Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Artificial Intelligence 147 (2003) 5–34

Artificial Intelligence

www.elsevier.com/locate/artint

On the undecidability of probabilistic planning and related stochastic optimization problems

Omid Madani [a,*], Steve Hanks [b], Anne Condon [c]

[a] *Department of Comp. Sci., University of Alberta, Edmonton, AB T6G 2E8, Canada*
[b] *Institute of Technology, University of Washington, Tacoma, WA 98402-3100, USA*
[c] *Department of Computer Science, 2366 Main Mall, University of British Columbia, Vancouver, BC V6T 1Z4, Canada*

Received 22 June 2001

Abstract

Automated planning, the problem of how an agent achieves a *goal* given a repertoire of *actions*, is one of the foundational and most widely studied problems in the AI literature. The original formulation of the problem makes strong assumptions regarding the agent's knowledge and control over the world, namely that its information is complete and correct, and that the results of its actions are deterministic and known. Recent research in planning under uncertainty has endeavored to relax these assumptions, providing formal and computation models wherein the agent has incomplete or noisy information about the world and has noisy sensors and effectors. This research has mainly taken one of two approaches: extend the classical planning paradigm to a semantics that admits uncertainty, or adopt another framework for approaching the problem, most commonly the Markov Decision Process (MDP) model. This paper presents a complexity analysis of planning under uncertainty. It begins with the "probabilistic classical planning" problem, showing that problem to be formally undecidable. This fundamental result is then applied to a broad class of stochastic optimization problems, in brief any problem statement where the agent (a) operates over an infinite or indefinite time horizon, and (b) has available only probabilistic information about the system's state. Undecidability is established for policy-existence problems for partially observable infinite-horizon Markov decision processes under discounted and undiscounted total reward models, average-reward models, and state-avoidance models. The results also apply to corresponding approximation problems with undiscounted objective functions. The paper answers a significant open question raised by Papadimitriou and Tsitsiklis [Math. Oper. Res. 12 (3) (1987) 441–450] about the complexity of infinite horizon POMDPs.
© 2003 Elsevier Science B.V. All rights reserved.

* Corresponding author.
E-mail addresses: madani@cs.ualberta.ca (O. Madani), hanks@u.washington.edu (S. Hanks), condon@cs.ubc.ca (A. Condon).

0004-3702/03/$ – see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/S0004-3702(02)00378-8

Keywords: Probabilistic planning; Undecidability; Computability; Markov decision processes; Computational complexity; Infinity-horizon; Partial observability; Unobservability; Stochastic optimization; Discounted

1. Introduction

The *planning problem* is one of the oldest and most studied problems in the AI research literature; a vast body of work has been devoted to establishing theoretical results (the worst-case time complexity of various problem classes), finding representations and algorithms that effectively solve important classes of problems, and evaluating those representations and algorithms empirically.

Until recently planning was almost exclusively posed as a problem of logical deduction: given an initial state of the world, actions that effect state changes, and a set of states identified as *goal states*, find a sequence of actions that (provably) leave the world in (any) one of goal states provided that it begins in the initial state.

Strong assumptions implicitly underly this model: the initial state is known, the exact effects of each action is known, and there are no unknown exogenous changes in state. As a result, the agent is omniscient in its world (at least with respect to all variables relevant to achieving its goal), and finding a solution plan amounts to constructing a *deduction* that executing the plan in the known initial state would leave the world in a known goal state.

An interesting side effect of this model is that observing the world state during plan execution is unnecessary: no model of sensing the world is required because the agent always knows ahead of time what state the world will be in as it executes its plan. The agent's flawless ability to predict compensates for its inability to sense the world (and thus to make action decisions at run time).

There are obvious limitations to this problem definition as a model of human or machine agency: rarely can the world be so completely known or so thoroughly controlled.

To address these limitations, various alternative models have been introduced. One line of inquiry extends the classical logical model to admit incomplete information about the world state, extending the planning paradigm to include observation actions and contingency plans. These models represent incomplete information by modeling the agent's state of information as a set of states (the true world state must be one of the states in this set). These models allow reasoning about incomplete information about the world's initial state, but generally do not address issues associated with uncertainty about action effects or about exogenous change in the world (e.g., effected by other agents).

Taking this approach a step further, one can adopt a probabilistic semantics: the agent's state of information becoming a distribution over world states and the actions becoming stochastic state transitions. The Buridan work [24] presents this model under the classical planning assumption that the agent is unable to observe the world's state during plan execution; C-Buridan [17] extends the work to allow a plan to contain stochastic sensing actions.

A third line of work (e.g., [9,12,21,26,50]) discards the classical planning model altogether and studies control models—in particular Markov Decision Processes—as an alternative way to think about planning scenarios. In doing so, the logical semantics, omniscience assumptions, and strict goal orientation of classical planning are replaced

by a probabilistic semantics and an alternative set of assumptions. Most notable among these is the assumption of *full observability* taken from the most common MDP models (FOMDPs): although uncertainty prevents the agent from predicting the run-time state ahead of time, it has complete and accurate information about that state at execution time. While researchers have also applied partially observable (POMDP) models to AI planning problems, the apparent complexity of planning with partial observability has limited the practical application of this work.

There has been extensive analysis of the computational complexity of classical (deterministic) planning problems, e.g., [11], and likewise the complexity of common MDP problems is well understood: fully observable MDPs are analyzed by Papadimitriou and Tsitsiklis [40], and their work and later work by Mundhenk et al. [39] and Littman et al. [29] analyzes partially observable MDPs but provides results only for finite-horizon problems or bounded-length plans. See Section 5 for a discussion of previous work on infinite-horizon partially observable MDP's.

Probabilistic classical planning is not covered by either of these lines of analysis due to its probabilistic semantics and to the fact that goal-pursuit planning cannot be reduced to a problem of planning over a fixed finite horizon. That is, the length of a solution plan cannot in general be bounded *a priori*.

This paper begins with a complexity analysis of this probabilistic planning problem. Given:

- a set of states,
- a probability distribution over the identity of the initial state,
- a set of goal states,
- a set of operators that effect stochastic state transitions, and
- a rational threshold τ on the probability of plan success,

determine whether there is a sequence of operators that will leave the system in a goal state with probability at least τ.

The paper's main result proves the undecidability of this problem, using an existing result establishing the undecidability of an isomorphic problem in the area of Probabilistic Finite-State Automata (PFAs).

The result applies more widely, however, to a variety of problems in stochastic planning and control. For example, the result can be extended to resolve an open problem posed in [40] regarding the undecidability of infinite-horizon POMDPs and a problem about PFAs called the threshold isolation problem [41,44].

Fig. 1 depicts the problems analyzed in this paper. In the figure, the problems in bold rectangles are those established as undecidable in this paper. The oval is the existing result, and the rounded rectangles represent problems with well-studied complexity results discussed to put our results in context. Arrows point from "easier" to "harder" problems. The paper begins with the existing result regarding the undecidability of determining whether the set of input strings accepted by a PFA is empty. It then shows that an equivalent probabilistic planning problem is undecidable, and so too are more general classes of "unobservable" and partially observable MDP problems. This result applies

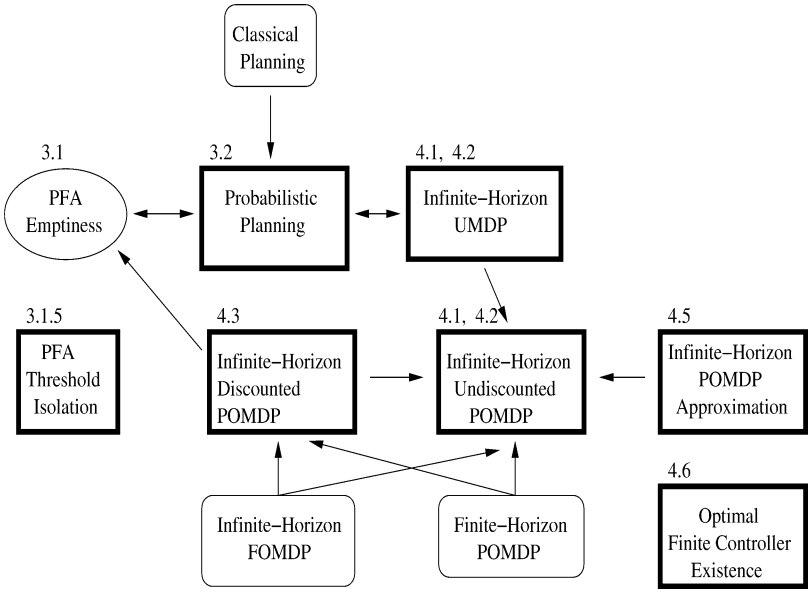

Fig. 1. Summary of Undecidability Results. Problems in bold rectangles are those established as undecidable in this paper, with the proofs starting from the result in the oval. In the rounded rectangles are related problems with previously known complexity results. Arrows point from "easier" to "harder" problems. Above each problem is the section number where the problem is addressed.

both to optimization and approximation versions of the problems, and to discounted and undiscounted objective functions.

The paper is organized as follows. Section 2 defines the relevant stochastic optimization problems: fully and partially observable MDPs, probabilistic planning, and PFAs. Section 3 begins by explaining the undecidability proof for the PFA emptiness problem, extends the result to the PFA threshold-isolation problem, then establishes the undecidability of the probabilistic planning problem. Section 4 then extends the result to various POMDP problems: discounted and undiscounted objective functions, related approximation problems, and questions of existence of optimal policies with special structure. Section 5 presents related work and conclusions.

2. Problem definitions

We make the standard assumption that the input parameters in all problems are rational numbers. We will be using matrices and vectors at several points, but only familiarity with elementary linear algebra is needed. We use boldface font to denote vectors. For a vector $\mathbf{x}$, $\mathbf{x}^{\mathrm{T}}$ denotes its transpose, and $\mathbf{x}[i]$ denotes its ith component. $\mathbf{M}[i, j]$ denotes the entry in row i and column j of matrix $\mathbf{M}$, and $\mathbf{M}[i, .]$ denotes the vector corresponding to row i of the matrix. We use the parenthesized superscript notation for sequences of objects such as

scalars and vectors. For example, $o^{(j)}$ denotes the jth object of the sequence $o^{(1)}, o^{(2)}, \ldots$, and $\{o^{(i)}\}$ refers to the whole sequence. Sequences may be finite or infinite.

We use Σ to denote a finite set of symbols (an alphabet). In this case, a sequence is treated as a juxtaposition (or concatenation) of the symbols of Σ, for example, $w = \sigma_1\sigma_2\ldots$, where $\sigma_i \in \Sigma$. A string is a finite sequence, possibly empty. If w and w' are two sequences, were w is finite, then ww' is the sequence corresponding to their concatenation, w^k denotes w concatenated with itself k times, and w^∞ denotes the infinite concatenation of w with itself, or $w^\infty = wwww\ldots$. With sequence $w = \sigma_1\sigma_2\ldots$, where $\sigma_i \in \Sigma$, w_i denotes the ith element of w, or $w_i = \sigma_i$, $|w|$ denotes the length of sequence w, i.e., the number of symbols in w, and $pre(w, i)$ denotes the length i prefix of w, thus $pre(w, i) = \sigma_1\sigma_2\ldots\sigma_i$. Σ^* denotes the set of all finite sequences (strings) of elements in Σ, including the empty string.

2.1. Probabilistic planning

This work was originally motivated by questions about the computability of probabilistic planning problems, such as the problems introduced in [9,24].

The probabilistic planning problem, studied by Kushmerick, Hanks, and Weld [24] for example, consists of

- a finite set of states (or equivalently, an assignment of truth values to a finite number of Boolean variables),
- a finite set of actions effecting stochastic state transitions,
- a start state (or probability distribution over states),
- a goal region of the state space (a subset of states designated as goal), and
- a threshold τ on the probability of success.

The problem is to find *any* sequence of actions that would move the system from the start state to a goal state with probability exceeding τ. While this abstract form of the problem, described in terms of system states and transition among the states, may seem dissimilar to what often appears in the literature on probabilistic planning in AI, most other variations can be seen to be at least as hard in terms of worst case computational complexity. When viewed from the point of the underlying states, the probabilistic planning problem, as just described, can be thought as a special *Markov decision process* problem.

2.2. Markov decision processes

A Markov decision process (MDP) model consists of a set Q of n states, and a finite set of actions, Σ. Time is discretized and at each time point t, $t = 0, 1, \ldots$, the system occupies a single state in Q, which is called the (current) state of the system at time t. The means of change in system state is action execution, and there is uncertainty about the outcome of actions. Consider the states indexed: $s_i \in Q$, $1 \leqslant i \leqslant n$. Associated with each action $a \in \Sigma$ is an $n \times n$ stochastic matrix $\mathbf{M_a}$ which specifies the state transition probabilities for action a. $\mathbf{M_a}[i, j]$ has the semantics that if the state of the system is s_i at a given time point t and action a is executed, with probability $\mathbf{M_a}[i, j]$ the state of the system

at time point $t+1$ is s_j. We refer to $\mathbf{M_a}$ as the transition matrix or the dynamics (matrix) of action a. Also associated with each action a is an $n \times 1$ vector of rewards $\mathbf{R_a}$, with the semantics that the decision maker gains $\mathbf{R_a}[i]$ (or incurs cost $\mathbf{R_a}[i]$ when $\mathbf{R_a}[i] < 0$), if the state of the system is s_i and action a is executed. MDP models satisfy the *Markov property*: the next state of the system, and other aspects such as the reward attained and the observation made (see below), depend only on the current state of the system and the action executed, and not, for example, on the history of previous system states and executed actions.

The states and actions (basically the transition matrices and the reward vectors) are completely specified as part of the problem instance. Informally, the problem of a decision maker, confronted with such a system, is to execute actions that maximize some measure of accumulated reward—for example, the expected total reward—over some fixed number of time steps or over an indefinite number of time steps. Observe that in making a choice of action, not only the immediate reward of an action is important, but also the state the action leads to may be significant in obtaining a high reward in the future.

The reader is referred to books on MDPs and POMDPs, such as Puterman [42], White [49], and Bertsekas [6], for a more comprehensive introduction to the models and motivational examples. We will next describe two important aspects, state observability, and the time horizon and value functions, over which model specifications vary, and define a few computational problems.

2.2.1. Observability

In the traditional *fully observable* MDP model, the system state at each time point is known to the decision maker at the time the next action is selected and executed. In the partially observable (POMDP) generalization, the decision maker may have only partial information about the system state, for example a probability distribution over possible states. Consequently, in the POMDP model, there are two sources of uncertainty: uncertainty about the outcome of actions and, at each time point, uncertainty over the current system state. In general, partial observability makes the life of the decision maker difficult, as the current system state may not be known, and, for example, the decision maker may need to act conservatively.

Generally in POMDPs, after each action execution, the decision maker obtains some feedback, possibly erroneous, on the current state of the system. In common POMDP models, a state s_i emits an *observation* (signal) $O_j \in \mathcal{O}$, where $\mathcal{O}$ is a finite observable set. The source of partial observability is the possibility that multiple states may emit the same observation. Traditionally, the observation is modeled as a random variable depending on the state or on both the state and the action executed. The distributions of the random variables are given as part of the problem instance. The fully observable MDP model corresponds to an extreme case where each state maps to a unique observable.

In this paper, we consider another extreme of partial observability, the unobservable MDP (UMDP) model, where all states map to the same observation. The UMDP model simplifies our analyses of computability and is closely related to the probabilistic finite automaton model defined in Section 2.3. Of course, the hardness results that we establish apply to the more general POMDP variations as well. Note also that the probabilistic

planning problem as described in Section 2.1 can also be viewed as a UMDP with a special value function (see Section 2.2.4).

We always assume that the decision maker is given a probability distribution over the initial state. For the undecidability results, the decision maker may even know the initial state. Let a state probability distribution at time t be denoted by the $1 \times n$ vector $\mathbf{x}^{(t)}$, $\mathbf{x}^{(t)}[i]$ is the probability that the system is in state s_i at time t. Given probability distribution $\mathbf{x}^{(t)}$ at time t, and execution of action a at time t, the probability distribution at time $t+1$ is $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}\mathbf{M_a}$. Thus by knowing current probability distribution and action, the next probability distribution is easily computed. Therefore in UMDPs, the distribution $\mathbf{x}^{(t)}$, computed from the initial probability distribution $\mathbf{x}^{(0)}$ and the sequence of actions executed till time t, is the only information available to the decision maker about the system's state at time t.

2.2.2. Infinite-horizon criteria

In this paper, we will be interested in *infinite-horizon* optimization objectives or criteria, that is, we assume the decision maker executes actions in the system over an indefinite or an unbounded number of time steps, and in case of UMDPs we will consider both finite and infinite sequences of actions. This contrasts with *finite-horizon* objectives, where the decision maker executes a fixed and known number of actions. The complexity of finite horizon MDPs and POMDPs has been extensively studied; see for example [26,38,40].

2.2.3. Measures of value of action sequences

In solving planning and MDPs problems, we need a measure of value of an action sequence to formulate optimization objectives and computational problems. We will sometimes refer to a choice of value measure as an *optimality criterion*. Once a value measure is defined, an optimization objective becomes selecting the action sequence with the highest value in the collection of possible action sequences. We will now discuss several value measures for action sequences and introduce notation for expressing them.

Most MDP models adopt an *additive* value model in which the overall value of executing a policy is expressed as a function on the sum of the rewards collected at each stage. In this model, execution of an action a, given a distribution $\mathbf{x}$ over system state, gives an expected reward of $\sum_{1 \leqslant i \leqslant n} \mathbf{x}[i]\mathbf{R_a}[i]$, which can be expressed compactly using vector product notation as $\mathbf{xR_a}$.

Let w denote a nonempty finite or infinite sequence of actions. By execution of the sequence w we mean that the ith action, w_i, is executed at time $t = i-1, i \geqslant 1$, so w_1 is executed at time 0, w_2 at time 1, and so on. Let us first express the total reward value of w, which is the total expected reward obtained when initial system distribution is $\mathbf{x}^{(0)}$ and w is executed. Let $\mathbf{R_{w_i}}$ and $\mathbf{M_{w_i}}$ denote the reward vector and the dynamics matrix of the ith action in w respectively. Then $\mathbf{x}^{(i-1)}\mathbf{R_{w_i}}$ is the expected reward from executing the ith action w_i at time $t = i-1$. Note that $\mathbf{x}^{(i)}$ is determined by w and $\mathbf{x}^{(0)} : \mathbf{x}^{(i)} = \mathbf{x}^{(i-1)}\mathbf{M_{w_i}}$. We use the function $\mathcal{V}$ to denote the expected reward of w when initial distribution is $\mathbf{x}^{(0)}$. We have:

$$\mathcal{V}\big(w, \mathbf{x}^{(0)}\big) = \sum_{i=1}^{|w|} \mathbf{x}^{(i-1)}\mathbf{R_{w_i}}, \tag{1}$$

where if $|w| = \infty$, it is understood that the limit is taken, if well defined. This expected total reward measure is called the *total reward criterion*.

A common and well-studied variation on the optimality criterion, called the *total discounted reward* value, or the *discounted criterion* for short, is always well defined in the presence of arbitrary reward and state transition structure. Under this criterion, a discount factor β, $0 \leqslant \beta < 1$, is used to discount future rewards. The value of a sequence w, at an initial distribution $\mathbf{x}^{(0)}$ is then expressed as:

$$\mathcal{V}_\beta\big(w, \mathbf{x}^{(0)}\big) = \sum_{i=1}^{|w|} \beta^{i-1} \mathbf{x}^{(i-1)} \mathbf{R}_{\mathbf{w_i}}. \tag{2}$$

The *average reward* criterion is another natural and common measure. The (expected) average reward of a finite action sequence is its expected total reward divided by its length. The average reward of an infinite sequence w is the limit of the average rewards of its finite prefixes as the prefix length grows:

$$\mathcal{V}_{\text{avg}}\big(w, \mathbf{x}^{(0)}\big) = \lim_{i \to \infty} \frac{1}{i} \mathcal{V}\big(pre(w, i), \mathbf{x}^{(0)}\big), \tag{3}$$

where $\mathcal{V}(pre(w, i), \mathbf{x}^{(0)})$ denotes the total reward of length i prefix of w, as defined above. There may be no such limit for some infinite action sequences: consider a single state deterministic MDP with two actions a and b with rewards of 1 and -1 respectively, and an infinite action sequence that basically alternates with consecutive sequences of a and b of increasing length, such that infinitely many prefixes of it have a total reward and therefore an average that is 0 (equal numbers of a's and b's), while infinitely many prefixes have an average that is close to 1 (we can append enough a actions to the prefixes of the first type). The limit in Eq. (3) is undefined for such a sequence (see for example Puterman [42] for further discussion).

2.2.4. Goal oriented optimality criteria

An important special case of the total-reward criterion is when a reward is obtained at most once upon entering a specially designated "goal" state. We call this special criterion the *goal-oriented* model or criterion. This criterion is always well defined and is basically equivalent to the criterion of maximizing the probability of reaching the (unique) goal state. The probabilistic planning problem falls under this criterion and all our undecidability results are based on reductions from goal-oriented problems.

2.2.5. Policies and computational problems

Under any optimality criterion a number of computational problems of interest suggest themselves. We formulate several such problems under the infinite-horizon discounted criterion. A discounted UMDP model is specified by the sextuple $\mathcal{M} = \{Q, \Sigma, \{\mathbf{M_a}\}, \{\mathbf{R_a}\}, \mathbf{x_0}, \beta\}$, where Q and Σ are the states and actions of the model, $\{\mathbf{M_a}\}$ and $\{\mathbf{R_a}\}$ are the sets of transition dynamics and rewards corresponding to the actions, $\mathbf{x_0}$ is the initial distribution, and β is the discount factor. A simple decision problem, which we call the *sequence* existence problem is then:

Problem 2.1. Given a discounted UMDP $\mathcal{M}$, and a threshold τ, is there a finite action sequence w, such that $\mathcal{V}_\beta(w, \mathbf{x}^{(0)}) > \tau$?

This sequence existence problem is a major problem shown undecidable in Section 4. Similar problems can be specified under other criteria (see Problem 4.1).

Let us next define optimal values and optimal action sequences under the discounted model. Let $\mathcal{X}$ be the set of $n \times 1$ vectors of probability distribution over system state: $\mathcal{X} = \{\mathbf{x} \mid \mathbf{x}[i] \geqslant 0,\ \sum_{i=1}^{n} \mathbf{x}[i] = 1\}$. Let the *optimal value function* $\mathcal{V}^* : \mathcal{X} \to R$, be defined as:

$$\mathcal{V}^*(\mathbf{x}) = \sup_{w \in \Sigma^*} \mathcal{V}_\beta(w, \mathbf{x}). \tag{4}$$

The function $\mathcal{V}^*$ is well defined over any $\mathbf{x} \in \mathcal{X}$, since $\mathcal{V}^*(\mathbf{x})$ is bounded from above and below at any $\mathbf{x}$: the execution of any action at any time point falls in $[-R, +R]$, where R denotes the highest reward in absolute value over components of action reward vectors, thus for any w, its value $\mathcal{V}_\beta(w, \mathbf{x})$ is bounded above and below by $\pm R \sum_{i=0}^{\infty} \beta^i = \pm R/(1-\beta)$. Put in words, $\mathcal{V}^*(\mathbf{x})$ is the highest value in expectation attainable if the initial distribution is $\mathbf{x}$. An optimization problem is then: given a UMDP and an initial distribution $\mathbf{x}^{(0)}$, compute $\mathcal{V}^*(\mathbf{x}^{(0)})$ or an approximate thereof.

It is also the case that there is some action sequence w, which we call an *optimal action sequence*, that yields the value $\mathcal{V}^*(\mathbf{x})$ (possibly in the limit if infinite) when executed with the initial distribution being $\mathbf{x}$. An action sequence for a UMDP is an example of a *policy*. A policy, in general, is a prescription of action as a function of the information available to the decision maker. In the UMDP case, we may think of action sequences as policies that specify the action to take as a function of the time point only. An optimization problem would then be: given a UMDP and an initial distribution $\mathbf{x}^{(0)}$, compute a finite representation of an optimal action sequence, or compute a finite prefix, for example the first action of the sequence.

A representation of a policy which is an elegant way of controlling a POMDP is a *finite controller*. Briefly, a finite controller is a finite state machine which prescribes an action to execute at each of its own states, and when the prescribed action is executed and an observation from the POMDP is observed, it changes state based on the observation and its current state. Algorithms generating finite controllers show promise in efficiently finding optimal or near optimal policies in many POMDP problems [21,23,36]. Techniques that compute other types of policies, for example the more general finite-memory policies and grid-based approaches, have demonstrated similar successes as well [32,50]. In case of UMDPs, because there is no observation, the output of a finite controller is just a *periodic* action sequence. Formally we define a periodic action sequence as follows: given an alphabet (action set) Σ, we call a sequence w periodic if $w = uv^\infty$, where $u, v \in \Sigma^*$ (note that a string—a finite sequence—is periodic under this definition). Not all POMDP problems have optimal finite controllers: a simple two-state 2-armed bandit counter-example can be constructed; see [33] for a 3-state UMDP counter-example. In the next section, we will view action sequences as the desired output for solving UMDPs, but the problems of whether an optimal finite controller or an optimal finite action sequence exist are also addressed in Section 4.6.

Throughout this paper we always assume that a UMDP problem includes the initial distribution over states in the problem specification.

2.3. Probabilistic finite automata

A probabilistic finite-state automaton (PFA) $\mathcal{M}$ is defined by a quintuple $\mathcal{M} = (Q, \Sigma, T, s_1, s_n)$ where Q is a set of n states, Σ is the input alphabet, T is a set of $n \times n$ row-stochastic transition matrices, one for each symbol in Σ, $s_1 \in Q$ is the initial state of the PFA, and $s_n \in Q$ is an *accepting* state. The PFA model shares the dynamics of the UMDP model: a PFA occupies one state from its states Q at any point in time, and at each stage transitions stochastically from one state to another. The state transition is determined as follows:

(1) the current input symbol a determines a transition matrix $\mathbf{M_a}$,
(2) the current state s_i determines the row $\mathbf{M_a}[i, .]$, a probability distribution over the possible next states, and
(3) the state changes according to the probability distribution $\mathbf{M_a}[i, .]$.

In the undecidability proof, we restrict attention to PFA's in which the accepting state s_n is absorbing: $\mathbf{M_a}[n, n] = 1.0, \forall a \in \Sigma$. This is equivalent to the assumption of the automaton halting upon entering the accepting state s_n.

We say a PFA $\mathcal{M}$ *accepts* the string $w \in \Sigma^*$ if the automaton ends in the accepting state upon reading the string w, otherwise we say it *rejects* the string. We denote by $p^{\mathcal{M}}(w)$ the acceptance probability of string w by PFA $\mathcal{M}$. Note that for any infinite sequence w, the acceptance probabilities of the prefixes $p^{\mathcal{M}}(pre(w, i))$ do not decrease with i as the accepting state is assumed to be absorbing, and consequently the limit $\lim_{i\to\infty} p^{\mathcal{M}}(pre(w, i))$ is well defined, and is defined naturally to be the acceptance probability $p^{\mathcal{M}}(w)$ of w.

The language accepted by a PFA $\mathcal{M}$ given a threshold τ, denoted by $L(\mathcal{M}, \tau)$, is the set of all strings that take the PFA to the accepting state with probability greater than τ:

$$L(\mathcal{M}, \tau) = \left\{ w \in \Sigma^* \colon\ p^{\mathcal{M}}(w) > \tau \right\}.$$

An important problem, which we call the *emptiness* problem, is whether, given a PFA $\mathcal{M}$ and threshold τ, the language accepted by $\mathcal{M}$ is empty.

While PFA's are models of language acceptors and probabilistic planning and UMDP problems are stochastic optimization problems, note that the underlying system models are the same with the alphabet corresponding to the actions and acceptance corresponding to goal achievement. This change in view allows us to see the connection between the PFA model and probabilistic planning and UMDPs and leads to the undecidability results.

3. Undecidability results for probabilistic planning problems

Now that we have described the models, we show that many problems of interest are undecidable by showing reductions from the emptiness problem.

3.1. Undecidability of the emptiness problem

The (language) *emptiness* problem for PFAs is as follows:

Problem 3.1. Given a PFA and a desired threshold τ, decide whether or not there is some input string $w \in \Sigma^*$ that the PFA accepts with probability exceeding threshold τ.

Both Paz [41] and Lipton and Condon [15] establish the undecidability of the problem:

Theorem 3.2 [15,41]. *The emptiness problem for PFAs is undecidable.*

Below, we give a high-level description of the properties of the reduction in [15] and then give a more detailed explanation of the proof. The details of the proof are also used to develop subsequent undecidability results on probabilistic planning and POMDP problems, most notably Theorem 4.4, which establishes the undecidability of optimal policy construction for discounted-total-reward infinite-horizon POMDPs.

3.1.1. Properties of the reduction

In [15], the (undecidable) question of whether a Turing machine (TM) accepts the empty string is reduced to the question of whether a PFA accepts any string with probability exceeding a given threshold. The PFA constructed by the reduction tests whether its input is a concatenation of *accepting sequences*. An accepting sequence is a legal sequence of TM configurations beginning at the initial configuration and terminating in an accepting configuration.

The reduction has the property that if the TM is accepting, i.e., it accepts the empty string, then the PFA accepts sufficiently long concatenations of accepting sequences with high probability. But if the TM is *not* accepting, the PFA accepts all strings with low probability. We next formalize these properties and use them in subsequent undecidability results. The following section explains how the PFA generated by the reduction has these properties.

Theorem 3.3. *There exists an algorithm which, given a two-counter TM as input and any rational $\varepsilon > 0$ and integer $K \geqslant 1$, outputs a PFA M satisfying the following*:

(1) *if the TM does not accept the empty string, the PFA M accepts no string with probability exceeding ε, and*
(2) *if the TM does accept the empty string, then let string w represent the accepting sequence of the TM. We have $\lim_{i\to\infty} p^M(w^i) = 1 - (1/2)^K$, and $\forall i$, $p^M(w^i) < 1 - (1/2)^K$.*

We conclude this section making two additional points about the emptiness problem.

- Due to the separation between the acceptance probability of the PFA in the two cases of the TM accepting the empty string or otherwise, the emptiness problem remains

undecidable if the strict inequality in the description of the existence problem is replaced by a weak equality ($\geqslant$) relation.
- Although the problem is posed in terms of the existence of (finite) strings, the result holds even if the sequences have infinite length.

3.1.2. Details of the reduction

The class of TMs used in the reduction in [15] are *two-counter* TMs, which are as powerful as general TMs [22]. A two-counter TM has a read-only input tape with a two-way head, and two counters. In a transition, the TM may change state, add 1 to a counter, subtract 1 from a counter that has value greater than 0, and move the head on the input tape, as a function of the current state of the TM, the symbol under the input tape, and whether each of the counters is 0 or not. The constructed PFA should detect whether a sequence of computations represents a valid accepting computation (accepting sequence) of the TM. This task reduces to the problem of checking the legality of each transition from one configuration of the TM to the next, which amounts to verifying that

- the first configuration has the machine in the start state,
- the last configuration has the machine in the accepting state, and
- each transition is legal according to the TM's transition rules.

All of these checks can be carried out by a deterministic finite state automaton, except the check as to whether the TM's counter contents remain valid across consecutive configurations. For example, checking that the state of the ith configuration is consistent with the transition function requires that the state and symbol under the input head in the $(i-1)$st configuration be recorded in the finite state, along with two bits that indicate whether each counter is 0 or not. The PFA rejects immediately if any of the easily verifiable transition rules are violated, which leaves only the problem of validating the counters' contents across each transition.

On each computation step taken by a two-counter TM the counters' contents either stay the same, get incremented by 1, or get decremented by 1. Assuming without loss of generality that the counter contents are represented in unary, this problem reduces to checking whether two strings have the same length: given a string $a^i b^j$, does $i = j$?

Although this question cannot be answered exactly by any PFA, a *weak equality* test developed in [15] and inspired by the work of Freivalds [19] can answer it in a strict and limited sense which is nonetheless sufficient to allow the reduction. The weak equality test, shown in Fig. 2, works as follows. The PFA scans its input string $a^i b^j$, and with high probability enters an *Indecision* state (equivalently we say the outcome of the test is Indecision). In the figure, the event of Indecision is given a thick edge to denote its high probability. With some low probability the PFA enters one of two "decisive" states. If the substrings have equal length (Fig. 2(a)) the PFA either enters a *Correct* state or a *Suspect* state. It enters these two states equiprobably. However, suppose that the PFA enters a decisive state but the input string is composed of *unequal*-length substrings ($i \neq j$, Fig. 2(b)). In this case the *Suspect* outcome is at least 2^k (for $k \geqslant 2$) times more likely than the *Correct* outcome, where the *discrimination factor* k can be made as large as desired by increasing the size of the PFA. This test is described in more detail in Appendix A.

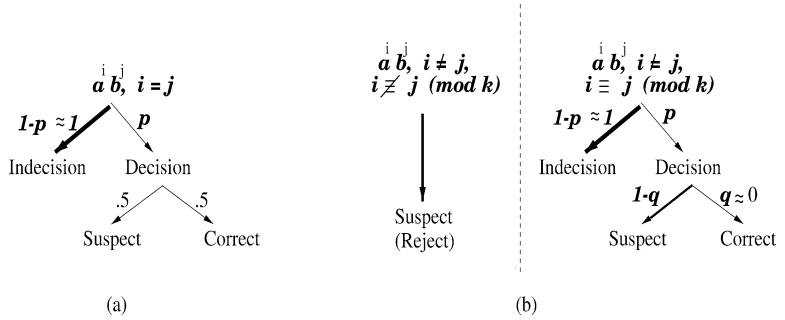

Fig. 2. Outcomes of the weak equality test when $a^i b^j$ is read, (a) when $i = j$, (b) when $i \neq j$. While the probability of making a decision is minute, if a decision is made, the chances of Suspect and Correct is the same when $i = j$, but Suspect is much more likely when $i \neq j$.

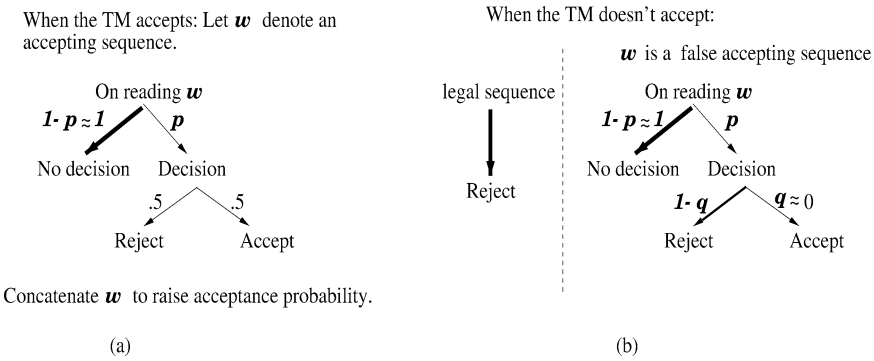

Fig. 3. (a) When the TM is accepting, on an accepting sequence w, while the probability of making an all decisive decision is minute, the PFA will accept or reject with equal probability. By concatenating w, the acceptance probability can be raised to as close to $1/2$ as desired. (b) When the TM is not accepting, on a false accepting sequence w, if the PFA makes a decision, with very high probability it is rejection.

The PFA of the reduction carries out a *global* test of its own, not unlike the weak equality test (Fig. 3), on a candidate accepting sequence for the TM, and uses the weak equality test to check for counter increments or decrements on consecutive configurations. Given a candidate accepting sequence, if the outcome of *all* the tests are decisive and *correct*, the PFA accepts the input. If the outcome of *all* the tests are *suspect*, the PFA rejects the input. The probability of an all decisive test is, of course, very small, but positive. The PFA remains in the *global-indecision* state until it detects the start of the next candidate accepting sequence (start configuration of the TM), or until it reaches the end of the input. It follows from the outcome of the individual tests that, if the PFA is given an illegal candidate accepting sequence, where the counter contents are not changed legally, given that the outcome of all test are decisive, the PFA rejects with much higher probability than it accepts (Fig. 3(b)). If the PFA is in the global-indecision state at the end of the input, it rejects.

If the original TM accepts the empty string, observe that the probability that the PFA accepts can approach the upper limit $1/2$ on an input string consisting of a concatenation

of sufficiently many accepting sequences (Fig. 3(a)). If the TM does not accept the empty string, it follows from the properties of the weak equality test that the probability that the PFA rejects is at least 2^k times more likely than the probability that it accepts. We can increase the acceptance probability of the PFA when the TM is accepting by adding a finite counter to the PFA, as discussed next.

3.1.3. Increasing the PFA acceptance probability

By making a minor adjustment to the PFA, the acceptance probability of the PFA when the TM accepts the empty string can be made arbitrarily close to 1: instead of rejecting or accepting if it sees an all *suspect* or an all *correct* outcome on a single candidate accepting sequence, the PFA can instead increment an *all-decisive* counter which has a finite upper limit K. The PFA accepts its input if and only if the all-decisive counter reaches K, *and* it has seen an all *correct* on a candidate sequence. Hence, if the TM is accepting, the PFA accepts a concatenation of sufficiently many accepting sequences with probability arbitrarily close to $1 - (1/2)^K$. In addition, for the cases when the TM is not accepting, the acceptance probability of the PFA can be made as small as desired for a given counter upper limit K, by choosing the discrimination factor k of the weak equality test to be large.

3.1.4. Undecidability of approximations

The question of approximability is an important one, especially when computing an optimal answer is impossible. Unfortunately, it follows from the next corollary that approximations, such as computing a string which the PFA accepts with probability within an additive constant or multiplicative factor $\varepsilon < 1$ of the maximum acceptance probability of the PFA[1] are also uncomputable.

Corollary 3.4. *For* any *fixed* ε, $0 < \varepsilon < 1$, *the following problem is undecidable*: *Given is a PFA M for which one of the two cases hold*:

(1) *the PFA accepts some string with probability greater than* $1 - \varepsilon$, *or*
(2) *the PFA accepts no string with probability greater than* ε.

Decide whether case (1) *holds*.

Proof. We reduce the undecidable problem of determining whether a two-counter TM accepts the empty string to the problem of distinguishing between cases (1) and (2) in the statement of the corollary. Fix any real ε, $0 < \varepsilon < 1$. Let ε' be a rational number with $0 < \varepsilon' < \varepsilon$. Let $K \geqslant 1$ be an integer with $1 - (1/2)^K > 1 - \varepsilon$. Given a two-counter TM, we use the algorithm whose existence is stated in Theorem 3.3 to reduce the TM, along with ε' and K, to a PFA M. Theorem 3.3 guarantees that if the TM does not accept the empty string, the PFA M accepts no string with probability greater than ε', and thus no string with probability greater than ε. Furthermore, if the TM does accept the

[1] The maximum acceptance probability is taken as the supremum over the acceptance probability over all strings.

empty string and string w is the accepting sequence of configurations of the TM, we have $\lim_{i\to\infty} p^M(w^i) = 1-(1/2)^K$. Therefore by our choice of K, for some i, $p^M(w^i) \geqslant 1-\varepsilon$ and so PFA M does accept some string with probability greater than $1-\varepsilon$. An algorithm to decide between the two cases for PFA M would thus contradict the undecidability of the problem of determining whether a two-counter TM accepts the empty string. □

3.1.5. Undecidability of the threshold-isolation problem

There might be some hope for decidability of the emptiness problem for special cases, for example for problems for which the given threshold is *isolated* for the given PFA:

Definition 1 [43]. Let M be a PFA. The threshold τ is ε-isolated with respect to M if $|p^M(x)-\tau| \geqslant \varepsilon$ for all $x \in \Sigma^*$, for some $\varepsilon > 0$.

Problem 3.5. The threshold-isolation problem[2] is, given a PFA M and a threshold τ, decide whether, for some $\varepsilon > 0$, the threshold τ is ε-isolated for the PFA M.

Isolated thresholds are interesting because PFAs with isolated thresholds have less expressive power than general PFAs, thus the corresponding decision problems are easier. General PFAs are powerful enough to accept even non-context-free languages [41]. However, Rabin [43] showed that PFA with isolated thresholds accept exactly the regular languages. A natural question then is: given a PFA and a threshold, is threshold is isolated for the PFA? If we can compute the answer and it is positive, then we can presumably compute the regular language accepted by the PFA, and see whether it is empty or not. That would afford at least the opportunity to recognize and solve a special case of the general emptiness problem.

The decidability of the isolation problem was raised by Rabin [44] (see also Paz [41, p. 163]), and was later shown undecidable by Bertoni [4,5] by a reduction from Post's correspondence problem. The reduction in [15] also shows that recognizing an isolated threshold is undecidable:

Corollary 3.6. *The threshold-isolation problem is undecidable.*

Proof. We can design the reduction of Theorem 3.3 with $\varepsilon = 1/3$, and $K = 1$. It follows that if the TM is not accepting, then there is no string that the PFA accepts with probability greater than $1/3$, while if the TM is accepting, there are (finite) strings that the PFA accepts with probability arbitrarily close to $1/2$. In other words, the threshold $1/2$ is isolated iff the TM is not accepting. □

3.2. Undecidable problems in probabilistic planning

The results of the previous section establish the uncomputability of probabilistic planing problems in the general case—when there is no restriction imposed on the

[2] In the PFA literature, such as [41] and [43], a threshold is referred to as a cutpoint, and the problem referred as cutpoint-isolation.

length of solution plans considered. Uncomputability follows when it is established that a sufficiently powerful probabilistic planning language can model any given PFA, so that any question about a PFA can be reformulated as a probabilistic planning problem.

This is the case for the probabilistic planning model investigated in Kushmerick, Hanks, and Weld [24] and many others. This model is based on STRIPS propositional planning [18] with uncertainty in the form of (conditional) probability distributions added to the action effects. It is established by Boutilier, Dean, and Hanks [9] that the propositional encoding of states is sufficient to represent any finite state space, and the extended probabilistic STRIPS action representation is sufficient to represent any stochastic transition matrix. Thus the emptiness problem ("is there any input string that moves the automaton from the start state to an accepting state with probability at least τ?") can be directly reformulated in the planning context ("is there any sequence of actions that moves the system from a start state to a goal state with probability at least τ?"). An algorithm that solved the planning problem would answer the question of whether or not such a plan exists by either generating the plan or terminating having failed to do so, thus solving the equivalent emptiness problem. Therefore, as a corollary of the undecidability of the emptiness problem for PFAs we obtain:

Theorem 3.7. *The plan-existence problem is undecidable.*

We also note that due to the tight correspondence between PFAs and probabilistic planning problems, the other undecidability results from the previous section apply as well:

- "Approximately satisficing planning", generating a plan that is within some additive or multiplicative factor of the threshold is undecidable.
- Deciding whether the threshold for a particular planning problem represents an isolated threshold for that problem is undecidable.

4. Undecidability results for POMDP problems

We noted earlier both that the probabilistic planning assumption of no observability was a restricted case of a probabilistic partial-observation model, and that the "goal pursuit" planning objective was a restricted case of more general objective functions.

It is therefore easy to establish that the strictly more general POMDP optimization problem (maximizing an arbitrary objective function over an infinite horizon) is undecidable. In this section we analyze some commonly studied restrictions to the objective function: positive bounded models, average-reward models, and discounted models, and show the corresponding optimization problems to be undecidable as well.

We first define the *policy-existence* (action sequence existence) problem for POMDPs, under any given optimality criterion. This is a generic version of Problem 2.1, which was formulated for the discounted criterion. The space of policies considered in the following definitions is an important consideration. All of the results hold when the space of policies includes any one or more of the following sets: finite action sequences of indefinite length,

infinite sequences, or algorithms that create such finite or infinite sequences (using some kind of implicit policy representation).

Problem 4.1. The policy-existence problem (with respect to an optimality criterion) is: given a POMDP and a threshold, does there exist a policy with expected value greater than the threshold?

4.1. Undecidability for positive-bounded models under total undiscounted reward

The most direct result involves a special case of infinite-horizon undiscounted total-reward models called the *positive bounded* models [42]. The essential feature of this model is that the reward structure and system dynamics for a problem must ensure that the total reward gathered is bounded both above and below, and is convergent (over action sequences) over an infinite horizon.

The PFA emptiness problem or the planning problem can easily be posed as a positive-bounded POMDP:

- the same observation o is received regardless of the state and action (non-observability),
- unit reward is gathered on the execution of *any* action on the goal state (Fig. 4(b)),
- the execution of *any* action at the goal state leads to an absorbing state: the system stays in that state and gathers no additional rewards (Fig. 4(b)), and
- all other states and actions incur no reward.

From this equivalence we can immediately establish the following:

Theorem 4.2. *The policy-existence problem for positive-bounded problems under the infinite-horizon total reward criterion is undecidable.*

Proof. We describe a reduction from the plan-existence problem. Since any planning problem can be posed as a positive-bounded POMDP, we can easily verify that an effective algorithm for that problem could be used to solve the plan-existence problem, and by Corollary 3.7 such an algorithm cannot exist. To see this, note that a plan, say a finite sequence of actions, exists for the planning problem with probability of reaching the goal (success probability) exceeding τ if and only if a finite sequence of actions exist with value exceeding τ for the corresponding UMDP model (as outlined above and in Fig. 4(b)): the success probability from executing a finite sequence of actions w in the planning problem

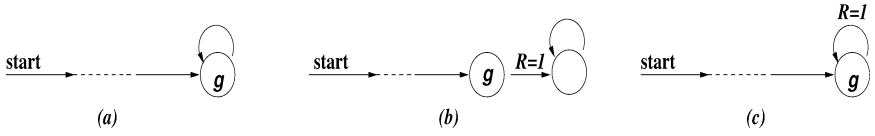

Fig. 4. (a) A PFA, or the criterion of maximizing the probability of reaching a goal state (g) in a probabilistic planning model. (b) The PFA emptiness problem modeled as a total reward criterion: The old accepting (goal) state g, on any action, now has a transition to an extra absorbing state giving reward of 1.0. All other rewards are zero. (c) Similarly, a PFA emptiness problem modeled as an average reward problem.

is $\sum_{1 \leqslant i \leqslant |w|} Pr(E_i)$, where E_i is the event that the absorbing goal state is entered on the executions of the ith action. We can also express the expected reward from executing any finite action sequence w on the model of Fig. 4(b) as $\sum_{1 \leqslant i \leqslant |w|-1} Pr(E_i)$. Therefore the success probability from executing a finite sequence of actions w in the planning problem equals the expected total reward of action sequence wa, for any action a, in the corresponding UMDP model. Conversely, if v is the value of a sequence w in the UMDP model, then the success probability of sequence w in the planning problem is at least v.

In the case of infinite action sequences, if there is an infinite action sequence with success probability greater than τ for the planning problem, then there must exist finite action sequences with success probability greater than τ, and we can use the above to deduce that there must exist a finite and therefore an infinite action sequence with total reward greater than τ in the corresponding UMDP model. The converse holds in the same way. Therefore a similar equivalence holds for the case of infinite action sequences. □

4.2. Undecidability under the average reward criterion

The indirect connection to PFAs allows extension of the previous result to *all* undiscounted total-reward models, and to average-reward models as well.

Theorem 4.3. *The policy-existence problem under the infinite-horizon average reward criterion is undecidable.*

Proof. The proof is complete once we observe that questions on the acceptance probability of strings for a given PFA can readily be turned to questions on the value of similar strings in a related UMDP model. This transformation is achieved by modeling the probability of reaching the accepting (goal) state g using rewards (Fig. 4(c)). It can be verified that there is a string accepted by the PFA M with probability exceeding τ if and only if there is a string with average reward greater than τ for the corresponding UMDP model. To see this, assume for some string w, $p^M(w) > \tau$, and denote by $\mathcal{V}(w)$ the average reward of w under the corresponding UMDP model. For any action a, the expected total reward of wa^k, $k \geqslant 1$, is at least $p^M(w)k$ (i.e., the probability of entering the goal state on executing w and then obtaining k rewards). Therefore, the average reward $\mathcal{V}(wa^k)$ is not less than $p^M(w)\frac{k}{k+|w|}$, and for sufficiently large k, $\frac{k}{k+|w|} > \frac{\tau}{p^M(w)}$. To show the converse, let E_i denote the event that the goal state g is entered on the ith time step on reading (executing) a string w. In this case, the reward obtained would be exactly $|w| - i$, so we have,

$$\mathcal{V}(w) = \frac{\text{total expected reward from } w}{|w|} = \frac{\sum_{1 \leqslant i \leqslant |w|} Pr(E_i)(|w| - i)}{|w|}$$
$$\leqslant \sum_{1 \leqslant i \leqslant |w|} \frac{P(E_i)(|w|)}{|w|} = \sum_{1 \leqslant i \leqslant |w|} P(E_i) = p^M(w).$$

Therefore, if $\mathcal{V}(w) > \tau$, then $p^M(w) > \tau$.

A similar equivalence holds for infinite sequences (see proof of Theorem 4.2). □

4.3. Undecidability of the discounted criterion

We turn now to the most commonly studied model: maximizing total expected discounted reward over an infinite horizon. Here, as in the proof of Theorem 4.3, we make a small change in the PFA constructed in the emptiness reduction.

Theorem 4.4. *The policy-existence problem under the infinite-horizon discounted criterion is undecidable.*

Proof. This result first appeared in [34]. We prove this again by reduction from the problem of deciding whether a two-counter TM accepts the empty string.

Given a two-counter TM, consider the PFA whose existence is stated in Theorem 3.3. We change this PFA to a *leaky* (or "discounted") PFA as follows, and then use Lemmas 4.5 and 4.6 given below. Let d (the leak quantity) be any rational value such that $0 < d < 1$. The leaky PFA, upon reading an input symbol, continues as the original PFA with probability $1 - d$. Otherwise, we say that it *leaks*, and in this case it makes a transition to either an absorbing rejection state or the absorbing accepting state, each with equal overall probability $d/2$. The equal leak probabilities "balance" each other, so that as we show in Lemma 4.6, the original PFA accepts a string with probability greater than $1/2$ if and only if its leaky version accepts a string with probability greater than $1/2$. □

Lemma 4.5 states that the leaky PFA is equivalent to a UMDP problem with a discounted criterion. The objective of maximizing the probability of reaching a state can be reformulated as a reward objective in an MDP problem (see for example the proof of Theorem 4.2) and the presence of a non-zero probability of entering an absorbing zero reward state on each state transition is equivalent to presence of a fixed discount factor less than one (see for example [42, pp. 126–127]). We summarize these observations in the following lemma.

Lemma 4.5. *For any leaky PFA as described above, there exists a discounted UMDP instance that is equivalent to it, in the sense that there is a string with acceptance probability p for the PFA if and only if there is a string (action sequence) with the same value p for the discounted UMDP instance.*

Proof. The UMDP instance has discount $\beta = 1 - d/2$, where d is the "leak" quantity defined in the proof of Theorem 4.4. We first construct the UMDP total reward model, call it model U_1, corresponding to the leaky PFA, as described in the proof of Theorem 4.2 and the explanations preceding that theorem and Fig. 4(b), and then remove the leak to the absorbing rejection state (replace it with discounting) and normalize the remaining transition probabilities to obtain the discounted model U_β as follows. The discounted UMDP model U_β has the same state and transition and reward structure as U_1, but with modified transition probabilities and a small modification in the reward structure. Therefore, the instance U_β has $n + 1$ states with state n corresponding to the accepting state of the leaky PFA, and state $n + 1$ being the new absorbing state. Execution of any action when the system is in state n causes a deterministic transition to state $n + 1$, with

a reward of $1.0/\beta$. All other action rewards are zero. The initial distribution is defined so that the initial state of the PFA has probability 1 and all other states have probability 0. For each action a, the transition matrix M'_a for the UMDP is defined as follows, where M_a is the transition matrix for the original PFA: $M'_a[n, n+1] = M'_a[n+1, n+1] = 1$, and for $i < n$, $M'_a[i, n] = (M_a[i, n](1-d) + d/2)/(1-d/2)$, and finally $M'_a[i, j] = M_a[i, j](1-d)/(1-d/2)$, when $i < n$ and $j < n$. Intuitively, the UMDP model U_β, compared to U_1 increases the probability that the accept state n is entered at each step by an amount that exactly balances the negative effect of the discount factor on the expected reward.

We can write a similar expression to the one in the proof of Theorem 4.2 for total expected reward under model U_β, but include this discounting in this case, as follows. In this discounted UMDP, the discounted reward of a string w is expresses as:

$$\mathcal{V}(w) = \sum_{1 \leqslant i \leqslant |w|-1} Pr(E_i)\beta^{i+1}\frac{1}{\beta} = \sum_{1 \leqslant i \leqslant |w|-1} Pr(E_i)\beta^i,$$

where E_i denotes the event that state n (original accepting or goal state) is entered on the ith action execution. Note that as another action is to be executed to obtain the reward of $1/\beta$, the discount factor is raised to $i+1$. On the other hand the value of w under the total reward model U_1 is

$$\mathcal{V}(w) = \sum_{1 \leqslant i \leqslant |w|-1} \left(1 - \frac{d}{2}\right)^i Pr(E_i) = \sum_{1 \leqslant i \leqslant |w|-1} Pr(E_i)\beta^i.$$

Therefore, by Theorem 4.2, the acceptance probability of the leaky PFA on any string w is equal to the expected discounted total reward of the discounted UMDP U_β on wa, for some action a. □

We can now focus on the relation between the acceptance probabilities for the original PFA and the corresponding leaky PFA. Only the first statement of the next lemma is needed to establish Theorem 4.4, but we will need the full result in Section 4.6, in order to show the undecidability of the problem of whether a discounted POMDP has an optimal action sequence that is finite.

Lemma 4.6. *Given a two-counter TM, if the TM is accepting* (*see Section* 3.1.1)*, then the corresponding leaky PFA accepts some strings with probability greater than* $1/2$*, while if the TM is not accepting, every finite string is accepted by the leaky PFA with probability less than* $1/2$*. Furthermore, if the TM is accepting, the maximum probability of acceptance by the leaky PFA is reached only in the limit on* w^∞*, where* w *denotes the accepting sequence of the TM.*

Proof. Let $K \geqslant 2$ and let ε satisfy the equality $1 - 1/2^K = 1/2 + \varepsilon$. Note that $\varepsilon < 1/2$. Given a two-counter TM, K, and ε, we take the PFA whose existence is stated in Theorem 3.3 and make it leaky.

Assume the TM is accepting, and let w be an accepting sequence of the TM. We assume the original PFA constructed in the reduction accepts only after $K \geqslant 2$ decisive outcomes

(the all-decisive counter limit K is explained in Section 3.1.3). We will show that some string must have acceptance probability greater than $1/2$ on the leaky PFA. We then use a similar argument to show that $p^M(w^{j+1}) > p^M(w^j), \forall j \geqslant K$.

Let q_j denote the probability that the PFA "halts" (i.e., goes into one of the absorbing states) on reading w^j. Let E_j be the event that the PFA has not leaked on w^j given that it has halted, so that it has made K decisions, and let $Pr(E_j)$ denote its probability. Hence, given that the PFA halts on w^j, the probability of acceptance is $p_j = (1 - Pr(E_j))/2 + Pr(E_j)(1 - 1/2^K)$, where the first term corresponds to the probability that it has leaked and accepted, given that it has halted, and the second is the probability that it has not leaked and has accepted, given that it has halted. Thus the overall probability of acceptance is $p^M(w^j) = q_j p_j$. Now, by our choice of ε, $1 - 1/2^K = 1/2 + \varepsilon$. Thus,

$$p_j = \big(1 - Pr(E_j)\big)/2 + Pr(E_j)(1/2 + \varepsilon) = 1/2 + Pr(E_j)\varepsilon.$$

We now show that the acceptance probability $p^M(w_j)$ exceeds $1/2$ for some j. First note that as j increases, the probability q_j that the leaky PFA halts on w^j approaches 1. Therefore, it follows from the above expressions for $p^M(w^j)$ and for p_j that $p^M(w_j)$ exceeds $1/2$ once we show that $Pr(E_j)$ is bounded from below by a constant greater than 0 for sufficiently large j. To complete the argument, we now show that $Pr(E_j) \geqslant Pr(E_K)$ for $j \geqslant K$. (Note that when $j < K$, $Pr(E_j) = 0$, but $Pr(E_K) > 0$ because with non-zero probability the PFA makes K decisions on w^K.) The event E_j, for $j \geqslant K$, is the union of the disjoint events E'_K (namely the event of halting but not leaking on the first K concatenations of w), and E_{j-K} (namely the event of not halting on the first K concatenations of w, but halting and not leaking on the remaining $j - K$ concatenations of w). We have $E_{j-K} > 0$. Consequently,

$$Pr(E_j) = Pr(E'_K) + Pr(E_{K-j}) = Pr(E_K) + Pr(E_{K-j}) > Pr(E_K), \quad \forall j \geqslant K.$$

We conclude that when the TM is accepting, then for sufficiently large j the leaky PFA accepts strings w^j with probability exceeding $1/2$.

We next extend the argument to show that $p^M(w^{j+1}) > p^M(w^j), \forall j \geqslant K$. The argument above shows that $P(E_{K+1}) > P(E_K)$. A similar argument shows $P(E_{K+2}) > P(E_{K+1})$, and in general $P(E_{K+i+1}) > P(E_{K+i})$, thus $Pr(E_{j+1}) > Pr(E_j)$ for any $j \geqslant K$. As $q_{j+1} \geqslant q_j$, we obtain $\forall j \geqslant K$, $p^M(w^{j+1}) > p^M(w^j)$.

Next, assume that the TM is not accepting. Let s be any input. Let q be the probability of halting on s and let p be the probability of leaking given that the PFA halts on s. The probability that s is accepted by the PFA is $q(p/2 + (1 - p)\varepsilon)$, where ε is the constant that we chose at the start of the proof to be $< 1/2$. The term $p/2$ in this expression accounts for the probability that the PFA leaks and accepts, and the term $(1 - p)\varepsilon$ accounts for the probability that the PFA does not leak and accepts. By Theorem 3.3, since the TM is not accepting, the probability that the PFA accepts, given that it does not leak, is at most ε. Clearly, $q(p/2 + (1 - p)\varepsilon)$ is less than $1/2$, and so we are done. This completes the proof of the first half of the lemma.

Finally, we complete the proof that the highest acceptance probability is achieved only in the limit, in case the TM is accepting. A candidate (accepting) sequence refers to a sequence of TM configurations where the first configuration is the initial TM configuration. Let w denote the true accepting sequence. Any input string s can be viewed as a

concatenation of $j \geqslant 0$ candidate sequences appended with a possibly empty string u where none of the prefixes of u is a candidate sequence: $s = w_1 w_2 \cdots w_j u,\ j \geqslant 0$. If $j < K$, the acceptance probability is less than $1/2$, therefore less than $p^M(w^j)$ for some $j \geqslant K$ (see above). Otherwise, if some w_i are false accepting sequences, it is more likely that the TM makes K rejection decisions on such a sequence than on $w^j u$, and thus $p^M(s) \leqslant p^M(w^j u)$. Finally we have $p^M(w^j u) \leqslant p^M(w^{j+i})$, where $|u| \leqslant |w^i|$: we need to compare the probability of accepting the sequence in the event that the PFA does not halt on the prefix w^j, but halts on the remainder. On $w^j u$, this is simply $1/2$, while on w^{j+i}, the acceptance event is more likely, as there is the additional possibility of making the Kth decision on the last w^i. We have shown that for any $j \geqslant K$, $p^M(w^j) < p^M(w^{j+1})$ and $p^M(w^j) \geqslant p^M(s)$, for any string s such that $|s| \leqslant |w^j|$, therefore the maximum acceptance probability is reached only in the limit on w^∞. □

The addition of the finite counter to the PFA in the reduction in [15] (as described in Section 3.1.3), while unnecessary to prove undecidability of emptiness, is crucial in making the above proof work. Without it, or when $K = 1$, the probability of PFA acceptance can not exceed $1/2$ whether or not the TM is accepting. On the other hand, infinite sequences corresponding to valid but non-halting TM computations would approach $1/2$ with our construction of the leaky PFA. Thus with $K = 1$, it would not be possible to distinguish between the acceptance probabilities of strings by the PFA in the cases when the TM is accepting versus when the TM is not accepting.

4.4. Undecidability under a negative model

The optimality criteria studied to this point involve maximizing the expected benefits of executing a policy. An alternative goal would be to choose a policy likely to avoid disaster. In these cases (*state-oriented negative* models, see for example [42]) the objective is to minimize the probability of entering one or more designated negative states over the infinite horizon. We use the reduction in the previous proof to establish the undecidability of this particular negative model; the technique should be applicable to other negative models as well.

Theorem 4.7. *Policy existence under the state-oriented negative model is undecidable.*

Proof. We reduce the string-existence question for the leaky PFA in reduction of Theorem 4.4 to this problem. Note that in the string-existence reduction for the leaky PFA, if the TM is accepting, there exist infinite (and therefore finite) sequences of symbols on which the probability of acceptance of the leaky PFA exceeds $1/2$. If the TM is rejecting, no infinite sequence has an acceptance probability greater than $1/2$ (an infinite sequence with acceptance probability equal to $1/2$ may exist.). Take the rejecting absorbing state of the leaky PFA to be the state to avoid and the (undecidable) question would be whether there is an infinite sequence that avoids the rejecting state with probability exceeding $1/2$. □

4.5. Inapproximability in POMDPs

An inapproximability result similar to the one for PFAs also holds for POMDPs under the total undiscounted reward and the average reward criteria. However, under the discounted criterion, the optimal value is approximable to within any $\varepsilon > 0$, due to the presence of the discount factor, and value iteration algorithms, in particular, take advantage of this [30].

4.6. Existence of optimal finite controllers and optimal strings

The question of whether an optimal finite controller exists for a given POMDP is an interesting one. A similar case to the threshold-isolation can be made for motivating such a question: given a POMDP and an initial distribution, if we could determine whether it has an optimal finite controller, and if the answer is positive, then possibly finding the controller would be easier, as only the space of finite controllers needs to be searched.

Recall from Section 2.2.5 that finite controllers are periodic action sequences for UMDPs. A simple construction, using example UMDPs or POMDPs that provably do not have an optimal periodic action sequence or finite controller, shows that the problem is undecidable under undiscounted total or average reward criteria [33]. The constructions make use of the gap in acceptance probability of the PFA corresponding to the cases of whether or not the TM is accepting. However, showing undecidability of the problem of determining whether an optimal finite controller exists for a POMDP under the discounted criterion is more difficult as there is no positive lower bound on the gap. Extensions of the proof of Theorem 4.4 do not seem to work either as we explain briefly next. This explanation also motivates our proof of the undecidability which follows.

In the proof of Theorem 4.4, when the TM is accepting, there exists an optimal periodic sequence for the constructed PFA. On the other hand, a nonempty subset of those TMs that are not accepting have aperiodic configuration sequences. However, it is not clear that for a TM with an aperiodic configuration sequence, the corresponding PFA's optimal action sequences (which have success probability $1/2$) need necessarily correspond to the TM's aperiodic legal configuration sequence and consequently be aperiodic. Note that the PFA can only reject when the sequence is composed of false candidate accepting sequences. If a sequence never ends in an accepting or rejecting configuration, has a few miscounts and is otherwise legal and ends in a periodic sequence of configurations, it may obtain the limiting acceptance probability of $1/2$ under the leaky PFA and therefore be optimal. In the following proof, we circumvent this difficulty by easing the condition under which the PFA accepts and rejects, so that when the corresponding TM does not halt and has an aperiodic configuration sequence, the optimal sequence for the PFA would provably be aperiodic. In the process, however, we give up the difference in acceptance probability of the PFA in the cases of TM accepting and rejecting.

Theorem 4.8. *The problem of whether an optimal finite controller exists for a discounted POMDP is undecidable.*

Proof. We construct a leaky PFA similar to those of Theorems 3.3 and 4.4 but with a modification to the conditions of acceptance and rejection. As a consequence of the modification, the maximum probability of acceptance of the PFA is 1/2 whether or not the corresponding TM is accepting.

Consider first the PFA in the proof of the undecidability of the emptiness problem. The modification is in the way the automaton behaves in case of a Decisive outcome of the weak equality test on the counter contents: If the outcome is Suspect on *any* test, the PFA rejects immediately, and if the outcome is Correct (again on any test), it accepts immediately, and otherwise, in case of Indecision, it continues. If it reaches an accepting or rejecting configuration, which indicates the end of the TM configuration sequence, it reinitializes to check for the start configuration and continues. Checking other conditions described in Section 3.1.2, such as that each transition is legal according to the TM's transition rules, remains the same. The PFA rejects if any of these conditions fail or if the PFA reaches the end of the input. The PFA does not keep a finite counter. Finally we make such a PFA leaky, just as in the proof of Theorem 4.4, to capture discounting.

Such a PFA tests only that individual transitions are legal, and the acceptance probability is not affected by whether the corresponding TM is accepting or not. Thus, if the corresponding TM halts, then the infinite concatenation of its accepting or rejecting configuration sequence is periodic, and the leaky PFA accepts this sequence with probability 1/2 in the limit. Similarly, if the TM does not halt but repeats a configuration, the corresponding configuration sequence is periodic, and the leaky PFA accepts this sequence with probability 1/2 in the limit. Finally, if the TM does not halt and does not repeat a configuration, the leaky PFA still has limiting acceptance probability 1/2 over the corresponding aperiodic configuration sequence. This aperiodic sequence is the only optimal sequence for the PFA, as the probability of acceptance of any other sequence is strictly less than 1/2: any illegal sequence would have an illegal transition, on which it is more likely that the PFA rejects given that it makes a decision on that particular test. (See Section 3.1.2 for details: if the transition is illegal because a counter is not properly updated, then the probability that the PFA rejects, given that it makes a decision, is greater than 1/2 with the exact value depending on the discrimination factor; if the transition is illegal because the state is not properly updated, then the probability that the PFA rejects is 1.)

Therefore, the optimal string accepted by the leaky PFA in the limit is periodic if and only if the TM halts on the empty string or the TM does not halt but repeats a configuration on the empty string. Now, consider the discounted UMDP corresponding to this leaky PFA as in the second paragraph of Theorem 4.4. The optimal string accepted by the leaky PFA in the limit is exactly the optimal action sequence for this UMDP. Thus, the UMDP has an optimal finite controller if and only if the optimal string accepted by the PFA in the limit is periodic. We conclude that the discounted UMDP has an optimal finite controller if and only if the TM halts or the TM does not halt but repeats a configuration.

We can now show that if the problem of determining whether a discounted UMDP has an optimal finite controller is decidable, then so is the halting problem for TMs. Given a TM, construct the UMDP as above. Use the hypothesized decision procedure to determine whether the discounted UMDP has an optimal finite controller. If not, we conclude that the TM does not halt (and in fact also never repeats a configuration). If the UMDP does

have an optimal finite controller, then simulate the TM on the empty string until either a halting configuration is reached, in which case we conclude that the TM halts, or until a configuration is repeated, in which case we conclude that the TM does not halt. One of these two possibilities must occur, and so this algorithm always halts.

It follows that the (two-counter) Turing machine halting problem reduces to the problem of whether an optimal finite controller exists for a discounted UMDP, and so the latter problem is undecidable. □

We note that, as a consequence of Theorem 4.4 the seemingly easier problem of whether there exists a string (finite sequence) that is optimal is also undecidable: we can add a single action, call it e, to the start state of the leaky PFA of Theorem 4.4. Action e leads to the accepting and rejecting states with equal 0.5 probability. Then the UMDP would have an optimal finite sequence (action e) if and only if the TM is not accepting, by Lemma 4.6. Consequently, we also see that the related problem of given a UMDP, an initial distribution and an action sequence, whether we can do better than the given action sequence, i.e., the UMDP has higher expected value than what the action sequence would provide, is undecidable, even in the discounted case.

5. Summary and related work

The main technical result in this paper established the undecidability of a wide class of stochastic optimization problems:

- probabilistic extensions to the classical planning problem with no observability and with stochastic observability;
- infinite-horizon POMDPs under a variety of optimization criteria, including discounted models, both for exact and approximate solutions;
- the existence of an optimal finite controller for discounted and undiscounted POMDPs, and related problems.

It should hardly be surprising that the general problem—finding the optimal policy for a partially observable system with an arbitrary objective function—was intractable, if for no other reason than that it would be impossible to bound the optimal policy's expected reward. More surprising is that none of the usual ways to restrict and simplify the problem: positive-bounded models, average-reward models, discounted models, explicit state representation, settling for approximate solutions—make the problem easier to solve in the worst case.

Related work includes complexity analyses for:

- (deterministic) classical planning;
- MDPs and restricted variants of POMDPs;
- restricted variants of probabilistic planning;
- probabilistic finite-state automata.

Complexity results for deterministic planning have been available for many years, beginning with Chapman's [13] first NP-hardness result for STRIPs planning, followed by additional analysis by Bylander [11]. These results are related to this paper only because of the historical connection through the planning problem: the earlier results rely on a combinatorial analysis quite different from the techniques used to establish the results in this paper.

Early papers on POMDP problem formulations include those by Drake [16], Astrom [3], and Aoki [2]. Sondik and Smallwood [46,47] are probably the first in addressing computational difficulties associated with solving POMDPs. Earlier references on applications and algorithms are the surveys [30,37].

Analysis of the complexity of solving MDPs and POMDPs began with the work of Papadimitriou and Tsitsiklis [40], in which they formally establish NP-hardness results for a variety of finite-horizon POMDP problems, but only conjecture as to the undecidability of the POMDP infinite-horizon case. The computational complexity of finite-horizon control problems has received considerable attention recently, see for example [10,20,39,48]. For the infinite-horizon case, two questions, the complexity of goal-state reachability with either nonzero probability or probability one, which reduce to reachability computations and are decidable, had been studied by Alur et al. [1] and Littman [26]. Other work considered infinite-horizon fully observable MDPs [40,48], fully observable MDPs with exponentially many states but with compact representations [27,28], and the complexity of infinite-horizon problems on stochastic games, which generalize MDPs [14,31].

Littman, Goldsmith, and Mundhenk [29] analyze the complexity of propositional probabilistic planning problems inspired by the same model of planning derived from the classical AI literature [24]. Their results are based on a restriction placed on solutions, however, in that they limit the analysis to plans that can be expressed in size polynomial in the size of the problem specification. As a result, their results parallel the results for finite-horizon POMDPs. Rintanen [45] analyzes the complexity of probabilistic planning problems under average reward criteria, and establishes undecidability results for the infinite-horizon criteria based on our results.

Early work on probabilistic finite automata include [41,43]. Our proof techniques build on the techniques developed by Freivalds and Condon and Lipton [15,19] and are different from previous techniques of Paz and Bertoni et al. [5,41], which were based on reductions from Post's correspondence problem. Recently, Blondel and Canterini [7] have shown that several PFA problems such as the emptiness and threshold isolation are undecidable even for a PFA with a constant number of states and a two letter alphabet (e.g., 46 states for the case of the emptiness problem). Their proof of the undecidability of the emptiness problem is by a reduction from Post's correspondence problem, which is known to be undecidable for 7 pairs of words [35]. It is currently open whether there is an effective procedure for the emptiness problem for a PFA with just two states and two letters. Probabilistic finite automata have found applications in several different contexts, including the study of the power of randomizing in algorithms, space bounded proofs and Arthur and Merlin games, rational series and semigroups of matrices, as well as Markov decision processes and probabilistic planning [8,15,19,25].

It is now well established that optimal planning without full observability is prohibitively difficult both in theory and practice [29,32,40]. These results suggest that it may

be more promising to explore alternative problem formulations, including restrictions on the system dynamics and the agent's sensing and effecting powers that are useful for realistic problem domains yet are more amenable to exact or approximate solution algorithms. The discovery that interesting problems are computationally intractable in the worst case should neither be surprising nor discouraging to the AI researcher at this point.

Acknowledgements

Thanks to Bill Rounds, who first pointed out to us the connection between PFAs and probabilistic planning. Thanks to Michael Littman and Eric Hansen for valuable discussions and input, and in particular raising the question of the existence of optimal finite controller problem. Thanks to the reviewers for many valuable comments and suggestions. Hanks and Madani were supported in part by ARPA/Rome Labs Grant F30602-95-1-0024, and in part by NSF grant IRI-9523649. Anne Condon was supported by NSF grants CCR-92-57241 HRD-627241 and by NSERC research grant.

Appendix A. The weak equality test

Here we describe in detail the algorithm, carried out by a PFA, for the weak equality test. To see that such an algorithm can be executed by a PFA, it is easier to think of the PFA model as the equivalent one-way TM machine model, where the machine has a read-only input tape and a finite memory, and can also flip a fair coin on each step and base its transition on the coin flip in addition to its input and its state. Given is the input $a^i b^j$, and the question is whether $i = j$. The outputs of the algorithm (the state the PFA ends in) are *Indecision*, *Suspect* or *Correct* as described in Section 3.1.2. The PFA performs the following independent computations as it scans the input:

(1) For each letter a, the algorithm flips two coins.
(2) For each letter b, the algorithm flips two coins.
(3) For each letter a and for each letter b, the algorithm flips one coin.
(4) For each letter a and for each letter b, the algorithm flips one coin.
(5) The algorithm checks whether $i \equiv j \mod k$,

where $k \geqslant 1$ is a constant. It can be easily verified that the algorithm can be carried out by a PFA for the operations described. If $i \not\equiv j \pmod{k}$, then the outcome is Suspect. Otherwise, let event A be true if computations 1 or 2 get only heads, and let event B be true if computations 3 or 4 get only heads. The algorithm outputs Indecision (the PFA goes into indecision state) if both A and B are false (the common case), or both are true. Otherwise, in case of a *Decision* outcome, the algorithm outputs Suspect if A is true and B is false, and it outputs Correct if A is false and B is true. We will next show that if $i = j$, then A and B have the same probability of being true, while if $i \neq j$, in case of a Decision outcome, and with $k \geqslant 2$, the probability of outputting Suspect is much higher than outputting Correct.

We have $p_A = Pr(A) = 2^{-2i} + 2^{-2j} - 2^{-2i-2j}$, and $p_B = Pr(B) = 2^{-i-j} + 2^{-i-j} - 2^{-2i-2j}$. Therefore, if $i = j$, $p_A = p_B$ and in case of a Decision outcome, Suspect and Correct outcomes are equally likely.

If $i \neq j$, assume without loss of generality that $j = i + lk$ for some integer $l \geqslant 1$. First note that with $k \geqslant 2$ or $l \geqslant 1$, p_A is larger than p_B as $p_A > 2^{-2i}$, while $p_B < 2^{-2i-2lk+1}$. The probability of the Suspect outcome given Decision is

$$Pr(A \mid \bar{A}B \text{ or } A\bar{B}) = Pr(A\bar{B} \mid A\bar{B} \text{ or } \bar{A}B) = \frac{Pr(A\bar{B} \text{ and } (\bar{A}B \text{ or } A\bar{B}))}{Pr(\bar{A}B \text{ or } A\bar{B})}$$
$$= \frac{Pr(A\bar{B})}{Pr(\bar{A}B \text{ or } A\bar{B})} = \frac{p_A(1-p_B)}{Pr(\bar{A}B \text{ or } A\bar{B})}.$$

Similarly, the probability of Correct given Decision is

$$Pr(B \mid A\bar{B} \text{ or } \bar{A}B) = \frac{p_B(1-p_A)}{p_A(1-p_B) + (1-p_A)p_B}.$$

Consequently, given a Decision outcome, the probability of Suspect is at least 2^k times higher than the probability of Correct:

$$\frac{Pr(A \mid \bar{A}B \text{ or } A\bar{B})}{Pr(B \mid A\bar{B} \text{ or } \bar{A}B)} = \frac{p_A(1-p_B)}{p_B(1-p_A)} > \frac{p_A}{p_B} > \frac{2^{-2i}}{2^{-2i-2lk+1}} = 2^{2k-1}.$$

In summary, we have shown that on input $a^i b^j$, if $i = j$ then the outcomes Suspect or Correct are equally likely, and if $i \neq j$ then the outcome Suspect is at least 2^k times more likely than the outcome Correct. Thus, the PFA meets the criteria listed in Section 3.1.2.

References

[1] R. Alur, C. Couroubetis, M. Yannakakis, Distinguishing tests for nondeterministic and probabilistic machines, in: Proc. of 27th STOC, 1995, pp. 363–372.
[2] M. Aoki, Optimal control of partially observable Markovian systems, J. Franklin Inst. 280 (1965) 367–386.
[3] K.J. Ästrom, Optimal control of Markov processes with incomplete state information, J. Math. Anal. Appl. 10 (1965) 174–205.
[4] A. Bertoni, The solution to problems relative to probabilistic automata in the frame of the formal languages theory, in: G. Goos, J. Hartmanis (Eds.), Vierte Jahrestagung der Gesellschaft für Informatik, Springer, Berlin, 1975, pp. 107–112.
[5] A. Bertoni, G. Mauri, M. Torelli, Some recursively unsolvable problems relating to isolated cutpoints in probabilistic automata, in: Automata, Languages and Programming, in: Lecture Notes in Computer Science, Springer, Berlin, 1977, pp. 87–94.
[6] D.P. Bertsekas, Dynamic Programming: Deterministic and Stochastic Models, Prentice-Hall, Englewood Cliffs, NJ, 1987.
[7] V.D. Blondel, V. Canterini, Undecidable problems for probabilistic finite automata of fixed dimension, Technical Report, University of Louvain (Belgium), 2001. Available from: http://www.inma.ucl.ac.be/~blondel/publications/.
[8] V.D. Blondel, J.N. Tsitsiklis, A survey of computational complexity results in systems and control, Automatica 36 (9) (2000) 1249–1274.
[9] C. Boutilier, T. Dean, S. Hanks, Decision theoretic planning: Structural assumptions and computational leverage, J. Artificial Intelligence Res. 11 (2000) 1–94.

[10] D. Burago, M. De Rougemont, A. Slissenko, On the complexity of partially observed Markov decision processes, Theoret. Comput. Sci. (1996) 161–183.
[11] T. Bylander, The computational complexity of propositional STRIPS planning, Artificial Intelligence 69 (1994) 161–204.
[12] A.R. Cassandra, Exact and approximate algorithms for partially observable Markov decision processes, PhD Thesis, Brown University, 1998.
[13] D. Chapman, Planning for conjunctive goals, Artificial Intelligence 32 (3) (1987) 333–377.
[14] A. Condon, The complexity of simple stochastic games, Inform. and Comput. 96 (2) (1992) 203–224.
[15] A. Condon, R. Lipton, On the complexity of space bounded interactive proofs, in: Proc. 30th Annual Symposium on Foundations of Computer Science, 1989.
[16] A.W. Drake, Observation of a Markov process through a noisy channel, PhD Thesis, Massachusetts Institute of Technology, 1962.
[17] D. Draper, S. Hanks, D. Weld, Probabilistic planning with information gathering and contingent execution, in: Proceedings of the Second International Conference on Artificial Intelligence Planning Systems, AAAI Press, Menlo Park, CA, 1994, pp. 31–36.
[18] R.E. Fikes, N.J. Nilsson, STRIPS: A new approach to the application of theorem proving to problem solving, Artificial Intelligence 2 (3–4) (1971) 189–208.
[19] R. Freivalds, Probabilistic two way machines, in: Proc. International Symposium on Mathemarical Foundations of Computer Science, Vol. 118, Springer, Berlin, 1981, pp. 33–45.
[20] J. Goldsmith, M. Mundhenk, Complexity issues in Markov decision processes, in: Proc. IEEE Conference on Computational Complexity, 1998, pp. 272–280.
[21] E.A. Hansen, Finite memory control of partially observable systems, PhD Thesis, University of Massachusetts, Amherst, 1998.
[22] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 1979.
[23] T.L. Dean, K. Kim, N. Meuleau, Approximate solutions to factored Markov decision processes via greedy search in the space of finite controllers, in: Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling, 2000, pp. 323–330.
[24] N. Kushmerick, S. Hanks, D.S. Weld, An algorithm for probabilistic planning, Artificial Intelligence 76 (1995) 239–286.
[25] S. Moran, L. Babai, Arthur–Merlin games: A randomized proof system, and a hierarchy of complexity classes, J. Computer System Sci. 36 (1988) 254–276.
[26] M.L. Littman, Algorithms for sequential decision making, PhD Thesis, Brown, 1996.
[27] M.L. Littman, Probabilistic propositional planning: Representations and complexity, in: Proceedings of the 14th National Conference on AI, AAAI Press, 1997.
[28] M.L. Littman, T.L. Dean, L.P. Kaebling, On the complexity of solving Markov decision problems, in: Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence, 1995.
[29] M.L. Littman, J. Goldsmith, M. Mundhenk, The computational complexity of probabilistic planning, J. Artificial Intelligence Res. 9 (1998) 1–36.
[30] W. Lovejoy, A survey of algorithmic methods for partially observable Markov decision processes, Ann. Oper. Res. (1991) 47–66.
[31] W. Ludwig, A subexponential randomized algorithm for the simple stochastic game problem, Inform. and Comput. 117 (1995) 151–155.
[32] C. Lusena, J. Goldsmith, M. Mundhenk, Nonapproximability results for partially observable Markov decision processes, J. Artificial Intelligence Res. 14 (2001) 83–103.
[33] O. Madani, Complexity results for infinite-horizon Markov decision processes, PhD Thesis, University of Washington, 2000.
[34] O. Madani, S. Hanks, A. Condon, On the computability of infinite-horizon partially observable Markov decision processes, in: Proc. of 16th National Conference in Artificial Intelligence, 1999, pp. 541–548.
[35] Y. Matiyasevich, G. Senizergues, Decision problems for semi-thue systems with a few rules, in: Proc. 11th Annual IEEE Symposium on Logic in Computer Science, 1996, pp. 523–531.
[36] N. Meuleau, L. Peshkin, K. Kim, L.P. Kaelbling, Learning finite-state controllers for partially observable environments, in: K.B. Laskey, H. Prade (Eds.), Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-99), Morgan Kaufmann, San Francisco, CA, 1999, pp. 427–436.

[37] G. Monahan, A survey of partially observable Markov decision processes: Theory, models, and algorithms, Management Sci. 28 (1) (1982) 1–15.
[38] M. Mundhenk, J. Goldsmith, E. Allender, The complexity of policy evaluation for finite-horizon partially-observable Markov decision processes, in: Proc. 22nd Mathematical Foundations of Computer Science, 1997, pp. 129–138.
[39] M. Mundhenk, J. Goldsmith, C. Lusena, E. Allender, Complexity results for finite-horizon Markov decision processs problems, J. ACM 47 (4) (2000).
[40] C.H. Papadimitriou, J.N. Tsitsiklis, The complexity of Markov decision processes, Math. Oper. Res. 12 (3) (1987) 441–450.
[41] A. Paz, Introduction to Probabilistic Automata, Academic Press, New York, 1971.
[42] M.L. Puterman, Markov Decision Processes, Wiley Interscience, 1994.
[43] M.O. Rabin, Probabilistic automata, Inform. and Control 6 (3) (1963) 230–245.
[44] M.O. Rabin, Lectures on classical and probabilistic automata, in: E.R. Caianiello (Ed.), Automata Theory, Academic Press, New York, 1966.
[45] J. Rintanen, Complexity of probabilistic planning under average rewards, in: Proc. IJCAI-01, Seattle, WA, Morgan Kaufmann, San Francisco, CA, 2001, pp. 503–508.
[46] R. Smallwood, E.J. Sondik, The optimal control of partially observable Markov processes over a finite horizon, Oper. Res. 21 (1973) 1071–1088.
[47] E.J. Sondik, The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs, Oper. Res. 26 (2) (1978) 282–304.
[48] P. Tseng, Solving H-horizon stationary Markov decision process in time proportional to $\log(H)$, Oper. Res. Lett. 9 (5) (1990) 287–297.
[49] D.J. White, Markov Decision Processes, Wiley, New York, 1993.
[50] W. Zhang, Algorithms for partially observable Markov decision processes, PhD Thesis, Hong Kong University of Science and Technology, 2001.