Complexity of Finite-Horizon Markov Decision Process Problems

Martin Mundhenk
and
Judy Goldsmith
and
Christopher Lusena
and
Eric Allender

Supported in part by the Office of the Vice Chancellor for Research and Graduate Studies at the University of Kentucky, and by the Deutsche Forschungsgemeinschaft (DFG), grant Mu 1226/2-1, and by the Deutsche Akademische Austauschdienst (DAAD) grant 315/PPP/gü-ab, and by NSF grants CCR-9315354, CCR-9610348, and CCR-9509603.
Portions of this work were performed while the fourth author was at the Institute of Mathematical Sciences, Chennai (Madras), India, and at the Wilhelm-Schickard Institut für Informatik, Universität Tübingen (supported by DFG grant TU 7/117-1).
Preliminary versions of some of this work appeared as Mundhenk, Goldsmith, and Allender [1997] and Goldsmith and Mundhenk [1998].
Name: Martin Mundhenk
Address: Universität Trier, FB IV – Informatik, D-54286 Trier, Germany, mundhenk@ti.uni-trier.de
Affiliation: Universität Trier

Name: Judy Goldsmith
Address: Dept. of Computer Science, University of Kentucky, Lexington KY 40506-0046, goldsmit@cs.engr.uky.edu
Affiliation: University of Kentucky

Name: Christopher Lusena
Address: Dept. of Computer Science, University of Kentucky, Lexington KY 40506-0046, lusena@cs.engr.uky.edu
Affiliation: University of Kentucky

Name: Eric Allender
Address: Dept. of Computer Science, Rutgers University, Piscataway, NJ 08855, USA, allender@cs.rutgers.edu
Affiliation: Rutgers University

Controlled stochastic systems occur in science, engineering, manufacturing, social sciences, and many other contexts. If the system is modeled as a Markov decision process (MDP) and will run *ad infinitum*, the optimal control policy can be computed in polynomial time using linear programming. The problems considered here assume that the time that the process will run is finite, and based on the size of the input. There are many factors that compound the complexity of computing the optimal policy. For instance there are many factors that compound the complexity of this computation. For instance, if the controller does not have complete information about the state of the system, or if the system is represented in some very succinct manner, the optimal policy is provably not computable in time polynomial in the size of the input. We analyze the computational complexity of evaluating policies and of determining whether a sufficiently good policy exists for a MDP, based on a number of confounding factors, including the observability of the system state; the succinctness of the representation; the type of policy; even the number of actions relative to the number of states. In almost every case, we show that the decision problem is complete for some known complexity class. Some of these results are familiar from work by Papadimitriou and Tsitsiklis and others, but some, such as our PL-completeness proofs, are surprising. We include proofs of completeness for natural problems in the as yet little-studied classes $\mathrm{NP}^{\mathrm{PP}}$.

Categories and Subject Descriptors: I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Plan execution, formation, and generation*; F.1.3 [**Computation by abstract devices**]: Reducibility and completeness; G.3 [**Probability and Statistics**]: *Markov processes*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*

General Terms: Markov decision processes, Partially observable Markov decision processes, Computational complexity

Additional Key Words and Phrases: Succinct representations, NP, PSPACE, $\mathrm{NP}^{\mathrm{PP}}$, PL

1. INTRODUCTION

Some argue that it is the nature of man to attempt to control his environment. However, control policies, whether of stock market portfolios or the water supply for a city, sometimes fail. One problem in controlling complex systems is that the actions chosen rarely have determined outcomes: how the stock market or the weather behaves is something that we can, at best, model probabilistically. Another problem in many controlled stochastic systems is that the sheer size of the system makes it infeasible to thoroughly test or even analyse alternatives. Furthermore, a controller must take into account the costs as well as rewards associated with the actions chosen. This paper considers the computational complexity of the basic tasks facing such a controller.

We consider here a mathematical model of controlled stochastic systems with utility called *Markov decision processes* (MDPs). We also consider a model that takes into account that the exact state of the system may at times be hard or impossible to recognize. Such a system can be modeled by *partially observable Markov decision processes* (POMDPs).

Given such a model, a controller needs to be able to find and evaluate control policies. We consider the computational complexity of these two problems. Unsurprisingly to either algorithm designers or theorists, we show that incomplete information about the state of the system hinders computation. However, the patterns formed by our results are not as regular as one might expect (see Tables

1–3).

One major contribution of this paper is a natural problem that is complete for the class NP^{PP}. This class captures the notion of guessing a solution and checking it probabilistically; the complete problem is of the form, "Is there a good policy for this MDP?" The type of MDP considered for this theorem generalizes a variety of models used by the Planning community, problems whose complexity was not fully understood until [Goldsmith et al. 1997]. Until then, they were simply classified as NP-hard and in PSPACE. With the advent of NP^{PP}-completeness proofs, algorithms have begun to be developed for such problems [Majercik and Littman 1998b; Majercik and Littman 1998a; Littman 1999b].

In this paper, we consider problems of interest when a controller has been given a MDP or POMDP modeling some system. We assume that the system has been described in terms of a finite state space and a finite set of actions, with full information about the probability distributions and rewards associated with being in each state and taking each action.

In Operations Research, MDPs are usually assumed to be continuous. We do not address any of the complexity issues for the continuous case here. In addition, there is a large literature on acquiring the actual MDPs or POMDPs, usually by some form of automated learning or by knowledge elicitation. While there are fascinating questions of algorithms, complexity and psychology (see for instance [van der Gaag et al. 1999]) associated with these processes, we do not address them here.

In the remainder of this section, we discuss the basic models, MDPs, POMDPs, and unobservable Markov decision processes; we consider the most important parameters that affect the complexity of computational problems for MDPs, etc., and we present an overview of the results, and of the rest of the paper.

Markov Decision Processes. MDPs model sequential decision-making problems. At a specified point in time, a decision maker observes the state of a system and chooses an action. The action choice and the state produce two results: the decision maker receives an immediate reward (or incurs an immediate cost), and the system evolves probabilistically to a new state at a subsequent discrete point in time. At this subsequent point in time, the decision maker faces a similar problem. The observation made from the system's state now may be different from the previous observation; the decision maker may use the knowledge about the history of the observations. The goal is to find a policy of choosing actions (dependent on the observations of the state and the history) which maximizes the rewards after a certain time.

Finding an optimal policy for a fully-observable MDP — i.e. an MDP which allows one to observe its current state — is a well studied topic in optimization and control theory (see e.g. Puterman [1994] as one recent books). Tractable algorithmic solutions use dynamic or linear programming techniques. The inherent complexity was first studied in [Papadimitriou and Tsitsiklis 1987; Smallwood and Sondik 1973; Sondik 1971]. However, computers are used in more and more areas of daily life, and real life systems lack the convenient property of being fully observable. In many models, there are properties of a state that are not exactly observable, so the observations only specify a subset of the states to which the current state belongs. The model of *partially-observable* MDPs (abbreviated POMDPs) was introduced by Smallwood and Sondik [1973].

Only for small models can one find exact solutions with feasible resources [Cassandra et al. 1994; Madani et al. 1999; Monahan 1982]. Even the known ε-approximations are useful only on systems with tens of states [Cassandra et al. 1995; Cassandra et al. 1997; Lovejoy 1991; Parr and Russell 1995; White 1991; Zhang et al. 1999; Zhang and Liu 1997]. Since already simple tasks can need models with thousands of states (see [Simmons and Koenig 1995]), the hardness of the problem is well-known. However, despite the work of Papadimitriou and Tsitsiklis [1987] and a few others, there were many variants of POMDP problems for which it had not been proved that finding tractable exact solutions or provably good approximations is hard. There is a growing literature on heuristic solutions for POMDP (see [Cassandra 1998; Hansen 1998b; Hauskrecht 1997; Lovejoy 1991; Lusena et al. 1999; Meuleau et al. 1999; Peshkin et al. 1999; Platzman 1977; Smallwood and Sondik 1973], for instance.). Since these algorithms do not yield guaranteed optimal or near-optimal solutions, we leave a discussion of them to other sources.

In this paper, we address the computational complexity, given a process and specifications for a decision maker, of (a) evaluating a given policy, and (b) finding a good policy for choosing actions. We abstract the first of these to the *policy evaluation problem*, "Does this policy have expected reward > 0?" The second of these was abstracted by Papadimitriou and Tsitsiklis [1987]to the *policy existence problem*, "Is there a policy with expected reward equal to 0?" The underlying assumption in [Papadimitriou and Tsitsiklis 1987] is that all rewards are negative or 0, and thus a policy with reward 0 is optimal. In contrast, we consider two kinds of policies, (a) policies with only *nonnegative* rewards, and (b) policies with both positive and negative rewards. POMDPs with nonnegative rewards tend to be computationally much simpler, since in this case rewards cannot cancel each other out. We abstract the policy existence problems as follows: "Is there a policy with expected reward > 0?" We have chosen these decision problems because they can be used, along with binary search, to calculate the exact value of the given or the optimal policy[1].

It could be argued that the best way to evaluate a policy is to test it. There are two problems with this approach. One is that it gives us a single possible outcome, rather than a probabilistic analysis. The second is that some systems should *not* be run to test a hypothesis or policy: for instance, a military scenario. Although any POMDP can be simulated, this only gives us an approximation to the expected outcome. Therefore, it is important that we be able to evaluate policies directly.

The complexity of the decision problems depends on specifications on the decision maker and the representation of the process. The specifications on the decision maker include the amount of feedback that the decision maker gets from the system (observability), restrictions on her/his computing resources (type of policy), and the length of time that the process will run (horizon). The straightforward process representation consists of a function for the transition probabilities and a function

[1]Binary search involves creating a series of new POMDPs from the given one by appending an initial state such that any action from that state yields a positive or negative reward equal to the shift from the last value considered. It seems that this process requires both negative and positive rewards.

for the rewards. The usual way is to give these functions as tables. When the system being modelled has sufficient structure, one may use a more consice representation e.g. via circuit, to efficiently specify processes whose table representation would be intractable.

Observability. We consider three models of Markov Decision Processes: plain or *fully-observable* MDPs, which we refer to simply as *MDP*s; processes where the decision maker may have incomplete knowledge of the state of the system, called *partially-observable* MDPs or *POMDPs*, and processes where the decision maker has no feedback at all about the present state, which we call *unobservable* MDPs, or *UMDPs.* (More formal definitions are given in Section 2.) Note that MDPs and UMDPs are special cases of POMDPs.

The only difference between MDPs and POMDPs is the observability of the system. Whereas with MDPs the exact state of the system can always be observed by the decision maker, this is not the case with POMDPs. The effects of observability on the complexity of calculating an optimal policy for a process are drastic. For MDPs, dynamic programming is used to calculate an optimal policy (for a finite number of decision epochs, or a finite *horizon*) in time polynomial in the size of the system. Moreover, optimal policies can be expressed as functions of the system's current state and the number of previous decision epochs. Papadimitriou and Tsitsiklis [1987] showed that in this case, related decision problems are P-complete. In contrast, the optimal policies for POMDPs are generally expressible as functions from initial segments of the history of a process (a series of observations from previous and current times). This is unfortunate because the problem of finding optimal policies for POMDPs is PSPACE-hard. Moreover, specifying those policies explicitly may require space exponential in the process description. Hence, it is intractable (again, in the worst case) for a decision maker to choose actions according to an optimal policy for a POMDP.

Policy types. The computational restrictions of the decision maker are expressed in terms of different types of policies. The simplest policy takes only the actual observation into account (*stationary policy*), whereas the *history-dependent policy* makes use of all observations collected during the run-time of the process. Intermediate lies the *time-dependent policy*, which makes its choice of actions dependent on the actual observation and on the number of steps the process performed already.

For infinite horizon MDP policies with discounted rewards, the optimal value is achieved by a stationary policy. The optimal policy for an MDP with finite horizon is a time-dependent policy, under discounted or total expected rewards. For POMDPs, the optimal policy under finite or infinite horizons, average, total expected, or discounted rewards, is history-dependent. Since UMDPs cannot distinguish histories except as blind sequences of actions, history-dependent policies are equivalent to time-dependent policies.

Representation. A POMDP consists of a finite number of states, a function which maps each state to the observation made from this state, a probabilistic transition relation between states, and a reward function on states; these last two are dependant on actions. The straightforward representation of a POMDP is by a set of tables for the transition relation – one table for each action – and similar tables for the reward function and for the observation function. There is interest in so-called "structured" POMDPs [Boutilier et al. 1995; Bylander 1994; Littman 1997a]. These representations arise when one can make use of structures in the state space to provide small descriptions of very large systems [Boutilier et al. 1995]. In many cases, when a MDP or POMDP is learned, there is an *a priori* structure imposed on it. For instance, consider the example cited in [van der Gaag et al. 1999]: the system modeled is for diagnosis and treatment of cancer of the oesophagus. The actual POMDP has been elicited from physicians. A list of significant substates, or variables, was elicited: characteristics of the carcinoma, depth of invasion into the oesophagus wall, state of metastases, etc. An actual state of the system is described by a value for each variable. Then interrelations of the variables and probabilities of transitions were elicited.

There are many ways to model a structured MDP or POMDP, such as two-phase temporal Bayes nets (2TBNSs) or probabilistic STRIPS operators (PSOs) (see [Blythe 1999] and [Boutilier et al. 1999] for a discussion of a richer variety of representations); each of these representations may represent a very large system compactly. We choose a different representation that captures the succinctness of all of the representations common in AI, but does not necessarily have the semantic transparency of the others. We use circuits to describe the computation of probabilities, without specifying any internal representation that reflects the semantics of the situation. While the other representations with which we are familiar can be transformed (via easily computed reductions) to circuits, there may also be instances of MDPs or POMDPs that can be represented succinctly using circuits, even if we do not know a set of variables to represent the states.

There is a polynomial-time algorithm that takes any set of probabilistic STRIPS operators (PSOs) or, equivalently, 2TBNs [Littman 1997b] representing an MDP or POMDP and produces a circuit representation as described here. Thus, the circuit representation is at least as powerful as the PSO and 2TBN representations. The problems described here for circuit representations have the same complexity as those for PSO or 2TBN representations (see [Littman et al. 1998]), but this does not mean the all of the representations are equivalent. There are MDPs with 2^n states that can be represented using poly(n)-size circuits that require PSOs of size $\mathcal{O}(2^n)$. For instance, any action whose consequences depend on the parity of the number of propositions that are true will require an PSO exponential in the number of propositions but only a polynomial-size circuit.

Recently, a variant of 2TBNs has been introduced using ADDs (arithmetic decision diagrams) instead of tables or trees [Hoey et al. 1999]. While the ADDs improve the performance of the Bayes nets algorithms, they do not affect the underlying complexity of the problems: the hardness results for 2TBNs carry over immediately, and it is not hard to show that the complexity upper bounds for 2TBNs also hold. For any ADD, there is an equivalent circuit of approximately the same size. ADDs can encode parity problems succinctly, but there are well-known instances of functions with small circuits that have exponentially large BDDs and thus ADDs [Bryant 1991]. One example is the function $f(x_1 ... x_n) = x_a$, where $a = \Sigma_1^n x_i$. In order to remember each x_a, each of the 2^n paths through the tree must be disjoint, yielding an exponential-size tree.

The circuits described in this paper take as input a state s, an action a, and a potential next state s', and output the probability that action a performed when the system is in state s will take the system to state s'. This assumes that the circuit designer has such information about the model, and that that is the information that the policy maker uses. Other researchers have modeled succinctness by circuits that take as input a state s and an action a and randomly output a next state with the appropriate probability (relative to the random string) [Sutton and Barto 1998]. This assumes a knowledge of the system rather than the model, and that the policy maker bases its decisions on simulations rather than on the underlying model.

There are arguments for both the simulation-based and the model-based approaches. We consider both approaches valid and interesting, but address only the latter in this paper.

Our results show that succinctness is not a panacea for "the curse of dimensionality," [Sondik 1971], although there are certainly instances of succinct POMDPs where the optimal policies are easy to find. Thus, future research on efficient policy-finding algorithms for succinctly represented POMDPs must focus on structural properties of the POMDPs that limit the structure of the circuits in ways that force our reductions to fail.

In fact, the reductions that we use can be translated into results for a variety of representations. The fact that we use circuits to present these POMDPs is primarily a matter of convenience, for showing that the problems in question are members of their respective complexity classes. It also ties into complexity theory literature on succinct representations (c.f. [Galperin and Wigderson 1983; Wagner 1986]).

Applications

MDPs were described by Bellman [1957] and Howard [1960] in the '50's. The first known application of MDPs was to road management in the state of Arizona in 1978 ([Golabi et al. 1982], as cited in [Puterman 1994]). The states represented the amount of cracking of the road surface; the actions were the types of removal, resurfacing, or crack sealing. Puterman also reports on applications of MDPs to wildlife management, factory maintenance, and warehousing and shipping.

In many instances of controlled stochastic systems, the state of the system is not fully observable. In an industrial or robotic setting, this could be because of sensor failure or simply that sensors cannot distinguish between all the states modeled. For instance, in the robot location problem, the robot can only see its immediate surroundings, which may look just like some other location, at least to the robot. In the organic settings, the controller may simply not have access to full information. For instance, a doctor rarely, if ever, has complete information about a patient's state of health, yet she is called on to make decisions and choose actions based on what she knows, in order to optimize the expected outcome. POMDPs model such systems, and were developed in the '60's by Astrom [1965], Aoki [1965], Dynkin [1965] and Streibel [1965]; the final formalization was done by Sondik [1971], and Smallwood [Smallwood and Sondik 1973].

A rich source of information on applications of MDPs and POMDPs is the wealth of recent dissertations, for instance by: Littman [1996a], Hauskrecht [1997], Cassandra [1998], Hansen [1998a], and Pyeatt [1999].

Any controlled system represented by Bayesian networks or other similar graphical representations of stochastic systems, as well as any systems represented by probabilistic STRIPS-style operators is an MDP or POMDP. It is beyond the scope of this paper to give a full survey of applications of MDPs, POMDPs, and their variants. To give some sense of the breadth of possible applications, we mention those that came up at a recent conference, Uncertainty in AI '99: medical diagnosis and prediction systems [Arroyo-Figueroa and Sucar 1999; van der Gaag et al. 1999]; requirements engineering [Barry and Laskey 1999]; intelligent buildings [Boman et al. 1999]; auctions [Boutilier et al. 1999]; intelligent email handling and receiver alerting [Horvitz et al. 1999]; poker [Korb et al. 1999]; educational assessment [Mislevy et al. 1999]; video advising (choosing the right video to watch tonight) [Nguyen and Haddawy 1999]; dependability analysis [Portinale and Bobbio 1999]; robot location [Shatkay 1999], and waste management [Welch and Smoth 1999]. (We apologize to any authors from that conference whose applications we have missed.)

While we show conclusively that simply compressing the representation of an MDP or POMDP does not buy more computational power, many of the recent applications have been in the form of these so-called "factored" representations, where different parameters of the states are represented as distinct variables, and actions are represented in terms of the dependence of each variable on others. It has been argued that people do not construct large models without considerable semantic content for the states. One way to represent this content is through such variables. Although there seems to be much more algorithmic development for "flat" or straightforward representations, it seems that there are more examples of systems modeled by compressed or factored representations. Thus, a very fruitful area of research is to develop policy-finding algorithms that take advantage of such factored representations. (For a survey of the state of the art, see [Boutilier et al. 1999].)

Results overview. Our main result for policy evaluation problems (Theorem 4.5) shows that the general problem is complete for the class PL, probabilistic logarithmic space. Problems in this class can be computed in polynomial time, but more efficiently in parallel than problems in P generally can. The problems we show to be PL-complete were already known to be in P [Papadimitriou and Tsitsiklis 1987] under a slightly different formulation; our work yields additional information about their inherent complexity. This yields PP-completeness (Theorem 4.9) and PSPACE-completeness (Theorem 5.2) for succinctly represented POMDPs.

For policy existence problems, in most cases, we show that they are complete for classes thought or known to be above P, such as NP, PSPACE, and even as high as EXPSPACE. Boutilier, Dearden, and Goldszmidt [1995] conjectured that finding optimal policies for structured POMDPs is infeasible. We prove this conjecture by showing that in many cases the complexity of our decision problems increases exponentially if succinct descriptions for POMDPs are considered. For example, policy existence problems for POMDPs with nonnegative rewards are NL-complete under straightforward descriptions. Using succinct descriptions, the completeness increases to PSPACE. We also consider a new intermediate notion of compressed representations and get intermediate complexity results, e.g. NP-completeness for the above example. We observe that there is no general pattern which determines the complexity trade-offs between different restrictions. For example, we compare the complexity of policy existence problems for POMDPs with nonnegative rewards to that of POMDPs with both positive and negative rewards. Whereas for general (history-dependent) policies the complexity contrasts NL completeness (Corollary 4.19) and PSPACE completeness (Theorem 4.17), for stationary policies both problems are complete for NP (Theorems 4.13 and 4.20).

In Theorems 6.2, 4.28 and 4.29 we prove $\mathrm{NP}^{\mathrm{PP}}$ completeness of several policy existence problems. In spite of its strange looking definition, $\mathrm{NP}^{\mathrm{PP}}$ turns out to be a natural class between the Polynomial Time Hierarchy [Toda 1991] and PSPACE which deserves further investigation.

Problems that we show to be PSPACE- or $\mathrm{NP}^{\mathrm{PP}}$-complete are therefore computable in polynomial space. Problems in EXP, NEXP, or EXPSPACE are expected or known to not have polynomial *space* algorithms, an apparently more stringent condition than not having polynomial time algorithms. Thus, the immediate contributions this paper makes to the fields of control theory, economics, medicine, etc. are largely negative: it is unlikely that there are efficient algorithms for finding *the optimal* policies for the general POMDP problems.

However, there is good news in this bad news. Besides classifying the hardness of known hard problems, our work has useful corollaries. Once the corresponding decision problem is shown to be hard for a particular complexity class, the known approximation heuristics for equi-complex optimization problems can be applied. For instance, there are many NP-optimization heuristics (see [Hochbaum 1997] for a reasonable introduction) used in practice, and a growing body of heuristics for PSPACE-hard optimization problems. There have been a variety of algorithms proposed for problems we now know are complete for $\mathrm{NP}^{\mathrm{PP}}$, but until recently [Majercik and Littman 1998a; Majercik and Littman 1998b], there were no heuristics that explicitly addressed these optimization problems. Because $\mathrm{NP}^{\mathrm{PP}}$-complete problems seem to be widespread, at least in AI contexts, we expect and hope to see more heuristics for these problems in the near future.

In addition to pointers to optimization heuristics, exact complexity results often give information about resource bound tradeoffs. For instance, Papadimitriou and Tsitsiklis [1987] compared P-completeness results for MDP policy existence problems with faster parallel algorithms for some deterministic variants. The parallel algorithms were NC algorithms, i.e., required polynomially many processors and poly-log time in the size of the input. As Papadimitriou and Tsitsiklis note, most researchers consider it unlikely that P-complete problems have such algorithms (unless $\mathrm{P} = \mathrm{NC}$). Thus, we expect that any P-complete problems will not have poly-log time parallel algorithms that run on at most polynomially many processors.

As we have observed, our results in some cases differ from others because we consider slightly different decision problems. Another difference between our work and many others' is the particular formalization we have chosen for representing succinctly representable POMDPs.

Most of the completeness results of this paper are summarized in Tables 1–3. The history-dependent cases for fully-observable and UMDPs have been omitted, since those cases are identical to the time-dependent ones.

There are two recent surveys on the complexity of MDPs and of stochastic control, respectively, that are related to this work. The article by Littman, Dean, and Kaelbling [1995] surveys the complexity of algorithms for fully-observable MDPs;

The complexity of policy evaluation

encoding, horizon, and observability properties

Table 1	flat (short horizon)	compressed (short horizon)	compressed (long horizon)
	unobservable		
stationary	PL	PP	PSPACE
time-dependent	PL	PP	n/a
	fully-observable or partially-observable		
stationary	PL	n/a	n/a
time-dependent	PL	n/a	n/a

policy types

The complexity of policy existence

Table 2	flat (short horizon)	compressed (short horizon)	compressed (long horizon)
	unobservable		
stationary	PL	NP^{PP}	PSPACE
time-dependent	NP	NP^{PP}	NEXP
	fully-observable		
stationary	(P-hard)	(PSPACE-hard)	(EXP-hard)
time-dependent	P	PSPACE	EXP
	partially-observable		
stationary	NP	NEXP	NEXP
time-dependent	NP	NEXP	NEXP
history-dependent	PSPACE	PSPACE	EXPSPACE

The complexity of policy existence with nonnegative rewards

Table 3	flat (short horizon)	compressed (short horizon)	compressed (long horizon)
	unobservable or fully-observable		
any policy type	NL	NP	PSPACE
	partially-observable		
stationary	NP	NP	NEXP
time-dependent	NL	NP	PSPACE
history-dependent	NL	NP	PSPACE

Fig. 1. Summary of complexity results

Blondel and Tsitsiklis [1998] article, discusses both the complexity of algorithms and of the underlying problems, but the section on POMDPs does not go into the level of detail provided here.

The following Section 2 presents a short overview of the complexity classes we use, and gives the formal definitions of partially-observable Markov Decision Processes, their different specification parameters, and the related decision problems. The subsequent Section 3 overviews complexity results for POMDPs and some variants from the literature. In each of the next two sections, we consider policy evaluation and existence problems: Section 4 presents the complexity results for flat and compressed POMDPs with horizon bounded by the size of the POMDP representation; Section 5 presents the complexity results for compressed POMDPs with horizon bounded exponentially by the size of the POMDP representation. In Section 6.1, we also consider the question, does a given POMDP have a policy with expected reward *exactly* r?

2. DEFINITIONS AND PRELIMINARIES

2.1 Complexity Classes

For definitions of complexity classes, reductions, and standard results from complexity theory we refer to [Papadimitriou 1994]. In the interest of completeness, in this section we give a short description of the complexity classes we use in this work.

It is important to note that most of the classes we consider are *decision classes.* This means that the problems in these classes are "yes/no" questions. Thus, for instance, the traveling salesperson problem is in NP in the form, "Is there a TSP tour within budget b?" The question of finding the *best* TSP tour for a graph is technically not in NP, although the decision problem can be used to find the optimal value via binary search. Although there is an optimization class associated with NP, there are not common optimization classes associated with all of the decision classes we reference. Therefore, we have phrased our problems as decision problems in order to use known complete problems for these classes.

The sets decidable in polynomial time form the class P. Decision problems in P are often said to be "tractable," or more commonly, problems that cannot be solved in polynomial time are said to be intractable. A standard complete problem for this class is the *circuit value problem (CVP):* given a Boolean circuit and an input, does the circuit output a 1?

The class EXP consists of sets decidable in exponential time. This is the smallest class considered in this paper which is known to properly contain P. The complete set we use for EXP is the *succinct circuit value problem (sCVP).* This is a similar problem to CVP, except that the description of the circuit is given in terms of two (much smaller) circuits, one that takes as input i and j and outputs a 1 if and only if i is the parent of j, and a second circuit that takes as input i and a gate type t, and outputs 1 if and only if i is of type t (i.e., AND, OR, NOT, or INPUT 1 or INPUT 0).

The nondeterministic variants of P and EXP are NP and NEXP. The complete sets used here are primarily 3SAT (is this Boolean formula in 3-CNF form satisfiable) and succinct SAT (the formula is again given by a circuit).

The class NL is the class of sets which are decidable by nondeterministic logarithmically space-bounded Turing machines. Such machines have limits on the space used as "scratch space," and have read-only input and write-only output. Complete problems for NL are, e.g., the reachability problem for directed graphs and the satisfiability problem for Boolean formulae in conjunctive normal form with at most two literals in each clause.

Nondeterministic computation is essential for the definitions of probabilistic and counting complexity classes. The class #L [Àlvarez and Jenner 1993] is the class of functions f such that, for some nondeterministic logarithmically space-bounded machine N, the number of accepting paths of N on x equals $f(x)$. The class #P is defined analogously as the class of functions f such that, for some nondeterministic *polynomial-time*-bounded machine N, the number of accepting paths of N on x equals $f(x)$.

Probabilistic logspace, PL, is the class of sets A for which there exists a nondeterministic logarithmically space-bounded machine N such that $x \in A$ if and only if the number of accepting paths of N on x is greater than its number of rejecting paths. In apparent contrast to P-complete sets, sets in PL are decidable using very fast parallel computations [Jung 1985]. The set of integer matrices with determinant greater than 0 is complete for PL [Jung 1984]. (See also [Allender and Ogihara 1996].) Probabilistic polynomial time, PP, is defined analogously. A classic PP-complete problem is MAJSAT: given a Boolean formula, do the majority of assignments satisfy it?

For polynomially space-bounded computations, PSPACE equals probabilistic PSPACE, and #PSPACE (defined analogously to #L and #P) is the same as the class of polynomial-space-computable functions [Ladner 1989]. Note that some functions in #PSPACE produce output of exponential length. The most common PSPACE-complete problem is QBF, the set of true quantified Boolean formulas with no free variables. Another variant that is also complete for PSPACE is stochastic satisfiability, where QBF-style formulas alternate existential and random quantifiers.

For any complexity classes $\mathcal{C}$ and $\mathcal{C}'$, the class $\mathcal{C}^{\mathcal{C}'}$ consists of those sets that are $\mathcal{C}$-*Turing reducible* to sets in $\mathcal{C}'$, i.e., sets that can be accepted with resource bounds specified by $\mathcal{C}$, using some problem in $\mathcal{C}'$ as a subroutine (oracle) with instantaneous output. For any class $\mathcal{C} \subseteq \mathrm{PSPACE}$, it is the case that $\mathrm{NP}^{\mathcal{C}} \subseteq \mathrm{PSPACE}$, and therefore $\mathrm{NP}^{\mathrm{PSPACE}} = \mathrm{PSPACE}$; see Papadimitriou's textbook [Papadimitriou 1994].

Another useful result is that the class $\mathrm{NP}^{\mathrm{PP}}$ equals the "$\leq_m^{\mathrm{NP}}$" closure of PP [Torán 1991], which can be seen as the closure of PP under polynomial-time disjunctive reducibility with an exponential number of queries (each of the queries computable in polynomial time from its index in the list of queries). Hence, as for each member of an NP set there is a short certificate of its membership which can be checked in polynomial-time, for each member of an $\mathrm{NP}^{\mathrm{PP}}$ set there is a short certificate of its membership which can be checked in probabilistic polynomial time.

For the complexity classes mentioned, the following inclusions hold.

$$\mathrm{NL} \subseteq \mathrm{PL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PP} \subseteq \mathrm{NP}^{\mathrm{PP}} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{NEXP} \subseteq \mathrm{EXPSPACE}$$

$$\mathrm{NL} \subseteq \mathrm{PL} \subset \mathrm{PSPACE} = \mathrm{NPSPACE} \subset \mathrm{EXPSPACE}; \quad \mathrm{P} \subset \mathrm{EXP}; \quad \mathrm{NP} \subset \mathrm{NEXP}$$

2.2 Partially-Observable Markov Decision Processes

A partially-observable Markov Decision Process (POMDP) describes a controlled stochastic system by its states and the consequences of actions on the system. It is denoted as a tuple $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, t, o, r)$, where

—$\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ are finite sets of *states*, *actions* and *observations*,

—$s_0 \in \mathcal{S}$ is the *initial state*,

—$t : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the *state transition function*, where $t(s, a, s')$ is the probability that state s' is reached from state s on action a (where $\Sigma_{s' \in \mathcal{S}} t(s, a, s') \in \{0, 1\}$ for every $s \in \mathcal{S}, a \in \mathcal{A}$),

—$o : \mathcal{S} \to \mathcal{O}$ is the *observation function*, where $o(s)$ is the observation made in state s, and

—$r : \mathcal{S} \times \mathcal{A} \to \mathbf{Z}$ is the *reward function*, where $r(s, a)$ is the reward gained by taking action a in state s.

If states and observations are identical, i.e. $\mathcal{O} = \mathcal{S}$ and o is the identity function (or a bijection), the POMDP is called *fully-observable* and denoted as MDP. Another special case is the *unobservable* POMDPs, where the set of observations contains only one element; i.e. in every state the same observation is made and therefore the observation function is constant. We denote unobservable POMDPs as UMDPs. Without restrictions on the observability, a POMDP is called *partially-observable*.[2]

Many algorithms, especially for POMDPs, assume that the initial state, or a probability distribution over states, is given with the description of the POMDP and policy. There is a disagreement amongst practitioners over whether it is valid to assume such an initial distribution, or to allow it to vary. We fall firmly on the side of a known initial state in this work, without directly addressing the philosophical or computational ramifications of this choice.

2.3 Policies and Performances

A policy describes how to act depending on observations. We distinguish three types of policies.

—A *stationary policy* π_s (for M) is a function $\pi_s : \mathcal{O} \to \mathcal{A}$, mapping each observation to an action.

—A *time-dependent policy* π_t is a function $\pi_t : \mathcal{O} \times \mathbf{N} \to \mathcal{A}$, mapping each $\langle$observation, time$\rangle$ pair to an action.

—A *history-dependent policy* π_h is a function $\pi_h : \mathcal{O}^* \to \mathcal{A}$, mapping each finite sequence of observations to an action.

Notice that for a UMDP, a history-dependent policy is equivalent to a time-dependent one. Thus, our theorems do not explicitly address history-dependent policies for UMDPs.

[2]Note that making observations probabilistically does not add any power to POMDPs. Any probabilistically observable POMDP is equivalent to one with deterministic observations which is only polynomially larger. Thus, for the purpose of complexity analysis we can ignore this variant.

Let $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, t, o, r)$ be a POMDP. A *trajectory θ for M* is a finite sequence of states $\theta = \sigma_1, \sigma_2, \ldots, \sigma_m$ ($m \geq 0$, $\sigma_i \in \mathcal{S}$). The probability $\mathrm{prob}_\pi(M, \theta)$ of a trajectory $\theta = \sigma_1, \sigma_2, \ldots, \sigma_m$ under policy π is

—$\mathrm{prob}_\pi(M,\theta) = \Pi_{i=1}^{m-1} t(\sigma_i, \pi(o(\sigma_i)), \sigma_{i+1})$, if π is a stationary policy,

—$\mathrm{prob}_\pi(M,\theta) = \Pi_{i=1}^{m-1} t(\sigma_i, \pi(o(\sigma_i), i), \sigma_{i+1})$, if π is a time-dependent policy, and

—$\mathrm{prob}_\pi(M,\theta) = \Pi_{i=1}^{m-1} t(\sigma_i, \pi(o(\sigma_1) \cdots o(\sigma_i)), \sigma_{i+1})$, if π is a history-dependent policy.

The *reward* $\mathrm{rew}_\pi(M, \theta)$ of trajectory θ under policy π is the sum of its rewards, i.e. $\mathrm{rew}_\pi(M,\theta) = \Sigma_{i=1}^{m-1} r(\sigma_i, \pi(\cdot))$ dependent on the type of policy. The *performance of a policy* π for finite horizon k with initial state σ is the expected sum of rewards received on the next k steps by following the policy π, i.e. $\mathrm{perf}(M, \sigma, k, \pi) = \sum_{\theta \in \Theta(\sigma,k)} \mathrm{prob}_\pi(M,\theta) \cdot \mathrm{rew}_\pi(M,\theta)$, where $\Theta(\sigma, k)$ is the set of all length k trajectories beginning with state σ.

2.4 Representations and Computational Questions

In this discussion, we will use "POMDP" to refer to any form of MDP, UMDP, or POMDP.

There are various ways a POMDP can be represented. We begin by considering problem instances that are represented in a straightforward way, which we call "flat POMDPs". Namely, a POMDP with m states is represented by a set of $m \times m$ tables for the transition function – one table for each action – and similar tables for the reward function and for the observation function. Encodings of UMDPs and of MDPs may omit the observation function.

For a flat POMDP M we let $|M|$ denote the number of bits used to write down M's tables. Note that the number of states and the number of actions of M is at most $|M|$. In the same way, a stationary policy π can be encoded as a list with at most $|M|$ entries, each entry of at most $|M|$ bits. A time-dependent policy for horizon k can be encoded as a table with at most $k \times |M|$ entries each of at most $|M|$ bits. Note that for a POMDP M, any stationary policy and any time-dependent policy for horizon $|M|$ can be specified with at most $|M|^3$ bits.

The standard computation problems associated with MDPs are the computation of optimal or nearly optimal policies. However, our analysis requires the computation of Boolean functions. Papadimitriou and Tsitsiklis [1987] considered the question for MDPs with nonpositive rewards, of whether there was a policy with expected performance greater than 0. We consider slightly different MDPs and questions.

Let XMDP be any of POMDP, MDP, and UMDP. The *α policy evaluation problem for XMDPs* is the set of all pairs (M, π) consisting of a XMDP $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, o, t, r)$ and an α policy π with performance $\mathrm{perf}(M, s_0, |M|, \pi) > 0$.

The *α policy existence problem for XMDPs* is the set of all XMDPs $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, o, t, r)$, for which there exists an α policy π with positive performance, i.e. $\mathrm{perf}(M, s_0, |M|, \pi) > 0$.

There are $|\mathcal{O}|^h$ possible histories for a POMDP M with time horizon h. If $|\mathcal{O}| > 1$, this means that one cannot fully specify a history-dependent policy with horizon $|M|$ in space polynomial in the size of M. Therefore, we do not generally consider the history-dependent policy evaluation problem in this paper. When we assume that $h \leq |M|$, time-dependent policies do not cause such problems.

When the system being modelled has sufficient structure, there is no need to represent the POMDP by complete tables. This is true, for instance, when a system is modelled as a Bayes belief net, or when actions are given in the STRIPS model [Bylander 1994; Goldsmith et al. 1997; Littman 1996b].

Changing the way in which POMDPs are represented may change the complexities of the considered problems too. In this section we consider two circuit-based models. The first, which we call *succinct representations,* is a variant of a model introduced independently in [Galperin and Wigderson 1983; Wagner 1986] to model other highly structured problems. A succinct representation of a string t of length n is a Boolean circuit with $\lceil \log n \rceil$ input gates and one output gate which on input i outputs the ith bit of t. Note that any string of length n can have such a circuit of size $O(n)$, and the smallest such circuit requires $O(\log n)$ bits to simply represent i. The transition tables for a POMDP can be represented succinctly in a similar way by a circuit which on input (s, a, s', i) outputs the ith bit of the transition probability $t(s, a, s)$.

The process of extracting a probability one bit at a time is less than ideal in some cases. The advantage of such a representation is that the number of bits for the represented probability can be exponentially larger than the size of the circuit. If we limit the number of bits of the probability, we can use a related representation we call *compressed.* (The issue of bit-counts in transition probabilities has arisen before; it occurs, for instance, in [Beauquier et al. 1995; Tseng 1990]. It is also important to note that our probabilities are specified by single bit-strings, rather than as rationals specified by two bit-strings.) We let the transition tables of a POMDP with m states and actions be represented by a Boolean circuit C with $3\lceil \log m \rceil$ input bits such that $C(s, a, s')$ outputs $t(s, a, s')$. Note that such a circuit has many output gates. Encodings by circuits are no larger than the straightforward table encodings, but for POMDPs with sufficient structure the circuit encodings may be much smaller, namely $O(\log m)$, i.e., the logarithm of the size of the state space. (It takes $\lceil \log m \rceil$ bits to specify a state as input to the circuit.)

It is easy to see that in going from straightforward to compressed encodings, the complexity of the corresponding decision problems increases at most exponentially. More importantly (and less obviously), there are many problems for which this exponential increase in complexity is inherent (see [Balcázar et al. 1992] for general conditions for such an increase).

Note that there are apparently similar models discussed in the reinforcement learning literature. In those models, however, the circuits take as input the initial state and action and output at random another state with probability equal to the probability of that transition. (A so-called "random circuit" is in fact a deterministic circuit that takes an additional set of inputs which are chosen uniformly at random; the given input plus the random inputs determine the output.) Although these two models are not apparently polynomial-time equivalent (we know of no easy transformation from one to the other), the proofs given in this paper can easily be transformed: membership proofs can be modified to show that the corresponding problems for their representations are in the same complexity classes as ours, and reductions from complete problems produce simple POMDPs that can as easily be represented in their format as ours. Given the vast array of succinct representations considered, ours was chosen for its familiarity to complexity theorists. However, this does not prevent it from yielding information about problems under other representations. See, for instance, [Littman et al. 1998] for an example of related work on planning domains inspired by the work described here.

The *α policy evaluation problem for compressed (succinct) XMDPs* is the set of all pairs (M, π) consisting of a compressed (succinctly) encoded XMDP $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, o, t, r)$ and an α policy π, such that $\text{perf}(M, s_0, |M|, \pi) > 0$.

The policy existence problems are expressed similarly.

3. SKETCH OF PREVIOUS COMPLEXITY RESULTS

Given the widespread use and consideration of POMDPs, there has been surprisingly little work on the computational complexity of finding good policies. There have certainly been algorithms presented and analyzed, but we will not generally survey them here. For that, we recommend [Lovejoy 1991; Puterman 1994] as a starting place.

The complexity of POMDPs was first considered by Papadimitriou and Tsitsiklis [1986, 1987]. They considered POMDPs with nonpositive rewards (costs) and investigated the complexity of the decision problem: given a POMDP, does there exist a time-dependent or history-dependent policy with expected sum of rewards *equal to* 0? Their maximization problem can be stated as a minimization problem for POMDPs with nonnegative rewards.

They showed an NP-completeness result for the stationary, finite-horizon case [Papadimitriou and Tsitsiklis 1986, p. 645]. For time-dependent policies they showed that the policy existence problem for (fully-observable) MDPs is P-complete (for finite or infinite horizon). For history-dependent policies they proved that the policy existence problem for POMDPs is PSPACE-complete and for UMDPs is NP-complete [Papadimitriou and Tsitsiklis 1987].

Note that the minimization problem investigated by Papadimitriou and Tsitsiklis cannot be used in a binary search to determine the maximal performance of a POMDP. (The use of such a search depends on both positive and negative rewards.) Our work extends theirs significantly, both in the problems considered, and in the scope of the results. Because of the subtle differences between the decision problems, our complexity results on similar problems sometimes differ greatly from the results in [Papadimitriou and Tsitsiklis 1987]. For example, our decision problem for POMDPs with nonnegative rewards under history-dependent policy is NL-complete. (Cf. Corollary 4.19, compare to PSPACE-completeness in [Papadimitriou and Tsitsiklis 1987, Theorem 6].) Unobservable POMDPs were also considered in [Burago et al. 1996]. Beauquier, Burago, and Slissenko [1995] considered different optimality criteria, and thus their results are not directly comparable with ours.

Most of the related AI papers, especially [Bäckström 1995; Bylander 1994; Chapman 1987; Erol et al. 1996; Erol et al. 1995], consider computationally simpler problems without probabilistic transitions. Thus, the complexity of their problems is generally lower than of ours (except [Goldsmith et al. 1997; Littman 1996b] and one theorem in [Bylander 1994]). Many of the results are for succinctly represented systems, described by 2-Phase Temporal Bayes Nets [Boutilier et al. 1995;

Boutilier et al. 1995], sequential-effects-tree representation (ST) [Littman 1997a], probabilistic state-space operators (PSOs) [Kushmerick et al. 1995], or other related representations. For these problems, they consider the complexity of finding policies that are of size polynomial in either the size of the representation, or the number of states — leading to different complexities, in certain cases. (For instance, Littman showed that the general plan existence problem is EXP-complete, but if plans are limited to size polynomial in the size of the planning domain, then the problem is PSPACE-complete [Littman 1997a]; the PSPACE-completeness was proved independently, and published in [Goldsmith et al. 1997].) Sometimes the stochasticity of the system does not add computational complexity. For instance, Bylander [1994] showed that the time-dependent policy existence problem for succinct UMDPs with nonnegative rewards is PSPACE-complete, even if the transitions are restricted to be deterministic. In Theorem 5.10 we show PSPACE-completeness of the stochastic version of this problem.

4. SHORT TERM HORIZON

Consider a model of emergency medical services. On television, people seem to only spend an hour (minus commercial breaks) in the emergency room. In real life, most patients are there somewhat longer. However, emergency medical treatment is supposed to be fast and short-term, just enough for basic diagnosis and to help the patient last until longer-term care is available.

In this section we consider policies for POMDPs with horizon polynomial in the size of the representation. Hence, for flat POMDPs the horizon is — roughly speaking — bounded by a polynomial in the size of the state space, and for compressed or succinct POMDPs it is bounded logarithmically in the size of the state space.

4.1 Policy Evaluation

The policy evaluation problem asks whether a given POMDP M has performance greater than 0 under a given policy π after $|M|$ steps. Using binary search, an algorithm for this problem can be used to determine the exact performance of a POMDP under a policy. For the evaluation problem, the observability of the POMDP does not matter. Therefore, we state the results only for general partially-observable Markov decision processes.

4.1.1 *Flat Representations.* We consider the complexity of the policy evaluation problem for POMDPs under stationary and time-dependent policies. History-dependent policies are functions from all possible histories to actions, and hence have a range of size exponential in the representation size of the POMDP. Therefore, the explicit specification of a history-dependent policy is intractable, and we leave out complexity considerations of evaluating those policies. For a discussion of the complexity of evaluating compressed history-dependent policies, see [Mundhenk 1998].

The standard polynomial time algorithm for evaluating a given policy for a given POMDP uses dynamic programming [Papadimitriou and Tsitsiklis 1987; Puterman 1994]. We show that for POMDPs this evaluation can be performed quickly in parallel. Eventually this yields the policy evaluation problem being complete for PL. We begin with a technical lemma about matrix powering, and show that each entry $(T^m)_{(i,j)}$ of the mth power of a nonnegative integer square matrix T can be computed in #L, if the power m is at most the dimension n of the matrix.

Lemma 4.1. *(cf. [Vinay 1991]) Let T be an $n \times n$ matrix of nonnegative binary integers, each of length n, and let $1 \leq i, j \leq n$, $0 \leq m \leq n$. The function mapping (T, m, i, j) to $(T^m)_{(i,j)}$ is in* #L.

We review the relations between the function class #L and the class of decision problems PL. GapL is the class of functions representable as differences of #L functions, GapL $= \{g - h \mid g, h \in \#\mathrm{L}\}$, and PL is the class of sets A for which there is a GapL function f such that for every x, $x \in A$ if and only if $f(x) > 0$ (see [Allender and Ogihara 1996]). We use these results to prove the following.

Lemma 4.2. *The stationary policy evaluation problem for POMDPs is in* PL.

Proof. Let $\hat{M} = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, \hat{t}, o, \hat{r})$ be a POMDP, and let π be a stationary policy, i.e. a mapping from $\mathcal{O}$ to $\mathcal{A}$. We show that $\mathrm{perf}_s(\hat{M}, s_0, |M|, \pi) > 0$ can be decided in PL.

We transform $\hat{M}$ into a UMDP M with the same states and rewards as $\hat{M}$, having the same performance as $\hat{M}$ under policy π. Since π is a stationary policy, this can be achieved by "hard-wiring" the actions chosen by π into the POMDP. Then $M = (\mathcal{S}, s_0, \{a\}, \mathcal{O}, t, o, r)$ where $t(s, a, s') = \hat{t}(s, \pi(o(s)), s')$ and $r(s, a) = \hat{r}(s, \pi(o(s)))$. Since M has only one action, the only policy to consider is the constant function mapping each observation to that action a. Then $\hat{M}$ under policy π has the same performance as M under (constant) policy a, i.e. $\mathrm{perf}(\hat{M}, s_0, |M|, \pi) = \mathrm{perf}(M, s_0, |M|, a)$. This performance can be calculated using a recursive definition of perf, namely $\mathrm{perf}(M, i, m, a) = r(i, a) + \sum_{j \in \mathcal{S}} t(i, a, j) \cdot \mathrm{perf}(M, j, m-1, a)$ and $\mathrm{perf}(M, i, 0, a) = 0$.

The state transition probabilities are given as binary fractions of length h. In order to get an integer function for the performance, define the function p as $p(M, i, 0) = 0$ and $p(M, i, m) = 2^{hm} r(i, a) + \sum_{j \in \mathcal{S}} p(M, j, m-1) \cdot 2^h t(i, a, j)$. One can show that

$$\mathrm{perf}(M, i, m, a) = p(M, i, m) \cdot 2^{-hm} \quad .$$

Therefore, $\mathrm{perf}(M, s_0, |M|, a) > 0$ if and only if $p(M, s_0, |M|) > 0$.

In order to complete the proof using the characterization of PL mentioned above, we have to show that the function p is in GapL.

Let T be the matrix obtained from the transition matrix of M by multiplying all entries by 2^h, i.e. $T_{(i,j)} = t(i, a, j) \cdot 2^h$. The recursion in the definition of p can be resolved into powers of T, and we get

$$p(M, i, m) = \sum_{k=1}^{m} \sum_{j \in \mathcal{S}} (T^{k-1})_{(i,j)} \cdot r(j, a) \cdot 2^{(m-k+1) \cdot h} \quad .$$

Each $T_{(i,j)}$ is logspace computable from the input $\hat{M}$. From Lemma 4.1 we get that $(T^{k-1})_{(i,j)} \in \#\mathrm{L}$. The reward function is part of the input too, thus r is in GapL (note that rewards may be negative integers). Because GapL is closed under multiplication and polynomial summation (see [Allender and Ogihara 1996]), it follows that $p \in$ GapL. □

In the following hardness proof, we simulate a Turing machine computation by a POMDP which has only one action. Clearly, if there is no choice of actions, there is only one policy: take this action. Therefore, this hardness proof also applies for unobservable and fully-observable POMDPs.

LEMMA 4.3. *The stationary policy evaluation problem for POMDPs is* PL*-hard.*

PROOF. Consider $A \in \mathrm{PL}$. Then there exists a probabilistic logspace machine N accepting A, and a polynomial p such that each computation of N on x uses at most $p(|x|)$ random decisions [Jung 1985]. Now, fix some input x. We construct a UMDP $M(x)$ with only one action, which models the behavior of N on x. Each state of $M(x)$ is a pair consisting of a configuration of N on x (there are polynomially many) and an integer used as a counter for the number of random moves made to reach this configuration (there are at most $p(|x|)$ many). Also, we add a final "trap" state reached from states containing a halting configuration or from itself. The state transition function of $M(x)$ is defined according to the state transition function of N on x, so that each halting computation of N on x corresponds to a length $p(|x|)$ trajectory of $M(x)$ and vice versa. The state transition function is defined so that each halting computation of N on x corresponds to a length $p(|x|)$ trajectory of $M(x)$ and vice versa. The reward function is chosen such that $\mathrm{rew}(\theta) \cdot \mathrm{prob}(\theta)$ equals 1 for trajectories θ corresponding to accepting computations (based on the number of random moves made, as recorded in the counter), or -1 for rejecting computations, or 0 otherwise. Since $x \in A$ if and only if the number of accepting computations of N on x is greater than the number of rejecting computations, it follows that $x \in A$ if and only if the number of trajectories θ of length $|M(x)|$ for $M(x)$ with $\mathrm{rew}(\theta) \cdot \mathrm{prob}(\theta) = 1$ is greater than the number of trajectories θ with $\mathrm{rew}(\theta) \cdot \mathrm{prob}(\theta) = -1$, which is equivalent to $\mathrm{perf}(M(x), s_0, |M(x)|, a) > 0$. □

COROLLARY 4.4. *The stationary policy evaluation problem for UMDPs and for MDPs is* PL*-hard.*

THEOREM 4.5. *The stationary and time-dependent policy evaluation problems for POMDPs are* PL*-complete.*

PROOF. Completeness of the stationary case follows from the above Lemmas 4.2 and 4.3. Because every stationary policy is a time-dependent policy too, hardness of the time-dependent case follows from Lemma 4.3. It remains to show that the time-dependent case is contained in PL. Let $\hat{M} = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, \hat{t}, \hat{o}, \hat{r})$ be a POMDP, and let π be a time-dependent policy, i.e. a mapping from $\mathcal{O} \times \{0, \ldots, |\hat{M}|-1\}$ to $\mathcal{A}$. Essentially, we proceed in the same way as in the above proof. The main difference is that the transformation from $\hat{M}$ to M is more involved. We construct M by making $|\hat{M}|$ copies of $\hat{M}$ such that all transitions from the ith copy go to the $i+1$st copy, and all these transitions correspond to the transition chosen by π in the ith step. The rest of the proof proceeds as in the proof of Lemma 4.2. □

COROLLARY 4.6. *The stationary and time-dependent policy evaluation problems for UMDPs and for MDPs are* PL*-complete.*

Earlier work, for instance Papadimitriou and Tsitsiklis [1987], considered POMDPs with *nonpositive* rewards. We consider the policy evaluation problems — and later the existence problems — for POMDPs with *nonnegative* rewards, which fits better to our question for a positive performance. This is generally easier than for POMDPs with unrestricted rewards. In fact, these problems reduce in a straightforward way to graph accessibility problems. It suffices to find one trajectory with positive probability through the given POMDP that is consistent with the given policy and yields reward > 0 in at least one step. Moreover, the transition probabilities and rewards do not need to be calculated exactly. Thus, our question has a "there exists" flavor, whereas the question by Papadimitriou and Tsitsiklis [1987] mentioned above has a "for all" flavor.

THEOREM 4.7. *The stationary and time-dependent policy evaluation problems for POMDPs with nonnegative rewards are* NL*-complete.*

PROOF. A POMDP M (any observability) and a stationary policy π can be interpreted as a directed graph: the vertices are states of the POMDP, and an edge exists from s_i to s_j if and only if $t(s_i, \pi(o(s_i)), s_j) > 0$. Then $\operatorname{perf}(M, s_0, |M|, \pi) > 0$ if and only if a vertex of the set $R = \{s_i : \ r(s_i, \pi(o(s_i))) > 0\}$ is reachable from s_0. This is an instance of the standard NL-complete graph reachability problem. Because a path from s_0 to a vertex in R contains any node at most once, the same idea holds for time-dependent policies.

The graph reachability problem can easily be transformed to an existence problem for a POMDP. Any graph is transformed to a POMDP with only *one* action (all edges from a given vertex have equal probability). The source of the graph is the initial state of the POMDP, and the sink of the graph is the only state of the POMDP whose incoming edges yield reward 1 — all other rewards are 0. The POMDP has only one policy, and that policy has positive expected performance if and only if the sink node is reachable in the original graph. Since there is only one possible policy, observability is irrelevant. □

4.1.2 *Compressed Representations.* In this section we focus on the complexity of stationary and time-dependent policies. Even for flat POMDPs, a history-dependent policy specification is intractable, i.e. it takes exponentially many bits in the POMDP description. (This problem can be addressed either by computing only the *value* of the policy, or by considering only *succinctly represented* policies. However, those approaches are beyond the scope of this paper.)

Because the number of observations may be exponential in the representation size of a compressed POMDP, specifying a stationary policy is intractable. Moreover, an evaluation problem consisting of pairs of POMDPs and a very large policy description has "pathologically" very low complexity, since the complexity is related to the size of those pairs. The only exception are policies for compressed UMDPs. Each stationary policy consists of one action only (i.e. a mapping from the only observation class to an action), and each time-dependent policy consists of a sequence of actions. Those policies can be specified using only polynomially many bits in the description size of the compressed UMDP and hence do not "pathologically" decrease the complexity of evaluation problems. The complexity for these problems is intermediate between those for flat POMDPs and for succinct POMDPs.

Remember that #P is the class of functions f for which there exists a nondeterministic-polynomial-time bounded Turing machine N such that $f(x)$ equals the number of accepting computations of N on x, and that $\mathrm{FP} \subseteq \#\mathrm{P} \subseteq \mathrm{FPSPACE}$.

The matrix powering problem for integer matrices with "small" entries is in #P, using the same proof idea as for Lemma 4.1.

LEMMA 4.8. *Let T be a $2^m \times 2^m$ matrix of nonnegative integers, each consisting of m bits. Let T be represented by a Boolean circuit C with $2m + \lceil \log m \rceil$ input gates, such that $C(a, b, r)$ outputs the r-th bit of $T_{(a,b)}$. For $1 \le i, j \le 2^m$, and $0 \le s \le m$, the function mapping (C, s, i, j) to $(T^s)_{(i,j)}$ is in* #P.

As a consequence, the complexity of the policy evaluation problems can be shown to be intermediate between NP and PSPACE.

THEOREM 4.9. *The stationary and time-dependent policy evaluation problems for compressed UMDPs are* PP*-complete.*

The proof, which relies on Lemma 4.8, is similar to that of Theorem 4.5.

For those readers who feel adrift in complexity classes, we note that PP is apparently much more powerful than NP, although it is contained in PSPACE. Once again, we see that limiting the type of rewards can considerably simplify the computation.

THEOREM 4.10. *The stationary and time-dependent policy evaluation problems for compressed UMDPs with nonnegative rewards are* NP*-complete.*

PROOF. Remember that the horizon is roughly the size of the input, so that a trajectory can be guessed in polynomial time and then checked for consistency, positive probability, and positive reward.

The proof of NP-hardness is a reduction from SAT. Given a Boolean formula ϕ with m variables, we create an MDP with $2^m + 2$ states $\{s_0, s_1\} \cup \{s_w \mid w \in \{0,1\}^m\}$ and one action, a. The state s_0 is the initial state, state s_1 is a final sink state, and each of the other 2^m states represents an assignment to the m variables. From s_0, all of these 2^m states are accessed on action a, each with equal probability 2^{-m} and without reward. In each of the "assignment states," reward 1 is obtained on action a, if the assignment satisfies ϕ, and reward 0 is obtained otherwise. From each of the "assignment states," the sink state is accessed with probability 1. The process remains in the sink state without rewards. The unique policy $\pi = a$ has expected reward > 0 if and only if there is a satisfying assignment for ϕ. □

4.2 Policy Existence

The policy existence problem asks whether a given POMDP M has positive performance under any policy with horizon $|M|$. Unlike policy evaluation, it is feasible to consider whether a good enough history-dependent policy exists. We will see that in order to answer the existence problem for history-dependent policies, it is not necessary to specify a policy, and therefore the problem does not become automatically intractable. Instead, structural properties of POMDPs are used. Also unlike the policy evaluation problem, the observability of the POMDP has an effect on the complexity. Hence, we have (at least) three parameters — type of policy, type of observability, type of representation — which determine the complexity. The complexity trade-offs between these parameters are interesting, and it is important for a system designer to know them.

4.2.1 *Flat Representations.* The computation of optimal policies for fully-observable MDPs is a well-studied optimization problem. The maximal performance of any infinite-horizon stationary policy for a fully-observable Markov decision process can be solved by linear programming techniques in polynomial time. Dynamic programming can be used for the finite horizon time-dependent and history-dependent policy cases.

THEOREM 4.11. *The time-dependent and history-dependent policy existence problems for MDPs are* P*-complete.*

Because the proof is a straightforward modification of the proof of Theorem 1 in [Papadimitriou and Tsitsiklis 1987], we omit it here.

The exact complexity of the policy existence problem for stationary, fully-observable MDPs with finite horizon is not known. From [Papadimitriou and Tsitsiklis 1987] it follows that it is P-hard, and it is easily seen to be in NP.

THEOREM 4.12. *The stationary policy existence problem for MDPs is* P*-hard and in* NP*.*

Let us for a moment consider policy existence problems with different parameters on observability, representation, or rewards and how the type of policy influences the complexity. We will see that there is no pattern which allows one to draw conclusions for the fully-observable case.

For instance, for UMDPs and for stationary policies, the evaluation and the existence problem are of the same complexity, whereas this is not the case for time-dependent policies. For fully-observable and partially-observable MDPs, the policy existence problems have the same complexity for both stationary and time-dependent policies.

THEOREM 4.13. *The stationary policy existence problem for POMDPs is* NP*-complete.*

PROOF. Membership in NP is straightforward, because a policy can be guessed and evaluated in polynomial time. To show NP-hardness, we reduce 3SAT to it. Let $\phi(x_1, \ldots, x_n)$ be a formula with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$, where clause $C_j = (l_{v(1,j)} \vee l_{v(2,j)} \vee l_{v(3,j)})$ for $l_i \in \{x_i, \neg x_i\}$. We say that variable x_i *0-appears (resp. 1-appears) in* C_j if $\neg x_i$ (resp. x_i) is a literal in C_j. Without loss of generality, we assume that every variable appears at most once in each clause. The idea is to construct a POMDP $M(\phi)$ having one state for each appearance of a variable in a clause. The set of observations is the set of variables. Each action corresponds to an assignment of a value to a variable. The transition function is deterministic. The process starts with the first variable in the first clause. If the action chosen in a certain state satisfies the corresponding literal, the process proceeds to the first variable of the next clause, or with reward 1 to the final state, if all clauses were considered. If the action does not satisfy the literal, the process proceeds to the next variable of the clause, or with reward 0 to the final state. The partition of the state space into observation classes guarantees that the same assignment is made for every appearance of the same variable. Therefore, the maximal performance of $M(\phi)$ equals 1 if ϕ is satisfiable, and it equals 0 otherwise.

Formally, from ϕ, we construct a POMDP $M(\phi) = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, t, o, r)$ with

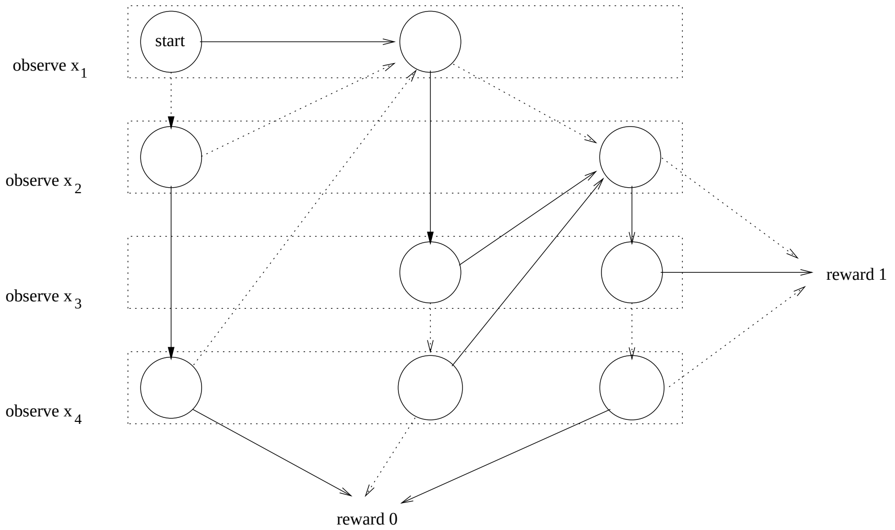

Fig. 2. $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$.

$$
\begin{aligned}
\mathcal{S} &= \{(i,j) \mid 1 \leq i \leq n, 1 \leq j \leq m\} \cup \{F, T\} \\
s_0 &= (v(1,1),1), \qquad \mathcal{A} = \{0,1\}, \qquad \mathcal{O} = \{x_1, \ldots, x_n, F, T\} \\
t(s,a,s') &= \begin{cases}
1, & \text{if } s = (v(i,j),j), s' = (1, j+1), j < m, 1 \leq i \leq 3, \\
 & \qquad \text{and } x_{v(i,j)} \text{ } a\text{-appears in } C_j \\
1, & \text{if } s = (v(i,m),m), s' = T, 1 \leq i \leq 3, \\
 & \qquad \text{and } x_{v(i,m)} \text{ } a\text{-appears in } C_m \\
1, & \text{if } s = (v(i,j),j), s' = (v(i+1,j),j), 1 \leq i < 3, \\
 & \qquad \text{and } x_{v(i,j)} \text{ } (1-a)\text{-appears in } C_j \\
1, & \text{if } s = (v(3,j),j), s' = F, \\
 & \qquad \text{and } x_{v(3,j)} \text{ } (1-a)\text{-appears in } C_j \\
1, & \text{if } s = s' = F \text{ or } s = s' = T \\
0, & \text{otherwise}
\end{cases} \\
r(s,a) &= \begin{cases} 1, & \text{if } t(s,a,T) = 1, s \neq T \\ 0, & \text{otherwise} \end{cases}, \qquad o(s) = \begin{cases} x_i, & \text{if } s = (i,j) \\ T, & \text{if } s = T \\ F, & \text{if } s = F \end{cases}.
\end{aligned}
$$

Note that all transitions in $M(\phi)$ are deterministic, and every trajectory has reward 0 or 1. Each assignment to ϕ can be interpreted as a stationary policy for $M(\phi)$ and vice versa: the value assigned to a variable equals the action assigned to an observation. Policies under which $M(\phi)$ has performance 1 correspond to satisfying assignments for ϕ, and vice versa. Therefore, ϕ is satisfiable if and only if there exists a stationary policy under which $M(\phi)$ has performance > 0.

Figure 2 sketches the construction of $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$. Solid lines indicate transitions on action 1, and dotted lines transitions on action 0. All actions have reward 0 unless reward 1 is indicated. □

Note that Theorem 4.13 is in stark contrast with the result in [Madani et al. 1999] that finding an optimal history-dependent, infinite horizon policy for a POMDP is not a computable problem. There are two significant differences between that undecidable problem and our NP-complete problem: we consider stationary policies and finite horizons. Either significantly simplifies the problem. Thus an "NP-complete" problem begins to look almost tractable! Unfortunately, stationary policies for POMDPs can be arbitrarily far from optimal.

For UMDPs, the stationary policy existence problem is as hard as the policy evaluation problem.

THEOREM 4.14. *The stationary policy existence problem for UMDPs is* PL-*complete.*

PROOF. First we show that the problem is in PL. Let M be a UMDP with set of actions $\mathcal{A}$. Because M is unobservable, every policy for M is a constant function $a \in \mathcal{A}$. Then there exists a policy under which M has performance greater than 0 if and only if for some $a \in \mathcal{A}$, M under a has performance greater than 0. Thus the stationary policy existence problem for UMDPs logspace disjunctively reduces to the stationary policy *evaluation* problem for UMDPs. From Lemma 4.2 and the closure of PL under logspace disjunctive reductions (see [Allender and Ogihara 1996]), it follows that the policy existence problem is in PL.

In order to show PL-hardness, note that for POMDPs with only one action, there is no difference between the complexity of the policy evaluation problem and that of the policy existence problem. In the proof of Lemma 4.3, every PL computation was logspace reduced to a UMDP with exactly one action only. This proves PL-hardness of the policy existence problem. □

A time-dependent policy for a UMDP allows one to choose one action for each step, instead of one action which is performed on all steps of a stationary policy. This alters the complexity of the policy existence problem from PL to NP (assuming those classes are distinct).

THEOREM 4.15. *The time-dependent policy existence problem for UMDPs is* NP-*complete.*

Papadimitriou and Tsitsiklis proved a similar theorem [Papadimitriou and Tsitsiklis 1987]. Their POMDPs had only non-positive rewards, and their formulation of the decision problem was whether there is a policy with reward 0. However, our result can be proven by a proof very similar to theirs, by showing a reduction from 3SAT.

PROOF. That it is in NP follows from the fact that a policy with performance > 0 can be guessed and checked in polynomial time. NP-hardness follows from the following reduction from 3SAT. At the first step, a clause is chosen randomly. At step $i + 1$, the assignment of variable i is determined. Because the process is unobservable, it is guaranteed that each variable gets the same assignment in all clauses. If a clause was satisfied by this assignment, it will gain reward 1, if not, the reward will be $-m$, where m is the number of clauses of the formula. Therefore, if all clauses are satisfied, the maximal time-dependent performance of the UMDP is positive, otherwise negative.

We formally define the reduction. Let ϕ be a formula with n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. We say that x_i 1-*appears in* C_j, if $x_i \in C_j$, and x_i

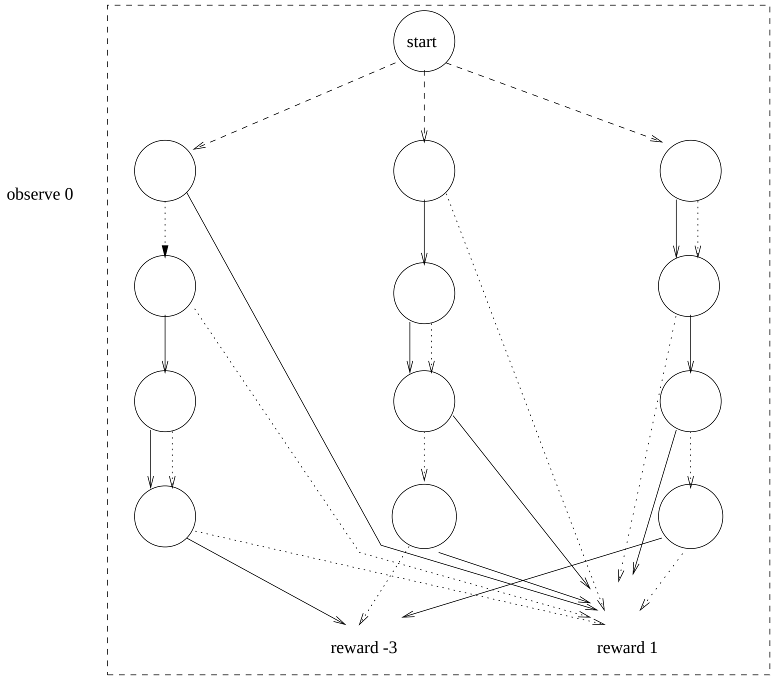

Fig. 3. $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$.

0-*appears*, if $\neg x_i \in C_j$. Define the UMDP $M(\phi) = (\mathcal{S}, s_0, \mathcal{A}, t, r)$ where

$$
\begin{array}{lcl}
\mathcal{S} & = & \{(i,j) \mid 1 \leq i \leq n, 1 \leq j \leq m\} \cup \{s_0, T, F\} \\
\mathcal{A} & = & \{0,1\} \\
t(s,a,s') & = & \begin{cases} \frac{1}{m}, & \text{if } s = s_0, a = 0, s' = (1,j), 1 \leq j \leq m \\ 1, & \text{if } s = (i,j), s' = T,\ x_i\ a\text{-appears in } C_j \\ 1, & \text{if } s = (i,j), s' = (i+1,j), i < n,\ x_i \text{ does not } a\text{-appear in } C_j \\ 1, & \text{if } s = (n,j), s' = F,\ x_n \text{ does not } a\text{-appear in } C_j \\ 1, & \text{if } s = s' = F \text{ or } s = s' = T, a = 0 \text{ or } a = 1 \\ 0, & \text{otherwise} \end{cases} \\
r(s,a) & = & \begin{cases} 1, & \text{if } t(s,a,T) > 0 \text{ and } s \neq T \\ -m, & \text{if } t(s,a,F) > 0 \text{ and } s \neq F \\ 0, & \text{otherwise.} \end{cases}
\end{array}
$$

The correctness of the reduction follows by the above discussion.

Figure 3 sketches the construction of $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$. Note that ϕ is the same as in Figure 2. The lines have the same meaning as in Figure 2. The dashed lines from the initial state indicate a transition with probability $\frac{1}{3}$ under action 0. $\square$

Since UMDPs are POMDPs, the same hardness proof as in Theorem 4.15 applies for the time-dependent policy existence problem for POMDPs.

COROLLARY 4.16. *The time-dependent policy existence problem for POMDPs is* NP-*complete.*

For (fully-observable) MDPs the optimal time-dependent and the optimal history-dependent policy have the same performance. Hence, the respective policy existence problems have the same complexity. For POMDPs the complexity is different. Instead, POMDPs seem to obtain their expressive power by history-dependent policies.

The following theorem is in stark contrast to the infinite-horizon case, where the problem is simply uncomputable [Madani et al. 1999]. Problems in PSPACE are certainly computable, albeit potentially slowly: we have no evidence yet that there are PSPACE-complete problems with deterministic time complexity significantly below exponential time! Thus, limiting the horizon reduces an uncomputable problem to an intractable one. However, it does leave open the hope of reasonable heuristics.

THEOREM 4.17. *The history-dependent policy existence problem for POMDPs is* PSPACE-*complete.*

The proof of Theorem 4.17 is a straightforward modification of the proof by Papadimitriou and Tsitsiklis [1987, Theorem 6], where the rewards for reaching the satisfying and unsatisfying final states are changed appropriately.

Papadimitriou and Tsitsiklis [1987] investigated POMDPs with all rewards ≤ 0 and considered the complexity of the question, whether a time-dependent or history-dependent policy exists under which the POMDP has expected reward equal 0. This is equivalent to the question, whether for a POMDP with all rewards ≥ 0 (we call this *nonnegative rewards*) there exists a policy with performance equal 0. In this context, the question of Papadimitriou and Tsitsiklis is a *minimization* question, whereas our question for policies with positive performance are *maximization* questions. We have seen that proof techniques from [Papadimitriou and Tsitsiklis 1987] — which in this light prove the complexity of minimization questions for POMDPs with nonnegative rewards — can be adapted for maximization questions for POMDPs with unrestricted (both negative and nonnegative) rewards (cf. Theorem 4.15 and Theorem 4.17). Now, we consider POMDPs with nonnegative rewards. As it turns out, for those the complexity of maximization is apparently smaller than that of minimization.

THEOREM 4.18. *The stationary, time-dependent and history-dependent policy existence problems for MDPs and UMDPs with nonnegative rewards are* NL-*complete.*

PROOF. Because each reward is at least 0, the expected reward for a policy π is greater than 0 if and only if at least one trajectory $\sigma_0, \ldots, \sigma_m$ consistent with π exists for which the action chosen by π for σ_m yields a reward greater than 0. It is clear that in such a trajectory, no state needs to appear more than once. This bounds the length of the trajectories to be considered by the number of the states of the POMDP. Therefore, the policy existence problems can be shown to be equivalent to graph reachability problems, from which we can conclude NL-completeness. □

For POMDPs the same argument works for time- or history-dependent policy existence problems. In order to create a trajectory with positive reward that is consistent with some unspecified policy, at any step of the process the next action can be chosen without taking former observations and actions into account.

COROLLARY 4.19. *The time-dependent and history-dependent policy existence problems for POMDPs with nonnegative rewards are* NL*-complete.*

For stationary policies, the same action must be chosen whenever the same observation is made. Therefore, the policy must be stored. Logspace is not sufficient to handle this problem, but the resources of NP suffice to guess a policy and to evaluate it. On the other hand, a stationary policy takes the same action whenever a certain observation is made during the process. In a sense, each observation "stores" the action taken when the observation was made for the first time. This allows us to express NP-complete problems.

Contrary to the results for time- and history-dependent policies, surprisingly, the complexity of the stationary policy existence problem for POMDPs does not depend on whether the rewards are nonnegative or unrestricted.

THEOREM 4.20. *The stationary policy existence problem for POMDPs with nonnegative rewards is* NP*-complete.*

PROOF. In Theorem 4.13 it was shown that the stationary policy existence problem for POMDPs with unrestricted rewards is NP-complete. In the proof of NP-hardness, a reduction from SAT was given by a construction of a POMDP with rewards 0 and 1 from a Boolean formula. Hence, the same proof applies for POMDPs with nonnegative rewards. □

4.2.2 *Compressed Representations.* For (fully-observable) flat MDPs, we have P-completeness for the time-dependent and for the history-dependent existence problems, and we have P-hardness and containment in NP for the stationary problem. The roles of P and NP are taken by PSPACE and NEXP in the case of compressed MDPs. It is noteworthy that the jump from P to PSPACE is potentially smaller than that from NP to NEXP. The first is a jump from time to space within the same polynomial resource bounds, whereas the latter increases the resource bounds exponentially.

LEMMA 4.21. *The stationary, time-dependent and history-dependent policy existence problems for compressed MDPs are* PSPACE*-hard.*

PROOF. To prove hardness, we show a polynomial time reduction from QBF, the validity problem for quantified Boolean formulae.

From a formula Φ with n quantified variables, we construct a fully-observable MDP with $2^{n+1}-1$ states, where every state represents an assignment of Boolean values to the first i variables ($0 \leq i \leq n$) of Φ. Transitions from state s can reach the two states representing assignments that extend s by assigning a value to the next unassigned variable. If this variable is bound by an existential quantifier, then the action taken in s assigns a value to that variable; otherwise the transition is random and independent of the action. Reward 1 is gained for every action after a state representing a satisfying assignment for the formula is reached. If a state representing an unsatisfying assignment is reached, reward $-(2^n)$ is gained. Then the maximal performance of this MDP is positive if and only if the formula is true. A compressed representation of the MDP can be computed in time polynomial in the size of the formula Φ.

Since every state except the last one appears at most once in every trajectory, this construction proves the same lower bound for any type of policy. □

LEMMA 4.22. *The history-dependent policy existence problem for compressed MDPs is in* PSPACE.

PROOF. We show that this problem is in NPSPACE = PSPACE. In PSPACE, an entire history-dependent policy cannot be specified for a horizon n equal to the size of the input. However, a policy can be guessed stepwise.

The set of possible histories to consider forms a tree of depth n. A policy can be specified by labelling the edges of that tree with actions. Such a policy can be guessed one branch at a time; in order to maintain consistency, one need only keep track of the current branch.

To evaluate a policy, for each branch through the policy tree, one must compute the value of each node under that policy. The probability of each transition can be represented with n bits; the product of n such requires n^2 bits. Rewards require at most n bits, and are accumulated (potentially) at each transition. The total reward for a given trajectory, therefore, requires at most n^3 bits.

There are at most 2^n states, so there are at most 2^{n^2} trajectories. The value of each is bounded by 2^{n^3}, so the sum is bounded by $2^{n^2+n^3}$, and thus can be represented by polynomially many bits. □

As in the case for flat MDPs, one can argue that the optimal action chosen when state s is reached in step i does not depend on how state s was reached (i.e. on the history of the process). Hence, the maximal performance of a compressed MDP under history-dependent policies equals its maximal performance under time-dependent policies. Therefore, the time-dependent policy existence problem for this type of MDP has the same complexity as the history-dependent problem.

LEMMA 4.23. *The time-dependent policy existence problem for compressed MDPs is in* PSPACE.

Combining Lemma 4.22 and Lemma 4.23 with Lemma 4.21, we show that the complexity of policy existence problems makes a jump from polynomial time to polynomial space if we consider compressed MDPs instead of flat MDPs. One could argue that this is a good indication that P is different from PSPACE. However, this remains an open question.

THEOREM 4.24. *The time-dependent and history-dependent policy existence problems for compressed MDPs are* PSPACE-*complete.*

Note that — similarly to flat MDPs — the exact complexity of the stationary policy existence problem for compressed MDPs is open.

For compressed POMDPs, we could expect to obtain similar complexity jumps as from flat to compressed fully-observable MDPs. Surprisingly, the complexities of the different policy existence problems relate totally differently. The history-dependent existence problem does not alter its complexity from flat to compressed representation. It becomes the easiest of the three problems. However, the stationary and the time-dependent existence problems make a full exponential complexity jump.

THEOREM 4.25. *The stationary policy existence problem for compressed POMDPs is* NEXP*-complete.*

PROOF. Membership in NEXP follows from the standard guess-and-check approach. To show NEXP-hardness, we sketch a reduction from the succinct version of 3SAT, shown to be NEXP-complete in [Papadimitriou and Yannakakis 1986], to the stationary existence problem. Instances for succinct 3SAT are encodings of Boolean formulae in conjunctive normal form, in which all clauses consist of exactly three literals, and each literal appears at most three times. Hence, if the formula has n variables, it has $O(n)$ clauses. The encoding of the formula is a circuit which on input $i; k$ outputs the ith literal in the kth clause. The reduction is similar to that of Theorem 4.15, where a formula was transformed into a POMDP which "checks" all clauses of the formula in parallel. Here, we put all states which stand for appearances of the same variable into one observation class and leave out considerations of variables in a clause where they do not appear. Let m be the number of clauses of the formula. Then each of the m trajectories consists of the initial state, (at most) three states for the literals in one clause and a final sink. Each stationary policy corresponds to an assignment to the variables. If the assignment is satisfying, each trajectory will yield reward 1. If the assignment is not satisfying, then at least one trajectory corresponds to a unsatisfied clause and yields reward $-m$.

Figure 4 sketches the construction for our standard example. □

The time-dependent policy existence problem for compressed POMDPs has the same complexity. Interestingly, as in the flat case we need more random transitions to show hardness for the time-dependent existence problem than for the stationary one.

THEOREM 4.26. *The time-dependent policy existence problem for compressed POMDPs is* NEXP*-complete.*

PROOF. Containment in NEXP follows from the standard guess-and-check approach. Hardness for NEXP is an extension of the above construction. Let $M_{(j,b)}$ be the POMDP as constructed above, but whenever observation x_j is made (i.e. the process expects an assignment to variable x_j) then reward -2^{n+m} is obtained if action b is *not* chosen. Let $s_{(j,b)}$ be the initial state of $M_{(j,b)}$. The new process we construct has a new initial state. Let l be the number of variables in the formula. From the initial state with equal probability the new states s_i (for $1 \leq i \leq n$) are reachable. State s_i has observation x_i, and the meaning of s_i is to fix an assignment to x_i. On action $a \in \{0, 1\}$, the process goes deterministically from state s_i to state $s_{(i,a)}$. Notice that after the next transition the observations are in $\{x_1, \ldots, x_l\}$, and henceforth the time-dependent policy cannot remember which variable it assigned on the first step. Assume that at some step the policy acts differently on observation x_i than in the first step, where it chose action b. Then it will obtain such a big negative reward in the subprocess $M_{(i,b)}$ that the expected reward of the whole process will be negative. Therefore, the only policy that obtains a positive

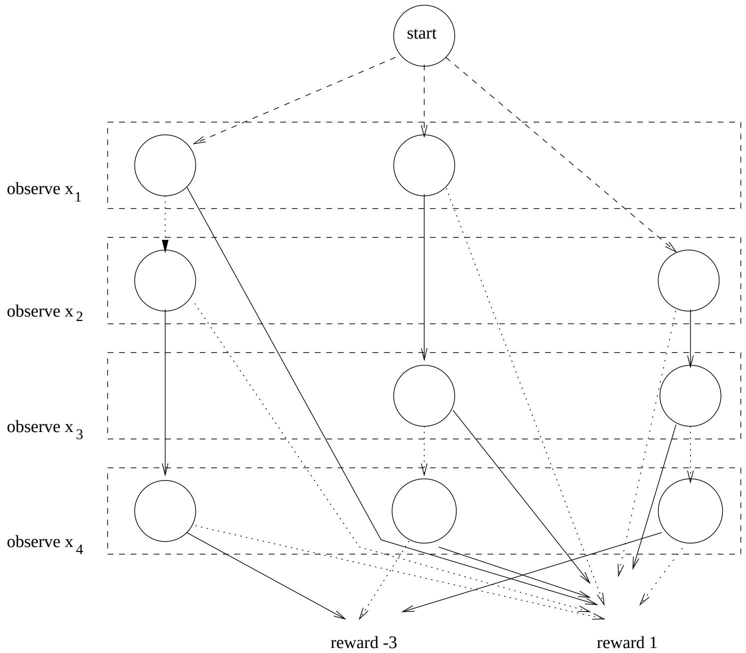

Fig. 4. $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$.

expected reward is the one that behaves consistently, i.e. which chooses the same action whenever the same observation is made. As in the above proof, it now follows that a time-dependent policy with positive expected reward exists if and only if the formula is satisfiable.

Figure 5 sketches the construction for our standard example. □

Full or partial observability does not affect the complexity of the history-dependent existence problem.

Theorem 4.27. *The history-dependent policy existence problem for compressed POMDPs is* PSPACE*-complete.*

Proof. The argument that the problem is in PSPACE is a straightforward modification of a proof by Papadimitriou and Tsitsiklis [1987, Theorem 6] (see also Theorem 4.17). Hardness for PSPACE follows from Lemma 4.21. □

For flat POMDPs, the complexity of policy existence is equal for partial observability and unobservability. We show that observability influences the complexity of compressed POMDPs. The policy existence problems for compressed UMDPs are simpler than for compressed MDPs. Later, we will show that the number of possible actions is important. Remember that $\mathrm{NP} \subseteq \mathrm{PP} \subseteq \mathrm{NP}^{\mathrm{PP}} \subseteq \mathrm{PSPACE}$.

Theorem 4.28. *The stationary policy existence problem for compressed UMDPs is complete for* $\mathrm{NP}^{\mathrm{PP}}$.

Proof. To see that this problem is in $\mathrm{NP}^{\mathrm{PP}}$, remember that the corresponding policy evaluation problem is PP-complete, by Theorem 4.9. For the existence

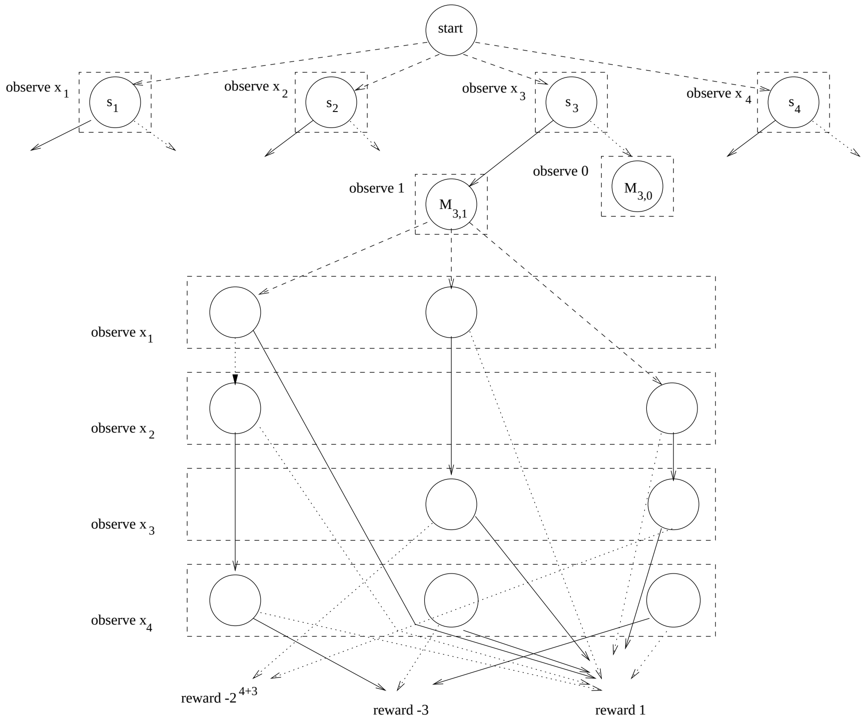

Fig. 5. $M(\phi)$ for $\phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$.

question, one can guess a (polynomial-sized) policy, and verify that it has expected reward > 0 by consulting a PP oracle.

To show $\mathrm{NP}^{\mathrm{PP}}$-hardness, one needs that $\mathrm{NP}^{\mathrm{PP}}$ equals the $\leq_m^{np}$ closure of PP [Torán 1991]. That can be seen as the closure of PP under polynomial-time disjunctive reducibility with an exponential number of queries. Each of these queries is computable in polynomial time from its index in the list of queries.

Let $A \in \mathrm{NP}^{\mathrm{PP}}$. By Theorem 4.9, there is q polynomial-time computable two-parameter function f and a polynomial p, such that for every x, $x \in A$ if and only if there exists a y of length $p(|x|)$ such that $f(x,y)$ outputs a positive instance (M,π) of the stationary policy evaluation problem for compressed UMDPs. We fix an x. For $f(x,y) = (M,\pi)$, we let $M_{x,y}$ be that UMDP obtained from M by hard-wiring policy π into it. I.e. $M_{x,y}$ has the same states as M, but whenever an observation b is made in M, $M_{x,y}$ on any action a behaves like M on action $\pi(b)$. Hence, we do not need any observations in $M_{x,y}$, and M has positive expected reward under π if and only if $M_{x,y}$ has positive expected reward after the given number of steps under all policies. After this given number of steps, $M_{x,y}$ goes into a sink state in which it earns no rewards. Now, we construct a new UMDP M_x from the union of all $M_{x,y}$ and a new initial state. From its initial state, M_x on action a reaches the initial state of $M_{x,a}$ with probability 1 and reward 0. Therefore, M_x has positive expected reward under some policy if and only if for some a, $M_{x,a}$ has positive expected reward under constant policy a if and only if for some a,

$f(x, a)$ is a positive instance of the policy evaluation problem under consideration if and only if $x \in A$. Because f is polynomial time computable, and M_x has at least $2^{p(|x|)} + 1$ states, it follows that M_x is a compressed UMDP. Thus the policy existence problem is hard for $\mathrm{NP}^{\mathrm{PP}}$. □

Note that only the action chosen in the first step determined which reward will be obtained. Therefore, we obtain the same hardness for time-dependent policies. Membership in $\mathrm{NP}^{\mathrm{PP}}$ follows from Theorem 4.9.

THEOREM 4.29. *The time-dependent policy existence problem for compressed UMDPs is* $\mathrm{NP}^{\mathrm{PP}}$*-complete.*

For compressed POMDPs with nonnegative rewards, all policy existence problems are equally hard. Note that for stationary policies, the complexity does not increase going from flat to compressed POMDPs.

THEOREM 4.30. *The stationary, time-dependent and history-dependent policy existence problems for compressed MDPs, compressed POMDPs or compressed UMDPs, with nonnegative rewards are* NP*-complete.*

PROOF. Membership in NP follows using similar arguments as those used for Theorem 4.18. Hardness for NP follows from the fact that an NP computation tree can be represented as a compressed POMDP with only one action, where nondeterministic computation steps are probabilistic transitions between states. Reaching an accepting state yields a positive reward. Because the POMDP has only one action, observability does not matter. □

5. LONG TERM HORIZON

We turn to planning problems for horizons exponential in the size of the problem description. One can argue that, for compressed or succinct representations, this is the natural analogue of a horizon that is polynomial in the size of a flat representation. If an example of a short-horizon policy is emergency medical care, then a long-term horizon would be the ongoing policy of, say, a family practice doctor. Or, for another example, consider radioactive waste management: congressional policy may be short-term, but a reasonable and safe policy should be extremely long-term. (We won't even mention, for instance, U.S. funding policies for basic research.)

The *long term α policy evaluation problem for POMDPs* is the set of all pairs (M, π) consisting of a POMDP $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, o, t, r)$ and an α policy π, such that $\mathrm{perf}(M, s_0, 2^{|M|}, \pi) > 0$. As before, we consider POMDPs in different encodings. The *long term policy existence problems for POMDPs* are defined similarly.

For long term problems, compressed POMDPs are more interesting than flat POMDPs, because in the long term case the number of states is exponential in the size of the input, so a trajectory can visit each state once. Remember that we have two circuit representations of POMDPs. A *succinct POMDP* consists of circuits calculating the state transition probability t, the observation function o, and the reward function r. Unlike compressed POMDPs, these circuits only have one output bit. They take the index of the function value's bit as additional input. For example, the state transition probability function t is represented by a circuit which on input (s, a, s', l) outputs the lth bit of $t(s, a, s')$. This allows for function values which are exponentially long in the size of the function's circuit representation.

5.1 Policy evaluation for compressed representations

As in the short term problems for compressed POMDPs, specification of long term policies for compressed POMDPs is not tractable. Moreover, because the horizon is exponential in the size of the POMDP's description, even time-dependent policies for UMDPs cannot be efficiently specified. Therefore, we consider the complexity of policy evaluation only for UMDPs under stationary policies.

Put succinctly, the techniques from Section 4.1.1 translate to succinctly represented POMDPs. The first result which we translate is Lemma 4.1. In Lemma 4.1, we considered the problem of matrix powering for $m \times m$ matrices; here, we consider powering a succinctly-represented $2^m \times 2^m$ matrix. In space logarithmic in the size of the matrix, we can apply the algorithm posited in Lemma 4.1; since the representation has size $\Omega(m)$, the problem is located in the function class #PSPACE.

LEMMA 5.1. *Let T be a $2^m \times 2^m$ matrix of nonnegative integers, each consisting of 2^m bits. Let T be represented by a Boolean circuit C with $3m$ input bits, such that $C(a, b, r)$ outputs the r-th bit of $T_{(a,b)}$. For $1 \leq i, j \leq 2^m$, and $0 \leq s \leq m$, the function mapping (C, s, i, j) to $(T^s)_{(i,j)}$ is in* #PSPACE.

As a consequence, we can prove the long term policy evaluation problem for compressed UMDPs is exponentially more complex than for the flat ones.

THEOREM 5.2. *The long term stationary policy evaluation problem for compressed UMDPs is* PSPACE*-complete.*

PROOF. In order to show that the problem is in PSPACE, we can use the same technique as in the proof of Lemma 4.2, yielding here that the problem is in PPSPACE (= probabilistic PSPACE). Ladner [Ladner 1989] showed that FPSPACE = #PSPACE, from which it follows that PPSPACE = PSPACE.

Showing PSPACE-hardness is even easier than showing PL-hardness (as in the proof of Theorem 4.3), because here we deal with a deterministic class. We can consider the computation of a polynomial space-bounded machine on input x as a directed graph with configurations as nodes and the transition relation between configurations as arcs. This graph is the skeleton of a compressed POMDP. The only action means "perform a configuration transition." If an accepting configuration is reached, reward 1 is obtained; all other rewards are 0. Hence, the expected reward is greater than 0 if and only if an accepting configuration can be reached from the initial configuration. □

Note that we only used rewards 0 and 1 in the above hardness proof. Therefore, unlike the flat case, the same completeness result follows for POMDPs with nonnegative rewards.

THEOREM 5.3. *The long term stationary policy evaluation problem for compressed UMDPs with nonnegative rewards is* PSPACE*-complete.*

5.2 Policy existence problems for compressed representations

The results from Section 4.2.1 for short term policy existence problems for flat POMDPs translate to compressed POMDPs. The arguments used to prove the following results are direct translations of those used in the related proofs for flat POMDPs.

THEOREM 5.4. *The long term time- or history-dependent policy existence problems for compressed MDPs are* EXP*-complete.*

PROOF. Given a compressed MDP M, one can write down its flat representation M' in time $O(2^{|M|})$. Now, the short term time-dependent policy existence problem for M' is equivalent to the long term time-dependent policy existence problem for M, and takes time polynomial in $|M'|$ by Theorem 4.16. Hence, the long term time-dependent policy existence problem is in EXP.

To show P-hardness of the short term time-dependent policy existence problem for MDPs, we used P-hardness of the circuit value problem. Since the *succinct circuit value problem* is hard for EXP, we obtain EXP-hardness of the long term time-dependent policy existence problem for MDPs. □

THEOREM 5.5. *(1) The long term stationary and time-dependent policy existence problems for compressed POMDPs are* NEXP*-complete.*

(2) The long term history-dependent policy existence problem for compressed POMDPs is EXPSPACE*-complete.*

Note that NEXP-hardness also follows from Theorem 4.26.

THEOREM 5.6. *The long term stationary policy existence problem for compressed UMDPs is* PSPACE*-complete.*

PROOF. A stationary policy for an UMDP consists of one action. Guessing an action and evaluating whether the UMDP's performance under this policy is positive can be performed in $\mathrm{NP}^{\mathrm{PSPACE}} = \mathrm{PSPACE}$.

Hardness for PSPACE follows as in the hardness part of the proof of Theorem 5.2. □

THEOREM 5.7. *The long term time-dependent policy existence problem for compressed UMDPs is* NEXP*-complete.*

The following results can be proven using the same ideas as for short term flat POMDPs.

THEOREM 5.8. *The long term stationary policy existence problem for compressed POMDPs with nonnegative rewards is* NEXP*-complete.*

THEOREM 5.9. *The long term time-dependent and history-dependent policy existence problems for compressed POMDPs with nonnegative rewards are* PSPACE*-complete.*

THEOREM 5.10. *The long term stationary, time-dependent, and history-dependent policy existence problems for compressed MDPs and for compressed UMDPs with nonnegative rewards are* PSPACE*-complete.*

6. SOME SPECIAL CASES

In Theorem 4.29 we showed $\mathrm{NP}^{\mathrm{PP}}$-completeness of the time-dependent policy existence problem for compressed UMDPs. It turns out that the the number of actions of the UMDP affects the complexity. In the proof of Theorem 4.29 we needed a number of actions exponential in the size of the UMDP. When we restrict this number to be polynomial in the size of the UMDP, we obtain a lower complexity. Those UMDPs will be called *UMDPs with few actions.*

Theorem 6.1. *The stationary policy existence problem for compressed UMDPs with few actions is* PP*-complete.*

Proof. Hardness for PP follows from the hardness proof of Lemma 4.3. To show containment in PP, we use arguments similar to those in the proof of Lemma 4.2. It is convenient to use the class GapP, which is the class of functions that are the difference of two #P functions. We make use of the fact that $A \in \mathrm{PP}$ if and only if there exists a GapP function f such that for every x, $x \in A$ if and only if $f(x) > 0$ (see [Fenner et al. 1994]). One can show that the function p from the proof of Lemma 4.2 is in GapP, because the respective matrix powering is in GapP (see the proofs of Lemmas 4.8 and 4.1), and GapP is closed under multiplication and summation. Finally, PP is closed under polynomial-time disjunctive reducibility [Beigel et al. 1995], which completes the proof. □

The complexity gap between the stationary and the time-dependent policy existence problems for compressed UMDPs with few actions is as big as that for flat POMDPs, but the difference no longer depends on the number of actions.

Theorem 6.2. *The time-dependent policy existence problem for compressed UMDPs with few actions is* $\mathrm{NP}^{\mathrm{PP}}$*-complete.*

Proof. In the proof of Lemma 4.28, we consider a polynomial-time computable two-parameter function f and a polynomial p, such that for every x, $x \in A$ if and only if there exists a y of length $p(|x|)$ such that $f(x, y)$ outputs a positive instance (M, π) of the stationary policy evaluation problem for compressed UMDPs. The UMDP $M_{x,y}$ has exactly one action, and the $M_{x,y}$s are joined so that action y in step 1 transitions to $M_{x,y}$.

We now modify that construction by allowing $p(|x|)$ steps to get to y, and 2 actions (0 and 1); a time-dependent policy specifies a sequence of 0s and 1s (which we call "y") for the first $p(|x|)$ steps, and then both actions have the same effect within $M_{x,y}$.

Thus, $x \in A$ if and only if there is some y such that $f(x, y) = M_{x,y}$ has positive expected value if and only if there is some time-dependent policy for M_x that has positive expected value. □

6.1 Exact Policy Existence

So far, the policy existence problems we have considered have been inexact: Is there a policy with value ≥ 0? There are times when we want to know whether we can spend the budget exactly.

The *exact α policy existence problem for β MDPs* is the set of all pairs (M, k) of a β MDP $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, o, t, r)$ and a rational k, for which there exists an α policy π with performance $\mathrm{perf}(M, s_0, |M|, \pi) = k$.

There are several notable instances where the *exact* policy existence problem has a different complexity than the general policy existence problem. For instance, we have been unable to determine the complexity of the finite horizon stationary policy existence problem for fully-observable MDPs. (The infinite horizon case is known to be P-complete [Papadimitriou and Tsitsiklis 1987]; as noted in the introduction, the finite horizon problem is only known to be P-hard and in NP.) However, we can determine the complexity of the *exact* policy existence problem as follows.

THEOREM 6.3. *The exact stationary policy existence problem for fully-observable (flat) MDPs is* NP*-complete.*

PROOF. It is clear that the problem is in NP. In order to show its completeness, we give a reduction from the NP-complete problem PARTITION. The instances of PARTITION are functions $f : [n] \to \mathbf{N}$ for arbitrary $[n] = \{1, 2, \ldots, n\}$. Such an instance belongs to PARTITION if and only if for a subset $A \subseteq [n]$ it holds that $2\sum_{a\in A} f(a) = \sum_{b\in[n]} f(b)$.

Let $g : [m] \to \mathbf{N}$ be an instance of PARTITION. From g, we construct a fully-observable MDP M with state set $\mathcal{S} = [m + 1]$, initial state $s_0 = 1$, actions $\mathcal{A} = \{in, out\}$, transition function

$$t(s, a, s') = \begin{cases} 1, & \text{if } s' = \min\{s + 1, m + 1\},\ a \in \{in, out\} \\ 0, & \text{otherwise,} \end{cases}$$

and reward function

$$r(s, a) = \begin{cases} g(s), & \text{if } a = in \text{ and } s \le m \\ 0, & \text{if } a = out \text{ or } s = m + 1. \end{cases}$$

Every stationary policy π for M determines a subset $A = \{i \mid \pi(i) = in\}$ of $[m]$, such that $\mathrm{perf}(M, s_0, |M|, \pi) = \sum_{a\in A} g(a)$. Therefore, $g \in$ PARTITION if and only if M has a stationary policy with reward exactly $\frac{1}{2}\sum_{b\in[n]} g(b)$. □

Notice that, in the MDP constructed in this proof, transitions depend only on time, not on the action chosen. Thus, the proof holds for unobservable and partially-observable MDPs, and also for time-dependent policies.

COROLLARY 6.4. *The exact stationary and time-dependent policy existence problems partially- and fully-observable (flat) MDPs and the time-dependent policy existence problem for UMDPs are* NP*-complete.*

7. DISCUSSION AND OPEN QUESTIONS

One general lesson that can be drawn from our results is that there is no simple relationship among the policy existence problems for stationary, time-dependent, and history-dependent policies. Although it is trivially true that if a good stationary policy exists, then good time- and history-dependent policies exist, is is not always the case that one of these problems is easier than the other. For instance, in Theorems 4.17 and 4.13 and Corollary 4.16, we see that the policy existence problem for history-dependent policies can be more difficult than for stationary and time-dependent policies, and in Theorems 4.14 and 4.15, the time-dependent case is more difficult than the stationary case. This contrasts with the situation in Corollary 4.19 and Theorem 4.20 and in Theorems 4.25 and 4.27, where the policy-existence problem is more difficult for stationary policies than for history-dependent policies.

There are a few problems that have not yet been categorized by completeness. However, the major open questions are of the form: What now? Now that we have proved that these problems are difficult to compute, heuristics are needed in order to manage them. The only heuristics for $\mathrm{NP}^{\mathrm{PP}}$-complete problems in the literature so far are in [Majercik and Littman 1998a]. Surprisingly, $\mathrm{NP}^{\mathrm{PP}}$-complete problems arise frequently, at least in the AI/planning literature. (See [Goldsmith et al. 1997] for more examples of $\mathrm{NP}^{\mathrm{PP}}$-complete planning problems, and [Littman 1999a] for experimental data on nondeterministically and probabilistically constrained formulas.)

Acknowledgements

We would like to thank Anne Condon, Andy Klapper, Matthew Levy, and especially Michael Littman for discussions and suggestions on this material.

REFERENCES

ALLENDER, E. AND OGIHARA, M. 1996. Relationships among PL, #L, and the determinant. *RAIRO Theoretical Informatics and Applications 30*, 1, 1–21.

ÀLVAREZ, C. AND JENNER, B. 1993. A very hard log-space counting class. *Theoretical Computer Science 107*, 3–30.

AOKI, M. 1965. Optimal control of partially observed Markovian systems. *Journal of the Franklin Institute 280*, 367–368.

ARROYO-FIGUEROA, G. AND SUCAR, L. E. 1999. A temporal Bayesian network for diagnosis and prediction. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

ASTROM, K. 1965. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications 10*, 174–205.

BÄCKSTRÖM, C. 1995. Expressive equivalence of planning formalisms. *Artificial Intelligence 76*, 17–34.

BALCÁZAR, J., LOZANO, A., AND TORÁN, J. 1992. The complexity of algorithmic problems on succinct instances. In R. BAEZA-YATES AND U. MANBER Eds., *Computer Science*, pp. 351–377. Plenum Press.

BARRY, P. AND LASKEY, K. B. 1999. An application of uncertain reasoning to requirements engineering. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

BEAUQUIER, D., BURAGO, D., AND SLISSENKO, A. 1995. On the complexity of finite memory policies for Markov decision processes. In *Math. Foundations of Computer Science* (1995), pp. 191–200. Lecture Notes in Computer Science #969: Springer-Verlag.

BEIGEL, R., REINGOLD, N., AND SPIELMAN, D. 1995. PP is closed under intersection. *Journal of Computer and System Sciences 50*, 191–202.

BELLMAN, R. 1957. *Dynamic Programming.* Princeton University Press.

BLONDEL, V. AND TSITSIKLIS, J. 1998. A survey of computational complexity results in systems and control. Available from `http://web.mit.edu/jnt/www/publ.html` or `http://web.mit.edu/~jnt/survey.ps.` (postscript, 785K), to appear in Automatica.

BLYTHE, J. 1999. Decision-theoretic planning. *AI Magazine 20*, 2, 37–54.

BOMAN, DAVIDSSON, AND YOUNES. 1999. Artificial decision making in intelligent buildings. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

BOUTILIER, C., DEAN, T., AND HANKS, S. 1995. Planning under uncertainty: Structural assumptions and computational leverage. In *Proceedings of the Second European Workshop on Planning* (1995).

BOUTILIER, C., DEAN, T., AND HANKS, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of AI Research 11*, 1–94.

BOUTILIER, C., DEARDEN, R., AND GOLDSZMIDT, M. 1995. Exploiting structure in policy construction. In *14th International Conference on AI* (1995).

BOUTILIER, C., GOLDSZMIDT, M., AND SABATA, B. 1999. Continuous value function approximation for sequential bidding policies. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

BRYANT, R. E. 1991. On the complexity of VLSI implementations and graph representations of boolean functions with application to integer multiplication. *IEEE Transactions on Computers C-40*, 2, 205–213.

Burago, D., de Rougemont, M., and Slissenko, A. 1996. On the complexity of partially observed Markov decision processes. *Theoretical Computer Science 157*, 2, 161–183.

Bylander, T. 1994. The computational complexity of propositional STRIPS planning. *Artificial Intelligence 69*, 165–204.

Cassandra, A. 1998. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes.* Ph. D. thesis, Brown University.

Cassandra, A., Kaelbling, L., and Littman, M. 1994. Acting optimally in partially observable stochastic domains. In *Proceedings of AAAI-94* (1994).

Cassandra, A., Kaelbling, L., and Littman, M. 1995. Efficient dynamic-programming updates in partially observable Markov decision processes. Technical Report CS-95-19, Brown University, Providence, Rhode Island.

Cassandra, A., Littman, M., and Zhang, N. 1997. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In D. Geiger and P. P. Shenoy Eds., *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI-97)* (1997), pp. 54–61. Morgan Kaufmann Publishers.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence 32*, 333–379.

Dynkin, E. 1965. Controlled random sequences. *Theory and Probability Appl. X*, 10, 1–14.

Erol, K., Hendler, J., and Nau, D. 1996. Complexity results for hierarchical task-network planning. *Annals of Mathematics and Artificial Intelligence 18*, 69–93.

Erol, K., Nau, D., and Subrahmanian, V. 1995. Complexity, decidability and undecidability results for domain-independent planning. *Artificial Intelligence 76*, 75–88.

Fenner, S., Fortnow, L., and Kurtz, S. 1994. Gap-definable counting classes. *Journal of Computer and System Sciences 48*, 1, 116–148.

Galperin, H. and Wigderson, A. 1983. Succinct representation of graphs. *Information and Control 56*, 183–198.

Golabi, K., Kulkarni, R., and Way, G. 1982. A statewide pavement management system. *Interfaces 12*, 5–21.

Goldsmith, J., Littman, M., and Mundhenk, M. 1997. The complexity of plan existence and evaluation in probabilistic domains. In *Proc. 13th Conf. on Uncertainty in AI* (1997). Morgan Kaufmann Publishers.

Goldsmith, J. and Mundhenk, M. 1998. Complexity issues in Markov decision processes. In *Proceedings of IEEE Conference on Computational Complexity* (1998).

Hansen, E. 1998a. *Finite-Memory Control of Partially Observable Systems.* Ph. D. thesis, Dept. of Computer Science, University of Massachussets at Amherst.

Hansen, E. 1998b. Solving POMDPs by searching in policy space. In *Proceedings of AAAI Conference Uncertainty in AI* (1998).

Hauskrecht, M. 1997. *Planning and Control in Stochastic Domains with Imperfect Information.* Ph. D. thesis, Massachusetts Institute of Technology.

Hochbaum, D. 1997. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company.

Hoey, J., St-Aubin, R., Hu, A., and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 279–288.

Horvitz, E., Jacobs, A., and Hovel, D. 1999. Attention-sensitive alerting in computing systems. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

Howard, R. 1960. *Dynamic Programming and Markov Processes.* MIT Press.

Jung, H. 1984. On probabilistic tape complexity and fast circuits for matrix inversion problems. In *Proceedings 11th ICALP* (1984), pp. 281–291. Lecture Notes in Computer Science #172.

Jung, H. 1985. On probabilistic time and space. In *Proceedings 12th ICALP* (1985), pp. 281–291. Lecture Notes in Computer Science #194: Springer-Verlag.

Korb, K. B., Nicholson, A. E., and Jitnah, N. 1999. Bayesian poker. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

KUSHMERICK, N., HANKS, S., AND WELD, D. 1995. An algorithm for probabilistic planning. *Artificial Intelligence 76*, 239–286.

LADNER, R. 1989. Polynomial space counting problems. *SIAM Journal on Computing 18*, 1087–1097.

LITTMAN, M. 1996a. *Algorithms for Sequential Decision Making*. Ph. D. thesis, Brown University.

LITTMAN, M. 1996b. Probabilistic STRIPS planning is EXPTIME-complete. Technical Report CS-1996-18 (November), Duke University Department of Computer Science.

LITTMAN, M. 1997a. Probabilistic propositional planning: Representations and complexity. In *Proc. 14th National Conference on Artificial Intelligence* (1997). AAAI Press/The MIT Press.

LITTMAN, M. 1997b. Probabilistic propositional planning: Representations and complexity. In *Proc. 14th National Conference on AI* (1997). AAAI Press / MIT Press.

LITTMAN, M., DEAN, T., AND KAELBLING, L. 1995. On the complexity of solving Markov decision problems. In *Proceedings of the 11th Annual Conference on Uncertainty in AI* (1995), pp. 394–402.

LITTMAN, M., GOLDSMITH, J., AND MUNDHENK, M. 1998. The computational complexity of probabilistic plan existence and evaluation. *The Journal of AI Research 9*, 1–36.

LITTMAN, M. L. 1999a. Initial experiments in probabilistic satisfiability. In *Proceedings of AAAI-99* (1999). In preparation for conference submission.

LITTMAN, M. L. 1999b. Initial experiments in stochastic satisfiability. In *Proceedings of Sixteenth National Conference on Artificial Intelligence* (1999).

LOVEJOY, W. 1991. A survey of algorithmic methods for partially observed Markov decision processes. *Annals of Operations Research 28*, 47–66.

LUSENA, C., LI, T., SITTINGER, S., WELLS, C., AND GOLDSMITH, J. 1999. My brain is full: When more memory helps. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 374–381.

MADANI, O., HANKS, S., AND CONDON, A. 1999. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence* (1999).

MAJERCIK, M. AND LITTMAN, M. 1998a. Maxplan: A new approach to probabilistic planning. In *Artificial Intelligence and Planning Systems* (1998), pp. 86–93.

MAJERCIK, S. M. AND LITTMAN, M. L. 1998b. Using caching to solve larger probabilistic planning problems. In *Proceedings of Fifteenth National Conference on Artificial Intelligence* (1998), pp. 954–959.

MEULEAU, N., KIM, K.-E., KAELBLING, L. P., AND CASSANDRA, A. R. 1999. Solving POMDPs by searching the space of finite policies. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 417–426.

MEULEAU, N., PESHKIN, L., KIM, K.-E., AND KAELBLING, L. P. 1999. Learning finite-state controllers for partially observable environments. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 427–436.

MISLEVY, R. J., ALMOND, R. G., YAN, D., AND STEINBERG, L. S. 1999. Bayes nets in educational assessment: Where the numbers come from. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

MONAHAN, G. 1982. A survey of partially observable Markov decision processes: Theory, models, and algorithms. *Manag. Sci. 28*, 1–16.

MUNDHENK, M. 1998. The complexity of optimal small policies. To appear in Mathematics of Operations Research.

MUNDHENK, M., GOLDSMITH, J., AND ALLENDER, E. 1997. The complexity of the policy existence problem for partially-observable finite-horizon Markov decision processes. In *Proc. 25th Mathematical Foundations of Computer Sciences* (1997), pp. 129–138. Lecture Notes in Computer Science #1295: Springer-Verlag.

NGUYEN, H. AND HADDAWY, P. 1999. The decision-theoretic interactive video advisor. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

Papadimitriou, C. 1994. *Computational Complexity.* Addison-Wesley.

Papadimitriou, C. and Tsitsiklis, J. 1986. Intractable problems in control theory. *SIAM Journal of Control and Optimization 24*, 4, 639–654.

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

Papadimitriou, C. and Yannakakis, M. 1986. A note on succinct representations of graphs. *Information and Control 71*, 181–185.

Parr, R. and Russell, S. 1995. Approximating optimal policies for partially observable stochastic domains. In *Proceedings of IJCAI-95* (1995).

Peshkin, L., Meuleau, N., and Kaelbling, L. P. 1999. Learning policies with external memory. In *Proceedings of the Sixteenth International Conference on Machine Learning* (1999).

Platzman, L. 1977. *Finite-memory Estimation and Control of Finite Probabilistic Systems.* Ph. D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

Portinale, L. and Bobbio, A. 1999. Bayesian networks for dependability analysis: an application to digital control reliability. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

Puterman, M. 1994. *Markov Decision Processes.* John Wiley & Sons, New York.

Pyeatt, L. 1999. *Integration of Partially Observable Markov Decision Processes and Reinforcement Learning for Simulated Robot Navigation.* Ph. D. thesis, Colorado State University.

Shatkay, H. 1999. Learning hidden Markov models with geometrical constraints. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

Simmons, R. and Koenig, S. 1995. Probabilistic robot navigation in partially observable environments. In *Proceedings of IJCAI-95* (1995).

Smallwood, R. and Sondik, E. 1973. The optimal control of partially observed Markov processes over the finite horizon. *Operations Research 21*, 1071–1088.

Sondik, E. 1971. *The Optimal Control of Partially Observable Markov Processes.* Ph. D. thesis, Stanford University.

Streibel, C. 1965. Sufficient statistics in the optimal control of stochastic systems. *J. Math. Anal. Appl. 12*, 576–592.

Sutton, R. S. and Barto, A. G. 1998. *Reinforcement Learning: An Introduction.* The MIT Press.

Toda, S. 1991. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing 20*, 865–877.

Torán, J. 1991. Complexity classes defined by counting quantifiers. *Journal of the ACM 38*, 3, 753–774.

Tseng, P. 1990. Solving h-horizon, stationary Markov decision problems in time proportional to $\log h$. *Operations Research Letters 9*, 5 (September), 287–297.

van der Gaag, L., Renooij, S., Witteman, C., Aleman, B., and Tall, B. 1999. How to elicit many probabilities. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 647–654.

Vinay, V. 1991. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proc. 6th Structure in Complexity Theory Conference* (1991), pp. 270–284. IEEE.

Wagner, K. 1986. The complexity of combinatorial problems with succinct input representation. *Acta Informatica 23*, 325–356.

Welch, R. L. and Smoth, C. 1999. A process control algorithm for concentrating mixed-waste based on Bayesian cg-networks. In *Proc. 15th Conference on Uncertainty in Artificial Intelligence* (1999). Morgan Kaufman.

White, C., III. 1991. Partially observed Markov decision processes: A survey. *Annals of Operations Research 32*, 215–230.

ZHANG, N. AND LIU, W. 1997. A model approximation scheme for planning in partially observable stochastic domains. *Journal of Artificial Intelligence Research 7*, 199–230.

ZHANG, N. L., LEE, S. S., AND ZHANG, W. 1999. A method for speeding up value iteration in partially obserable Markov decision processes. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 696–703.