# The complexity of unobservable finite-horizon Markov decision processes *

Martin Mundhenk † Judy Goldsmith ‡ Eric Allender §

December 3, 1996

**Abstract**

Markov Decision Processes (MDPs) model controlled stochastic systems. Like Markov chains, an MDP consists of states and probabilistic transitions; unlike Markov chains, there is assumed to be an outside controller who chooses an action (with its associated transition matrix) at each step of the process, according to some strategy or *policy*. In addition, each state and action pair has an associated reward. The goal of the controller is to maximize the expected reward. MDPs are used in applications as diverse as wildlife management and robot navigation control. Optimization and approximation strategies for these models constitute a major body of literature in mathematics, operations research, and engineering.

We consider the complexity of the following decision problem: for a given MDP and type of policy, is there such a policy for that MDP with positive expected reward? The complexity of this problem depends on at least half a dozen factors, including the information available to the controller, the feedback mechanism, and the succinctness of the representation of the system relative to the number of states.

This paper, together with [6], shows variations of the decision problem to be complete for NL, PL, P, NP, PP, $NP^{PP}$, PSPACE, EXP, NEXP and EXPSPACE, and [6] also shows that some NP-complete problems are not $\varepsilon$-approximable. This paper focuses on the proofs of completeness for PL, PP, and $NP^{PP}$. All of the problems considered here are for MDPs that run for a fixed, finite time, either equal to the number of states of the system or the size of the representation of the system.

Papadimitriou and Tsitsiklis showed that the most straightforward MDP decision problem is P-complete, and other variants are NP- or PSPACE-complete [13]. Several others have shown other MDP problems are complete for P, NP, PSPACE, or EXP [12, 4, 9, 10]. We consider a slightly different decision problem than they do; our decision problem allows self-reductions to find the optimal policy, whereas theirs does not.

*Correspondence address: Judy Goldsmith, University of Kentucky, Dept. of Computer Science, 773 Anderson Hall, Lexington KY 40506-0046, goldsmit@cs.engr.uky.edu.

†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.

‡Supported in part by NSF grant CCR-9315354.

§Supported in part by NSF grant CCR-9509603.

# 1 Introduction

Markov Decision Processes (MDPs) are ubiquitous in the world of mathematical modeling, used by ecologists, economists, engineers, and even computer scientists, to name a few. Although they were only introduced in the early 1960's, there is a rich literature on them, spanning the mathematics, operations research, engineering, scientific, and social science communities. This literature concentrates on applications of MDPs, and algorithms for computing or approximating them. Very little of the literature discusses the inherent computational complexity of the model in its myriad variations, although there is certainly analysis of particular algorithms.

MDPs model controllable stochastic processes: there is a set of *states;* a controller chooses among some number of *actions,* each of which has an associated probability matrix of state transitions; associated with each state and action pair is a *reward.* The basic goal, given such a model, is to find a strategy or *policy* for choosing actions, that maximizes the total expected reward over some fixed time horizon. Policies may consider the current state only (stationary), the state and time (time-dependent), or the full decision history and current state (history-dependent).

We consider the complexity of the following decision problem: for a given MDP and type of policy, is there such a policy for that MDP with positive expected reward? If negative and nonnegative rewards are allowed, the optimization problem is reducible to the decision problem via binary search. First, one can find the expected value of the optimal policy, by appending a new initial state, with only one action which has a reward of -$c$, where $c$ is the value being tested for optimality. The number of iterations needed to find the optimal value is linear in the number of significant bits of $c$. Once one has the optimal value, one can test hypotheses about the policy itself by fixing one action (for a given state, state-time pair, or history sequence) at a time, and computing whether there is a policy of value $c$ for that reduced MDP.

We have considered finite horizon MDPs, and in fact only horizons of $n$ and $\log n$, where $n = |\mathcal{S}|$ is the size of the state space of the MDP. The question of $\log n$ horizons arises in the consideration of MDPs which have succinct representations such as circuits that generate the transition and reward tables, where the representation of the system may have size $O(\log n)$. (We differentiate between those succinctly represented systems where the rewards and transition probabilities can be written in $n$ bits, and in $\log n$ bits, when this distinction affects the complexity of the problem.)

To the best of our knowledge, the following papers are the only ones that give complexity results for finite horizon MDP problems. Papadimitriou and Tsitsiklis [12, 13] have considered MDPs with non-positive rewards, and asked whether there is a policy with expected value of 0. This problem does not allow the same self-reducibility as outlined above. Although many of their results and proof techniques extend to the problem we consider, not all do. They considered partially observable MDPs (POMDPs), and showed an NP-completeness result for the stationary, finite-horizon case [12]. Then they showed for history-dependent policies, that the fully observable MDP problem is P-complete (for finite or infinite horizon, and also for stationary policy), that the POMDP problem is PSPACE-complete, and that the unobservable case is NP-complete [13]. Tseng considered fully observable stationary *discounted* MDPPs. (This is a criterion for evaluating the performance of a policy, and is more commonly applied to infinite horizon MDPs.) His only result for undiscounted MDPs is an $\varepsilon$-approximation for the (not succinct) case. Littman showed that a restricted version of the stationary, $n$-horizon, POMDP problem is NP-complete [9], and that other variants are complete for PSPACE and EXP [10]. Beauquier, Burago and Slissenko examined the complexity of finite memory policies for POMDPs [3], and Burago, Rougemont and Slissenko showed a non-approximability result for unobservable MDPs [4].

As we have shown, here and elsewhere [6], there are many factors that contribute to the com-

| policy | observation | succinct | reward | horizon | actions | completeness | |
|---|---|---|---|---|---|---|---|
| s/t/h | null/full | – | + | $n$ | $n$ | NL | [6] |
| t/h | partial | – | + | $n$ | $n$ | NL | [6] |
| s | null | – | $-/+$ | $n$ | $n$ | PL | $*$ |
| s/t/h | full | – | $-/+$ | $n$ | $n$ | P | [13] |
| s | partial | – | + | $n$ | $n$ | NP | [6] |
| t/h | null | – | $-/+$ | $n$ | $n$ | NP | $*$ |
| s/t | partial | – | $-/+$ | $n$ | $n$ | NP | [6] |
| s/t/h | full/null/part. | $n$ | + | $\log n$ | $n$ | NP | [6] |
| s | null | $\log n$ | $-/+$ | $\log n$ | $\log n$ | PP | $*$ |
| t/h | null | $\log n$ | $-/+$ | $\log n$ | $\log n$ | $\mathrm{NP}^{\mathrm{PP}}$ | $*$ |
| s/t/h | null | $\log n$ | $-/+$ | $\log n$ | $n$ | $\mathrm{NP}^{\mathrm{PP}}$ | $*$ |
| h | partial | – | $-/+$ | $n$ | $n$ | PSPACE | [13, 6] |
| s/t/h | null | $n$ | + | $n$ | $n$ | PSPACE | $*$ |
| s/t/h | full | $n$ | + | $n$ | $n$ | PSPACE | [6] |
| t/h | partial | $n$ | + | $n$ | $n$ | PSPACE | [6] |
| h | full | $\log n$ | $-/+$ | $\log n$ | $n$ | PSPACE | [6] |
| h | partial | $\log n$ | $-/+$ | $\log n$ | $n$ | PSPACE | [6] |
| s/t/h | full | $n$ | $-/+$ | $n$ | $n$ | EXP | [6] |
| s | partial | $n$ | + | $n$ | $n$ | NEXP | [6] |
| s | partial | $n$ | $-/+$ | $\log n$ | $n$ | NEXP | [6] |
| s/t | partial | $n$ | $-/+$ | $n$ | $n$ | NEXP | [6] |
| t/h | null | $n$ | $-/+$ | $n$ | $n$ | NEXP | $*$ |
| h | partial | $n$ | $-/+$ | $n$ | $n$ | EXPSPACE | [6] |

Table 1: Completeness results

plexity of the decision problem, including: the scope of the policy; the length of time (horizon) under consideration; the amount of information – none, some, or complete – about the current state of the system (observability); the size of the representation of the system relative to the number of states; whether negative rewards are used; the number of actions relative to the size of the input; and the number of significant bits in the rewards and transition probabilities.

In all but one case, we have shown the decision problems to be complete for their respective classes. These classes range from NL to NEXP. In the case of some NP-complete problems, we have also shown that they are not $\varepsilon$-approximable for any $\varepsilon$ (unless P $=$ NP) [6].

The most interesting results are about the succinctly represented, unobservable, $\log n$-horizon MDPs. For these MDPs, the complexity depends on the length of the strings representing transition probabilities and rewards (we assume finite binary strings in both cases), and the number of possible actions. With few actions, the decision problem is PP-complete. With only slightly more actions, it is $\leq_m^p$-complete for the $\leq_m^{np}$-closure of PP.

We summarize our results from this paper, and that of Goldsmith, Lusena, and Mundhenk [6], in table 1. Each line stands for the completeness of (some) MDP problems. The number of states of the MDP is denoted by $n$. In the "policy" column, "s" stands for stationary, "t" for time-dependent, and "h" for history-dependent. The "succinct" column says whether the encoding of the MDP is not succinct (–), or whether the encoding is succinct and the rewards and transition probabilities have $\log n$ bits or $n$ bits.

We then give formal definitions in Section 2, and proofs of the starred results from the table.

## 2 Definitions and Preliminaries

For definitions of complexity classes, reductions, and standard results from complexity theory we refer to [11].

### 2.1 Markov Decision Processes

A Markov decision process (MDP) 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 $s$,
- $r : \mathcal{S} \times \mathcal{A} \to \mathbf{Z}$ 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), then the MDP is called *fully-observable*. Another special case is *unobservable MDPs*, 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. Without restrictions on the observability, an MDP is called *partially-observable*.[1]

### 2.2 Policies and Performances

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

- A *stationary policy* $\pi_s$ (for $M$) is a function $\pi_s : \mathcal{O} \to \mathcal{A}$, mapping each observation to an action.
- A *time-dependent policy* $\pi_t$ is a function $\pi_t : \mathcal{O} \times \mathbf{N} \to \mathcal{A}$, mapping each pair $\langle$observation, time$\rangle$ to an action.
- A *history-dependent policy* $\pi_h$ is a function $\pi_h : \mathcal{O}^* \to \mathcal{A}$, mapping each finite sequence of observations to an action.

Let $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, t, o, r)$ be an MDP. A *trajectory $\theta$ 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 $\text{prob}_\pi(M, \theta)$ of a trajectory $\theta = \sigma_1, \sigma_2, \ldots, \sigma_m$ under policy $\pi$ is

[1] Note that making observations probabilistically does not add any power to MDPs. Any probabilistically observable MDP can be turned into one with deterministic observations with only a polynomial increase in its size.

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

The *reward* $\text{rew}_\pi(M,\theta)$ of trajectory $\theta$ under $\pi$ is the sum of its rewards, i.e. $\text{rew}_\pi(M,\theta) = \Sigma_{i=1}^{m-1} r(\sigma_i, \pi(\cdot))$ dependent on the type of policy. The *performance of a policy* $\pi$ for finite-horizon $k$ with initial state $\sigma$ is the expected sum of rewards received on the next $k$ steps by following the policy $\pi$, i.e. $\text{perf}(M,\sigma,k,\pi) = \sum_{\theta\in\Theta(\sigma,k)} \text{prob}_\pi(M,\theta)\cdot\text{rew}_\pi(M,\theta)$, where $\Theta(\sigma,k)$ is the set of all length $k$ trajectories beginning with state $\sigma$. The *$\alpha$-value* $\text{val}_\alpha(M,k)$ of $M$ for horizon $k$ is $M$'s maximal performance under a policy $\pi$ of type $\alpha$ for horizon $k$ when started in its initial state, i.e. $\text{val}_\alpha(M,k) = \max_{\pi\in\Pi_\alpha} \text{perf}(M,s_0,k,\pi)$, where $\Pi_\alpha$ is the set of all $\alpha$ policies.

For simplicity, we assume that the size of an MDP is determined by the size $|\mathcal{S}|$ of its state space. We assume that there are no more actions than states, and that each state transition probability is given as binary fraction with $|\mathcal{S}|$ bits and each reward is an integer of $|\mathcal{S}|$ bits. This is no real restriction, since adding unreachable "dummy" states allows one to use more bits for transition probabilities and rewards. Also, it is straightforward to transform an MDP $M$ with non-integer rewards to $M'$ with integer rewards such that $\text{val}_\alpha(M,k) = c\cdot\text{val}_\alpha(M',k)$ for some constant $c$.

To encode an MDP one can use the "natural" encoding of functions by tables. Thus, the description of an MDP with $n$ states and $n$ actions requires $O(n^4)$ bits. For MDPs with sufficient structure, we can use the concept of *succinct representations* introduced by Papadimitriou and Yannakakis [14] (see also [18, 2]). In this case, the transition table of an MDP with $n$ states and actions is represented by a Boolean circuit $C$ with $4\lceil\log n\rceil$ input bits such that $C(s,a,s',l)$ outputs the $l$-th bit of $t(s,a,s')$. Encodings of those circuits are no larger than "natural" encodings, and may be much smaller, namely of size $O(\log n)$. A further special case of "small" MDPs are those with $n$ states where the transition probabilities and rewards need only $\log n$ bits to be stored. They can be represented by circuits as above which have only $3\lceil\log n\rceil + \lceil\log\log n\rceil$ input bits.

The *$\alpha$ $\beta$ Markov decision process problem* $\mathcal{M}[f(n), g(n)]$ is the set of all $\beta$-observable MDPs $M = (\mathcal{S},\mathcal{O},s_0,\mathcal{A},o,t,r)$, where $|\mathcal{A}| \leq g(|\mathcal{S}|)$ and $\text{val}_\alpha(M,s_0,f(|\mathcal{S}|)) > 0$, where $\alpha$ is a type of policy. For $\mathcal{M}$ we use MDPP if the MDP is in standard encoding, $s$MDPP for succinctly encoded instances, and $s_{\log}$MDPP for succinctly encoded instances where each transition probability and reward takes $\log|\mathcal{S}|$ many bits. $\mathcal{M}^+$ denotes the restriction to nonnegative rewards.

# 3 Unobservable MDPs under Stationary Policies

Each stationary policy for an unobservable MDP is one action. Thus, whether an unobservable MDP with nonnegative rewards has a positive stationary value reduces to a connectivity problem. The graph is obtained from the MDP by choosing an action, taking states as vertices, drawing an edge between vertices whose corresponding states have transition probability $> 0$ under the chosen action, marking all vertices corresponding to states which yield a reward $> 0$ under the chosen action. The question is whether a marked vertex can be reached from the initial vertex.

**Theorem 3.1** *1. The stationary unobservable* MDPP$^+[n,n]$ *is* NL*-complete.*

*2. The stationary unobservable* $s$MDPP$^+[\log n, n]$ *is* NP*-complete.*

*3. The stationary unobservable* $s$MDPP$^+[n,n]$ *is* PSPACE*-complete.*

If we allow positive and negative rewards, the situation becomes more complicated. Whereas in the above cases we had to find only one trajectory with positive reward, we now consider all trajectories.

We begin with some technical lemmas about matrix powering, and show that each element of the power of a nonnegative integer square matrix can be computed in #L, if the power is at most the dimension of the matrix.

**Lemma 3.2** *(cf. [17]) Let $p$ be a polynomial. Let $T$ be an $n \times n$ matrix of nonnegative binary integers, each of length $p(n)$, 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.

**Proof** Let $T, m, i, j$ be as stated. We claim that $\#acc_N(T, m, i, j) = (T^m)_{(i,j)}$ for the following nondeterministic machine $N$.

> **input** $T, m, i, j$
> **repeat** $m$ times
>   **guess** $k$, $1 \leq k \leq n$
>   split nondeterministically into $T(k, j)$ paths
>   $j := k$
> **end**
> *accept* iff $i = j$

$N$ runs in logspace. To prove the correctness of $N$, observe that on input $T, 0, i, j$, there is only one computation path, and this path is accepting iff $i = j$. Thus $\#acc_N(T, 0, i, j) = (T^0)_{(i,j)}$. For the induction step, note that $\#acc_N(T, m+1, i, j) = \sum_{k=1}^{n} \#acc_N(T, m, i, k) \cdot T(k, j)$. Using the induction hypothesis, it follows that the latter equals $\sum_{k=1}^{n} (T^m)_{(i,k)} \cdot T(k, j)$, which proves the claim. ■

**Lemma 3.3** *The stationary unobservable* MDPP$[n, 1]$ *is in* PL.

**Proof** Let $M = (\mathcal{S}, s_0, \{a\}, \mathcal{O}, t, o, r)$ be an unobservable MDP with only one action $a$. We show that $\mathrm{val}_s(M, |\mathcal{S}|) > 0$ can be decided in PL.

Since $M$ has only one action, the only policy to consider is the constant function mapping onto that action $a$. Therefore $\mathrm{val}_s(M, |\mathcal{S}|) = \mathrm{perf}(M, s_0, |\mathcal{S}|, a)$, and to compute $\mathrm{val}_s(M, |\mathcal{S}|)$, one can use dynamic programming. For the dynamic programming approach one uses 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 $h = |\mathcal{S}|$. Define the function $v$ as $v(M, i, 0) = 0$ and $v(M, i, m) = 2^{hm} r(i, a) + \sum_{j \in \mathcal{S}} v(M, j, m-1) \cdot 2^h t(i, a, j)$. Using induction, one can show that $\mathrm{perf}(M, i, m, a) = v(M, i, m) \cdot 2^{-hm}$. Therefore, $\mathrm{val}_s(M, |\mathcal{S}|) > 0$ iff $v(M, s_0, |\mathcal{S}|) > 0$.

We now show that the function $v$ is in GapL. Let $T$ be the matrix obtained from the transition matrix of $M$ by multiplying all entries by $2^h$, i.e. $T_{(i,j)} = t(i, a, j) \cdot 2^h$.

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

To prove the claim, we use induction. Since $v(M, i, 0) = 0$, the base case is clear. We continue with the induction step.

$$\begin{aligned}
v(M,i,m+1) &= 2^{(m+1)h} r(i,a) + \sum_{j\in\mathcal{S}} v(M,j,m)\cdot T(i,j) \\
&= 2^{(m+1)h} r(i,a) + \sum_{j\in\mathcal{S}} T(i,j)\cdot \sum_{k=1}^{m}\sum_{j'\in\mathcal{S}} (T^{k-1})_{(j,j')}\cdot r(j',a)\cdot 2^{(m-k+1)\cdot h} \\
&= 2^{(m+1)h} r(i,a) + \sum_{k=2}^{m+1}\sum_{j'\in\mathcal{S}} (T^{k-1})_{(i,j')}\cdot r(j',a)\cdot 2^{((m+1)-k+1)\cdot h} \\
&= \sum_{k=1}^{m+1}\sum_{j'\in\mathcal{S}} (T^{k-1})_{(i,j')}\cdot r(j',a)\cdot 2^{((m+1)-k+1)\cdot h} \qquad .
\end{aligned}$$

We claim that $v \in$ GapL. Each $T_{(i,j)}$ is logspace computable from the input MDP. From Lemma 3.2 we get that $(T^{k-1})_{(i,j)} \in$ GapL. Because the reward is part of the input, $r$ is also in GapL. Because GapL is closed under multiplication and polynomial summation, the claim follows. Finally, the characterization of PL as the class of all sets $A$ for which there is a GapL function $f$ such that for all $x$, $x \in A \Leftrightarrow f(x) > 0$ (see [1]) completes the proof. ■

**Lemma 3.4** *The stationary unobservable* MDPP$[n,1]$ *is* PL*-hard.*

**Proof** Consider $A \in$ 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 [7]. Now, fix some input $x$. We construct an MDP $M(x)$ modeling 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 counter for the number of random decisions 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 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 $\theta$ corresponding to accepting computations (independent of their length), or $-1$ for rejecting computations, or 0 otherwise. Let $M(x) = (\mathcal{S}, s_0, \mathcal{A}, \{o\}, t, o, r)$ be an unobservable MDP, where

$$\begin{aligned}
\mathcal{S} &= \{(c,j) \mid c \text{ is a configuration of } N \text{ on } x,\ 0\le j\le p(|x|)\} \cup \{q_{end}\} \\
s_0 &= (c_0, 0), \text{ where } c_0 \text{ is the initial configuration of } N \text{ on } x \\
\mathcal{A} &= \{a\} \\
t((c,i),a,(c',j)) &= \begin{cases} 1, & \text{if } c' \text{ is reachable from } c \text{ in one step by } N \text{ with prob. } 1, i=j \\ \frac{1}{2}, & \text{if } c' \text{ is reachable from } c \text{ in one step by } N \text{ with prob. } \frac{1}{2}, i+1=j \\ 0, & \text{otherwise} \end{cases} \\
t((c,i),a,q_{end}) &= \begin{cases} 1, & \text{if } c \text{ is a halting configuration} \\ 0, & \text{otherwise} \end{cases} \\
t(q_{end},a,q_{end}) &= 1 \\
r((c,i),a) &= \begin{cases} 2^i, & \text{if } c \text{ is an accepting configuration} \\ -(2^i), & \text{if } c \text{ is a rejecting configuration} \\ 0, & \text{otherwise.} \end{cases}
\end{aligned}$$

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 from the above arguments that $x \in A$ if and only if the number of trajectories $\theta$ of length $|\mathcal{S}|$ for $M(x)$ with $\mathrm{rew}(\theta) \cdot \mathrm{prob}(\theta) = 1$ is greater than the number of trajectories $\theta$ with $\mathrm{rew}(\theta) \cdot \mathrm{prob}(\theta) = -1$, which is equivalent to $\mathrm{val}_s(M(x), |\mathcal{S}|) > 0$. ■

**Theorem 3.5** *The stationary unobservable* MDPP$[n, 1]$ *and* MDPP$[n, n]$ *are* PL*-complete.*

**Proof** MDPP$[n, n] \in$ PL follows from Lemma 3.3 and the closure of PL under logspace disjunctive reducibility (see [1]): from an MDP with a set of actions $\mathcal{A} = \{a_1, \ldots, a_n\}$ of size $n$ one can produce $n$ MDPs $M_1, \ldots, M_n$ with the same state space, where each of these MDPs has exactly one action. There is a policy $\pi = a_i$ such that $\mathrm{perf}(M, s_0, |\mathcal{S}|, \pi) > 0$ if and only if $\mathrm{val}_s(M_i, |\mathcal{S}|) > 0$ for some $i$. Lemma 3.4 completes the proof. ■

Next we consider succinctly represented MDPs. To perform calculations on their transition probabilities and rewards in polynomial space, we need to restrict the number of bits in the representations of the probabilities and rewards. The number of actions which can be chosen also affects the complexity of the problem. Again, we first consider the complexity of powering succinctly represented matrices.

**Lemma 3.6** *Let $T$ be a $2^n \times 2^n$ matrix of nonnegative integers, each consisting of $n$ bits, $1 \leq i, j \leq 2^n$, and $0 \leq m \leq n$. Let $T$ be represented by a Boolean circuit $C$ with $2n + \lceil \log n \rceil$ input bits, such that $C(a, b, r)$ outputs the $r$-th bit of $T_{(a,b)}$. The function mapping $(T, m, i, j)$ to $(T^m)_{(i,j)}$ is in* #P.

**Proof** The proof is similar to the proof of Lemma 3.2. The splitting into $T(a, j)$ paths can be done iteratively, reading the bits of $T(a, j)$. Since the power is bounded by $n$, the computation tree has depth polynomial in $n$, which is a lower bound on the size of the input, i.e., the circuit $C$. ■

**Lemma 3.7** *The stationary unobservable* $s_{\log}$MDPP$[\log n, 1]$ *is* PP*-hard.*

**Proof** We show a reduction from the PP-complete set MAJSAT. Let $\phi(x_1, \ldots, x_n)$ be a Boolean formula. From $\phi$, construct an MDP $M(\phi)$ with one action $a$ and states $\mathcal{S} = \{0, 1\}^{\leq n} \cup \{q\}$, where $q$ is a final (trap) state. Let $\varepsilon$ be the initial state. For each state $u \in \{0, 1\}^{<n}$, the transition function $t$ is defined by $t(u, a, w) = \frac{1}{2}$ if $w \in \{u0, u1\}$, and $t(u, a, w) = 0$ otherwise. The reward is $r(u, a) = 0$. For states $u \in \{0, 1\}^{=n}$, the transitions are $t(u, a, q) = 1$ and $t(u, a, w) = 0$ for $w \neq q$. The reward is $r(u, a) = 1$, if $\phi(u) = 1$, and $r(u, a) = -1$, if $\phi(u) = 0$.

Then $\phi \in$ MAJSAT iff $\mathrm{val}_s(M(\phi), \log |\mathcal{S}|) > 0$. ■

**Lemma 3.8** *The stationary unobservable* $s_{\log}$MDPP$[\log n, \log n]$ *is in* PP.

**Proof** To show that $s_{\log}$MDPP$[\log n, 1] \in$ PP, we use arguments similar to those in the proof of Lemma 3.3. We make use of the fact that $A \in$ PP if and only if there exists a GapP funtion $f$ such that for every $x$, $x \in A \leftrightarrow f(x) > 0$ (see [5]). For the function $v$ from the proof of Lemma 3.3, one can show that it is in GapP under these circumstances, because the respective matrix powering is in #P (Lemma 3.6), and GapP is closed under multiplication and summation. Finally, PP is closed under polynomial-time disjunctive reducibility, which completes the proof. ■

From Lemma 3.8 and Lemma 3.7 follows

**Theorem 3.9** *The stationary unobservable* $s_{\log}$MDPP$[\log n, 1]$ *and* $s_{\log}$MDPP$[\log n, \log n]$ *are* PP*-complete.*

Omitting the restriction on the number of actions, the complexity of the $s_{\log}$MDPP rises from PP to $\mathrm{NP}^{\mathrm{PP}}$.

**Lemma 3.10** *The stationary unobservable* $s_{\log}\mathrm{MDPP}[\log n, n]$ *is in* $\mathrm{NP}^{\mathrm{PP}}$.

**Proof** We give a short sketch of the reduction. Let the circuit $C$ (describing an MDP) be given. Guess an action $a$, and produce an MDP by "hard-wiring" $a$ into the transition table of $C$. This new MDP with only one action behaves like the one described by $C$ for policy $a$. Thus, $s_{\log}\mathrm{MDPP}[\log n, n] \leq_m^{np} s_{\log}\mathrm{MDPP}[\log n, 1]$. From Theorem 3.9, the claim follows. ■

**Lemma 3.11** *The stationary unobservable* $s_{\log}\mathrm{MDPP}[\log n, n]$ *is hard for* $\mathrm{NP}^{\mathrm{PP}}$.

**Proof** Following a result from [16], $\mathrm{NP}^{\mathrm{PP}}$ equals the $\leq_m^{np}$ closure of PP. Thus for any $A$ in $\mathrm{NP}^{\mathrm{PP}}$, by Lemma 3.7 there exists a polynomial $q$ and an FP function $f$ such that for every $x \in \Sigma^*$

- $f(x, y)$ is the succinct description of an unobservable MDP $F_{x,y}$ having only one action, for every $y \in \Sigma^{q(|x|)}$, and
- $x \in A$ if and only if $\exists y \in \Sigma^{q(|x|)}$ such that $\mathrm{val}_s(F_{x,y}) > 0$.

For simplicity and w.l.o.g. we can assume that $F_{x,y}$ has states $\mathcal{S}_{F_{x,y}} = \{0,1\}^{p(|x|)}$, initial state $0^{p(|x|)}$, and each entry in the transition table has $p(|x|)$ bits, for some polynomial $p$. Also, let $a_0$ be the only action which can be taken in any $F_{x,y}$. Furthermore, we assume that after at most $\log|\mathcal{S}|$ steps the process reaches a trap state in which it stays with probability 1 without receiving any additional rewards. This can be achieved by making $\log|\mathcal{S}|$ copies of each state $s$, such that the $i$-th copy of $s$ is reached after $i$ steps if $s$ is reached after $i$ steps in the original MDP. Each transition from the last copy of each $s$ leads to the trap state.

We define an MDP $M(x)$ which under policy $a$ behaves exactly as $F_{x,a}$. Thus $\mathrm{val}_s(M(x), \log n) > 0$ iff $\mathrm{val}_s(F_{x,a}, \log n) > 0$ for some $a \in \Sigma^{p(|x|)}$, and therefore $x \in A$ iff $M(x) \in s_{\log}\mathrm{MDPP}[\log n, n]$. Let $M(x) = (\mathcal{S}, s_0, \mathcal{A}, \{o\}, t, o, r)$ be an unobservable MDP whith

$$
\begin{array}{rcl}
\mathcal{S} & = & s_0 \cup \{(y, s) \mid y, s \in \{0,1\}^{p(|x|)}\} \\
\mathcal{A} & = & \{0,1\}^{p(|x|)} \\
t(s_0, a, (s, s')) & = & \begin{cases} t_{F_{x,a}}(0^{p(|x|)}, a_0, s'), & \text{if } a = s \\ 0, & \text{otherwise} \end{cases} \\
t((s, s'), a, (t, t')) & = & \begin{cases} t_{F_{x,s}}(s', a_0, t'), & \text{if } s = t \\ 0, & \text{otherwise} \end{cases} \\
r(s, a) & = & \begin{cases} r_{F_{x,a}}(0^{p(|x|)}, a_0), & \text{if } s = s_0 \\ r_{F_{x,s'}}(t', a_0), & \text{if } s = (s', t') \\ 0, & \text{otherwise,} \end{cases}
\end{array}
$$

where $t_{F_{x,s}}$ and $r_{F_{x,s}}$ are the transition and reward functions for $F_{x,s}$, whose succinct descriptions are computed by $f(x, s)$. From this each bit of the transition probabilities and rewards can be computed.

From this description, it follows that $M(x)$ has a succinct description which can be computed from $x$ in polynomial time. ■

**Theorem 3.12** *The stationary unobservable* $s_{\log}\mathrm{MDPP}[\log n, n]$ *is* $\leq_m^p$*-complete for* $\mathrm{NP}^{\mathrm{PP}}$.

From Toda's result that $\mathrm{PH} \subseteq \mathrm{P}^{\mathrm{PP}}$ [15] and Theorem 3.12 we have the following.

**Corollary 3.13** *The stationary unobservable* $s$MDPP$[\log n, n]$ *is* $\leq_m^p$*-hard for* PH.

The techniques used to show PL-completeness translate to succinctly represented MDPs with small transition probabilities and rewards.

**Lemma 3.14** *The stationary unobservable* $s_{\log}$MDPP$[n, 1]$ *is* PSPACE*-hard.*

**Proof** Let $A = L(N)$ for a polynomial-space bounded Turing machine. For input $x$, construct an MDP $M(x)$ which has the possible configurations of $N$ on $x$ as states and one action $a$. The transition function "simulates" the configuration transition of $N$. I.e. $t(s, a, s') = 1$ if configuration $s'$ can be reached from $s$ in one step by $N$. A reward 1 is received when a state corresponding to an accepting configuration is reached. There is exactly one trajectory of $M(x)$ which corresponds to the computation of $N$ on $x$ and has reward 1 iff $N$ accepts $x$. ∎

**Lemma 3.15** *The stationary unobservable* $s_{\log}$MDPP$[n, n]$ *is in* PSPACE.

**Proof** We can use the same technique as in the proof of Lemma 3.3 to show that $s$MDPP$[n, 1]$ is in PPSPACE. Here it is important that the transition probabilities and rewards are logarithmically small in the number of states. Ladner [8] showed that FPSPACE = #PSPACE, from it follows that PPSPACE = PSPACE. Thus, $s_{\log}$MDPP$[n, 1]$ is in PSPACE. Because PSPACE is closed under "or," the claim follows. ∎

**Theorem 3.16** *The stationary unobservable* $s_{\log}$MDPP$[n, 1]$ *and* $s_{\log}$MDPP$[n, n]$ *both are* PSPACE*-complete.*

Note that the same problems for MDPs with nonnegative rewards are also PSPACE-complete (see Theorem 3.1).

## 4 Unobservable MDPs under time-dependent policies

For unobservable MDPs, there is no difference between their value under history-dependent policies and under time-dependent policies. Because in every step the same observation is made, only the number of observations yields information.

Theorem 3.1 translates to time-dependent policies.

**Theorem 4.1**

1. *The time-dependent unobservable* MDPP$^+[n, n]$ *is* NL*-complete.*
2. *The time-dependent unobservable* $s$MDPP$^+[\log n, n]$ *is* NP*-complete.*
3. *The time-dependent unobservable* $s$MDPP$^+[n, n]$ *is* PSPACE*-complete.*

**Theorem 4.2** *The time-dependent unobservable* MDPP$[n, n]$ *is* NP*-complete.*

Papadimitriou and Tsitsiklis proved a similar theorem [13]. Their MDPs had only non-positive rewards, and their formulation of the decision problem was whether there was a policy with reward 0. However, our result can be proved by a proof very similar to theirs. The reduction is from 3SAT. The fact that the problem is in NP follows from the fact that a time-dependent policy with performance $> 0$ can be guessed and checked in polynomial time.

**Theorem 4.3** *The time-dependent unobservable* $s_{\log}$MDPP$[\log n, \log n]$ *is* $\mathrm{NP}^{\mathrm{PP}}$*-complete.*

**Proof** Let $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, t, o, r)$ be an unobservable MDP. Each time-dependent policy for $n = |\mathcal{S}|$ steps is a list $\pi = a_0, \ldots, a_{n-1}$ of $n$ actions from $\mathcal{A}$. For this policy, we can construct an MDP $M_\pi$ with only one action, which has the same performance after $n$ steps as $M$. Essentially, $M_\pi$ consists of $n$ layers of $M$, such that each transition leads from a layer to its successor. From the last layer only the trap-state can be reached. Let $M_\pi = (\mathcal{S}, s_0, \mathcal{A}, \{o\}, t, o, r)$ be an unobservable MDP whith

$$
\begin{array}{lcl}
\mathcal{S}' & = & \{(s,j) \mid 0 \leq j \leq |\mathcal{S}|, s \in \mathcal{S}\} \cup \{q_{end}\} \\
s_0' & = & (s_0, 0) \\
\mathcal{A}' & = & \{a\} \\
t'(s,a,s') & = & \begin{cases} t(u, a_i, v), & \text{if } s = (u,i),\ s' = (v, i+1) \\ 1, & \text{if } s \in \{(u, |\mathcal{S}|), q_{end}\},\ s' = q_{end} \\ 0, & \text{otherwise} \end{cases} \\
r'(s,a) & = & \begin{cases} r(u, a_i), & \text{if } s = (u,i) \\ 0, & \text{otherwise.} \end{cases}
\end{array}
$$

We claim that $\mathrm{perf}(M, s_0, n, \pi) = \mathrm{val}_t(M_\pi, |\mathcal{S}'|)$. Thus $\mathrm{val}_t(M, n) = \max_{\pi \in \Pi} \mathrm{val}_t(M_\pi, |\mathcal{S}'|)$, where $\Pi$ is the set of all time-dependent policies for $n$ steps for $M$. Since the time-dependent policy has length polynomial in $n$, it follows that the time-dependent unobservable $s_{\log}\textsc{Mdpp}[\log n, \log n]$ $\leq_m^{np}$-reduces to the stationary unobservable $s_{\log}\textsc{Mdpp}[\log n, 1]$. Since the latter is in PP (Lemma 3.8), it follows that the time-dependent unobservable $s_{\log}\textsc{Mdpp}[\log n, \log n]$ is in $\mathrm{NP}^{\mathrm{PP}}$.

$\mathrm{NP}^{\mathrm{PP}}$ hardness follows from Theorem 3.12. ■

Essentially the same proof works for an unrestricted number of different actions.

**Theorem 4.4** *The time-dependent unobservable* $s_{\log}\textsc{Mdpp}[\log n, n]$ *is* $\mathrm{NP}^{\mathrm{PP}}$*-complete.*

**Theorem 4.5** *The time-dependent unobservable* $s\textsc{Mdpp}[n, n]$ *is* NEXP*-complete.*

**Proof** Since succint 3SAT is complete for NEXP (see [11]), the proof of Theorem 4.2 can be translated to its succinct version. ■


## Acknowledgements

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


## References


[1] E. Allender and M. Ogihara. Relationships among PL, #L, and the determinant. *Theoretical Informatics and Applications*, 30(1):1–21, 1996.

[2] J.L. Balcázar, A. Lozano, and J. Torán. The complexity of algorithmic problems on succinct instances. In R. Baeza-Yates and U. Manber, editors, *Computer Science*, pages 351–377. Plenum Press, 1992.

[3] D. Beauquier, D. Burago, and A. Slissenko. On the complexity of finite memory policies for Markov decision processes. In *Mathematical Foundations of Computer Science*, 1995.

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

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

[6] J. Goldsmith, C. Lusena, and M. Mundhenk. The complexity of deterministically observable finite-horizon Markov decision processes. Technical Report 269-96, University of Kentucky Department of Computer Science, 1996.

[7] H. Jung. On probabilistic time and space. In *Proceedings 12th ICALP*, pages 281–291. Lecture Notes in Computer Science, Springer-Verlag, 1985.

[8] R. Ladner. Polynomial space counting problems. *SIAM Journal on Computing*, 18:1087–1097, 1989.

[9] M.L. Littman. Memoryless policies: Theoretical limitations and practical results. In *From Animals to Animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*. Lecture Notes in Computer Science, MIT Press, 1994.

[10] M.L. Littman. Probabilistic strips planning is exptime-complete. Technical Report CS-1996-18, Duke University Department of Computer Science, November 1996.

[11] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[12] C.H. Papadimitriou and J.N. Tsitsiklis. Intractable problems in control theory. *SIAM Journal of Control and Optimization*, pages 639–654, 1986.

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

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

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

[16] J. Torán. Complexity classes defined by counting quantifiers. *Journal of the ACM*, 38(3):753–774, 1991.

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

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