Chapter 1

BOUNDED ERROR PROBABILISTIC FINITE STATE AUTOMATA

Anne Condon
The Department of Computer Science
University of British Columbia
Vancouver, B.C. Canada
condon@cs.ubc.ca

1. INTRODUCTION

What power does randomness confer to computing devices? In this article, we focus on this question in what is perhaps its simplest form, namely when the computing device is a finite state automaton.

Some of the oldest studies of probabilistic computations, dating as far back as the 40's, implicitly concern probabilistic finite state devices. A beautiful theory of probabilistic finite state automata was developed starting in the early 60's [Rabin, 1963, Paz, 1971]. This work primarily concerned automata with 1-way heads on the input tape, where an input w is considered to be accepted if the probability of reaching the accept state from the initial configuration is greater than some threshold, say $1/2$. The class of languages thus accepted is known as the stochastic languages.

A pfa for a stochastic language may err by rejecting inputs in the language with probability that approaches $1/2$ as the input size increases. It is natural to consider the language-recognition power of pfa's whose probability of error is bounded away from $1/2$. The theory of so-called *bounded error* (or isolated threshold) pfa's was initiated by Rabin [Rabin, 1963], but much progress in understanding these pfa's has been made more recently, and so we focus on bounded error pfa's in this survey. The study of bounded error pfa's is motivated by questions such as the following: if one wants to recognize patterns with small degree of error,

can it be done with fewer states then when recognizing a pattern exactly? What kinds of patterns can be recognized when a small probability of error is allowed, but that cannot be recognized exactly?

The purpose of this survey is to describe a coherent set of results on bounded error pfa's. In Section 2. we describe pfa's and define associated language classes. In Section 3. we present a result of Freivalds that pfa's with a 2-way input head can accept nonregular languages. In particular, the language $\{a^n b^n \mid n \geq 0\}$ is accepted with bounded error by a 2pfa, although the 2pfa has worst case expected running time that is exponential in the length of its input. We also describe work by Ravikumar [Ravikumar, 1992] that builds on Freivalds work to show two classes of languages that are recognized with bounded error by 2pfa's.

Freivald's work raises the question: if the expected running time of a pfa is limited to polynomial time, can nonregular languages still be recognized? In Section 4. we show that the answer is no. Rabin [Rabin, 1963] showed pfa's with a 1-way input head accept exactly the regular languages. Dwork and Stockmeyer [Dwork and Stockmeyer, 1992] and independently Kaneps and Freivalds [Kaneps and Freivalds, 1990] showed that 2pfa's that run in subexponential expected time only accept the regular languages. The techniques used to prove this result include fundamental results on Markov chains that are interesting in their own right [Dwork and Stockmeyer, 1990, Greenberg and Weiss, 1986, Leighton and Rivest, 1983]. We conclude this section with a brief summary of results on succinctness of 2pfa's in terms of number of states needed to recognize languages compared with deterministic or nondeterministic finite automata.

Section 5. concerns undecidability results for pfa's. Freivalds showed that the problem of determining whether a bounded error 2pfa with known acceptance threshold accepts the empty language is undecidable. Freivalds' work was generalized to prove a related undecidability result for 1pfa's [Condon and Lipton, 1989]. At the end of this section, we describe how these undecidability results were applied to show that problems in probabilistic planning and Markov decision processes are undecidable [Madani et al., 1999].

2. PFA MODEL

We assume that the reader is familiar with the definitions of deterministic and nondeterministic finite state automata (dfa's and nfa's). We denote the class of languages accepted by such automata, with either 1-way or 2-way input heads, as **Regular**.

A 2-way head probabilistic finite state automaton (pfa or 2pfa), like a 2-way dfa or nfa, has a finite set of states including an initial state and zero or more accept states, a finite input alphabet, and a transition function. Also associated with a pfa are a left endmarker, #, and a right endmarker, $, which are used to mark the left and right ends, respectively, of the input and are not in the input alphabet. The transition function maps (state, symbol) pairs to a (real-valued) probability distribution over (state, head direction) pairs, where the symbol may be from the input alphabet or an endmarker and the head direction determines whether the head moves one position left (L) on the input tape, one position right (R), or remains in the same position (N). Transitions are defined so that the head never moves left from the left endmarker or right from the right endmarker. For example, the pfa of Figure 1 has input alphabet $\{a\}$, left endmarker #, and right endmarker $. On input symbol a, state 2 from state 1 is reached with probability 1/5 and state 1 is reached with probability 4/5; in both cases the head moves right.

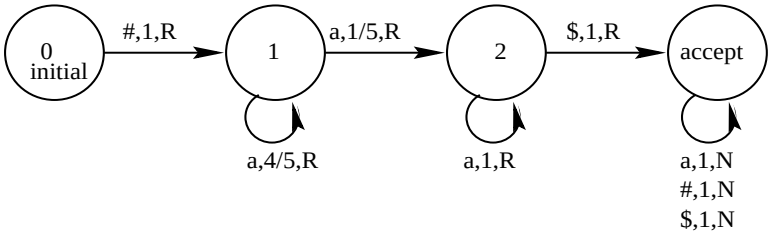

Figure 1.1 Example pfa with initial state 0. An edge labeled (σ, p, X) denotes a transition with probability p in which σ is under the tape head. When $X = R$, the head moves right and when $X = N$ the head does not move; since on no transition does the head move left, this is a 1pfa. The pfa has an additional reject state which is not shown, and all transitions not shown are to this reject state. For example, from state 0 on either $ or a, the reject state is reached with probability 1.

Throughout, we will use the following conventions. We assume that a pfa has a unique accepting state which is halting, that is, all transitions from the accept state lead back to this state without moving the head. Similarly, there is a unique halting rejecting state. A 1pfa is a pfa with no transitions that cause the tape head to move left. Whenever we refer to an input w of a pfa, we mean that the pfa has $\#w\$$ on its tape.

We say that input w is *accepted* by the pfa if the probability that the accept state of the pfa is eventually reached, starting with the head in the initial state on the left endmarker, is greater than 1/2. The language accepted by pfa A is denoted by $L(A)$. It is not hard to see that the pfa of Figure 1 accepts the language $\{a^k \mid k \geq 4\}$. The class

of languages accepted by pfa's is known as the stochastic languages, and we denote it by **Stochastic**. Replacing the acceptance threshold by any real number in the range $(0, 1)$ in the definition of string acceptance does not change the class **Stochastic** [Paz, 1971]. Rabin [Rabin, 1963] showed that **Stochastic** contains nonregular languages. For examples of other nonregular stochastic languages, including unary languages, see [Dwork and Stockmeyer, 1990, Paz, 1971, Salomaa and Soittola, 1978]. The class Stochastic was initially defined for 1pfa's only, but Kaneps [Kaneps, 1989], building on work of Turakainen [Turakainen, 1969], showed that 2pfa's accept exactly those languages accepted by 1pfa's. Macarie [Macarie, 1998] showed that the question of whether a given string x is accepted with probability $> 1/2$ by a given 1pfa P which has rational transition probabilities can be decided in deterministic logarithmic space.

Consider a pfa A with the property that for all inputs in $L(A)$, the probability of reaching the accept state is in fact greater than $1/2 + \gamma$ for some constant $\gamma > 0$ (independent of the input length), while the probability that the accept state is reached on inputs not in $L(A)$ remains at most $1/2$. We say that such a pfa accepts its language with bounded error, and we refer to $1/2 + \gamma$ as the *acceptance threshold* of the pfa. We denote the languages accepted by bounded error 1pfa's by 1PFA and the languages accepted by bounded error 2pfa's by 2PFA. The class of languages accepted by pfa's which are restricted to run in polynomial expected time is denoted by 2PFA-polytime.

Finally, we say that a pfa is *coin-flipping* if all transition probabilities are in the set $\{0, 1/2, 1\}$. We note that for any pfa with rational transition probabilities there is an equivalent coin-flipping pfa, in the sense that acceptance probabilities on all strings are the same. Figure 2 shows a coin-flipping pfa that is equivalent to that of Figure 1. All of the 2pfa constructions described in Sections 3. and 5. are coin-flipping.

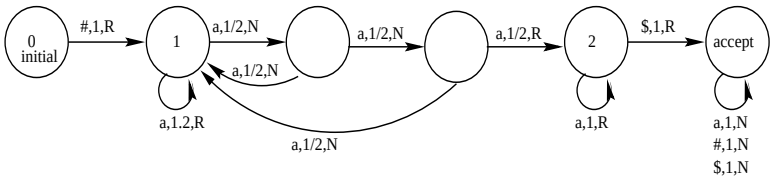

Figure 1.2 Coin-flipping pfa that is equivalent to that in Figure 1.

3. 2PFA CONTAINS NON-REGULAR LANGUAGES

Before describing results on the limitations of pfa's, it is useful to first see a pfa that does something surprising. We describe here a coin-flipping 2pfa of Freivalds [Freivalds, 1981] that accepts the language $\{a^n b^n \mid n > 0\}$ with bounded error.

Freivalds' 2pfa does the following. First, (deterministically) check that the input is of the form $a^n b^m$ for some $n > 0, m > 0$ and that $n = m \bmod (k+1)$ for some constant k and if not, reject. Here, k controls the error probability of the 2pfa, as we discuss later.

Then, while scanning the input repeatedly, do the following. On each scan, flip a fair coin for each a and b; call these a-flips and b-flips, respectively. A scan is said to be a success for the a's if all a-flips are heads but at least one b-flip is a tail, and is a success for the b's if all b-flips are heads but at least one a-flip is a tail. If there are L successes for the a's before any success for the b's or vice versa, then reject, else accept. Here again L is a constant that controls the error probability, as discussed later.

If $n = m$ then on each scan the probability of success for the a's is the same as the probability of success for the b's. However, if $n > m+k$, then the probability of success for the b's is at least 2^k times the probability of success for the a's. A similar statement holds if $m > n + k$.

Thus, if $n = m$, the chance of L successes for the a's before any success for the b's is at most $(1/2)^L$ (and vice versa), and so the 2pfa accepts a string in the language with probability at least $1 - (1/2)^{L-1}$. For example, to obtain a machine with acceptance threshold $3/4$, it is sufficient to choose $L = 3$.

But if $n > m + k$, then in a single trial,

$$\frac{\text{Prob[success for the } a\text{'s]}}{\text{Prob[success for the } a\text{'s or } b\text{'s]}} \leq \frac{1}{2^k + 1}.$$

Therefore, the probability that the first L successful trials are all successes for the b's, and thus that the 2pfa rejects, is at least $(1 - 1/(2^k + 1))^L$. By choosing k sufficiently large (depending on L), this constant can be made as close to 1 as desired. For example, if $L = 3$ and $k = 2$ then $(1 - 1/(2^k + 1))^L = .512$ and so A rejects inputs not in L with probability at least $1/2$.

B. Ravikumar [Ravikumar, 1992] extended Freivald's construction to show that all bounded semilinear languages and all languages recognizable by deterministic blind counter machines are contained in 2PFA. These language include non-context free languages such as $\{a^n b^n c^n \mid n \geq 0\}$.

4. LIMITS ON THE POWER OF TIME-RESTRICTED PFA'S

The expected running time of Freivalds' 2pfa is exponential in the input size, since the probability that a single scan is a success for the a's or b's is exponentially small in the length of the input. If one is interested in efficient finite state pattern recognizers, it is necessary to restrict attention to pfa's that run in polynomial expected time. As it turns out, such pfa's accept only the regular languages. The earliest result along these lines is due to Rabin [Rabin, 1963] and is for 1pfa's. We describe this first, and then return to the class 2PFA-polytime.

4.1 1PFA = REGULAR

Rabin's proof uses the following fundamental property of the regular languages. Define two strings x and x' to be Myhill-Nerode-equivalent with respect to a given language L if for all strings y, xy is in L if and only if $x'y$ is in L. Then L is regular if and only if the number of Myhill-Nerode equivalence classes with respect to L is finite (see the text by Hopcroft and Ullman [Hopcroft and Ullman, 1979]). The proof of one direction of the Myhill-Nerode theorem is easy: if A is a dfa for L, define x and x' to be A-equivalent if x and x' lead to the same state of A from the initial state. A-equivalence implies Myhill-Nerode equivalence, and hence the number of Myhill-Nerode equivalence classes is at most the number of states of A.

We'd like to use the behavior of a 1pfa P on two strings x and x' to define some notion of P-equivalence between x and x', such that if x and x' are P-equivalent, then they are Myhill-Nerode-equivalent. A natural approach is to base a notion of P-equivalence on the probability distribution of states reached when the head moves to the right off x or x', when starting in the initial configuration. For simplicity, assume without loss of generality that the 1pfa P does not enter the accept or reject state unless the head reaches the right end-marker. However P may loop forever without reaching either the accept or reject state, perhaps by cycling among several states without ever moving its head. Let the non-halting states of 1pfa P be numbered $1, \ldots, c$ and let $p(x)$ be a c-vector whose ith entry is the probability of being in state i when the tape head moves off x. Intuitively, if the vectors $p(x)$ and $p(x')$ are close then for any string y, the probability that xy is accepted is close to the probability that $x'y$ is accepted.

More precisely, suppose that $||p(x) - p(x')|| \leq \epsilon$, where $\epsilon > 0$ and $||x||$ denotes the max of the absolute values of the entries of the vector x. Let $q(y)$ be the column c-vector with ith entry equal to the probability that

the accept state of P is reached from a starting configuration in which P is in state i and the head of P is on the first (leftmost) symbol of $y\$$. Then, the inner product $p(x)q(y)$ is the probability that P accepts string xy and $p(x')q(y)$ is the probability that P accepts $x'y$. Therefore,

$$\begin{aligned} &|\text{Prob}[P \text{ accepts } xy] - \text{Prob}[P \text{ accepts } x'y]| \\ &\quad \leq ||p(x) - p(x')q(y)|| \leq c\epsilon. \end{aligned} \tag{1.1}$$

Let $\epsilon > 0$ be sufficiently small so that the acceptance threshold of P is $> 1/2 + c\epsilon$. Then if P accepts xy, Prob[P accepts xy] $> 1/2 + c\epsilon$. Combining this with inequality (1.1), we have that Prob[P accepts $x'y$] $> 1/2$. Hence, P must also accept $x'y$.

Now, partition the space of all vectors in $[0,1]^c$ into a constant number of cells of dimension at most ϵ, so that for any pair of vectors v, w in the same cell, $||v - w|| \leq \epsilon$. Define x and x' to be P-equivalent if $p(x)$ and $p(x')$ are both in the same cell. The argument of the previous paragraph shows that if x and x' are P-equivalent (with ϵ chosen appropriately depending on the acceptance threshold of P), then for any string y, xy is in L if and only if $x'y$ is in L. Thus, the number of equivalence classes of the language accepted by P is bounded by the number of cells in the partition of $[0,1]^c$, and thus is finite. By the Myhill-Nerode theorem, the language accepted by P is therefore regular.

4.2 2PFA-POLYTIME = REGULAR

Dwork and Stockmeyer [Dwork and Stockmeyer, 1992] and independently Kaneps and Freivalds [Kaneps and Freivalds, 1990] generalized Rabin's result to show that only the regular languages are recognized by 2pfa's that are restricted to have expected running time that is subexponential. In particular, 2PFA-polytime = Regular. We describe their techniques in this section. These build on earlier work of Greenberg and Weiss [Greenberg and Weiss, 1986], who showed that exponential time is needed to recognize $\{a^n b^n\}$.

Let P be a 2pfa that halts in polynomial expected time; in particular P halts with probability 1, and assume again for simplicity that the accept or reject states are only entered when the head is on the right endmarker. As in Rabin's proof, a key idea is to model the behavior of P on x in such a way that if the models for x, x' are close (according to some measure), then xy is accepted if and only if $x'y$ is. In the case of a 1pfa, the behavior on x is modeled by the vector $p(x)$, but this is inadequate for 2pfa's, since, during a computation of a 2pfa on input xy, the tape head may repeatedly cross left from y back onto x. Thus, to fully capture the behavior of P on x, in addition to including the probabilities in $p(x)$, we need to include, for each pair of nonhalting

states q and q', the probability that, starting from q with the head on the right end of $\#x$, P is in state q' when its head moves off the right end of $\#x$ for the first time. Note that these probabilities are independent of whatever string is to the right of x.

It is convenient to represent these probabilities as weights on edges of a graph, which we denote by $M[x]$. $M[x]$ has $2c+1$ nodes and directed weighted edges. Of these nodes, c are of the form $(q,1), 1 \leq q \leq c$, representing the configuration in which P's head is on the right end of x in state q. Also, c are of the form $(q',2)$, representing the configuration in which the state of P is q' and the head is on the first symbol of the string that is to the right of $\#x$. The weight of the edge $(q,1) \rightarrow (q',2)$ is the probability that, starting from configuration $(q,1)$, the first configuration reached, among those configurations that are represented as nodes of $M[x]$ of the form $(*,2)$, is $(q',2)$. The remaining node in the graph is Initial, denoting the initial configuration of P on $\#x$. The weight of the edge Initial $\rightarrow (q',2)$ denotes the probability that the first configuration reached from Initial, among those configurations that are represented as nodes of $M[x]$ of the form $(*,2)$, is $(q',2)$. Similarly, we can model the behavior of P on $y\$$ as a graph $M[y]$ with $2c+2$ nodes. Of these nodes, c are of the form $(q,1)$, representing the configuration in which the head is on the symbol just to the left of $y\$$, c are of the form $(q,2)$, representing the configuration in which the head is on the leftmost symbol of $y\$$, and the weight of edge $(q',2) \rightarrow (q,1)$ is the probability that, starting from $(q',2)$, the first configuration reached, among those represented as nodes of $M[y]$ that are not of the form $(*,2)$, is $(q,1)$. The two additional nodes are Accept and Reject, where edge $(q',2) \rightarrow$ Accept (Reject) has weight equal to the probability of reaching the accept (reject) state from configuration $(q',2)$ before reaching any configuration of type $(*,1)$. Also, there is an edge of weight 1 from Accept to itself and from Reject to itself.

Let $M[x,y]$ denote the graph with $2c+3$ nodes obtained from the "union" of the graphs $M[x]$ and $M[y]$. That is, the set of nodes (resp. edges) of $M[x,y]$ is the union of the set of nodes (resp. edges) of $M[x]$ and $M[y]$. Note that the sum of the weights of edges from any node of $M[x,y]$ is 1 since P halts with probability 1. The probability of eventually reaching Accept from Initial in $M[x,y]$, while transitioning between states according to the edge probabilities, equals the probability that the accept state of P is reached from the initial configuration on input xy.

Now, consider a sequence of random variables $X_0, X_1, \ldots$ over the nodes of $M[x,y]$, where $X_0 =$ Initial and for $i \geq 1$, Prob$[X_i = N \mid X_{i-1} = N']$ is the weight of edge $N \rightarrow N'$. This sequence is a (discrete, time-

inhomogeneous) Markov chain; we also use $M[x, y]$ to refer to this Markov chain. Associated with a Markov chain M is a square transition probability matrix $[p_{ij}]$ with dimension equal to the number of states of M. Entry p_{ij} is the probability of reaching state j from state i in one step. To define equivalence between strings x and x', we need to know how small perturbations in the transition probabilities of a Markov chain affect the probability of eventually reaching Accept from Initial.

Let's look at two examples, to get some intuition on this. In our first example, let M have states 1, 2, and 3, with 1 being the initial state and states 2 and 3 being absorbing, that is, the probability of reaching 2 from 2 is 1 and similarly for 3. Let $a(M)$ and $a(M')$ denote the probability that state 2 is eventually reached, starting from state 1. Let p_2, p_3 be the probabilities of going from state 1 to state 2 and from state 1 to state 3, respectively. The probability of looping at state 1 is thus $1 - p_2 - p_3$ and $a(M)$ is $p_2/(p_2 + p_3)$. Let M' be the same as M, except that p_i is replaced by p'_i everywhere. Suppose that $p_2 = p_3$, $p'_2 = 2p_2$, and $p'_3 = p_3$. Then $a(M) = 1/2$ and $a(M') = 2/3$. By choosing p_2 small enough, the quantity $|p'_2 - p_2| = 2p_2 - p_2 = p_2$ can be made arbitrarily small, yet the difference $a(M) - a(M')$ remains equal to $3/2 - 1/2$.

This example suggests that the *ratio*, rather than the difference, of corresponding transition probabilities needs to be close to 1 in order for two Markov chains M and M' to have similar absorbtion probabilities at their (common) absorbing states.

Let $\beta \geq 1$. We say that two real numbers p and p' are β-close if either (i) $p = p' = 0$, or (ii) $p > 0, p' > 0$, and $\beta^{-1} \leq p/p' \leq \beta$. Let M and M' be two Markov chains with associated transition probability matrices $[p_{ij}]$ and $[p'_{ij}]$. We say that M and M' are β-close if for all i, j, p_{ij} and p'_{ij} are β-close.

Let M and M' be β-close Markov chains over the same state space and suppose that $a(M)$ and $a(M')$ are the probabilities of reaching a (common) absorbing state. We would like an upper bound on the ratio $a(M)/a(M')$, as a function of β and m, where m is the number of states of M and M'. The following proposition provides such a bound. It was first proved by Greenberg and Weiss [Greenberg and Weiss, 1986] and follows from the Markov Chain Tree Theorem of Leighton and Rivest [Leighton and Rivest, 1983].

Proposition 1 *Let M and M' be β-close. Then $a(M)$ and $a(M')$ are β^{2m}-close.*

For example, suppose that M has m states, of which $m - 1$ and m are absorbing, and 1 is the initial state. Let $a(M)$ be the probability of eventually reaching $m - 1$ from 1. Let the probability of going from

state i to $i+1, 1 \leq i \leq m-2$, be p, where $0 < p < 1/2$. Let the probability of going from state i to state m, $1 \leq i \leq m-2$, be $1-p$. Thus, $a(M) = p^{m-2}$. If M' is the same as M except that p is replaced everywhere by $2p$, then $a(M')$ is $(2p)^{m-2}$. In this example, M and M' have m states and are β-close for $\beta = 2$, and the ratio $a(M)/a(M')$ is β^m.

Proposition 1 does not seem too promising: for M and M' to be close, the difference between logs of (non-zero) transition probabilities needs to be small. But since non-zero transition probabilities lie in the range (0,1], the logs of these probabilities lie in the range $(-\infty, 1]$. Thus, it is not possible to partition the space of Markov chains into a finite number of classes such that all of those in the same class are β-close, where β is a small constant.

One ray of hope is that if x has length at most n, then the non-zero transition probabilities of $M[x]$ cannot be less than 2^{-cn-1}. (Recall that $M[x]$ is that part of the graph $M[x,y]$ with $2c+1$ states determined by x.) To see this, note that, for example, if configuration $(q', 1)$ is reachable from configuration $(q, 2)$ on some execution of the pfa P on input $\#x$, then there is an execution for which $(q', 1)$ is reachable from $(q, 2)$ in at most $cn-1$ steps. Since P is a coin-flipping pfa, the probability of this execution is at least 2^{-cn-1}.

If $|x| \leq n$, then the non-zero transition probabilities of $M[x]$ lie in the range $[2^{-cn-1}, 1]$ and so the logs (to base 2) of these probabilities lie in the range $[-cn-1, 0]$. As Theorem 1 below makes precise, it is possible to partition the graphs $M[x]$, and thus the strings x of length $\leq n$, into a number of equivalence classes that is bounded by a polynomial function of n, such that if x and x' are in the same equivalence class then xy is accepted by P if and only if $x'y$ is accepted by P. There is some hope that this might be a useful step towards the goal of showing that P accepts only regular languages, if it can be combined with a quantitative version of the Myhill-Nerode theorem. Roughly, the latter would provide a lower bound on the number of words which must be distinguished by any recognizer of the strings of length at most n in a regular language.

Theorem 1 *Let P be a 2pfa that accepts its language with bounded error. There is a partition of the strings of length at most n into a number of classes that is bounded by a polynomial in n, such that if x and x' are in the same class then for all y, xy is accepted by P if and only if $x'y$ is.*

Since $M[x,y]$ models a 2pfa that always halt with probability 1, the following sketch of the proof of Theorem 1 applies only to such 2pfa's, but the techniques can easily be extended to apply to 2pfa's that do not

necessarily halt with probability 1. Partition the interval $[-cn-1, 1]$ into subintervals of length at most ϵ, for some constant $\epsilon < 0$. Define x and x' to be ϵ-close if for each pair of corresponding edge weights p and p' in $M[x]$ and $M[x']$, either $p = p' = 0$ or $\log_2 p$ and $\log_2 p'$ are both in the same subinterval. Note that ϵ-closeness is an equivalence relation, and the number of equivalence classes is at most $(\lceil (cn+1)/\epsilon \rceil + 1)^{(2c+1)^2}$.

Let $a(xy)$ and $a'(xy)$ be the probability of reaching state Accept from Initial in $M[x, y]$ and $M[x', y]$, respectively. From the fact that x and x' are ϵ-close, it follows that $M[x, y]$ and $M[x', y]$ are 2^{ϵ}-close. By Proposition 1, it follows that $a(xy)$ and $a(x'y)$ are $2^{2(2c+3)\epsilon}$-close. Therefore,

$$\frac{a(xy)}{a(x'y)} \geq 2^{-2(2c+3)\epsilon}.$$

If $x'y$ is accepted then $a(x'y) > 1/2 + \delta$, for some constant δ. Hence,

$$a(xy) \geq (1/2 + \delta)2^{-2(2c+3)\epsilon}.$$

By choosing ϵ sufficiently small (depending on the number c of states of P and on δ), the right hand side of the above inequality can be made greater than $1/2$, and so xy must be accepted by P.

Corollary 1 *The language* $\{w\&w \mid w \text{ is in } \{a, b\}^*\}$ *is not in 2PFA.*

To see this, suppose that P is a 2pfa that accepts this language with bounded error. Let $p(n)$ be the polynomial of Theorem 1. Let n be such that $2^n > p(n)$. Consider the 2^n distinct strings of length n. All of these must be in different classes (see statement of Theorem 1), which is impossible, contradiction.

However, we cannot hope to strengthen it to show that *no* nonregular language is in 2PFA, since Freivalds has shown that $\{a^n b^n \mid n \geq 0\}$ is in 2PFA. In order to prove something about 2PFA-polytime, we need to strengthen Theorem 1 in the case that P halts in polynomial expected time. The following result does this.

Theorem 2 *Let P be a 2pfa that accepts its language with bounded error. Let the expected running time of P be bounded by a polynomial $t(n)$. Then there is a partition of the strings of length at most n into a number of classes that is bounded by a polynomial in* $\log n$*, such that if x and x' are in the same class then for all y such that $|xy| \leq n, |xy'| \leq n$, xy is accepted by P if and only if $x'y$ is.*

Roughly, the proof of Theorem 2 is as follows. Let (s, s') be a transition in $M[x, y]$ with probability p where $p \leq t(n)^{-2}$. Since P halts in

expected time $t(n)$, it is unlikely that $M[x, y]$ ever transitions from s to s' before halting. Let $M'[x]$ be the graph obtained by changing transition probabilities of $M[x]$ that are $< t(n)^{-2}$ to 0. Since the edge weights of $M'[x]$ are in the range $[t(n)^{-2}, 1]$, it is possible to partition the strings x of length at most n into a number of classes that is polylogarithmic in n, such that if x and x' are in the same class then for all y with $|xy| \leq n$ and $|x'y| \leq n$, xy is accepted by P if and only if $x'y$ is.

We can use Theorem 2 to show that 2PFA-polytime accepts only regular languages if we can show that for some function $g(n)$ that grows faster than any polylogarithmic function of n, for all nonregular languages L, for infinitely many n, there is a set of strings of size at least $g(n)$, each of length at most n, such that each pair of strings x, x' in the set is *n-dissimilar* with respect to L. By this, we mean there is some y for which $|xy| \leq n, |xy'| \leq n$, and xy is in L if and only if $x'y$ is not in L.

Towards this end, we define a more quantitative measure of the "nonregularity" of a language L than the notion of inequivalence defined by Myhill and Nerode. Let $N_L(n)$ be the maximum k such that there exist k distinct words which are pairwise n-dissimilar with respect to L. It is not hard to show that $N_L(n) = O(1)$ if and only if L is regular. The following theorem states that if L is non-regular, then $N_L(n)$ is bounded below by a linear function of n for infinitely many n. For related results, see the work of Shallit et al. [Glaister and Shallit, 1998, Pomerance et al., 1997, Shallit and Breitbart, 1996].

Theorem 3 *If L is non-regular then for infinitely many n, $N_L(n) \geq n/2 + 1$.*

The proof of Theorem 3 is based on an old result of Moore [Moore, 1956]. We say that a dfa D recognizes the initial n-fragment of L if $L(M) \cap \Sigma_{\leq n} = L \cap \Sigma_{\leq n}$. Let $\phi_L(n)$ be the size of a minimal 1dfa that recognizes the initial n-fragment of L. Kaneps and Freivalds [Kaneps and Freivalds, 1990] showed that $N_L(n) = \phi_L(n)$.

Karp [Karp, 1967] showed that if L is non-regular, then for infinitely many n, $\phi_L(n) > n/2 + 1$. To see this, it is sufficient to show that, for any positive integer r, there is $n > r$ such that $\phi_L(n) > n/2+1$. Given r, let n $(n > r)$ be the unique integer such that $\phi_L(r) = \phi_L(n-1) < \phi_L(n)$.

Such an n exists, since $\phi_L()$ is monotone and unbounded. Let minimal finite-state machines M and N be chosen so that M recognizes the initial $(n-1)$-fragment of L and N recognizes the initial n-fragment of L. Note that M and N cannot be identical, since M does not have enough states needed to recognize the initial n-fragment of L, and so they disagree on a string z of length n. Moore [Moore, 1956] proved the following result,

which provides an upper bound on the length of a string on which M and N disagree as a function of the number of their states.

Proposition 2 *Let M and N be dfa's with j and k states, respectively, such that $L(M) \neq L(N)$. Then there is a string z of length at most $j + k - 2$ such that z is in $L(M)$ if and only if z is not in $L(N)$.*

It follows that there is a string z length at most $\phi_L(n-1) + \phi_L(n) - 2$ such that z is in $L(M)$ if and only if z is not in $L(N)$. Hence, $\phi_L(n-1) + \phi_L(n) - 2 \geq n$. Since we also have that $\phi_L(n-1) \leq \phi_L(n) - 1$, we obtain

$$(\phi_L(n) - 1) + \phi_L(n) > n + 1, \text{ or } \phi_L(n) > n/2 + 1,$$

as required.

The fact that 2PFA-polytime = Regular follows immediately from Theorems 2 and 3.

4.3 RELATED RESULTS

Let 2PFA(rat) be the class of languages recognized with bounded error by 2pfa's that have rational transition probabilities (or equivalently, are coin-flipping 2pfa's). Wang [Wang, 1992] showed that 2PFA(rat) is contained in the class of deterministic, context-sensitive languages; we don't include details of his proof here but it is based on the techniques of the Markov Chain Tree Theorem.

Finally, a note on succinctness of polynomial time bounded 2pfa's. Dwork and Stockmeyer [Dwork and Stockmeyer, 1990] extended the above techniques to address the question of whether bounded error 2pfa's that run in polynomial expected time can recognize languages with fewer states than dfa's or nfa's. Roughly, they showed that if P is a bounded error 2pfa with c states that runs in polynomial expected time, then there is a 1dfa for $L(P)$ that has $c^{O(c^2)}$ states, where the constant hidden in the big-O notation depends on the acceptance threshold of P and the degree of the polynomial bound on P's running time. They also showed that the c^2 in the exponent cannot be replaced by a function of c which grows more slowly than $\sqrt{c/\log c}$, even if 1dfa is replaced by 2nfa.

5. UNDECIDABILITY RESULTS FOR BOUNDED ERROR PFA'S AND THEIR CONSEQUENCES

We now turn to the following question: given a pfa P, does it accept a regular language? In particular, does P accept the empty language?

In this section, we describe some results on this question, known as the emptiness problem for finite state automata. Later in this section, we describe applications of these results to show undecidability of probablistic planning problems and problems on partially observable and unobservable Markov decision processes.

For dfa's or nfa's there is an efficient algorithm for the emptiness problem, since a dfa or nfa accepts the empty language if and only if there is no path from the initial state to an accept state in the transition graph for the automaton.

In the case of a 1pfa P that accepts its language with bounded error and the acceptance threshold is known, then the question is decidable: From Section 4.1, we know that there is a 1nfa D that accepts the same language as P and has a number of states at most exponential in the size of P, where the exponential bound can be calculated explicitly from the acceptance threshold γ and the number of states c of P. Denote this bound by $B(\gamma, c)$. Thus, a naive procedure to solve the emptiness problem in this case is to enumerate all nfa's D of size at most $B(\gamma, c)$; for each, determine whether $L(D) = L(P)$ and, if so, whether D accepts the empty language. Note that if $L(D) \neq L(P)$ then Moore's result (Proposition 2 above) provides a bound on the length of a witness to this fact. Thus, by enumerating all potential witnesses and checking whether each is accepted by P and by D, it is possible to determine whether $L(D) = L(P)$.

5.1 UNDECIDABILITY OF THE EMPTINESS PROBLEM FOR 2PFA'S

Since 2pfa's accept nonregular languages, it is perhaps not surprising that the emptiness problem for 2pfa's is undecidable. This result was proved by Freivalds [Freivalds, 1981].

Theorem 4 *The following problem is undecidable. Given a natural number $k \geq 2$ and a 2pfa P recognizing a language L with acceptance threshold $1/2 + 1/k$, determine whether the language L is empty.*

The proof of Theorem 4 is via a reduction from the halting problem for Turing machines on empty input. A first try at a reduction would be to use the description of a Turing machine M to produce a 2pfa $A = A(M)$ that can recognize (on its input tape) halting computations of M on the empty string. This 2pfa may err, as long as the error is bounded away from $1/2$. Assuming that M has a single, 1-way infinite tape, a standard representation of a computation of M is a string of the form $C_0 \& C_1 \& \ldots \& C_t$, where each C_i describes a configuration of M. If

at the ith move of M on empty input the contents of M's tape is uv, the head is placed on the rightmost symbol of u, and the state of M is s, then C_i is the string usv.

Note that C_{i+1} differs from C_i in a constant number of places around the tape head, and a dfa can check that these differences between C_{i+1} and C_i correspond to a valid transition of M. A dfa can also check that C_0 represents the initial configuration of M on the empty string and that C_t is a halting configuration. The remaining task of the 2pfa is easily reducible to repeatedly recognizing strings of the form $w\&w$. Unfortunately, from Corollary 1, a 2pfa cannot recognize strings of the form $w\&w$. Unless, of course, w is a string over a unary alphabet.

Are Turing machines with a unary worktape alphabet equivalent in power to general Turing machines? It turns out that if the Turing machine has two worktapes, the answer is yes [Hopcroft and Ullman, 1979]. Specifically, the emptiness problem for the following model, called the 2-counter machine, is undecidable. A 2-counter machine is a Turing machine with two one-way infinite read-only worktapes (in addition to a read-only input tape). The leftmost symbol on both worktapes is \$ and all remaining symbols are blank. In effect, the 2-counter machine can use the worktapes as counters, simulating an increment or decrement by moving the head right or left, respectively, and testing for zero by checking if \$ is under the tape head.

Given a description of a 2-counter machine M, which is an instance of the halting problem for 2-counter machines, the reduction constructs a 2pfa $P = P(M)$ with the following property. P accepts with probability $\geq 1 - \epsilon$ any string that represents a halting computation of M on the empty input, and rejects with probability $1-\epsilon$ any string that represents a non-halting computation of M on the empty input, where ϵ may be any constant in the range $(0, 1)$. A computation of M is represented as a string of the form

$$u_1 c^{k_1} d^{m_1} \& u_2 c^{k_2} d^{m_2} \& \ldots \& u_t c^{k_t} d^{m_t},$$

where each u_i represents the state of M, and k_i, m_i denote the values of the 2 counters of M at the ith step of the computation. A dfa can check that the states u_i are in accordance with the transitions of the 2-counter machine, that $u_1 c^{k_1} d^{m_1}$ is the initial configuration, and that u_t is a halting state. The remaining task is to check that the counters are updated correctly. Since a counter's value changes by at most 1 on each transition, this task is easily reduced to that of recognizing the following language:

$$A^* = \{0^{n_1} 1^{n_1} 0^{n_2} 1^{n_2} \ldots 0^{n_k} 1^{n_k}, \mid k \geq 0 \text{ and each } n_i \text{ is a positive integer}\}.$$

The proof that this language can be recognized by a 2pfa with bounded error is a natural extension of Freivald's algorithm for $\{a^n b^n\}$ and can be found in [Freivalds, 1981].

5.2 UNDECIDABILITY OF THE EMPTINESS PROBLEM FOR 1PFA'S

The emptiness problem for 1pfa's is as follows: given a 1pfa A and a constant $\epsilon, 0 < \epsilon < 1/2$, is there some input that the machine accepts with probability at least $1 - \epsilon$?

Theorem 5 *The emptiness problem for unbounded error 1pfa's is undecidable.*

This was proved by Paz [Paz, 1971]. The proof presented here is different than that of Paz, and is due to Lipton [Condon and Lipton, 1989, Lipton, 1989], and actually shows the following stronger result.

Theorem 6 *The following promise problem is undecidable: Given a constant $\epsilon, 0 < \epsilon < 1$ and a 1pfa P that either accepts some string with probability at least $1 - \epsilon$ or accepts all strings with probability at most ϵ, decide which is the case.*

Recall that the key to Freivalds' proof was a bounded error 2pfa for the language

$$A^* = \{0^{n_1}1^{n_1}0^{n_2}1^{n_2}\ldots 0^{n_k}1^{n_k}, \mid k \geq 0 \text{ and } n_i \text{ is a positive integer}\}.$$

At the heart of the proof is a simple game that is played on a string of the form 0^i1^j and is inspired by Freivald's work. The game works as follows. If $i \neq j \bmod k$ where k is some constant, then the outcome is "bad". Otherwise, four tests are performed: (i) flip a coin for each 0 and 1; (ii) flip a coin for each 0 and 1; (iii) flip two coins for each 0; and (iv) flip two coins for each 1. Say that sums win if all coins in either (i) or (ii) are heads and say that doubles win if all coins in either (iii) or (iv) are heads. If sums win, but not doubles, the outcome is good, and if doubles win but not sums, the outcome is bad. Otherwise the outcome is inconclusive. A key property of this game is that if $i = j$, then the outcomes good and bad are equally likely, but if i is not equal to j then the outcome bad is $f(k)$ times more likely than the outcome good, where $f(k)$ goes to infinity as k goes to infinity.

Lipton devised a 1pfa, call it R, that plays the above game independently on successive strings of the form 0^i1^j and has the following property. R has an associated constant parameter $C > 0$ which can be chosen

to be arbitrarily large. R has three outcomes: accept, reject, or inconclusive. Also, if x is in A^*, then Prob[R accepts x] = Prob[R rejects x] and if x is not in A^*, then Prob[R rejects x] > CProb[R accepts x].

We can now describe the reduction from the halting problem for 2-counter machines to the emptiness problem for 1pfa's. Let M be a 2-counter machine. The reduction produces a 1pfa $P = P(M)$ which works as follows. If the input to P is of the form $E_1\&E_2\&E_3\&\ldots E_t$, then, working from left to right, P does a probabilistic check that successive E_i's represent halting computations of M on the empty input. This check has three outcomes: good, bad, or inconclusive. If E_i is indeed a halting computation of M, then the probability that the outcome of P on E_i is good equals the probability that the outcome of P on E_i is bad. But if E_i is not a valid halting computation of M, then the probability that the outcome of P on E_i is bad is C times as great as the probability that the outcome of P on E_i is good. This check can easily be implemented using the 1pfa R described above, similar to Freivalds' proof. If k bad outcomes occur before any good outcome, then P rejects the input; otherwise P accepts the input.

Let $0 < \epsilon < 1$. If C and k are chosen appropriately, then the 1pfa P has the following properties: If M halts on the empty input, then P accepts some input with probability at least $1 - \epsilon$, namely the input that is the concatenation of sufficiently many halting computations of M on the empty input. However, if M does not halt on the empty input, then P accepts any input with probability at most ϵ.

5.3 MARKOV DECISION PROCESSES

In this section we show how undecidability results for 1pfa's lead to undecidability results for central problems on unobservable Markov decision processes (MDP's). Such problems are are abstractions of general planning problems; finding good heuristics for solving planning problems is an active research area in Artificial Intelligence [Boutilier et al., 1999, Littman, 1997, Papadimitriou and Tsitsiklis, 1987].

The following description of a Markov decision process is largely from Derman [Derman, 1970]. Consider a finite state system with state space S that is observed at times 1,2, ... Let $Y_0, Y_1, Y_2, \ldots$ be the sequence of observed states. After each observation of the system, one of a possible number of actions are taken. Let $A_0, A_1, \ldots, A_t, \ldots$ denote an infinite sequence of actions. A rule, or policy, or strategy, is a prescription for taking actions at each point in time. The most general type of strategy for taking an action at time t may be a function of the entire "history" of the system up to time t and may also involve randomization. Thus,

a strategy is a set of functions

$$D_a(H_{t-1}, Y_t),$$

where H_t denotes the history up to time $t-1$, that is, the sequence $Y_0, A_0, Y_1, A_1, \ldots Y_t, A_t$, a is a possible action when the system is in state $Y_t, t = 0, 1, \ldots, 0 \leq D_a \leq 1$, and the sum of the D_a is 1. The interpretation is: if H_{t-1} denotes the history up to time $t-1$ and Y_t denotes the observed state at time t, then the probability of taking action a at time t is $D_a(H_{t-1}, Y_t)$.

The system has an associated transition function that maps (state, action) pairs to a probability distribution over (state, head direction) pairs. If $Y_{t-1} = i$ and action $A_t = a$, then the probability that $Y_t = j$ is determined by the transition function. Given a distribution over the initial observed states of the system and a strategy R, the sequence $\{Y_t, A_t, t = 0, 1, \ldots\}$ is a stochastic process called a Markov decision process.

A certain reward structure is imposed on a Markov decision process: whenever the system is in state i and action a is taken, we assume that a known reward w_{ia} is incurred. In order to consider the accumulated rewards of a process over time, we need the following notation. Let W_t be the random variable which is equal to w_{ia} if $Y_t = i$, $A_t = a$. Let $E_R[W_t]$ be the expected reward of W_t when the strategy is R. That is,

$$E_R[W_t] = \sum_{i,a} \text{Prob}[Y_t = i, A_t = a \mid \text{strategy is} R] w_{ia}$$

The following quantities are among those most commonly studied for Markov decision processes.

- *Finite Horizon Reward:* Let M be a MDP. Let

  $$\text{Reward}(M, T) = \max_R \{\sum_{t=0}^{T} E_R[W_t]\}$$

  denote the maximum total expected reward of M that can be accumulated over the first T steps, where the maximum is taken over all strategies R.

- *Total (Infinite Horizon) Reward:* The quantity

  $$\text{Total-Reward}(M, T) = \lim_{T \to \infty} \text{Reward}(M, T),$$

  if it exists, is called the total reward of the MDP M. Note that the limit may be infinity.

- *Discounted Reward*: Let $0 \leq \alpha < 1$; α is referred to as the discount factor. Let

$$D(M, R, \alpha) = \sum_{t=0}^{\infty} \alpha^t E_R[W_t]$$

be the expected discounted reward of M with strategy R. Let

$$\text{Discounted-Reward}(M, \alpha) = \sup_R D(M, R, \alpha)$$

denote the supremum, over all strategies R, of the expected discounted reward of M with strategy R.

A variant of the MDP model is the unobservable MDP, or UMDP, model. In this model, the strategy is restricted so that the choice of action may depend on t and also on the history of actions up to time $t-1$, but may not depend on the history of states. Thus, a strategy for a UMDP is an infinite sequence of actions. Both UMDP's and MDP's are a special case of POMDP's or Partially Observable Markov Decision Processes, in which the strategy may depend on partial information about the history of states, in addition to the history of actions.

The Total Reward Problem for UMDP's is: given a UMDP M, decide whether Total-Reward$(M) \geq 1/2$ (where in the definition of Total Reward, Reward(M, T) is maximized over all strategies that are restricted as for UMDP's). Similarly, the Discounted Reward Problem for UMDP's is: given a UMDP M and a discount factor α, decide whether Discounted-Reward$(M) \geq 1/2$.

UMDP's and 1pfa's are closely related, as we now explain. Let P be a 1pfa and without loss of generality assume that P always moves its head to the right at every step. Furthermore, assume that P always rejects if the initial symbol under the input head is not # or if # is read when the head of P has moved right off the leftmost #, and that P enters an absorbing state (accept or reject) when \$ is reached. Let Σ be the alphabet of P. Now let M be a UMDP with the same state space as P with the following properties. The number of actions of M associated with each state is $|\Sigma| + 2$ (the "+2" includes the endmarkers), and the transition function of M is identical to that of P. Any transition that causes M to enter the accept state from a state other than the accept state itself has a reward of 1 and all other transitions have a reward of 0. Think of the accept state as a goal state; then the total reward of M on a given strategy is the probability of eventually reaching the goal state on that strategy. Note that there is a 1-1 correspondence between inputs $\#x\$$ of P and strategies of M that start with $\#x\$$. Let R_x be any strategy corresponding to input string x. Then the probability that P accepts x equals the total reward of M on strategy R_x.

Theorem 7 *The Total Reward problem for UMDP's is undecidable.*

Theorem 7 follows immediately from the following stronger result.

Theorem 8 *The following promise problem for UMDP's is undecidable: Given a constant $\epsilon, 0 < \epsilon < 1$ and a UMDP M that either has total-reward at least $1 - \epsilon$ or at most ϵ, decide which is the case.*

This result follows directly from the relationship between 1pfa's and UMDP's described above, together with Theorem 6. Let $\epsilon, 0 < \epsilon < 1$ be given and let P be a 1pfa that either accepts some string with probability at least $1 - \epsilon$ or accepts all strings with probability at most ϵ. Let M be the corresponding Markov decision process as described above. Then, P accepts some string with probability at least $1 - \epsilon$ if and only if the unobservable total reward of M is at least $1 - \epsilon$, and P accepts all strings with probability at most ϵ if and only if the unobservable total reward of M is at most ϵ.

Theorem 9 *The Discounted Reward Problem for UMDP's is undecidable.*

The proof is a simple modification of the argument for the Total Reward problem. Let $1/2 < d < 1$, and consider the discounted reward of a UMDP obtained in the previous reduction. Effectively, the discount d acts as a penalty at every step, analogous to entering the reject state with probability $1 - d$ at each step. A natural idea, then, is to try to *balance out this penalty by adding a bonus* at each transition of the UMDP. Specifically, let M' be obtained from M by decreasing the probability of each transition of M from all states by a factor of $(1 - d)$ and then increasing the probability of entering the accept (goal) state by d.

Fortuitously, this modification has the desired effect: if the UMDP M has a strategy whereby the goal state is reached with probability greater than $1/2$, then the discounted reward of M' is greater than $1/2$. Furthermore, if on all strategies the total reward of M is less than $1/2$, then similarly on all strategies the discounted reward of M' is less than $1/2$. A key property of M that ensures this is that, on an input w, P may only enter the accept state upon reaching the right endmarker symbol, \$, for the first time.

Acknowledgments

This research was supported by National Sciences and Engineering Research Council (NSERC) of Canada.

References

[Boutilier et al., 1999] Boutilier, C., Dean, T., and Hanks, S. (1999). Decision theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94.

[Condon and Lipton, 1989] Condon, A. and Lipton, R. (1989). On the complexity of space bounded interactive proofs. In *Proceedings of the 30th Annual IEEE Symposium on the Foundations of Computer Science*, pages 462–467.

[Derman, 1970] Derman, C. (1970). *Finite State Markovian Decision Processes*, volume 67 of *Mathematics in Science and Engineering (Edited by Bellman)*. Academic Press.

[Dwork and Stockmeyer, 1990] Dwork, C. and Stockmeyer, L. (1990). A time-complexity gap for two-way probabilistic finite state automata. *SIAM J. Comput.*, 19:1011–1023.

[Dwork and Stockmeyer, 1992] Dwork, C. and Stockmeyer, L. (1992). Finite state verifiers i: the power of interaction. *J. ACM*, 39(4):800–828.

[Freivalds, 1981] Freivalds, R. (1981). Probabilistic two-way machines. In *Proc. of the International Symposium on Mathematical Foundations of Computer Science Springer-Verlag Lecture Notes in Computer Science, 188*, pages 33–45.

[Glaister and Shallit, 1998] Glaister, I. and Shallit, J. (1998). Automaticity iii: Polynomial automaticity and context-free languages. *Comput. complex.*, 7:371–387.

[Greenberg and Weiss, 1986] Greenberg, A. G. and Weiss, A. (1986). A lower bound for probabilistic algorithms for finite state machines. *J. Comput. Syst. Sci.*, 33:88–105.

[Hopcroft and Ullman, 1979] Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley.

[Kaneps, 1989] Kaneps, J. (1989). Stochasticity of the languages acceptable by two-way finite probabilistic automata. *Diskretnaya Matematika*, 1:63–77.

[Kaneps and Freivalds, 1990] Kaneps, J. and Freivalds, R. (1990). Minimal nontrivial space complexity of probabilistic one-way turing machines. In *Proc. of the Conference on Mathematical Foundations of Computer Science, Springer Verlag Lecture Notes in Computer Science, 452*, pages 355–361.

[Karp, 1967] Karp, R. M. (1967). Some bounds on the storage requirements of sequential machines and turing machines. *J. ACM*, 14(3):478–489.

[Leighton and Rivest, 1983] Leighton, F. T. and Rivest, R. L. (1983). The markov chain tree theorem. Technical Report MIT/LCS/TM-249, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA (Also in IEEE Transactions on Information Theory, IT-37(6), (1986) 733-742).

[Lipton, 1989] Lipton, R. J. (1989). Recursively enumerable languages have finite state interactive proofs. Technical Report CS-TR-213-89, Department of Computer Science, Princeton University, Princeton, N.J.

[Littman, 1997] Littman, M. L. (1997). Probabilistic propositional planning: Representations and complexity. In *Proc. Fourteenth National Conference on AI, AAAI Press*, pages 748–754.

[Macarie, 1998] Macarie, I. (1998). Space-efficient deterministic simulation of probabilistic automata. *SIAM J. Comput.*, 27(2):448–465.

[Madani et al., 1999] Madani, O., Condon, A., and Hanks, S. (1999). On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision process problems. In *Proc. Sixteenth Annual Conference on Artificial Intelligence.*

[Moore, 1956] Moore, E. F. (1956). *Gedanken-experiments on sequential machines*, pages 129–153. Annals of Math. Studies No. 34. Princeton University Press, Princeton, N. J.

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

[Paz, 1971] Paz, A. (1971). *Introduction to Probabilistic Automata.* Academic Press.

[Pomerance et al., 1997] Pomerance, C., Robson, J. M., and Shallit, J. (1997). Automaticity ii: Descriptional complexity in the unary case. *Theoretical Computer Science*, 180(1-2):181–201.

[Rabin, 1963] Rabin, M. O. (1963). Probabilistic automata. *Information and Control*, 6:230–245.

[Ravikumar, 1992] Ravikumar, B. (1992). Some observations on 2-way probabilistic finite automata. In *Lecture Notes in Computer Science*, volume 652, pages 392–403. Springer-Verlag.

[Salomaa and Soittola, 1978] Salomaa, A. and Soittola, M. (1978). *Automata-theoretic aspects of formal power series.* Texts and Monographs in Computer Science. Springer-Verlag, New York.

[Shallit and Breitbart, 1996] Shallit, J. and Breitbart, Y. (1996). Automaticity i: Properties of a measure of descriptional complexity. *J. Comput. Systems Sci.*, 53:10–25.

[Turakainen, 1969] Turakainen, P. (1969). Generalised automata and stochastic languages. In *Proc. American Math. Soc.*, volume 21(2), pages 303–309.

[Wang, 1992] Wang, J. (1992). A note on two-way probabilistic automata. *Information Processing Letters*, 43:321–326.