Distinguishing tests for nondeterministic and probabilistic machines

(Appeared in the Proceedings of the 27th ACM Symposium on Theory of Computing, pp. 363-372, 1995)

Rajeev Alur* Costas Courcoubetis † Mihalis Yannakakis*

Abstract

We study the problem of uniquely identifying the initial state of a given finite-state machine from among a set of possible choices, based on the input-output behavior. Equivalently, given a set of machines, the problem is to design a test that distinguishes among them. We consider nondeterministic machines as well as probabilistic machines. In both cases, we show that it is PSPACE-complete to decide whether there is a *preset* distinguishing strategy (i.e. a sequence of inputs fixed in advance), and it is EXPTIME-complete to decide whether there is an *adaptive* distinguishing strategy (i.e. when the next input can be chosen based on the outputs observed so far). The probabilistic testing is closely related to probabilistic games, or Markov Decision Processes, with incomplete information. We also provide optimal bounds for deciding whether such games have strategies winning with probability 1.

1 Introduction

Finite-state machines have been widely used to model systems in diverse areas of computer science. Our interest mainly stems from their use in modeling communication protocols, controllers, and digital hardware [11, 14, 17]. We consider Mealy machines whose state-transitions are triggered by inputs producing outputs. In testing problems, we are given a machine M about which we lack some information. We wish to design an experiment that allows us to deduce the missing information by providing inputs to the machine and observing the outputs produced. There is an extensive literature on testing finite state machines, starting from Moore's seminal paper on 'gedanken-experiments' [24], motivated first from circuits (see [11, 16]), and more recently from testing of communication protocols [2, 3, 4, 5, 9, 18, 19, 21, 22, 28, 29, 31, 32, 33]; see [20] for a survey and more complete bibliography.

For the testing problem considered in this paper, the machine M is completely specified except for its initial state. We are given a possible set of choices for the initial state, and the goal is to design an experiment that provides inputs to the machine so that the initial state can be uniquely inferred from the outputs produced by the machine. Such experiments are called *distinguishing strategies*. Equivalently, a distinguishing strategy allows distinguishing among a given set of two or more (initialized) machines. Given two machines M_1 and M_2 with initial states s_1 and s_2 respectively, we can form their disjoint union M and seek an experiment that distinguishes states s_1 and s_2. For example, M_1 could be a specification machine and M_2 a possible faulty realization.

If the sequence of inputs is to be fixed ahead of time, the experiment is called a *preset distinguishing strategy*, and if the input symbol at every step can depend on the previously observed outputs, the experiment is called an *adaptive distinguishing strategy*. Such distinguishing strategies for deterministic machines were first studied by Moore [24], and later by other authors (eg. [13, 16, 11]). The complexity of checking whether there exist distinguishing strategies for deterministic machines is fully resolved in [19].

Most of the past work on testing finite state machines concerns deterministic machines. In the first part of the paper, we study distinguishing strategies for nondeterministic machines. Nondeterminism is

*AT&T Bell Labs, Murray Hill, NJ 07974 (email: alur,mihalis@research.att.com).
†Univ. of Crete, ICS-FORTH, Heraklion, Greece (email: courcou@csi.forth.gr).

indispensable in the theory of verification of concurrent systems: it is useful as an abstraction of real systems, and plays an important role in modeling concurrency when the system consists of several independent coordinating components [23, 17]. Modeling of delays in real-time systems also introduces nondeterminism [1]. For nondeterministic machines, a state may produce several output sequences in response to a single input sequence. An experiment distinguishes conclusively between two states if the outputs possible from the two states are disjoint. Testing of nondeterministic machines can be formulated as a two-player game with incomplete information: the adversary chooses an initial state, and then, at each step, the existential player chooses an input, and the adversary produces an output, and updates the machine-state. The existential player wins if the initial state can be deduced correctly from the observed outputs.

We show that the problem of testing whether a nondeterministic machine has a preset strategy that distinguishes between a pair of states is PSPACE-complete, and thus, the length of the distinguishing input sequence needs to be exponentially large. The problem of testing whether a nondeterministic machine has an adaptive strategy that distinguishes between a pair of states is shown to be EXPTIME-complete. For all our results, the complexity stays unchanged when we consider distinguishing among a set of initial states, instead of a pair of initial states. Our bounds are in conformance with the complexity of deciding the existence of winning strategies for two-player games with incomplete information [27].

In the second part of the paper, we study distinguishing strategies for probabilistic machines. A probabilistic machine is a nondeterministic machine with probabilities attached to transitions. An experiment is almost-surely-distinguishing if the machine produces an output, with probability 1, that allows unique identification of the initial state. This qualitative question does not depend on the precise values of the transition probabilities. When viewed as a two-player game, the adversary is required to play randomly, instead of maliciously as in the nondeterministic testing. Thus, the probabilistic setting offers a more realistic modeling of choices in the context of testing. While certain testing problems have been studied for probabilistic systems [21, 22, 33], the complexity of distinguishing strategies has not been studied previously. Technically, the probabilistic case is much more challenging. The precise formulation of the problem is subtle, and the algorithms require careful proofs.

To understand the testing problems for probabilistic systems, we first study probabilistic games with incomplete information between an existential and a probabilistic player. The existential player chooses its moves based on the outputs observed so far, and the moves of the probabilistic player are random according to the transition-probabilities. A winning strategy for the existential player is one that almost-surely leads to one of the accepting states. The question is whether there is an almost-surely-winning strategy. Note that such qualitative questions regarding probabilistic games with partial information were not addressed in earlier work on probabilistic game automata and games against nature [6, 7, 25, 26] (as explained in Section 4), while the study of winning strategies for Markov Decision processes [10, 8, 30] assumes that the existential player has complete information regarding the state. We show that the decision problem of checking the existence of an almost-surely-winning strategy is EXPTIME-complete. The problem is PSPACE-complete for blindfold games where the existential player does not get any additional information during the game, and thus chooses a strategy in advance. Since such games arise in other contexts also, we believe these results to be interesting in their own right.

The probabilistic testing problems are reducible to the probabilistic games, however, at a cost of an exponential blow-up. We show that this blow-up can be avoided: deciding existence of a almost-surely-distinguishing preset strategy is PSPACE-complete, and almost-surely-distinguishing adaptive strategy is EXPTIME-complete.

2 Nondeterministic Systems

We use $s, t, u, \ldots$ for states, $a, b, c, \ldots$ for input symbols, and $p, q, r \ldots$ for output symbols. A (finite) word $a_0 a_1 \ldots a_n$ is denoted by $\overline{a}$; an infinite sequence $a_0 a_1 \ldots$ is denoted by $\mathbf{a}$.

A nondeterministic finite-state Mealy machine M has a finite nonempty set S of states, a finite nonempty set I of input symbols, and a finite nonempty set O of output symbols. The transitions are labeled with input-output pairs: $s \overset{a/p}{\to} t$ denotes a transition from s to t on input a with output p. The machine is assumed to be input-enabled: for every state s and every input symbol a, there exists an output symbol p and a state t such that $s \overset{a/p}{\to} t$. The transition relation is extended to words also: $s \overset{\overline{a}/\overline{p}}{\to} t$ means that state s can transition to state t producing output $\overline{p}$ as a response to input $\overline{a}$. The transition relation also gives the output-function λ: $\overline{p} \in \lambda(s, \overline{a})$ iff state s can produce output $\overline{p}$ on input $\overline{a}$, and $\mathbf{p} \in \lambda(s, \mathbf{a})$

iff the infinite output $\mathbf{p}$ is possible from s on input $\mathbf{a}$.

The machine M is *deterministic* iff given a state s and an input symbol a, there is exactly one output symbol p and exactly one state t such that $s \stackrel{a/p}{\to} t$. The machine M is *output-deterministic* iff given a state s, an input symbol a, and an output symbol p, there is exactly one state t such that $s \stackrel{a/p}{\to} t$.

2.1 Preset strategies

A *preset strategy* for a machine M is a finite word $\overline{a} \in I^*$. The preset strategy $\overline{a}$ is (s,t)-*distinguishing*, for two states s and t, iff the two sets $\lambda(s,\overline{a})$ and $\lambda(t,\overline{a})$ are disjoint. Thus, if $\overline{a}$ is (s,t)-distinguishing, then it is possible to decide from the output response to the input $\overline{a}$ whether the machine started in state s or in state t. The *preset-decision-problem* is to decide whether a given nondeterministic machine M has a (s,t)-distinguishing preset strategy.

Note that distinguishability of two states of a nondeterministic machine differs from the various notions of equivalence (eg. language, failures, testing equivalence etc.) that have been defined and studied in the literature [23, 15]. The difference amounts to whether we regard the machine that is being tested to be cooperating with the tester to show a difference between two states, or as an adversary that is trying to hide the difference from the tester.

When M is deterministic, the classical $O(n \log n)$ Hopcroft-algorithm is used to solve the preset-decision-problem. We present an algorithm for nondeterministic machines. When M is nondeterministic, construct a deterministic automaton N_M over the alphabet I as follows.

- The state-space of N_M is 2^{S^2}, i.e., a state of N_M is a set of pairs of states of M.
- The machine N_M has an edge $U \stackrel{a}{\to} V$ iff the set V consists of pairs (u',v') such that for some $(u,v) \in U$ and output p, $u \stackrel{a/p}{\to} u'$ and $v \stackrel{a/p}{\to} v'$.

Lemma 1 *A preset strategy $\overline{a}$ is (s,t)-distinguishing for M iff $\{(s,t)\} \stackrel{\overline{a}}{\to} \emptyset$ in the automaton N_M.*

The condition that the empty set is reachable in N_M from the initial-set $\{(s,t)\}$ can be checked in PSPACE. In the worst case, the length of a (s,t)-distinguishing preset strategy for M is exponential.

Proposition 2 *The preset-decision-problem is solvable in* PSPACE.

2.2 Adaptive strategies

An *adaptive strategy* for a machine M is a function $\mu : O^* \mapsto I$ [1]. For an infinite input sequence $\mathbf{a} \in I^\omega$ and an output sequence $\mathbf{p} \in O^\omega$, we write $\mathbf{p} \in \mu^{-1}(\mathbf{a})$ if the inputs in $\mathbf{a}$ are according to the strategy μ applied to the outputs in $\mathbf{p}$, that is, $a_0 = \mu(\epsilon)$, and for all $i \geq 0$, $a_{i+1} = \mu(p_0 p_1 \dots p_i)$.

The (s,t)-distinguishing adaptive strategy is best understood as a two-player game. The adversary chooses a state $u \in \{s,t\}$, and the protagonist attempts to find out whether $u = s$ or $u = t$ by providing inputs based on the outputs produced. If the output produced is possible only from one of the two initial states, then the protagonist can decide, and wins the game. If the game continues forever, the adversary wins. The (s,t)-distinguishing adaptive strategy is a winning strategy for the protagonist.

The adaptive strategy μ for the machine M is (s,t)-distinguishing, for two states s and t, iff for all input sequences $\mathbf{a} \in I^\omega$, the intersection $\mu^{-1}(\mathbf{a}) \cap \lambda(s,\mathbf{a}) \cap \lambda(t,\mathbf{a})$ is empty. The *adaptive-decision-problem* is to decide whether a given nondeterministic machine M has a (s,t)-distinguishing adaptive strategy. Our algorithm constructs a set X containing pairs of sets of states as follows. Initially X is empty.

1. for all nonempty sets $T \subseteq S$, add both $(T,\emptyset) \in X$ and $(\emptyset,T)$ to X.
2. Choose $U,V \subseteq S$ such that $(U,V) \notin X$, and there exists an input a such that for all outputs p, $(U',V') \in X$ for $U \stackrel{a/p}{\to} U'$ and $V \stackrel{a/p}{\to} V'$. Add (U,V) to X. If no such pair (U,V) exists then terminate.

Lemma 3 *There exists an (s,t)-distinguishing adaptive strategy iff $(\{s\},\{t\})$ is in X upon termination.*

Proposition 4 *The adaptive-decision-problem for a nondeterministic machine with n states is solvable in time $2^{O(n)}$.*

When the given machine M is output-deterministic, it suffices to consider only pairs of states, instead of pairs of state-sets:

Proposition 5 *The adaptive-decision-problem for a output-deterministic machine with n states and m transitions is solvable in time $O(n \cdot m)$.*

[1] An adaptive strategy, equivalently, may be considered as a function from strings over input-output pairs to input symbols.

2.3 Two-player games with incomplete information

The lower bounds on the preset- and adaptive-decision-problems follow from their relationship to two-player games with incomplete information.

A two-player $\exists\forall$-game G consists of a nondeterministic machine with game-positions S, inputs I, outputs O, and transition relation labeled with input-output pairs, together with a set $S^i \subseteq S$ of start-positions and a set $S^f \subseteq S$ of winning positions. The game starts at some start-position, and the $\exists$-player wins if the game is in a winning position. The $\exists$-player knows only the outputs, and an $\exists$-strategy is a function $\mu : O^* \mapsto I$. Consider a game-position s. In one move, the $\exists$-player chooses an input a by applying its strategy to the outputs seen so far. The $\forall$-player chooses an output p, and updates the game-position to t such that $s \stackrel{a/p}{\to} t$. If the game continues forever then the $\forall$-player wins.

The *outcome-problem* for a $\exists\forall$-game G is to decide whether there is a winning $\exists$-strategy. Reif [27] has studied the complexity of such games in a general setting. The outcome-problem for $\exists\forall$-games is EXPTIME-complete [27]. We reduce the outcome-problem to the adaptive-decision-problem for nondeterministic machines. Given a $\exists\forall$-game G, construct a nondeterministic machine M_G by adding two new states s^i and s^f, a new output p^f, and the following new transitions

$$
\begin{aligned}
& s \stackrel{a/p^f}{\to} s^f \text{ for all } s \in S^f \\
& s^i \stackrel{a/p}{\to} t \text{ for all } t \in S^i,\ a \in I,\ p \in O \\
& s^f \stackrel{a/p}{\to} s^f \text{ for all } a \in I,\ p \in O.
\end{aligned}
$$

The $\exists\forall$-game G has a winning $\exists$-strategy iff M_G has a (s^i, s^f)-distinguishing adaptive strategy.

Theorem 6 *The adaptive-decision-problem for nondeterministic machines is* EXPTIME*-complete.*

The $\exists\forall$-game G is a *blindfold* game if the set O is a singleton set. In this case, the inputs chosen by a $\exists$-strategy are fixed a priori. The outcome-problem for blindfold $\exists\forall$-games is PSPACE-complete [27]. The outcome-problem of blindfold $\exists\forall$-games is reducible to the preset-decision-problem of output-deterministic machines.

Theorem 7 *The preset-decision-problem for nondeterministic as well as output-deterministic machines is* PSPACE*-complete.*

2.4 Distinguishing among multiple states

Suppose we are given a set $T \subseteq S$ of possible initial states, and we want to infer the initial state by providing inputs. When T equals S, this problem is called the *state-identification* problem. A preset strategy $\overline{a}$ for a machine M is T-distinguishing iff the sets $\lambda(s, \overline{a})$ and $\lambda(t, \overline{a})$ are disjoint for all pairs of states $s, t \in T$. Thus, the output response to a T-distinguishing preset strategy identifies a unique initial state in T. The solution to the preset-decision-problem (i.e. for the case T contains only 2 states) generalizes to a decision procedure to check if a T-distinguishing preset strategy exists. The PSPACE lower bound holds even when the machine M is deterministic and T equal S [19].

Theorem 8 *Given a nondeterministic machine M and a subset T of its states, deciding the existence of a T-distinguishing preset strategy for M is* PSPACE*-complete.*

The adaptive strategy μ for a machine M is T-distinguishing, for $T \subseteq S$, iff for all input sequences $\mathbf{a} \in I^\omega$, and for all $s, t \in T$, the intersection $\mu^{-1}(\mathbf{a}) \cap \lambda(s, \mathbf{a}) \cap \lambda(t, \mathbf{a})$ is empty. The T-distinguishing adaptive strategy is again a winning strategy for the protagonist in a two-player game: the adversary chooses an initial state $u \in T$, and the protagonist guesses u by providing the inputs based on the outputs produced. The algorithm for distinguishing between two states can be modified to check for the existence of a T-distinguishing adaptive strategy.

Theorem 9 *Given a nondeterministic machine M and a subset T of its states, deciding the existence of a T-distinguishing preset strategy for M is* EXPTIME*-complete.*

Note that, when M is deterministic and T equals S, there is a polynomial-time algorithm for deciding the existence of a T-distinguishing adaptive strategy [19].

3 Probabilistic Systems

A (finite-state) probabilistic machine A has state-space S, inputs I, and outputs O. The transition-probabilities are specified by a function $\nu : S \times I \times O \times S \mapsto [0, 1]$ such that for every state s and every input a, the sum $\sum_{p,t} \nu(s, a, p, t)$ equals 1. For the qualitative problems, it only matters whether a value $\nu(s, a, p, t)$ is nonzero. We write $s \stackrel{a/p}{\to} t$ iff $\nu(s, a, p, t) > 0$. Thus, we consider the probabilistic machine A also as a nondeterministic machine.

The transition-probability function extends to input words also: $\nu(s, \overline{a}, \overline{p})$ is the probability that state s on input $\overline{a}$ produces output $\overline{p}$.

3.1 Preset strategies

A *preset strategy* for a probabilistic machine A is an infinite sequence $\mathbf{a} \in I^\omega$. For an initial state s and a preset strategy $\mathbf{a}$, the set of possible (infinite) output sequences is $\lambda(s, \mathbf{a})$. The probability distribution of A gives a probability measure over the sets of output sequences. For $\Gamma \subseteq O^\omega$, $\nu(s, \mathbf{a}, \Gamma)$ is the probability that state s produces an output in Γ on input $\mathbf{a}$. In particular, $\nu(s, \mathbf{a}, \lambda(s, \mathbf{a}))$ equals 1.

A preset strategy $\mathbf{a}$ *almost-surely-distinguishes* a state s from t iff $\nu(s, \mathbf{a}, \lambda(t, \mathbf{a})) = 0$. That is, the probability that on input $\mathbf{a}$, state s produces an output that is also possible from state t, is 0. A preset strategy $\mathbf{a}$ is (s,t)-*a.s.distinguishing* iff $\mathbf{a}$ a.s.distinguishes both s from t and t from s. The (s,t)-a.s.distinguishing preset strategy can be viewed as a game as described below. The adversary chooses a state $u \in \{s, t\}$, and the protagonist attempts to find out whether $u = s$ or $u = t$, by providing the input sequence $\mathbf{a}$, and observing the infinite output. If the output is *possible* from only one of the two states, then the protagonist wins. If the output belongs to the set $\Gamma = \lambda(s, \mathbf{a}) \cap \lambda(t, \mathbf{a})$ of common outputs, then the adversary wins. If $\mathbf{a}$ is (s,t)-a.s.distinguishing, then the set Γ has zero measure whether the system starts in s or whether the system starts in t, and the protagonist wins with probability 1. The next proposition shows that, in this case, the probability that the output observed after the prefix $a_0 \ldots a_i$ is possible from both the initial states goes to 0 as i goes to infinity, and hence the protagonist can almost-surely infer the initial state in finite time.

Proposition 10 *The preset-strategy* $\mathbf{a}$ *is* (s,t)-*a.s.distinguishing strategy, iff both the probabilities* $\sum_{\overline{p} \in \lambda(t, a_0 \ldots a_i)} \nu(s, a_0 \ldots a_i, \overline{p})$ *and* $\sum_{\overline{p} \in \lambda(s, a_0 \ldots a_i)} \nu(t, a_0 \ldots a_i, \overline{p})$ *converge to 0 with increasing* i.

Proof Sketch: The proof relies on the fact that the set $\Gamma = \lambda(s, \mathbf{a}) \cap \lambda(t, \mathbf{a})$ of common outputs is limit-closed, i.e. if every finite prefix of an infinite sequence $\mathbf{p}$ can be extended to a sequence in Γ then $\mathbf{p}$ itself is in Γ. ∎

The *probabilistic-preset-decision-problem* is to decide the existence of an (s,t)-a.s.distinguishing preset strategy for a given probabilistic machine A. Note that if the preset strategy $\overline{a}$ for A as a non-deterministic machine is (s,t)-distinguishing then any infinite sequence that extends $\overline{a}$ is a (s,t)-a.s.distinguishing preset strategy for A. Thus, if the answer to the preset-decision-problem (A, s, t) is YES then the answer to the probabilistic-preset-decision-problem (A, s, t) is also YES. The converse does not hold. For the machine A (see Figure 1), there is a (s,t)-a.s.distinguishing strategy, but no (s,t)-distinguishing strategy.

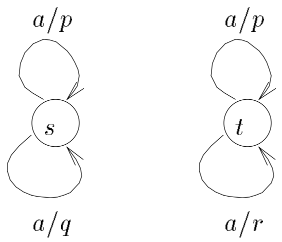

Figure 1: Machine A

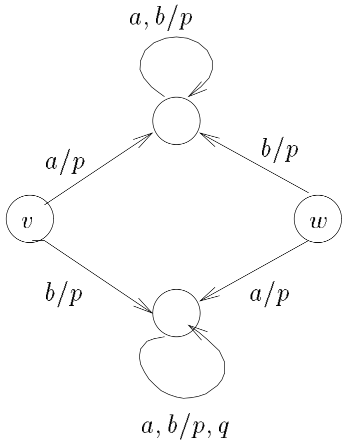

Figure 2: Machine B

It is possible that there is a strategy that a.s.distinguishes s from t, and there is another strategy that a.s.distinguishes t from s, but there is no (s,t)-a.s.distinguishing strategy. For the machine B (see Figure 2), for any preset strategy $\mathbf{c}$, the intersection $\lambda(v, \mathbf{c}) \cap \lambda(w, \mathbf{c})$ contains the single output p^ω. If $\mathbf{c}$ starts with a then $\nu(v, \mathbf{c}, p^\omega)$ is 1, and $\nu(w, \mathbf{c}, p^\omega)$ is 0, and $\mathbf{c}$ a.s.distinguishes w from v. If $\mathbf{c}$ starts with b then $\nu(v, \mathbf{c}, p^\omega)$ is 0, and $\nu(w, \mathbf{c}, p^\omega)$ is 1, and $\mathbf{c}$ a.s.distinguishes v from w. But there is no (v, w)-a.s.distinguishing strategy.

3.2 Adaptive strategies

The a.s.distinguishing adaptive strategy is similar to the game in the a.s.distinguishing preset case, except now the protagonist can decide the inputs on-the-fly after observing the outputs. Formally, an *adaptive strategy* for a probabilistic machine A is a function $\mu : O^* \mapsto I$. For an input word $\overline{a}$, the set $\mu^{-1}(\overline{a})$ of outputs consistent with the strategy is defined as before. For $i \geq 0$, let I^i be the set of input words of length i or less. Given two states s and t, for each $i \geq 0$, define

$$\nu_i(s,t) \;=\; \sum_{\overline{a} \in I^i} \; \sum_{\overline{p} \in \lambda(t,\overline{a}) \cap \mu^{-1}(\overline{a})} \nu(s, \overline{a}, \overline{p}).$$

Thus, $\nu_i(s,t)$ is the probability that state s produces an output that is possible from state t, when provided with inputs according to the strategy μ for i steps. The adaptive strategy μ a.s.distinguishes s from t iff this sequence $\nu_i(s,t)$ of probabilities converges to 0. The adaptive strategy μ is (s,t)-a.s.distinguishing iff μ a.s.distinguishes s from t and also t from s. The *probabilistic-adaptive-decision-problem* is to decide whether there exists an (s,t)-a.s.distinguishing adaptive strategy for a given probabilistic machine A.

Our observations regarding probabilistic preset strategies also apply to the adaptive case. As in the preset case, any (s,t)-distinguishing adaptive strategy is also a (s,t)-a.s.distinguishing adaptive strategy, but not vice versa. Also, there may be an adaptive strategy to a.s.distinguish s from t, an adaptive strategy to a.s.distinguish t from s, and yet no (s,t)-a.s.distinguishing adaptive strategy.

4 Probabilistic Games with Partial Information

Since testing of probabilistic systems is best understood as probabilistic games, we first study probabilistic games with incomplete information. The setting for the probabilistic game is similar to that of a $\exists\forall$-game, where the $\forall$-player is replaced by a randomizing player.

A $\exists R$-game H consists of a probabilistic machine with game-positions S, inputs I, outputs O, and transition-probability function ν, together with a set $S^i \subseteq S$ of start-positions and a set $S^f \subseteq S$ of winning positions. As in $\exists\forall$-games, the game starts in some start position, and $\exists$-player wins when the game is in a winning position. In one move, the $\exists$-player chooses an input according to its strategy, which is a function $\mu : O^* \mapsto I$, and then the output and the new position are chosen randomly according to the distribution. The strategy μ is *almost-surely-winning* if the $\exists$-player wins with probability 1. The *outcome-problem* for a $\exists R$-game H is to decide whether there is a a.s.winning $\exists$-strategy.

As in probabilistic testing, the answer to the outcome-problem depends only on the underlying transition-relation: let $s \overset{a/p}{\to} t$ iff $\nu(s,a,p,t)$ is positive. The $\exists R$-game H is called *blindfold* if the set O of outputs is a singleton set, and is a *complete-information* game if the transition-relation is output-deterministic. For complete-information games, the outcome-problem is solvable in polynomial-time [10, 6].

Our model of probabilistic games is related to space-bounded probabilistic game automata studied by Condon [6, 7]. While we consider the qualitative problem of winning with probability 1, Condon's problem is to determine if the $\exists$-player can win with probability 1/2. Our qualitative problem is easily reducible to Condon's quantitative problem (the random-player, at the beginning of the game, rejects with probability 1/2 else lets the $\exists$-player play its strategy). Condon proves an Exptime lower bound for the probability-1/2 problem, but no matching upper bound is known. Also, no bounds were provided for the blindfold case.

A $\exists R$-game is also a Markov decision process: the moves of the existential player correspond to the actions of the process, and the moves of the randomizing player are the random transitions of the process. Papadimitriou and Tsitsiklis [26] studied the partially observed Markov decision processes in a more general quantitative model, where transitions have associated costs and the objective is to minimize the total cost. They proved that in the finite horizon case, i.e. when the process lasts for a specified (finite) number of rounds, computing the optimal cost is Pspace-complete in the adaptive case and is NP-complete in the blindfold case. The authors say that the infinite horizon case in their model does not seem to be algorithmically solvable. Our results prove decidability—Exptime for the adaptive and Pspace for the blindfold—for an important qualitative case of the infinite horizon problem, where the objective is to reach one of the winning states (this can be easily expressed with costs).

4.1 Outcome-problem of blindfold $\exists R$-games

Consider a blindfold $\exists R$-game H. Since there is a single output, assume that transitions of H are labeled only with inputs. Since the game stops once a winning position is reached, without loss of generality, assume that if $s \xrightarrow{a} t$ and $s \in S^f$ then $t \in S^f$. A $\exists$-strategy is a preset sequence $\mathbf{a} \in I^\omega$. A preset strategy $\mathbf{a}$ is a.s.winning iff for every start-position $s \in S^i$, the probability $\sum_{t \notin S^f} \nu(s, a_0 \ldots a_i, t)$ that the $\exists$-player does not win the game after i steps goes to 0 as i goes to infinity.

Proposition 11 *For a blindfold $\exists R$-game H, let N_H be the deterministic automaton obtained from H by subset-construction. There is an a.s.winning $\exists$-strategy for the $\exists R$-game H iff there exists a state U of N_H and input words $\overline{a}$ and $\overline{b}$ such that*

1. *N_H has a path from S^i to U labeled with $\overline{a}$, and*
2. *N_H has a cycle from U to U labeled with $\overline{b}$, and*
3. *for all $u \in U$, H has a path from u to some winning-position labeled with $\overline{b}$.*

Proof Sketch: [*if*] Suppose that there exists a state U of N_H and input words $\overline{a}$ and $\overline{b}$ satisfying conditions 1–3. We show that the infinite sequence $\mathbf{c} = \overline{a}\,(\overline{b})^\omega$ is a.s.winning. Let $\overline{c}_i = \overline{a}\,(\overline{b})^i$ for $i \geq 0$. Consider a start-position $s \in S^i$. Define a sequence of probabilities as follows: for $i \geq 0$, let ν_i be the probability that s transitions to a non-winning position on input $\overline{c}_i$:

$$\nu_i = \sum_{u \in U - S^f} \nu(s, \overline{c}_i, u).$$

It suffices to prove that the sequence ν_i converges to 0.

The subset construction ensures that if $s \xrightarrow{\overline{c}_i} u$ then $u \in U$. Consider a state $u \in U$. From condition 3, there is a positive probability ν_u such that u goes to a winning position on input $\overline{b}$. Conditioned on the event that s transitions to u on $\overline{c}_i$, the input $\overline{c}_i\,\overline{b}$ leads s to a winning position with positive probability ν_u. Hence,

$$\nu_{i+1} \leq \sum_{u \in U - S^f} \nu(s, \overline{c}_i, u)(1 - \nu_u).$$

Since the ν_us are positive constants, there exists a positive $\varepsilon < 1$ such that $\nu_{i+1} \leq \varepsilon \nu_i$, and the sequence ν_i converges to 0.

[*only-if*] Suppose $\mathbf{a}$ is a.s.winning strategy for H. Define U_i such that $S^i \xrightarrow{a_0 \ldots a_i} U_i$. The automaton N_H has only finitely many states, and hence, there exists U such that $U_i = U$ for infinitely many i. Let $i_0 < i_1 < \ldots$ be a sequence such that $U_{i_j} = U$ for all $j \geq 0$. Set $\overline{c} = a_0 \ldots a_{i_0}$, and for $j > 0$, set $\overline{b}_j = a_{i_0+1} \ldots a_{i_j}$. We have $S^i \xrightarrow{\overline{c}} U$, and for all $j > 0$, $U \xrightarrow{\overline{b}_j} U$. We will show that one of the $\overline{b}_j$ words satisfies the condition 3. Consider a state $u \in U - S^f$. It suffices to show that there exists j such that $u \xrightarrow{\overline{b}_j} t$ for some $t \in S^f$. This follows from the fact there exists a j such that the probability that the game-position is non-winning after the prefix $\overline{c}\,\overline{b}_j$ is smaller than the probability that the game-position is u after $\overline{c}$. ∎

Proposition 11 allows us to reduce the outcome problem for the blindfold $\exists R$-game H to a search problem in the exponential-sized automaton N_H. It is easy to show the requirements 1–3 can be tested in space polynomial in the size of the game H. The PSPACE lower bound for the outcome problem of blindfold $\exists\forall$-games applies to the blindfold $\exists R$-games also.

Theorem 12 *The outcome-problem for blindfold $\exists R$-games is* PSPACE*-complete.*

4.2 Outcome-problem of $\exists R$-games

Consider an $\exists R$-game H. Construct a graph L_H that is deterministic in input-output pairs by subset-construction. The algorithm repeatedly deletes vertices and edges from L_H. At any stage a state v is said to be *reachable* from a state u iff there is a finite path $U_0 \xrightarrow{a_0/p_0} U_1 \cdots \xrightarrow{a_n/p_n} U_{n+1}$ in the (current) graph L_H such that $u \in U_0$, and $v \in U_{n+1}$, and there exist states $u_i \in U_i$ for $i = 1, \ldots n$ with $u \xrightarrow{a_0/p_0} u_1 \xrightarrow{a_1/p_1} \cdots \xrightarrow{a_n/p_n} v$. The algorithm applies the following steps repeatedly until the graph becomes stable:

1. Choose a vertex U of L_H with a state $u \in U$ such that for all $v \in S^f$, v is *not* reachable from u in L_H. If no such vertex exists then the graph L_H is stable, and terminate, let L_H^* be the stable graph upon termination.
2. If there is an edge from a vertex V to U labeled with some input a, then delete all edges starting from V labeled with the input a.
3. Delete the vertex U.

Proposition 13 *The $\exists R$-game H has an a.s.winning $\exists$-strategy iff L_H^* contains the vertex S^i.*

Proof Sketch: [*if*] We show that the $\exists$-player can win with probability 1. At every step of the game, the set of game-positions that the $\exists$-player considers possible depending on the input-outputs so far, corresponds to a vertex in L_H^*. Initially, this set is S^i. Consider a vertex U. If $U \subseteq S^f$, then the game is over. Otherwise, for each $u \in U$, let ν_u be the probability that the game is in position u conditioned on the input-outputs so far (our definition does not specify an initial distribution, each start-position can be considered equiprobable to compute these values). Choose $v \in U - S^f$ such that $\nu_v \geq \nu_u$ for all $u \in U - S^f$. The construction ensures that some $t \in S^f$ is reachable from v in L_H^*. Consider the path that leads v to t, and if a is the first input along this path, the $\exists$-player chooses a. Condition 2 ensures that for all outputs p, the graph L_H^* has a vertex V such that $U \stackrel{a/p}{\to} V$. This strategy ensures that the probability of not hitting a winning position goes to 0.

[*only-if*] Let μ be an a.s.winning $\exists$-strategy. Define a graph L as follows. A vertex $U \subseteq S$ belongs to L iff there exists an input word $\overline{a}$ and an output word $\overline{p} \in \mu^{-1}(\overline{a})$ such that $S^i \stackrel{\overline{a}/\overline{p}}{\to} U$ in the original graph L_H. Clearly, S^i is in L. An edge $U \stackrel{b/q}{\to} V$ belongs to L iff there exists an input word $\overline{a}$ and an output word $\overline{p} \in \mu^{-1}(\overline{a})$ such that $\mu(\overline{p}) = b$ and $S^i \stackrel{\overline{a}/\overline{p}}{\to} U$ in the original graph L_H. We need to show that (1) for every U in L and every u in U, a winning position is reachable from u in L, and (2) if $U \stackrel{a/p}{\to} V$ is an edge in L, then for every output q if the original graph L_H has an edge $U \stackrel{a/q}{\to} V'$ then so does L. This implies that L is a subgraph of L_H^*, and hence S^i belongs to L_H^*. ∎

It is clear that the complexity of the algorithm to compute L_H^* is exponential in the size of the game H. Furthermore, the proof of the EXPTIME lower bound for the outcome problem for $\exists\forall$-games can be modified to obtain an identical lower bound for $\exists R$-games.

Theorem 14 *The outcome-problem for $\exists R$-games is* EXPTIME*-complete.*

5 Complexity of Probabilistic Testing

5.1 From probabilistic testing to $\exists R$-games

Given a probabilistic machine A and two states s and t, construct an $\exists R$-game H_A whose game-positions are triples (u, U, ℓ) with $u \in S$, $U \subseteq S$, and $\ell \in \{0, 1\}$. The bit ℓ corresponds to whether the adversary chooses s or t as the initial state. There are two initial-positions: $(s, \{t\}, 0)$ and $(t, \{s\}, 1)$. A position (u, U, ℓ) is winning iff U is the empty set. The game H_A has a transition $(u, U, \ell) \stackrel{a/p}{\to} (v, V, \ell)$ iff $u \stackrel{a/p}{\to} v$ and $U \stackrel{a/p}{\to} V$ in A.

Lemma 15 *A $\exists$-strategy μ is an a.s.winning $\exists$-strategy for the $\exists R$-game H_A iff μ is (s, t)-a.s.distinguishing adaptive strategy for A.*

From the $\exists R$-game H_A, construct a blindfold $\exists R$-game H'_A by renaming the outputs of H_A to a single output. A $\exists$-strategy for the $\exists R$-game H'_A is simply a sequence $\mathbf{a} \in I^\omega$.

Lemma 16 *A sequence* $\mathbf{a}$ *is an a.s.winning $\exists$-strategy for the blindfold $\exists R$-game H'_A iff* $\mathbf{a}$ *is (s, t)-a.s.distinguishing preset strategy for A.*

5.2 Algorithm for the probabilistic-preset-decision-problem

The reduction from the probabilistic-preset-decision-problem to the outcome-problem for blindfold $\exists R$-games causes an exponential blow-up, and gives an EXPSPACE decision procedure. However, it is possible to save the extra exponential. Given a probabilistic machine A, construct a deterministic automaton N_A as in the algorithm for the preset-decision-problem (Lemma 1).

Proposition 17 *The probabilistic machine A has an (s, t)-a.s.distinguishing preset strategy iff either $\emptyset$ is reachable from $\{(s, t)\}$ in N_A, or there exists a state U of N_A and input words $\overline{a}$ and $\overline{b}$ such that*

1. *$\{(s, t)\} \stackrel{\overline{a}}{\to} U$, and*
2. *$U \stackrel{\overline{b}}{\to} U$, and*
3. *for all u such that $(u, v) \in U$ for some v, there exists an output word $\overline{p} \in \lambda(u, \overline{b})$ such that if $(u, v) \in U$ then $\overline{p} \notin \lambda(v, \overline{b})$, and*
4. *for all u such that $(v, u) \in U$ for some v, there exists an output word $\overline{p} \in \lambda(u, \overline{b})$ such that if $(v, u) \in U$ then $\overline{p} \notin \lambda(v, \overline{b})$.*

The proof of Proposition 17 is along the lines of the proof of Proposition 11, and is omitted due to the lack of space. The required conditions 1–4 can be checked in space polynomial in the size of the machine A, and PSPACE-hardness can be proved as in case of the preset-decision-problem.

Theorem 18 *The probabilistic-preset-decision-problem is* PSPACE*-complete.*

5.3 Algorithm for the probabilistic-adaptive-decision-problem

The reduction from the probabilistic-adaptive-decision-problem to the outcome-problem for $\exists R$-games gives a doubly-exponential decision procedure. Again, the extra exponential can be saved. Consider a probabilistic machine A, and two states s and t. Construct a graph L_A as follows. The vertices of L_A are pairs (U, V) for $U, V \subseteq S$. The edges are labeled with input-output pairs: $(U, V) \stackrel{a/p}{\to} (U', V')$ iff $U \stackrel{a/p}{\to} U'$ and $V \stackrel{a/p}{\to} V'$. A *path* in L_A is a finite sequence $(U_0, V_0) \stackrel{a_0/p_0}{\to} \cdots \stackrel{a_n/p_n}{\to} (U_{n+1}, V_{n+1})$. We say that the vertex (U_{n+1}, V_{n+1}) is *left-reachable* from a state $u \in U_0$ iff there exist states $u_i \in U_i$ for $i = 1, \ldots, n+1$ such that $u \stackrel{a_0/p_0}{\to} u_1 \stackrel{a_1/p_1}{\to} \cdots \stackrel{a_n/p_n}{\to} u_{n+1}$, and *right-reachable* from a state $v \in V_0$ iff there exist states $v_i \in V_i$ for $i = 1, \ldots, n+1$ such that $v \stackrel{a_0/p_0}{\to} v_1 \stackrel{a_1/p_1}{\to} \cdots \stackrel{a_n/p_n}{\to} v_{n+1}$. The algorithm repeatedly deletes vertices and edges from L_A till the graph becomes stable:

1. Choose a vertex (U, V) of L_A such that *either*

   there exists a state $u \in U$ such that for all $W \subseteq S$, the vertex $(W, \emptyset)$ is *not* left-reachable from u in L_A, *or*
   there exists a state $v \in V$ such that for all $W \subseteq S$, the vertex $(\emptyset, W)$ is *not* right-reachable from v in L_A.

   If no such vertex exists then the graph L_A is stable, and terminate.
2. If there is an edge from (U', V') to (U, V) labeled with some input a then delete all edges starting from (U', V') labeled with the input a.
3. Delete the vertex (U, V).

Proposition 19 *The probabilistic machine A has an (s,t)-a.s.distinguishing adaptive strategy iff the graph L_A, upon termination, contains the vertex $(\{s\}, \{t\})$.*

The proof of Proposition 19 is similar to the proof of Proposition 13. This gives an exponential algorithm for the probabilistic-adaptive-decision-problem. A matching lower bound can be proved by using the EXPTIME-hardness of the adaptive-decision-problem.

Theorem 20 *The probabilistic-adaptive-decision-problem is* EXPTIME*-complete.*

5.4 Distinguishing among multiple states

As in the nondeterministic case, we consider the problem of uniquely identifying an initial state among the possible choices $T \subseteq S$. A preset strategy $\mathbf{a}$ is T-a.s.distinguishing iff $\mathbf{a}$ is (s,t)-a.s.distinguishing for all pairs $s, t \in T$.

Theorem 21 *Given a probabilistic machine A and a subset T of its states, deciding the existence of a T-a.s.distinguishing preset strategy for A is* PSPACE*-complete.*

The adaptive strategy μ for the probabilistic machine M is T-distinguishing if for all $s \in T$, if the inputs are provided according to the strategy μ, s almost-surely produces an output that is not possible from any other state in T: for all $s \in S^i$, the probability

$$\sum_{\overline{a} \in I^i} \sum_{\overline{p} \in \mu^{-1}(\overline{a}) \cap (\cup_{t \in T - \{s\}} \lambda(t, \overline{a}))} \nu(s, \overline{a}, \overline{p})$$

converges to 0 as i goes to infinity.

Theorem 22 *Given a probabilistic machine A and a subset T of its states, deciding the existence of a T-a.s.distinguishing adaptive strategy for A is* EXPTIME*-complete.*

References

[1] R. Alur and D. Dill, A theory of timed automata. *Theoretical Computer Science* **126**, pp. 183-235, 1994.

[2] E. Brinksma, A theory for the derivation of tests. In *Proc. IFIP WG6.1 8th Intl. Symp. on Protocol Specification, Testing, and Verification*, pp. 63-74, 1988.

[3] E. Brinksma, J. Tretmans, and L. Verhaard, A framework for test selection. In *Proc. IFIP WG6.1 11th Intl. Symp. on Protocol Specification, Testing, and Verification*, pp. 233-48, 1991.

[4] W.Y.L. Chan, S.T. Vuong, and M.R. Ito, An improved protocol test generation procedure based on UIOs. In *Proc. SIGCOM*, pp. 283-294, 1989.

[5] T. S. Chow, Testing software design modeled by finite-state machines. *IEEE Trans. on Software Engineering* **SE-4** (3), pp. 178-87, 1978.

[6] A. Condon, *Computational models of games.* MIT Press, 1989.

[7] A. Condon and R.E. Ladner, Probabilistic game automata. *J. Computer and System Sciences* **36**, pp. 452-489, 1988.

[8] C. Courcoubetis and M. Yannakakis, Markov decision processes and regular events. In *Proc. 17th ICALP*, 1990.

[9] A. T. Dahbura, K. Sabnani, and M. U. Uyar, Formal methods for generating protocol conformance test sequences. *Proc. IEEE* **78** (8), August 1990.

[10] C. Derman, *Finite state Markov decision processes.* Academic Press, 1972.

[11] A. D. Friedman and P. R. Menon, *Fault Detection in Digital Circuits.* Prentice-Hall, 1971.

[12] S. Fujiwara, G. v. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi, Test selection based on finite state models. *IEEE Trans. on Software Eng.* **17**, pp. 591-603, 1991.

[13] A. Gill, State identification experiments in finite automata. *Information and Control*, pp. 132-154, 1961.

[14] G. J. Holzmann, *Design and Validation of Protocols.* Prentice-Hall, 1990.

[15] P.C. Kanelakis, S. A. Smolka, CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 1993.

[16] Z. Kohavi, *Switching and Finite Automata Theory.* 2nd Ed., McGraw-Hill, 1978.

[17] R.P. Kurshan, *Computer-Aided Verification of Coordinating Processes: The Automata-Theoretic Approach.* Princeton University Press, 1994.

[18] D. Lee, K. Sabnani, D. M. Kristol, S. Paul, and M. U. Uyar, Conformance testing of protocols specified as communicating FSM's. In *Proc. of IEEE INFOCOM*, pp. 115-127, 1993.

[19] D. Lee and M. Yannakakis, Testing finite state machines: state identification and verification. *IEEE Trans. on Computers* **43** (3), 1994.

[20] D. Lee and M. Yannakakis, Principles and methods of testing finite state machines - a survey, in preparation, invited to *Proc. IEEE.*

[21] S. Low, Probabilistic conformance testing of protocols with unobservable transitions. In *Proc. Intl. Conf. on Network Protocols*, pp. 368-375, 1993.

[22] M. Mihail and C. H. Papadimitriou, On the random walk method for protocol testing. In *Proc. 6th Intl. Conf. on Computer-Aided Verification*, pp. 132-141, 1994.

[23] R. Milner, *Communication and Concurrency.* Prentice Hall, 1989.

[24] E. F. Moore, Gedanken-experiments on sequential machines. In *Automata Studies*, Annals of Mathematics Studies 34, Princeton University Press, pp. 129-53, 1956.

[25] C. Papadimitriou, Games against nature. In *Proc. 20th FOCS*, pp. 446-450, 1983.

[26] C. Papadimitriou and J. Tsitsiklis, The complexity of Markov decision processes. *Mathematics of Operations Research* **12** (3), pp. 441-450, 1987.

[27] J. Reif, The complexity of two-player games with incomplete information. *J. Computer and System Sciences* **29**, pp. 274-301, 1984.

[28] K. K. Sabnani and A. T. Dahbura, A protocol test generation procedure. *Computer Networks and ISDN Systems* **15** (4), pp. 285-97, 1988.

[29] D. P. Sidhu and T.-K. Leung, Formal methods for protocol testing: a detailed study. *IEEE Trans. Soft. Eng.* **15** (4), pp. 413-26, 1989.

[30] M. Vardi, Automatic verification of probabilistic concurrent finite state programs. In *Proc. 26th STOC*, 1985.

[31] C. West, Protocol validation by random state exploration. In *Proc. IFIP WG6.1 6th Intl. Symp. on Protocol Specification, Testing, and Verification*, 1986.

[32] M. Yannakakis and D. Lee, Testing finite state machines: fault detection, to appear in *J. of Computer and System Sciences*, 1994; preliminary version in STOC 91.

[33] W. Yi and K. G. Larsen, Testing probabilistic and nondeterministic processes. In *Proc. IFIP WG6.1 12th Intl. Symp. on Protocol Specification, Testing, and Verification*, pp. 47-62, 1992.