decision Processes (MDPs) and their related problems [2, 10, 18] are very popular mathematical models, that capture the stochastic dynamics of a system over time. The existence of an external control (the signals) in MDPs makes the model an obvious candidate for the description of agent-environment interactions.

In this paper, we limit our attention to a specific kind of MDPs, namely stationary, fully finite, and discrete processes. The dynamics of such a system process is given by the tuple $\langle \mathcal{S}, A_s, \mathcal{T}, p_t, r_t \rangle$, where

- $\mathcal{S}$ is the set of possible system states;
- $\mathcal{A}_s$ is the set of all actions available that can affect the system state (this is the external control mentioned above);
- $\mathcal{T}$ is the number of epochs through which the system develops;
- $p_t : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ is the system transition distribution, which states the probability of reaching one state from another after a certain action was taken. This distribution describes the actual system dynamics.
- $r_t : \mathcal{S} \times \mathcal{A}_s \times \mathcal{S} \to \Re$ is the system transition evaluation, namely how good or bad, costly or profitable, the system transition from state to state was.
  However, most applications tend to use a *reward* function $r_t : \mathcal{S} \times \mathcal{A} \to \Re$, where $r_t(s,a) = \sum\limits_{s' \in \mathcal{S}} r_t(s,a,s') p_t(s,a,s')$.

Our focus on stationary, fully finite, and discrete processes imply that $\mathcal{S}, A_s$, and $\mathcal{T}$ are finite, and that the sets $\mathcal{S}$ and $\mathcal{A}_s$ are discrete. Moreover, the transition distribution p is invariant to the development epoch to which it applies.

MDP-related problems address the maximization of some optimality measure through the provision of a control policy. A *history dependent policy* $\pi = \{\pi_t\}_{0<t\leq\mathcal{T}}$, to which we limit our discussion, is a decision rule sequence, $\pi_t : H_t \to \mathcal{A}_s$, where the system development *history* is defined by $(X_1, Y_1, ..., X_t) = h_t \in H_t = \mathcal{S}^1 \times \mathcal{A}_s^1 \times ... \times \mathcal{S}^{t-1} \times \mathcal{A}_s^{t-1} \times \mathcal{S}^t$. We also limit our discussion to the *expected total reward* optimality measure, $v^\pi(s) = E_s^\pi \left\{ \sum\limits_{t=0}^{T} r_t(X_t, Y_t) \right\}$.

Although the MDP model is quite general, it relies heavily on the fact that the decision maker has complete knowledge of the current system state, and sometimes, due to history dependency, also knows past system states. Yet this assumption rarely matches real life demands — most of our sensors and knowledge bases are imperfect, and the system may provide access to only partial information about the world it models. It is possible to extend the MDP model so that it captures uncertainty in the state of the system. That is, a new set of possible observations Ω is introduced which is accompanied by a probability function $O(y \in \Omega | s, a, j)$, which describes the probability that the decision maker will observe y if the system performed a transition from state s at time $t \in T$ to state j at time $t+1$ under controlling action a. In this extended model (*Partially Observable Markov Decision Process (POMDP)*), a decision maker does not base his action selection on the system state, but rather on the observations that serve as a clue to the system reality.

2.1 Distributed MDPs

Although MDPs can be considered naturally applicable for describing agent-environment interactions, MDPs have undergone several adaptations for representing multi-agent systems (e.g., [11, 12]). In order to represent more realistic multi-agent systems, uncertainty about the system state needs to be taken into account, thus shifting the attention from MDPs to Partially Observable versions of MDPs. In this paper, we adopt the approach that considers multi-agent systems as Decentralized Partially Observable MDPs (Dec_POMDPs). A Dec_POMDP_k was formally defined by Bernstein et al. [4] as follows:

Definition 1. *Dec_POMDP_k is a tuple $\langle \mathcal{S}, \{\mathcal{A}^i\}_{i=1}^k, \{\Omega_i\}_{i=1}^k, \mathcal{O}, p, r, \mathcal{T} \rangle$, where*

- *k is the number of agents involved in the process;*
- *$\mathcal{S}$ is the set of system states;*
- *$\mathcal{A}^i$ is a set of actions available to agent i.*
- *Ω_i is a set of observations that are available to an agent;*
- *$\mathcal{O} : \mathcal{S} \times \mathcal{A}^1 \times ... \times \mathcal{A}^k \rightarrow \Pi(\Omega_1 \times ... \times \Omega_k)$ is a probability distribution. $\mathcal{O}(s, a^1, ..., a^k, s', o^1, ..., o^k)$ is the probability of agent i to see observation $o^i \in \Omega_i$ when the system arrives at state s' after the actions $a^1, .., a^k$ have been performed at state s;*
- *$p : \mathcal{S} \times \mathcal{A}^1 \times ... \times \mathcal{A}^k \rightarrow \Pi(\mathcal{S})$ is the system transition probability distribution;*
- *$r : \mathcal{S} \times \mathcal{A}^1 \times ... \times \mathcal{A}^k \times \mathcal{S} \rightarrow \Re$ is the reward function;*
- *$\mathcal{T}$ is the set of decision epochs.*

Limiting a Dec_POMDP to be fully joint observable, i.e., the combined information held by all agents determines the system state completely, results in a decentralized Markov decision process, i.e., a Dec_MDP.

Controlling a system of cooperative agents in a decentralized way is interpreted as finding a *decentralized joint policy*, that determines what policy will be followed by each agent. Bernstein et al. [4] showed that finding an optimal decentralized joint policy with respect to the expected total reward is NEXP-hard. In this paper, we extend and generalize this result by adopting a theoretical model known as Multi-Prover Interactive Protocols. This new approach to model multi-agent systems provides us with complexity results for the problem studied in this paper, i.e., finding approximated decentralized solutions for cooperative multi-agent systems.

3 Multi-Prover Interactive Protocols

Multi-Prover Interactive Protocols (MIPs) [1, 3, 8] are a computational model used to characterize complexity classes of languages. In this section, the notion of MIPs is presented together with the complexity results known for these protocols that will serve us in the next section. In section 4 we explain how an MIP can be simulated by a cooperative multi-agent system with decentralized control. Our main complexity result is obtained by this reduction.

Multi-Prover Interactive Protocols are given by a set of k provers and one verifier. Given an input word x, the provers try to convince the verifier that the word belongs to some language L. The verifier, on the other hand, has to decide whether the word x actually belongs to that language or not, based on the provers' proofs shown to him. Provers may cheat, but they cannot communicate with one another. This lack of communication among the provers makes it hard for these provers to coordinate their actions based on lying. This fact increases the probability of a verifier discovering the truth about x. Formally, an MIP is given by the following definition (from Babai et al. [1] and Ben-Or et al. [3]):

Definition 2. *Let $P_1, \ldots, P_k$ be Turing machines (*the provers*) which are computationally unbounded, and let V (*the verifier*) be a probabilistic polynomial time Turing machine. All machines share a read-only input tape. In addition, the verifier shares a communication tape with each prover P_i. There is no communication among the provers. $(P_1, ..., P_k, V)$ is a k-prover interactive protocol.*

The number of provers k can be polynomial in the size of the input, but it cannot be larger, otherwise the verifier will not be able to access all of the provers. The same logic applies to the length of the messages passed by provers to the verifier: the length of the messages cannot exceed a polynomial size of the input. A prover P_i can be seen as a function. Given the conversation that the prover has maintained so far with the verifier, and a current query from the verifier, it produces the prover's answer to the query.

Given an interactive protocol, we can characterize its corresponding language using the following definition.

Definition 3. *$(P_1, ..., P_k, V)$ form a multi-prover interactive protocol for a language L*

1. *If $x \in L$ then $Pr(P_1, ..., P_k$ make V accept $x) > 1 - 2^{-n}$, where $n = |x|$;*
2. *If $x \notin L$ then for all provers $P'_1, ..., P'_k$, $Pr(P'_1, ..., P'_k$ make V accept $x) < 2^{-n}$.*

The initials MIP also denote the class of all languages that have multi-prover interactive protocols. The following terminology is standard:

- The functions $P_1, \ldots, P_k$ will be called *the honest provers*. Any other collection of provers is *dishonest*;
- *The provers win* indicates that the verifier "accepts" the word given as input, i.e., he decides that the word belongs to the language L;
- Perfect completeness means that for all $x \in L$ honest provers always win. The following should also be noted: if $x \in L$ then dishonest provers have a positive probability of winning, yet this probability is significantly smaller than the probability of the honest provers to win.

Note that the power of a verifier over the provers comes from the lack of communication among the latter. If the provers are to convince the verifier of their truth, they need to be coordinated in advance because they will not have any other opportunity to agree upon the results of future queries. This 'pre-coordination' leads to an immense complexity of languages that can be decided by a multi-prover protocol.

Theorem 1. *The following identities among classes of languages exist:*

1. $MIP_{k=2} = MIP_{k\geq 2} = NEXP$ *[8, 1]*
2. $IP = MIP_1 = PSPACE$ *[20, 13]*

The languages that are included in the MIP class are also included in the complexity class NEXP, and vice versa. In other words, deciding whether a word is in a language that is in the MIP class will take non-deterministic exponential time. In particular, this is true for a multi-prover interactive protocol with only two provers. On the other hand, having a single prover, we can only decide PSPACE languages.

It has been proven that, even though MIPs can be extremely limited (e.g., by limiting the time or the space in which they are run), MIPs can nevertheless build protocols to decide all languages that are in the complexity class NEXP. For example, protocols that transmit constant (bit) length of messages and which include O(1) space bounded verifiers are as powerful as generally formulated MIPs. The following result is due to Feige[7, 6].

Theorem 2. *Let us denote by 'pfa' a verifier that uses O(1) working tape space.*[1] *Then MIP(pfa)=MIP.*
That is, for any language decidable by a multi-prover protocol in polynomial time, there exists a protocol with a constant space bounded verifier, that decides the same language.

This important result suffers from a minor weakness. When provers cheat, the protocol provided by Feige [7] has a small probability of breaking the polynomial time bound on the performed computations. Sacrificing logarithmic space for step counting easily cures the problem. Note that this protocol provides perfect completeness.

Multi-agent systems provide us with an interesting perspective on MIPs and vice versa. Agents in multi-agent systems, as well as provers in MIPs, all cope with a hostile adversary, namely the environment for MAS or the verifier for MIPs. Yet multi-agent systems comprise a more general term than MIPs. The additional power that multi-agent systems have over MIPs is engendered by the possibility of communication among the agents, along with their interactions with the environment, communication which is excluded from MIPs by definition. Moreover, the environment's behavior in multi-agent systems may be too complex to be captured by a polynomial time Turing machine as assumed in MIPs. However, in most practical applications the dynamics of the environment can be simulated by a time limited computation as represented by a verifier in MIPs. So MIPs can be seen as a model for a multi-agent system, limited in communication and environmental complexity, but quite practical.

[1] Note that probabilistic Turing machines limited to $O(1)$ working tape space are essentially probabilistic finite automata, that is 'pfa's [16, 7].

4 Approximate Solutions to Decentralized Control

In this section, we prove our main result concerning the complexity of finding ϵ-approximate joint policies for Dec_POMDPs. We analyze the complexity of deciding whether an ϵ-approximate joint policy exists for a given Dec_POMDP. Since attempting to find this policy cannot be easier than determining its existence, the complexity bound that was found holds for the problem in question as well.

In order to define an approximate joint policy, we assume that we want to obtain a reward value R, and we assume that there is a policy π that achieves a reward r. Then π is said to have relative error $E(R,\pi) = \frac{|r-R|}{\max\{r,R\}}$.

Definition 4. *A policy π is an ϵ-approximate policy if π yields an expected reward with relative error less than ϵ, i.e., $E(R,\pi) < \epsilon$.*

In the next theorem, we show that deciding whether a sub-optimal joint policy for the Dec_POMDP_2 exists, is equivalent to deciding whether a word belongs to a language that is in the NEXP complexity class. Given an input word x, we build a Dec_POMDP for two agents such that the provers' different reactions and the agents' local policies are isomorphic. This will identify the existence of a set of persuasive provers and the existence of an approximate decentralized joint policy for the cooperative two-agent system.

Theorem 3. *Deciding whether there exists an ϵ-approximate joint policy for Dec_POMDP_2 is NEXP-hard.*

Proof. Assume that a language L which belongs to the complexity class NEXP is given. Thus, for some input word(s) x, of length $n > 0$, deciding whether x belongs to L will take non-deterministically exponential time.

It is known (see section 3) that for any such language, there exists a multi-prover interactive protocol with two provers that can decide for any word x whether $x \in L$ or $x \notin L$. This $MIP_{k=2}$ has perfect completeness[2] and uses only logarithmic memory.

To prove this theorem we will show the following correspondence between deciding whether a word x is in an

[2] As explained in section 3, this means that honest provers always win for any word $x \in L$.

NEXP language and deciding whether there exists a sub-optimal joint policy π whose value is ϵ distant from the maximal total expected reward given a Dec_POMDP_2 that will be built corresponding to a $MIP_{k=2}$:

$$x \in L \iff \exists \pi E(1, \pi) < \epsilon.$$

Therefore, we decide that there exists such ϵ-approximate joint policy π, if and only if we decide that $x \in L$. This will be shown for any word x, therefore the reduction shown hereafter will be able to decide any language $L \in NEXP$. But then, deciding whether there is an approximate joint policy for a given Dec_POMDP is at least NEXP.

The proof consists of three parts. We first show how a Dec_POMDP can be built for two agents that corresponds to a multi-interactive protocol with two provers. Then, we show that this reduction has only polynomial time complexity and therefore cannot cause any increase in the complexity of the decision problem being studied. We finally demonstrate that deciding whether there exists an ϵ-approximate joint policy for the constructed Dec_POMDP is NEXP.

The construction of a Dec_POMDP_2 that corresponds to $MIP_{k=2}$ — We assume that the verifier is represented by the environment, that is the system notion of the Dec_POMDP. The provers correspond to the two agents acting in the system described by the Dec_POMDP. That is, the agents' actions correspond to the provers communicating with the verifier. The dynamics of the environment are affected by the agents' actions and by the behavior of the verifier that moves the system between its states.

The corresponding Dec_POMDP_2 is built as follows: $(\mathcal{S}, \mathcal{A}_{1,2}, \Omega_{1,2}, \mathcal{O}, p^x, r, \mathcal{T}^x)$ where,

- S is the set of states of the system. A state is given by a triple $(m^V_{P_1}, m^V_{P_2}, b^V)$, consisting of the messages that the verifier has sent to each one of the provers in one previous step, and the current belief of the verifier. The verifier can believe that the given word x belongs to the language L or not. This will be denoted by the values that b^V can attain, either *accept* or *reject*. Based on the MIPs definition (definition 2), the communication between the verifier and the provers is polynomially bounded. Therefore, for each language there is a polynomial bound that characterizes the length of the conversation between the verifier and the provers. We denote this bound by cn^q where $n = |x| > 0$ and c and q characterize the polynomial bound. The messages $m^V_{P_i}$ are composed from an alphabet Σ. The length of each message passed from the verifier to the provers and vice-versa is constant.

  Keeping a simple binary belief within the state is a mild simplification. To formalize our proof completely, along with these beliefs, it is necessary to keep more information within the system state description, e.g., a time stamp or time counter. The number of bits required to encode such extended information is logarithmic in n (see the previous section on MIPs and [7, 6]), so the number of possible states in our Dec_POMDP under this construction will be significantly large, but it still remains bounded by a polynomial in n.

  The initial state s_0 is given by $(\Box, \Box, accept)$. The first two parameters are two empty strings, since no messages were sent at stage zero. The last parameter of a state changes its value depending on the messages that the verifier receives from the provers during the process.

- A_1 and A_2 are the sets of actions that can be performed by the provers. An action a^i_V is given by the message that is sent by prover i to the verifier. These messages are taken from the same alphabet Σ as the messages $m^V_{P_i}$ sent by the verifier and have the same length.

- Ω_1 and Ω_2 are the sets of observations seen by the provers. These observations are given by the messages $m^V_{P_i}$ that the verifier sends to each one of the provers P_i.

- $O(s, a^1_V, a^2_V, s', o_1, o_2)$ denotes the probability of having each prover observe their corresponding observations $o_1 = m^V_{P_1}$ and $o_2 = m^V_{P_2}$, after having sent the messages a^i_V to the verifier at state s and having moved to state s'.

- $p^x(s, a^1_V, a^2_V, s')$ is the probability of moving to state s' after being in state s when the provers sent the messages a^i_V respectively. These messages were the provers' responses to the messages sent by the verifier that were observed in state s. The system moves to state s' where the verifier will send another pair of messages to the provers as a response to the pair of messages (a^1_V, a^2_V). Since in this construction, we assume that the verifier is the system, then this probability is given by the behavior of the verifier. We assume that the verifier, that is, the environment, is represented by a probabilistic finite automaton (pfa).[3] The states of the pfa machine are given by b^V, i.e., each state corresponds to a certain belief of the verifier. In addition, the automaton states are divided into two groups: "accepting" states and "rejecting" states, depending on whether the verifier believes that $x \in L$ (accept) or $x \notin L$ (reject). Therefore, p^x is completely defined by the probability transitions of the verifier in its pfa.

[3] Theorem 2 proved that a language that is in the NEXP class can be decided even if the verifier is implemented as a probabilistic finite automaton (pfa).

- The reward obtained from the provers sending to the verifier the message a_V^1 and a_V^2 and moving to another state s' in which the verifier sends his response to each one of the provers is given by the following formula:

$$r(s, a_V^1, a_V^2, s') = \begin{cases} \frac{1}{\mathcal{T}^x} - 1 & b_s = accept, b_{s'} = reject \\ \frac{1}{\mathcal{T}^x} & otherwise \end{cases}$$

.

  The agents are punished when the verifier (i.e., the system) changes its beliefs from acceptance to rejection. Note that once the 'rejection' belief is reached it never changes. The system just carries on until the time limit is reached.

  Since we assumed that the initial state of the system is given by $(\Box, \Box, accept)$, then in the best case, when the verifier does not change his beliefs during the $\mathcal{T}$ epochs, the reward will yield its maximal value, i.e., one. In the worst case, the verifier *accepts* that x is in L for some period of time, but changes his belief at some point, and then continues to *reject* the fact that x is in L. Then the minimal value that can be attained is zero.[4]

- $\mathcal{T}^x$ is the time horizon of the system. Since for each specific language and verifier, a polynomial time bound on the verifier and provers' conversation length is known, we may derive c and q such that for any x it will take cn^q steps to either accept or reject it with high certainty. Hence, we define the time horizon by $\mathcal{T}^x = cn^q$.

Notice that in the constructed Dec_POMDP, we cannot achieve fully joint observability because the provers can never obtain information about the verifier beliefs, which are part of the state definition.

The Reduction is Polynomial — Producing the Dec_POMDP as shown in this proof, for any word x is 1) polynomial in $n = |x|$, in time, and 2) logarithmic in n, in space. Thus the reduction itself is not a source of problem complexity.

First, we showed that computing the horizon of the Dec_POMDP is polynomial in time, because it requires the computation of cn^q. The space needed is logarithmic. Since we also assumed that the length of the messages transmitted in the process are of a constant length, then the size of the states are constant in the length of the word x. The belief of the verifier requires only one bit to describe it. Computing the sets of the actions and the observations, and the probability transitions, is also polynomially bounded in time, and logarithmically in space. All these sets refer to messages and all of the messages have a constant size. We may limit the number of coin tosses that the verifier does to move in his pfa and to determine the messages he will send. Therefore computing p^x is also polynomial in n.

Therefore the reduction made is polynomial in time and logarithmic in space. The reduction itself cannot be a source of an increase in complexity of the decision problem.

The Decision Problem — For each input word x, we construct a Dec_POMDP_2 decision problem, i.e., determine whether an ϵ-approximate joint policy for some reward value exists. The decision problem will have a solution if and only if $x \in L$.

Now consider the following special case of a decision problem for the previously defined Dec_POMDP: "Is there an ϵ-approximate joint policy π such that $E(1, \pi) < \epsilon$ (i.e., π's relative error with respect to the maximal optimal reward is at most ϵ)?". We show that the decision regarding this question depends only on whether $x \in L$ or not.

Theorem 3 is about a more general decision problem, i.e., deciding whether there is an ϵ-approximate joint policy for any value of R (i.e., not necessarily 1). This general problem is harder than the special problem we stated here. However, we show that the lower bound for the special case is NEXP, and therefore we can conclude that deciding the general problem stated in the theorem is also at least NEXP.

Assume that $x \in L$. Since the protocol assumed in the reduction is of perfect completeness, there exist two (functional) provers P_1, P_2 such that $P(V\ accepts) = 1$. Using these provers, we can design a history-dependent joint policy π^x that will reproduce, for each pair of observations (given by the messages sent by the verifier to the provers), a pair of actions (given by the messages sent by the provers to the verifier), so that the system (the verifier) will never change its beliefs from acceptance to rejection. We conclude that there exists a joint policy such that its expected reward value is greater or equal to $1 - \epsilon$, when $0 \leq \epsilon \leq \frac{1}{2}$, because $P(V\ accepts) = 1$ yields π^x to obtain the maximal reward of $1 \geq 1 - \epsilon$.

On the other hand, if $x \notin L$, then for any joint policy of agents we may produce a corresponding pair of provers with the same message response strategy. Yet, since $x \notin L$, the verifier will almost always reject, and agents will receive a negative reward with probability greater than $1 - 2^{-n}$,[5] which will lead to an expected reward less than 2^{-n}.[6] Thus, there is no joint policy that will produce an expected reward of at least $1 - \epsilon$, when $0 \leq \epsilon \leq \frac{1}{2}$.

[4] $\frac{\mathcal{T}-1}{\mathcal{T}} + \frac{1}{\mathcal{T}} - 1 = 0$

[5] Refer to section 3, $P(V\ rejects) \geq 1 - 2^{-n}$.

[6] With probability of $1 - 2^{-n}$ the system gets a reward of 0, and with a probability of 2^{-n} the verifier does not change his beliefs and the system yields a reward of 1, leading to an expected total reward of $(1 - 1/2^{-n}) * 0 + 2^{-n} * 1 = 2^{-n}$.

We conclude that determining the existence of a joint policy with expected reward at least $1-\epsilon$ is as hard as determining whether $x \in L$. Since $L \in NEXP = MIP_{k=2}$, we have that the existence of an ϵ-approximate joint policy decision problem for Dec_POMDP_2 is NEXP-hard. □

5 Conclusions

In this paper, we have introduced multi-prover interactive protocols (MIPs) as a model for multiagent systems. The modeling power of this approach is limited to multiagent systems with no inter-agent communication, and polynomial environmental complexity. Using known complexity results for MIPs, a reduction between the MIP model and the Dec_POMDP model has been presented. This reduction provides us with a lower complexity bound for solving a Dec_POMDP problem. Our reduction extends the results obtained in Bernstein et al. [4], concluding that both optimal and ϵ-approximate history-dependent joint policies are NEXP-hard to find in Dec_POMDPs.

We have shown a complexity lower bound for deciding whether a sub-optimal joint policy exists for a Dec_POMDP_2. Since this is a lower bound, we can conclude that for any $m \geq 2$, the decision for a Dec_POMDP_m is also NEXP (i.e., the decision problem cannot become easier). In addition, [4] have also proved that a Dec_MDP_3 can be reduced to a Dec_POMDP_2 and that deciding a Dec_MDP_3 is as hard as deciding a Dec_POMDP_2. With this result and the results for approximating Dec_POMDP_2 shown in this paper, it follows that approximating a Dec_MDP_3 is NEXP. Since this is a lower bound, we can conclude that for any $m \geq 3$, deciding whether a sub-optimal joint policy exists for any Dec_MDP_m is NEXP. Our results show that decentralized control of dynamic cooperative multiagent systems is NEXP-hard and not approximable. The source of this complexity lies mainly in the way the information about the system state is distributed among the agents.

MIPs as a model for multiagent systems demonstrates the complexity of information sharing in their two extremes: when there is no information sharing, and when there is full information sharing. In the first case, we see that the computational complexity rises to nondeterministic exponential (NEXP). In the latter, full information sharing practically deprives different agents of their independence and a group of them can be seen a single and sophisticated prover. Moreover, in the case of single prover, the system complexity drops to polynomial space (PSPACE). Similar results for these two extremes in the case of optimal policies were also shown in [19]. We can conclude from the results shown in this paper, that if no other assumptions are made about a multiagent system, the complexity of finding an optimal or *even* an approximate solution will lie between PSPACE and NEXP.

It is interesting to compare our results with those of Madani et al. [14] which deal with Markov decision processes with infinite time horizons. Madani et al. prove that finding an exact policy of any sort for infinite horizon POMDPs is undecidable, and the result naturally carries over to any extension, e.g., Dec_POMDP with infinite horizon. On the other hand, approximate solutions to POMDPs with infinite horizon are indeed tractable [5, 9]. Therefore, considering a process with an infinite horizon, it remains an open question whether approximating decentralized control is tractable as well.

Moreover, the implementation of MIPs as models for multiagent systems can guide us in other computational complexity-related topics not yet studied in multiagent systems.

Acknowledgments

We'd like to thank Prof. M. Ben-Or for his expert guidance through complexity theory, and members of the Theory Lab at the Hebrew University for their support and assistance.

We also thank Prof. S. Zilberstein and the members of the RBR Lab at the University of Massachusetts for fruitful discussions.

References

[1] Laszlo Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

[2] Richard Ernest Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[3] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the 20th ACM Symposium on Theory of Computing*, pages 113–131, 1988.

[4] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of MDPs. In *Proceedings of the 16th conference on uncertainty in artificial intelligence(UAI-2000)*, 2000.

[5] Vincent D. Blondel and John N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, September 2000.

[6] U. Feige and A. Shamir. Multi-oracle interactive protocols with space bounded verifiers. In *Proceedings of the 4th Annual Structure in Complexity Theory Conference*, pages 158–164, June 1989.

[7] Uriel Feige. *Alternative models for zero knowledge interactive proofs*. PhD thesis, The Weizmann Institute of Sience, Rehovot, Israel, 1990.

[8] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994. revising : Proc. 3rd Structure in Complexity Theory Conf., 1988, 156-161.

[9] Judy Goldsmith and Martin Mundhenk. Complexity issues in Markov decision processes. In *Proceedings of IEEE Conference on Computational Complexity*, 1998.

[10] Ronald A. Howard. *Dynamic programming and Markov processes*. MIT Press, Cambridge, MA, 1960.

[11] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *ML11: 11th international conference on machine learning*, pages 157–163, 1994.

[12] Michael L. Littman. Value-function reinforcement learning in Markov games. *Journal of Cognitive Research*, 2:55–66, 2001.

[13] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39:859–868, 1992.

[14] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 541–548, 1999.

[15] G. Owen. *Game Theory*. Academic Press, Orlando, FL, 2nd edition, 1982.

[16] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[17] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, August 1987.

[18] Martin L. Puterman. *Markov Decision Processes*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics Section. Wiley-Interscience Publication, New York, 1994.

[19] D. V. Pynadath and M. Tambe. Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. In *Proceedings of the 1st conference on autonomous agents and multiagent systems (AAMAS-2002*, 2002.

[20] A. Shamir. IP=PSPACE. In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Sciences*, pages 11–15, 1990.

[21] L. Shapley. Stochastic games. In *Proc. Natl. Acad. Sci. USA*, volume 39, pages 1095–1100, 1953.

[22] J. van der Wal. In *Stochastic Dynamic Programming*, volume 139 of *Mathematical Centre Tracts*. Mathematisch Centrum, Amsterdam, 1981.

[23] C. J. Watkins. *Models of Delayed Reinforcement Learning*. PhD thesis, Psychology Department, Cambridge University, 1989.

[24] C. J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.