UPPER BOUNDS ON THE COMPLEXITY OF SPACE BOUNDED INTERACTIVE PROOFS

by

Anne Condon and Richard J. Lipton

Computer Sciences Technical Report #841

April 1989

Upper Bounds on the Complexity of Space Bounded Interactive Proofs

Anne Condon *
Computer Science Department
University of Wisconsin-Madison

Richard J. Lipton †
Computer Science Department
Princeton University

April 1989

Abstract

We show that the class *IP(2pfa)* of languages accepted by interactive proof systems with finite state verifiers is contained in $ATIME(2^{2^{O(n)}})$. We also show that 2-prover interactive proof systems with finite state verifiers accept exactly the recursive languages. Our results generalize to other space bounds. We also obtain some results of independent interest on the rate of convergence of time-varying Markov chains and of non-Markov chains, called feedback chains, to their halting states.

1 Introduction

We prove new bounds on the class of languages accepted by space bounded interactive proof systems, in particular those with a 2-way probabilistic finite state verifier (*2pfa*). Interactive proof systems were introduced by Goldwasser et al. [8]; those with space bounded verifiers have been studied by Condon [2], Dwork and Stockmeyer [4] Killian [9] and Lipton [10]. An interactive proof consists of a pair of interacting computing processes, one called the prover, P and the other called the verifier, V. The pair shares an input; the more powerful prover wishes to convince the verifier, in this case a *2pfa*, that the input is in some language. The probability that a prover-verifier pair (P, V) accepts an input is the probability that the prover convinces the verifier that the input is in the language, taken over all the possible coin-tosses of the verifier.

Two different definitions of language acceptance for space bounded interactive proof systems have been proposed. Dwork and Stockmeyer [4] defined a prover-verifier pair (P, V) to be an interactive proof system for a language L with error probability $\epsilon < 1/2$ if

1. for all $x \in L$, (P, V) accepts x with probability at least $1 - \epsilon$ and
2. for all $x \notin L$ and all provers P^*, (P^*, V) rejects x with probability at least $1 - \epsilon$.

*Work supported by NSF grant number DCR-8402565
†Work supported by DARPA and ONR contracts N00014-85-C-0456 and N00014-85-K-0465, and by NSF Cooperative Agreement DCR-8420948

the condition 2 above by the following condition.

2′. For all $x \notin L$ and all provers P^*, (P^*, V) accepts x with probability at most ϵ.

It is not hard to see that for time bounded verifiers these definitions are equivalent. However, for space bounded verifiers Lipton [10] showed that the definitions are not equivalent. He proved the surprising result that the class of languages accepted by interactive proofs that satisfy the weaker condition is exactly the recursively enumerable languages, even when the verifier is a *2pfa*. In fact, the result holds for *one-way* interactive proofs, where the verifier never communicates with the prover except to request another communication symbol. The proof exploits the fact that the weaker interactive proof systems do not have to halt with high probability on inputs they do not accept. On the other hand, all languages in the class *IP(2pfa)* must be recursive. This is a direct consequence of the fact that these must halt with high probability on all inputs and is proved in Section 5.

In this paper we obtain the first non-trivial upper bound for the class *IP(2pfa)*. We show that any language in this class is contained in $ATIME(2^{2^{O(n)}})$. The best known lower bound, shown by Dwork and Stockmeyer [4], is that *IP(2pfa)* contains any language in $DTIME(2^{O(n)})$. Our result extends to yield upper bounds on the complexity of languages accepted by interactive proof systems with more general space bounds. The results on space bounded interactive proof systems are in Section 4.

We also contrast the power of multi-prover interactive proof systems with single prover interactive proof systems for space bounded verifiers. Multi-prover systems were introduced in Ben-Or et al. [1]. Informally, in a k-prover interactive proof system, the verifier communicates with k provers that cooperate with each other to convince the verifier that an input is in some language. The provers cannot communicate with each other during the execution of the verifier's protocol. Multi-prover interactive proof systems generalize the single-prover systems and hence are potentially more powerful; however in the case of polynomially time bounded verifiers it is an open problem whether they are more powerful or not. See Fortnow et al. [6] for results on the power of polynomial time bounded multi-prover interactive systems.

We show that any recursive language has a 2-prover interactive proof with a *2pfa* verifier that halts with probability 1. We also prove the other direction - that any language accepted by a multi-prover interactive proof system is recursive. Thus we obtain matching upper and lower bounds on the power of space bounded 2-prover interactive proof systems. These results were proven independently by Feige and Shamir [5]. As a consequence of these results, we can make the following informal observations about space bounded interactive proof systems. Two provers are much more powerful than one prover but three or more provers are no more powerful than two provers. These results are given in Section 5.

In proving our upper bound on single-prover interactive proof systems, we also obtain some results of independent interest on the rate of convergence of certain classes of discrete time-varying Markov chains to their halting, or absorbing, states. It turns out that one-way interactive proof systems – those in which all communication between the verifier and the prover is from the prover to the verifier – are closely related to certain time-varying Markov chains. Recall that in a stationary Markov chain, say over state space $\{1, \ldots, n\}$, the probability p_{ij} of a transition from i to j at the kth step is independent of k. Thus a single probability transition matrix P is sufficient to describe all the possible transitions of the chain at all time steps. In contrast, the transition probabilities of a time-varying Markov chain at the kth time step are a function of k. A time-varying Markov chain can be described by an infinite sequence of stochastic matrices $\{P_k\}, k \geq 1$, one per time step. In this paper, we consider the family of n-state time-varying Markov chains such that the matrices P_k are all from some finite set of stochastic matrices, say $\{A, B\}$. We assume that all the entries of A and B are rational, of the form p/q where p

and q are integers, $p \leq q \leq 2^n$. When A and B are fixed, we denote the family of such Markov chains with initial state 1 by $\mathcal{M}$.

A special case of our results on time-varying Markov chains can be stated simply as follows. Suppose that for all chains M in $\mathcal{M}$, n is a halting state which is eventually reached from the initial state with probability 1. Then the expected time to reach the halting state n is $2^{2^{O(n)}}$. A well known result for stationary Markov chains under similar conditions states that the expected time to reach a halting state n is $2^{O(n)}$. Our results on time-varying Markov chains are in Section 2.

We also show that this bound is tight in the following sense. Given any n, for some $m = O(n)$ there exist $m \times m$ matrices A and B with rational entries of the form p/q where $p \leq q \leq 2^m$, such that (i) for all chains M in $\mathcal{M}$, M eventually reaches the halting state and (ii) for some such chain, the expected time to do so is $2^{2^{\Omega(n)}}$. This latter result is easily explained using the intuitive framework of interactive proof systems. It provides evidence that the model of an interactive proof is a useful paradigm by which to approach other interesting open problems. As another example of this, Lipton [10] applied his result on weak interactive proof systems to show that the emptiness problem for 1-way probabilistic finite automata is undecidable.

From our results on time-varying Markov chains we derive upper bounds on the complexity of space bounded *one-way* interactive proof systems, where all communication between the verifier and the prover is from the prover to the verifier. In order to extend these results to two-way space bounded interactive proof systems where both the verifier and the prover send information to each other via the communication tape, we generalize our results on time-varying Markov chains to non-Markov chains, which we call *feedback* chains. In a feedback chain, there is a feedback symbol associated with each state. The probability of a transition from a state i to state j at the kth step depends not only on i, j and k but also on feedback of the states entered at steps $1, \ldots, k-1$. We describe feedback chains and our bounds on their convergence rates to halting states in Section 3. In Section 4 we apply these results to derive upper bounds on the complexity of space bounded interactive proof systems. Finally in Section 5 we contrast the power of single prover and multi-prover interactive proof systems for *2pfa* verifiers.

2 Time-varying Markov Chains

In this section we prove results on the rate of convergence to absorbing states of time varying Markov chains. A time-varying Markov chain is a sequence of random variables $X_1, X_2, \ldots, X_k, \ldots$ over state space $\{1, \ldots, n\}$ with the following property. For all integers i, j and k there is a real number $p_{ijk} \in [0, 1]$ such that p_{ijk} is the probability that the kth random variable, X_k, is j, given that the $(k-1)$st random variable is i. That is,

$$\text{Prob}[X_k = j | X_1 = a_1, \ldots, X_{k-2} = a_{k-2}, X_{k-1} = i] = p_{ijk}.$$

For all $k > 0$, let P_k be the matrix $[p_{ijk}], 1 \leq i, j \leq n$. The matrices P_k and the value of X_1 completely determine the chain. For any k, if $\text{Prob}[X_k = j] = p$, we say that the chain is in state j with probability p at time step k.

In this paper, we consider time-varying Markov chains where the matrices P_k come from a finite set of stochastic matrices. We assume that this set is of size two, say $\{A, B\}$ but our results generalize to any finite set. We let $\mathcal{M}$ denote the *family of n-state time-varying Markov chains* with initial state 1, where A and B have the following two properties. First, all of the entries of A and B are rational numbers of the form p/q where p and q are integers such that $p \leq q = 2^n$. Second, the nth row of both A

and B has 0's everywhere, except at the $(n,n)th$ position, which is 1. Thus n is an *absorbing*, or *halting*, state of $\mathcal{M}$; that is, whenever a chain enters the halting state n it never leaves that state. Let $\{A,B\}^\omega$ denote the set of infinite strings over the alphabet $\{A,B\}$. Then there is a one-to-one correspondence between the chains in $\mathcal{M}$ and the strings $\alpha = \alpha_1\alpha_2\ldots\alpha_k\ldots$ in $\{A,B\}^\omega$. M_α is the chain corresponding to α if and only if $P_k = \alpha_k$.

We next define what we mean by the halting probability of a family of time-varying Markov chains. Throughout, we denote the jth entry of an n-vector $\bar{x}$ by $(\bar{x})_j$ or $\bar{x}_j$. The vector $\bar{e}^i$ denotes the n-vector where all components are 0 except the ith, which is 1. Let $\alpha^{(k)}$ denote the matrix product $\prod_{i=1}^{k}\alpha_i$. Then $(\bar{e}^i\alpha^{(k)})_j$ is the probability that the value of the kth random variable in the chain M_α is j given that the chain is initially in state i. We say n is *reachable in k steps* from i on sequence α if $(\bar{e}^i\alpha^{(k)})_n > 0$. Similarly, we say that n is *reachable* from i on sequence α if for some k, n is reachable in k steps from i. Since n is a halting state, for any α and $\bar{x} \in [0,1]^n$, the sequence $\{(\bar{x}\alpha^{(k)})_n\}, k = 1,2,\ldots,$ is non-decreasing. Also every element of the sequence is bounded above by 1. Hence $\lim_{k\to\infty}(\bar{x}\alpha^{(k)})_n$ exists. We call this limit the probability that M_α halts, that is, reaches n from $\bar{x}$. When $\bar{x} = \bar{e}^1$ we call this the *halting probability* of M_α and denote it by p_α. We call $\inf_\alpha p_\alpha$ the *halting probability* of $\mathcal{M}$.

The goal of this section is to show that if the halting probability of $\mathcal{M}$ is at least p then for any α, the probability that M_α halts in $k2^{2^{O(n)}}$ steps is at least $p - 1/2^k$. We first give an overview of the proof. Consider any string α as composed of substrings of length 2^n and let $p_{\alpha,i}$ be the probability that M_α halts in $i2^n$ steps. An important part of the proof is to derive a lower bound on the difference $p_{\alpha,i+1} - p_{\alpha,i}$, which is the probability that M_α halts in $(i+1)2^n$ steps but *not* in $i2^n$ steps. This lower bound is obtained in two steps. Let S be the set of states in $[n-1]$ that are reachable from 1 in $i2^n$ steps of α and from which the halting state n is reachable in a further 2^n steps of α. That is, if α' is the string $\alpha_{i2^n+1}, \alpha_{i2^n+2}, \ldots$, which is the string α with the first $i2^n$ symbols removed, then

$$S = \{j \in [n-1] \mid (\bar{e}^1\alpha^{(i2^n)})_j > 0 \text{ and } (\bar{e}^j(\alpha')^{(2^n)})_n > 0\}.$$

The first step in obtaining a lower bound on $p_{\alpha,i+1} - p_{\alpha,i}$ is to show that if the chain M_α is in any state of S at step $i2^n$ with probability at least q, then the probability that the halting state is reached in a further 2^n steps is $\geq q/(2^n)2^n$. This follows immediately from Lemma 2.1. The second step in obtaining the lower bound is to show that with probability at least $p - p_{\alpha,i}$, M_α is in a state of S at step $i2^n$. This is proved in Lemma 2.3. Also in Lemma 2.3, the results of these two steps are combined to show that $p_{\alpha,i+1} - p_{\alpha,i} \geq (p - p_{\alpha,i})/(2^n)2^n$. Finally, this result is used in Lemma 2.4 to get an upper bound on the rate of convergence of $p_{\alpha,i}$ to p and to complete the proof.

For any set $S \subseteq [n]$ define the *weight* of S with respect to an n-vector $\bar{x}$ as $\sum_{j\in S}\bar{x}_j$ and denote it by $w_S(\bar{x})$.

Lemma 2.1 *Let S be any subset of $[n-1]$ and suppose that n is reachable from every $j \in S$ in k steps on M_α, where $\alpha \in \{A,B\}^\omega$. Then for any vector $\bar{x} \in [0,1]^n$, $(\bar{x}\alpha^{(k)})_n \geq \bar{x}_n + w_S(\bar{x})/(2^n)^k$.*

Proof: Write $\bar{x}$ as $\bar{u} + \bar{x}_n\bar{e}^n$, where $\bar{u}_n = 0$. Then $(\bar{x}\alpha^{(k)})_n = \bar{x}_n + (\bar{u}\alpha^{(k)})_n$, since n is halting. Also since all entries in the α_i and $\bar{x}$ are nonnegative, $(\bar{u}\alpha^{(k)})_n \geq \sum_{j\in S}\bar{x}_j(\bar{e}^j\alpha^{(k)})_n$.

Since n is reachable from every $j \in S$ in k steps, $(\bar{e}^j\alpha^{(k)})_n > 0$ by definition. Since the entries of the α_i are rational numbers of the form p/q where $p \leq q \leq 2^n$ it follows that every non-zero entry of $\alpha^{(k)}$, and hence of $\bar{e}^j\alpha^{(k)}$, is at least $1/(2^n)^k$. This can be proved easily by induction on k. Hence $(\bar{e}^j\alpha^{(k)})_n \geq 1/(2^n)^k$ and so $(\bar{u}\alpha^{(k)})_n \geq \sum_{j\in S}\bar{x}_j/(2^n)^k$. Therefore, $(\bar{x}\alpha^{(k)})_n \geq \bar{x}_n + w_S(\bar{x})/(2^n)^k$, as required. □

We define a set $S \subseteq [n-1]$ to be *k-safe* for any integer k, if for all sequences $\alpha \in \{A,B\}^\omega$, there exists $i \in S$ such that n is reachable from i on α in k steps. Similarly, a set $S \subseteq [n-1]$ is *safe* if for all α there exists $i \in S$ such that n is reachable from i on α. In Lemma 2.2 we prove that any safe set is 2^n-safe.

Lemma 2.2 *Let $S \subseteq [n-1]$. If S is safe then S is 2^n-safe.*

Proof: Suppose S is not 2^n-safe, so that n is not reachable from any state of S in 2^n steps on some sequence α. Let S_i be the set of states reachable from S in i steps of M_α. Since each S_i is a subset of $[n]$, by the pigeon hole principle $S_j = S_k$ for some j,k, $1 \leq j < k \leq 2^n$. If $\alpha' = \alpha_1 \ldots \alpha_j (\alpha_{j+1} \ldots \alpha_k)^*$, then the halting state is not reachable from any state of S on α'. This contradicts the fact that S is safe. □

Consider each string $\alpha \in \{A,B\}^\omega$ as composed of substrings of length 2^n and let $p_{\alpha,i}$ denote $(\bar{e}^1 \alpha(i2^n))_n$. Thus $p_{\alpha,i}$ is the probability that M_α halts, that is, reaches state n from state 1, in $i2^n$ steps. Recall that p_α denotes the halting probability of M_α . Therefore, $p_\alpha = \lim_{i \to \infty} p_{\alpha,i}$. Recall that $p = \inf_\alpha p_\alpha$ is the *halting probability* of $\mathcal{M}$.

Lemma 2.3 *Let the halting probability of $\mathcal{M}$ be p and for each α let $p_{\alpha,i}$ be the probability that M_α halts at step $i2^n$. Then for any α and any $i \geq 0$,*

$$p_{\alpha,i+1} - p_{\alpha,i} \geq (p - p_{\alpha,i})/(2^n)^{2^n}.$$

Proof: If $p - p_{\alpha,i} \leq 0$ the inequality is trivially true so we restrict our attention to the case where $p - p_{\alpha,i} > 0$.

Let $\bar{x} = \bar{e}^1 \alpha(i2^n)$. Let α' be the string $\alpha_{i2^n+1}, \alpha_{i2^n+2}, \ldots$; that is, the string α with the first $i2^n$ symbols removed. Let S be the set of states that are reachable from state 1 in $i2^n$ steps of α and which reach the halting state n in a further 2^n steps. Thus,

$$S = \{j \in [n-1] \mid \bar{x}_j > 0 \text{ and } (\bar{e}^j (\alpha')^{(2^n)})_n > 0\}.$$

Also let $\bar{S} = \{j \in [n-1] \mid \bar{x}_j > 0 \text{ and } (\bar{e}^j (\alpha')^{(2^n)})_n = 0\}$.

We claim that $\bar{S}$ is not 2^n-safe. This is because on sequence α', no state in $\bar{S}$ reaches the halting state in 2^n steps, by definition of $\bar{S}$. Then from Lemma 2.2, $\bar{S}$ is not safe. Thus there is some sequence $\alpha'' \in \{A,B\}^\omega$ such that n is not reachable from any state of $\bar{S}$ on α''.

We use the fact that $\bar{S}$ is not safe to argue that the weight of $\bar{S}$ with respect to $\bar{x}$ is less than $1-p$; that is, $w_{\bar{S}}(\bar{x}) = \sum_{j \in \bar{S}} \bar{x}_j < 1 - p$. Suppose to the contrary that $w_{\bar{S}}(\bar{x}) \geq 1 - p$. Then on the sequence obtained by concatenating the first $i2^n$ symbols of α with α'', the probability that the halting state n is eventually reached from state 1 is $< p$, contradicting the fact that p is the halting probability of $\mathcal{M}$. Hence

$$w_S(\bar{x}) = 1 - w_{\{n\}}(\bar{x}) - w_{\bar{S}}(\bar{x}) \geq 1 - p_{\alpha,i} - (1-p) = p - p_{\alpha,i}.$$

From Lemma 2.1 it follows that $(\bar{x}\ (\alpha')^{(2^n)})_n \geq \bar{x}_n + w_S(\bar{x})/(2^n)^{2^n}$. Note that $p_{\alpha,i+1} = (\bar{x}(\alpha')^{(2^n)})_n$, $p_{\alpha,i} = \bar{x}_n$ and $w_S(\bar{x}) \geq p - p_{\alpha,i}$. Substituting all of these values into the last inequality, $p_{\alpha,i+1} - p_{\alpha,i} \geq (p - p_{\alpha,i})/(2^n)^{2^n}$, as required. □

Lemma 2.4 *Let the halting probability of $\mathcal{M}$ be p and for every α let the probability that M_α halts at step $i2^n$ be $p_{\alpha,i}$. Then $p - p_{\alpha,i} \leq (1 - 1/\mu)^i$, where $\mu = (2^n)^{2^n}$.*

Also for any integer $k > 0$, the probability that M_α halts within $k2^n\mu$ steps of M_α is at least $p - 1/2^k$ for all α.

Proof: We prove the first part by induction on i. The basis, when $i = 0$, is trivial since then $p - p_{\alpha,i} = p \leq 1 = (1 - 1/\mu)^i$. Suppose that for some $i \geq 0$, $p - p_{\alpha,i} \leq (1 - 1/\mu)^i$. Then

$$\begin{aligned} p - p_{\alpha,i+1} &= (p - p_{\alpha,i}) - (p_{\alpha,i+1} - p_{\alpha,i}) \\ &\leq (p - p_{\alpha,i}) - (p - p_{\alpha,i})/\mu && \text{from Lemma 2.3} \\ &= (p - p_{\alpha,i})(1 - 1/\mu) \\ &\leq (1 - 1/\mu)^{i+1} && \text{(from the induction hypothesis).} \end{aligned}$$

This completes the first part of the proof. From this, the probability that M_α halts within $i2^n$ steps is $p_{\alpha,i} \geq p - (1 - 1/\mu)^i$. If $i = k\mu$ for some k then $(1 - 1/\mu)^i < 2^{-k}$. Hence the probability that M_α halts within $k2^n\mu$ steps is at least $p - 1/2^k$, as required. □

The bound in Lemma 2.4 is fairly tight, as the next lemma shows.

Lemma 2.5 *For any n, there is an integer $m = O(n)$ and a family $\mathcal{M}$ of m-state Markov chains such that the halting probability $\mathcal{M}$ is 1. Furthermore, for some chain in $\mathcal{M}$, the expected time to reach the halting state n is $2^{2^{\Omega(n)}}$.*

The proof of this lemma is postponed until the Section 4 because it is best described by constructing an interactive proof that runs in expected time $2^{2^{\Omega(n)}}$.

The following theorem combines Lemmas 2.4 and 2.5 to obtain our main result on the rate of halting of time-varying Markov chains.

Theorem 2.1 *Suppose $\mathcal{M}$ is a family of n-state time-varying Markov chains with halting probability p. Then for any chain in $\mathcal{M}$, the probability of halting in $2^{2^{O(n)}}$ steps is at least $p - o(1)$. Moreover, this bound is the best possible.*

3 Feedback Chains

In order to obtain our upper bounds on interactive proof systems, we need to generalize the definitions and results of Section 2 to non-Markov chains which we call *feedback chains*. The reason for this is that in an interactive proof, when the verifier communicates with the prover, it gives the prover information about its current state, and the prover's response to the verifier can depend on all of the information it receives from the verifier. To model this, we associate a feedback symbol with each state of a feedback chain; more than one state may be associated with the same feedback symbol. The value of the kth random variable of a feedback chain may depend not only on k and the value of the $(k-1)$st random variable, but also on the feedback symbols of the values of random variables $1, \ldots, k-1$. Without loss of generality we assume that the set of feedback symbols is $\{0, 1\}$; the results extend easily to larger sets of feedback symbols.

Formally, let S be a set of states $\{1, \ldots, n\}$ and let f be a function from S to $\{0, 1\}$, called the feedback function. Then a feedback chain is a sequence of random variables $X_1, X_2, \ldots, X_k, \ldots$ over state space $\{1, \ldots, n\}$ such that for all integers i, j and states $a_1, \ldots a_{k-1}$ there is a real number $p(i, j, f(a_1), \ldots, f(a_{k-1})) \in [0, 1]$ such that

$$\text{Prob}[X_k = j | X_1 = a_1, \ldots, X_{k-2} = a_{k-2}, X_{k-1} = i] = p(i, j, f(a_1), f(a_2), \ldots, f(a_{k-1})).$$

The number $p(i, j, b_1, \ldots, b_{k-1})$ is the probability of entering state j at time k given that the chain is in state i at time $k-1$ and the feedback symbols at steps $1, \ldots k-1$ are $b_1, \ldots, b_{k-1}$, respectively. Let $P_{b_1 \ldots b_k} = [p(i, j, b_1, \ldots, b_k)], 1 \leq i, j \leq m$ denote the probability transition matrix at step k when the feedback is $b_1, \ldots, b_k$.

As with the time-varying Markov chains, we consider the special case where the transition matrices come from the set $\{A, B\}$ where A and B are stochastic matrices whose entries are rational numbers of the form p/q where p and q are integers such that $p \leq q \leq 2^n$. We let $\mathcal{M}$ denote the family of all such feedback chains when $X_1 = 1$. Just as there is a one-to-one correspondence between the family of time-varying Markov chains in $\mathcal{M}$ and the strings in $\{A, B\}^\omega$, there is a similar correspondence between the feedback chains in $\mathcal{M}$ and infinite binary trees whose nodes are labeled A or B. We next describe this correspondence and introduce some useful notation. We illustrate these definitions in Figure 1.

Let $T(A, B)$ be the family of infinite binary trees α where every node of α has two children, each node is labeled either A or B and the left and right edges from a node are labeled 0 and 1 respectively. Denote the root of tree α by root(α) and denote the matrix labeling node η by $matrix(\eta)$. Define the level of a node η in a tree to be the number of edges on the path from the root to η. In particular, the level of the root is 0. Suppose that η is a node at level k of some tree α and that for $1 \leq i \leq k$, b_i is the label of the ith edge on the path from the root of α to η. Denote the string $b_1 \ldots b_k$ by $feedback(\eta)$.

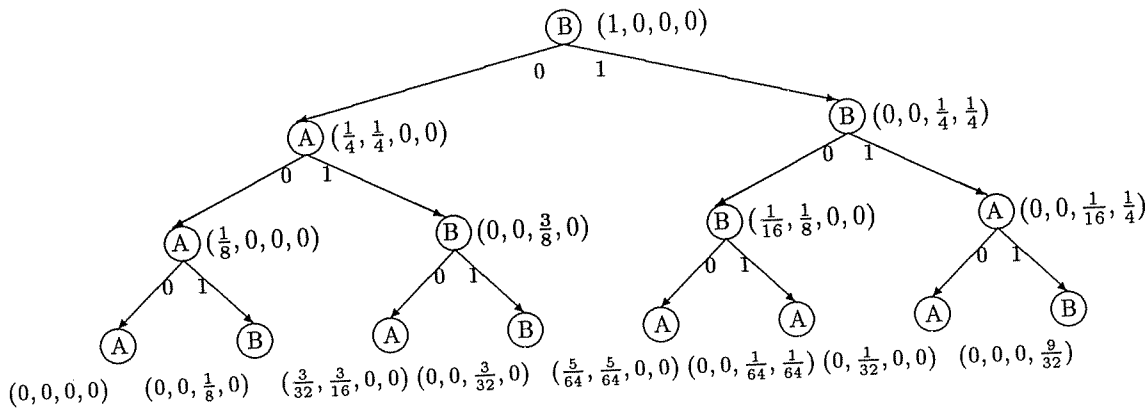
Then there is a one-to-one correspondence between the chains in $\mathcal{M}$ and the set of infinite binary trees $\alpha \in T(A, B)$, where the chain M_α corresponds to tree α if and only if for all $b_1 \ldots b_k \in \{0, 1\}^*$, $P_{b_1 \ldots b_k} = matrix(\eta)$, where $feedback(\eta) = b_1 \ldots b_k$.

For each i and each node η of tree α, we define a vector $\bar{x}(\eta, i)$ with the following property. Suppose that η is at level k of tree α. Then $(\bar{x}(\eta, i))_j$ is the probability that the $(k+1)$st random variable in the chain corresponding to α is j *and* that the feedback from the first k steps is $feedback(\eta)$, given that the chain starts in state i.

The vector $\bar{x}(\eta, i)$ is defined inductively as follows. The base case is root(α) and we define $\bar{x}$(root(α),i) $= \bar{e}^i$. Let F_0 be the matrix such that $(i, i) = 1$ if the feedback symbol $f(i)$ associated with state i is 0 and all other entries of F_0 are 0. Let $F_1 = I - F_0$. F_0 and F_1 are defined so that for any vector $\bar{v}$, the ith component of $\bar{v}F_0$ is $\bar{v}_i$ if $f(i) = 0$ and is 0 otherwise. Similarly, $(\bar{v}F_1)_i = \bar{v}_i$ if $f(i) = 1$ and is 0 otherwise. Then if η is any node with left and right children η_0 and η_1 respectively then $(\bar{x}(\eta_0, i))_j = (\bar{x}(\eta, i) matrix(\eta) F_0)_j$ and $(\bar{x}(\eta_1, i))_j = (\bar{x}(\eta, i) matrix(\eta) F_1)_j$.

Figure 1 depicts the first four levels of a tree in $T(A, B)$ that corresponds to a feedback chain with states $\{1, 2, 3, 4\}$. The feedback symbol of the first two states is 0 and of the second two is 1. The initial state is 1 and state 4 is the halting state. The vector adjacent to each node η is $\bar{x}(\eta, 1)$. Thus for example, the vector $(0, 0, \frac{1}{16}, \frac{1}{4})$ adjacent to the rightmost node at level three of the tree indicates that in the first two steps, the probability that the feedback is 1,1 *and* that the chain is in state 3 is $\frac{1}{16}$. Similarly, the probability that the feedback is 1,1 and the chain is in state 4 is $\frac{1}{4}$. Also, if the feedback is 1,1 then the probability of being in either state 1 or 2 is 0 since the feedback symbol of these two states is 0.

We define $\alpha^{(k)}$ to be the matrix whose (ij)th entry is the probability that $X_{k+1} = j$, given that $X_1 = i$. To do this, we let $\alpha^{(0)} = I$ and for $k \geq 1$, $\alpha_{ij}^{(k)} = \sum_\eta (\bar{x}(\eta, i))_j$, where the sum is taken over all nodes η at level k of tree α. If α is any extension of the tree of Figure 1, the vector $\bar{e}^1 \alpha^{(k)}$ is the sum of the vectors at level k. Hence for example, $\bar{e}^1 \alpha^{(3)} = (\frac{11}{64}, \frac{19}{64}, \frac{15}{64}, \frac{19}{64})$. Thus the probability of being in state 1 in three steps of M_α is 11/64.



$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 1: *The first three levels of a tree in $T(A,B)$ that corresponds to a feedback chain with states $\{1,2,3,4\}$. The feedback function f is defined by $f(1) = f(2) = 0$, $f(3) = f(4) = 1$. Beside each node η is the vector $\bar{x}(\eta, 1)$.*

The definitions of reachability for feedback chains is a natural extension of the definition for time-varying Markov chains. We say that n is reachable from i at η, where η is a node in tree α, if $(\bar{x}(\eta, i))_n > 0$. Similarly, n is reachable from i in k steps on α if $(\bar{e}^i \alpha^{(k)})_n > 0$; equivalently, if n is reachable from i at some node η at level k of tree α. Finally, n is reachable from i on α if n is reachable from i on α in k steps, for some k.

As before, assume that n is a halting state of A and B. Then for any tree α and $\bar{x} \in [0,1]^n$, the sequence $\{(\bar{x}\alpha^{(k)})_n\}, k = 0, 1, 2, \ldots$ is non-decreasing. Also every element of the sequence is bounded above by 1. Hence $\lim_{k\to\infty} (\bar{x}\alpha^{(k)})_n$ exists. We call this limit the probability of reaching n from $\bar{x}$ on the chain M_α. When $\bar{x} = \bar{e}^1$ we call this the halting probability of M_α and denote it by p_α. We call $\inf_\alpha p_\alpha$ the *halting probability* of $\mathcal{M}$.

Our goal is to show that if for all trees $\alpha \in T(A,B)$, the halting probability p_α of M_α is at least p then for any α, the probability of reaching n, or halting, in kt steps of M_α is at least $p - 1/2^k$, where $t = 2^n(2^n)2^n$. The structure of the proof is just as in Section 2.

Lemma 3.1 *Let $S \subseteq [n-1]$ and suppose that n is reachable from every j in S in k steps on tree $\alpha \in T(A,B)$. Then for any vector $\bar{x} \in [0,1]^n$, $(\bar{x}\alpha^{(k)})_n \geq \bar{x}_n + w_S(\bar{x})/(2^n)^k$.*

Proof: The proof is identical to Lemma 2.1, except in showing that all the entries of $\alpha^{(k)}$ are $\geq 1/(2^n)^k$. Again a straightforward proof by induction proves this. □

We define a set $S \subseteq [n-1]$ is said to be k-safe if for all trees α, there is some $i \in S$ such that n is reachable from i on α. We say the set S is safe if S is k-safe for some k.

Lemma 3.2 *If a set S is safe then it is 2^n-safe.*

Proof: Fix some safe set S. For each tree α and each node η of α we define the set $reachable(\eta) \subseteq [n]$ to be $\{j \in [n] | (\bar{x}(\eta, i))_j > 0$ for some $i \in S\}$. That is, j is in $reachable(\eta)$ if and only if j is reachable at η from some $i \in S$. Clearly n is reachable from i in k steps on α if and only if $n \in reachable(\eta)$ for some η at level k of α.

Now suppose that S is not 2^n-safe. Then for some α, for all nodes η of α in levels $0, \ldots, 2^n$, $n \notin reachable(\eta)$. We construct an infinite tree α' such that for all $i \in S$, n is not contained in any set labeling a node of α', proving that S is not safe.

We first prune α to obtain a finite tree that will be used as a template in the construction of α'. An example of such a template is given in Figure 2 following this lemma. Let $prune(\alpha)$ be any subtree of α with the following properties: (i) if η and η' are internal nodes of $prune(\alpha)$ then $reachable(\eta) \neq reachable(\eta')$; (ii) if η is a leaf of $prune(\alpha)$ then $reachable(\eta) = reachable(\eta'')$ for some internal node η'' of $prune(\alpha)$; and (iii) $prune(\alpha)$ is a full tree, that is, each node is either a leaf or has two children. A tree $prune(\alpha)$ exists whose depth is at most 2^n. This is because if $\eta_0, \ldots \eta_{2^n}$ are nodes forming a path of α starting at the root, then for two distinct nodes η_i, η_j on the path, $reachable(\eta_i) = reachable(\eta_j)$ by the pigeon-hole principle.

Now define α' as follows. Let $matrix(\text{root}(\alpha'))$ be $matrix(\text{root}(\alpha))$. For any node η' in α' suppose that $reachable(\eta')$ is the set I. Let η be the unique internal node of $prune(\alpha)$ such that $reachable(\eta) = I$ and let η_0 and η_1 be the left and right children of η. Then if η'_0 and η'_1 are the left and right children of η', define $matrix(\eta'_0) = matrix(\eta_0)$ and $matrix(\eta'_1) = matrix(\eta_1)$. From the definition of $reachable()$, $reachable(\eta'_l) = reachable(\eta_l)$ for $l = 0, 1$. By the construction of $prune(\alpha)$, there are internal nodes η_i, η_j in $prune(\alpha)$ such that $reachable(\eta'_0) = reachable(\eta_i)$ and $reachable(\eta'_1) = reachable(\eta_j)$. From this it follows that α' is well defined.

Also, for all nodes η' in tree α', $n \notin reachable(\eta')$. Hence n is not reachable on α' from any $i \in S$ and so S is not safe. This contradiction proves the lemma. □

To illustrate the proof of this lemma, we consider the example of Figure 1. Denote by L the left child of the root of the tree of Figure 1. Note that in the subtree rooted at L, the probability of reaching the halting state 4 is 0. Also the probability of being in states 1 and 2 at L is > 0. Let $S = \{1, 2\}$; we show that S is not safe. The tree of Figure 2 is derived directly from the left subtree of Figure 1 and is a template for the construction of an infinite tree α' such that n is never reached from set either state 1 or state 2 on α'. Each node η of the tree is labeled by $feedback(\eta)$. Note that all internal nodes have a unique label and all leaves are labeled by a set that also labels an internal node.

Let $p_{\alpha,i}$ denote $(\bar{e}^1 \alpha(i2^n))_n$. Thus $p_{\alpha,i}$ is the probability that M_α halts, that is, reaches state n from state 1, in exactly $i2^n$ steps. Recall that p_α denotes the halting probability of M_α . Therefore, $p_\alpha = \lim_{i \to \infty} p_{\alpha,i}$.

Lemma 3.3 *Let the halting probability of $\mathcal{M}$ be p and for each α let $p_{\alpha,i}$ be the probability that M_α halts in at most $i2^n$ steps. Let $\mu = (2^n)^{2^n}$. Then for any α and any $i \geq 0$,*

$$p_{\alpha,i+1} - p_{\alpha,i} \geq (p - p_{\alpha,i})\mu.$$

Proof: As in Lemma 2.3, if $p - p_{\alpha,i} \leq 0$ the inequality is trivially true so we restrict our attention to the case where $p - p_{\alpha,i} > 0$.

Let $\eta_1, \ldots, \eta_m$ be the nodes at level $i2^n$ of α. For $1 \leq i \leq m$ let α'_i be the tree rooted at η_i. Also let S^i be the set of states that are reachable at η_i from state 1 in $i2^n$ steps of α and which reach the halting

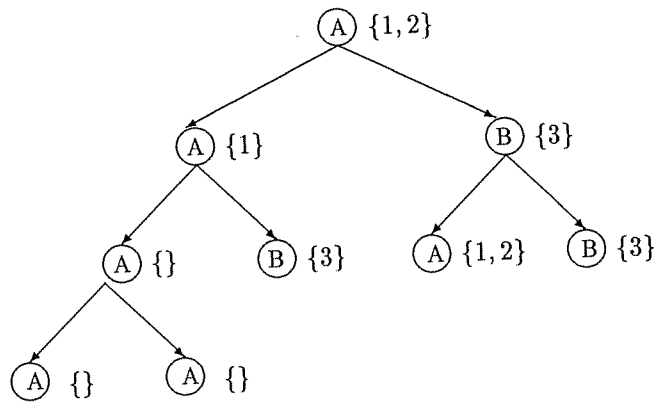


Figure 2: *Template that can be used to construct an infinite tree α' such that state 4 is not reachable from either of states 1,2 on α'.*

state n in a further 2^n steps. Thus,

$$S^i = \{j \in [n-1] \mid (\bar{x}(\eta_i, 1))_j > 0 \text{ and } (\bar{e}^j(\alpha_i')^{(2^n)})_n > 0\}.$$

Let $\bar{S}^i = \{j \in [n-1] \mid (\bar{x}(\eta_i, 1))_j > 0 \text{ and } (\bar{e}^j(\alpha_i')^{(2^n)})_n = 0\}$. Then $\bar{S}^i$ is not 2^n-safe. This is because on tree α_i', no state in $\bar{S}^i$ reaches the halting state in 2^n steps. Then from Lemma 3.2, $\bar{S}^i$ is not safe. Therefore, for each $i, 1 \leq i \leq m$, since $\bar{S}^i$ is not safe, there must exist a tree α_i'' such that n is not reachable from any state in $\bar{S}^i$ on α_i''.

In the rest of this proof, let $\bar{x}^i$ denote $\bar{x}(\eta_i, 1)$. We claim that $\sum_{i=1}^m w_{\bar{S}^i}(\bar{x}^i) < 1 - p$. Suppose to the contrary that $\sum_{i=1}^m w_{\bar{S}^i}(\bar{x}^i) \geq 1 - p$. Then on the tree obtained from α by replacing the tree rooted at η_i by α_i'', $1 \leq i \leq m$, the probability that the halting state n is eventually reached from state 1 is $< p$, contradicting the fact that p is the halting probability of $\mathcal{M}$. Therefore,

$$\sum_{1 \leq i \leq m} w_{S^i}(\bar{x}^i) = 1 - \sum_{1 \leq i \leq m} w_{\{n\}}(\bar{x}^i) - \sum_{1 \leq i \leq m} w_{\bar{S}^i}(\bar{x}^i) \geq 1 - p_{\alpha,i} - (1-p) = p - p_{\alpha,i}.$$

From Lemma 3.1, for all i, $(\bar{x}^i(\alpha_i')^{(2^n)})_n \geq \bar{x}_n^i + w_{S^i}(\bar{x}^i)/(2^n)2^n$. Hence,

$$\sum_{1 \leq i \leq m} (\bar{x}^i(\alpha_i')^{(2^n)})_n \geq \sum_{1 \leq i \leq m} \bar{x}_n^i + \sum_{1 \leq i \leq m} w_{S^i}(\bar{x}^i)/(2^n)2^n.$$

To complete the proof, note that $p_{\alpha,i+1} = \sum_{1 \leq i \leq m}(\bar{x}^i(\alpha_i')^{(2^n)})_n$, $p_{\alpha,i} = \sum_{1 \leq i \leq m}(\bar{x}^i)_n$ and as we have just shown, $\sum_{1 \leq i \leq m} w_S(\bar{x}^i) \geq p - p_{\alpha,i}$. Substituting all of these values into the last inequality, $p_{\alpha,i+1} - p_{\alpha,i} \geq (p - p_{\alpha,i})/\mu$, as required. □

Lemma 3.4 . *Let the halting probability of $\mathcal{M}$ be p and for every α let the probability that M_α halts in at most $i2^n$ steps be $p_{\alpha,i}$. Then $p - p_{\alpha,i} \leq (1 - 1/\mu)^i$, where $\mu = (2^n)2^n$.*

Also for any integer $k > 0$, the probability that M_α halts in at most $k2^n\mu$ steps of M_α is at least $p - 1/2^k$ for all α.

Proof: The proof is identical to the proof of Lemma 2.4. □

To complete this section, we state our main result for feedback chains.

Theorem 3.1 *Suppose $\mathcal{M}$ is a family of n-state feedback chains with halting probability p. Then for any chain in $\mathcal{M}$, the probability of halting in $2^{2^{O(n)}}$ steps is at least $p - o(1)$. Moreover, this bound is the best possible.*

4 Space Bounded Interactive Proofs

We apply the results of the previous sections to obtain upper bounds on the power of space bounded interactive proof systems. The following definition of an interactive proof system is similar to that used by Dwork and Stockmeyer [4]. An interactive proof system consists of a prover P and a verifier V. The verifier is a probabilistic Turing machine with a 2-way, read-only input tape, a read-write work tape and a source of random bits (a coin). The states of the verifier are partitioned into reading and communication states. In addition, the Turing machine is augmented with a special communication cell that allows the verifier to communicate with the prover.

A transition function describes the one-step transitions of the verifier. Whenever the verifier is in a reading state, the transition function of the verifier determines the next configuration of the verifier, based on the symbol under the tape heads, the state and the outcome of an unbiased coin toss. Whenever the verifier is in a communication state, the next configuration is determined as follows. Associated with each communication state is a symbol; without loss of generality we assume that the set of such symbols is $\{0,1\}$. When in communication state c, the verifier writes the symbol associated with c in the communication cell and in response, the prover writes a symbol in the cell. Based on the state and the symbol written by the prover, the verifier's transition function defines the next state of the verifier.

The prover P is specified by a prover transition function. This function determines what communication symbol is written by the prover in response to a symbol of the verifier, based on the input and the sequence of all past communication symbols written by the verifier. Without loss of generality we assume that all symbols written by the prover in the communication cell are from the set $\{a,b\}$ and that the input alphabet is Σ. Thus the prover's transition function is a mapping from $\Sigma^* \times \{0,1\}^*$ to $\{a,b\}$. (Unlike the definition of Dwork and Stockmeyer, the prover's transition function is deterministic, not probabilistic. This does not weaken the model; see Condon [2]).

Fix an input x. The probability that (P,V) accepts (rejects) x is the limit as $k \rightarrow \infty$ of the probability, (taken over all coin tosses of the verifier), that (P,V) reaches the accepting (rejecting) state on x in k steps.

The prover-verifier pair (P,V) is an interactive proof for L with error probability $\epsilon < 1/2$ if

- for all $x \in L$, the probability that (P,V) accepts x is $> 1-\epsilon$,
- for all $x \notin L$, and all provers P^*, the probability that (P^*,V) rejects x is $> 1-\epsilon$.

Define an interactive proof system to be *one-way* if the verifier writes the same symbol in the communication cell for all communication states. Thus in a one-way proof the kth symbol written in the communication cell by the prover is just a function of k and the input.

An interactive proof system (P,V) is said to be $s(n)$ space bounded if for all provers P^*, the number of work tape cells read or written by the verifier is $O(s(n))$, on any input of length n. If the number of work tape cells used by the verifier is $O(1)$, the verifier is a probabilistic 2-way finite state automaton

(2pfa). We denote the class of languages accepted by interactive proof systems that are $O(s(n))$ space bounded by $IP(space(s(n))$. To be consistent with the notation of Dwork and Stockmeyer, we denote $IP(space(O(1))$ by *IP(2pfa)*.

For convenience we assume that the initial, accept and reject states of the interactive proof are communication states and that whenever the verifier enters the accept and reject state it never leaves that state. We also assume that the number of communication states is exactly half the total number of states. Just as for Turing machines, a configuration of an interactive proof system for a fixed input is a tuple containing an encoding of the work tape, the positions of the tape heads on the input and work tapes, the state and the contents of the communication cell. If an interactive proof system (P, V) is $s(n)$ space bounded the length of any configuration of (P, V) is $\log n + O(s(n))$. This is because $\log n + O(1)$ space is required to write the position of the input tape head, the state and the contents of the communication cell and $O(s(n))$ space is required to write the work tape contents. Hence the number of configurations of (P, V) is $2^{\log n + O(s(n))}$.

In the next lemma we describe the relationship between space bounded interactive proof systems and feedback chains.

Lemma 4.1 *Let (P, V) be an $s(n)$ space bounded interactive proof system and let x be an input of length n. Then for some $m = 2^{\log n + O(s(n))}$, there exist $m \times m$ matrices A and B, whose entries are rational numbers of the form p/q where $p \leq q \leq 2^m$, such that the family $\mathcal{M}$ of feedback chains over $\{A, B\}$ has the following properties.*

There is a 1-1 correspondence between the set of all pairs (P^*, V) and the chains in $\mathcal{M}$, such that if (P^*, V) corresponds to M^*, then the probability that (P^*, V) enters a communication state k times, with the kth configuration equal to the rejecting configuration, equals the probability that the M^* reaches its halting state in k steps.

Proof: We call a configuration of (P, V) that contains a communication state or reading state a communication configuration or reading configuration, respectively. Without loss of generality, we assume that the number of configurations of (P, V) on x is $2(m-1)$ for some $m = 2^{\log n + O(s(n))}$, where $m-1$ are communication configurations and $m-1$ are reading configurations. Number the communication configurations $\{1, \ldots, m-1\}$. Assume that 1 is the initial configuration.

We define a family of feedback chains $\mathcal{M}$ with m states, $m-1$ of which correspond to the communication configurations of V. We first define the two $m \times m$ matrices A and B. For $i, j < m$ let p_{ija} be the probability of reaching configuration j from configuration i of V when the symbol a has just been written by a prover in the communication cell. Note that this probability is completely determined by i, j, a and the transition function of V. Also, we define p_{ima} to be the probability that V never reaches a communication configuration from i, that is, the probability that the verifier loops forever in states that are not communication states. Thus, for $i \neq m$ let $p_{ima} = 1 - \sum_{k \neq m} p_{ika}$, let $p_{mma} = 1$ and for $i \neq m$ let $p_{mia} = 0$. Let $A = [p_{ija}]$. Define $B = [p_{ijb}]$ similarly, replacing a everywhere by b.

All entries in A and B can be written as rational numbers of the form p/q where $p \leq q \leq 2^m$. This is because the computation of the verifier between two successive times it enters a communication state can be modeled by a stationary Markov chain, as follows. We describe this chain as a directed graph with probabilities labeling the edges. The graph has $2m-1$ vertices, corresponding to the states of the chain. Of these, $2(m-1)$ correspond to the configurations of V on x. The communication configurations of V are the halting states of the chain; that is, each has an edge labeled with probability 1 to itself. In addition there is one extra halting state and all states from which there is no path to a communication configuration have one edge labeled 1 to this additional halting state. Each other state corresponds to

a reading configuration from which a communication configuration is reachable and has two outgoing edges, to the possible configurations reachable from it in 1 step. The edges of the graph are labeled from the set $\{0, 1/2, 1\}$ and represent transitions. Therefore, the graph has m halting states and $m-1$ other states. It is well known that for any state in this chain, the probability of eventually reaching any halting state from any state can be represented as the quotient of two integers p, q where $p \leq q \leq 2^m$. (See Gill [7]) for one explanation of this).

The state of $\mathcal{M}$ that corresponds to the rejecting configuration of V is the halting state. Label this state of $\mathcal{M}$ the *reject* state. We now describe a 1-1 correspondence between the pairs (P^*, V) and the feedback chains in $\mathcal{M}$. To do this, it is sufficient to relate each pair (P^*, V) to a tree $\alpha \in T(A, B)$. Let δ^* be the transition function of P^* and let α^* be the infinite binary tree in $T(A, B)$ such that for all $\eta \in \alpha^*$,

$$matrix(\eta) = \begin{cases} A, & \text{if } \delta^*(x, feedback(\eta)) = a, \\ B, & \text{if } \delta^*(x, feedback(\eta)) = b. \end{cases}$$

Since δ^* is deterministic, α^* is well defined.

Let $M^* = M_{\alpha^*} = X_1^*, X_2^*, \ldots, X_k^*, \ldots$. From the definition of A and B, it is straightforward to show by induction on k that for $j < m$, the probability that $X_k^* = j$ is the probability that (P^*, V) enters k communication configurations and that the kth communication configuration is j. Similarly, the probability that X_k^* is m is the probability that (P^*, V) never enters k communication configurations on x. From this it follows that the probability (P^*, V) enters a communication state k times and rejects x equals the probability that the kth random variable of chain M^* is the reject state. □

As a special case of Lemma 4.1, there is a 1-1 correspondence between *one-way* interactive proof systems with space bounded verifiers on a fixed input and the family of time-varying Markov chains $\mathcal{M}$ for some matrices A and B. This follows from the fact that in one-way interactive proof systems, the kth symbol sent by the prover to the verifier is a function only of the input and k.

In the next theorem we use Lemma 4.1 and our results on feedback chains of Section 3 to bound the running time of a space bounded interactive proof system on inputs it rejects. We prove this and the other theorems of this section for *2pfa* verifiers, but they all generalize immediately to other space bounded verifiers.

Theorem 4.1 *Let (P, V) be an interactive proof for language L with error probability $\epsilon < 1/2$, where V is a* 2pfa *verifier. Let x be any string of length n that is* not *in L. Then for some $t = 2^{2^{O(n)}}$, for any prover P^* and any $k > 0$, the probability that (P^*, V) rejects x in $k^2 t$ steps is at least $(1-\epsilon) - 2^{-k} - 1/k$.*

Proof: Fix any input x of length n such that $x \notin L$. From Lemma 4.1, for some $m = 2^{\log n + O(1)} = O(n)$, there is a 1-1 correspondence between the pairs (P^*, V) and a family $\mathcal{M}$ of feedback chains, where A and B are $m \times m$ matrices with entries of the form p/q, $p \leq q \leq 2^m$.

Moreover, there is a halting state labeled *reject* of $\mathcal{M}$ such that for all provers P^*, if (P^*, V) corresponds to M^* then the probability that (P^*, V) enters a communication state k times and the kth communication configuration is rejecting, equals the probability that M^* reaches the reject state in k steps. Define the halting probability of $\mathcal{M}$ to be the probability of reaching the reject state. Since for every P^*, (P^*, V) rejects x with probability $> 1 - \epsilon$, the halting probability of $\mathcal{M}$ is at least $1 - \epsilon$. We now apply Lemma 3.4 to $\mathcal{M}$. Let $t' = 2^m(2^m)2^m$, so that $t' = 2^{2^{O(n)}}$. Then the probability of reaching the state labeled reject in at most kt' steps of any M^* is at least $(1 - \epsilon) - 1/2^k$.

Hence from Lemma 4.1, for any P^*, the probability that (P^*, V) enters a communication state kt' times, and the kt' configuration is the rejecting configuration, is at least $(1 - \epsilon) - 2^{-k}$. If (P^*, V)

reaches a communication configuration C_2 from communication configuration C_1 on some computation, via transitions only through reading configurations, then the expected time for (P^*, V) to reach C_2 from C_1 is at most $2^{O(n)}$, for any pair (C_1, C_2). Hence the expected time for (P^*, V) to reject x with probability $(1-\epsilon) - 1/2^k$ is $kt'2^{O(n)}$. Then from Markov's inequality, in $k^2t'2^{O(n)}$ steps of (P^*, V), the rejecting state is reached with probability at least $(1-\epsilon) - 1/2^k - 1/k$. Choose $t = 2^{2^{O(n)}}$ so that for sufficiently large n, $t \geq t'2^{O(n)}$. Then t satisfies the theorem. □

Theorem 4.2 *IP(2pfa)* $\subseteq$ ATIME($2^{2^{O(n)}}$).

Proof: Let (P, V) be an interactive proof for language L with error probability $\epsilon < 1/2$. Let $t = 2^{2^{O(n)}}$ be the constant of Theorem 4.1 and let k be a constant such that $(1-\epsilon) - 1/2^k - 1/k > 1/2$. We describe an interactive proof system (P, V') that accepts L with error probability $\epsilon + 1/2^k + 1/k$ and runs in time k^2t. From this the result follows easily, by applying a result of Condon and Ladner [3], that the class of languages accepted by interactive proof systems that are $t(n)$ time bounded is contained in the class *ATIME(poly(t(n)))*.

The verifier V' simulates the transitions of V, but in addition it keeps a count of the number of steps V has taken at each point of the computation. If V takes k^2t steps without ever halting, V' halts the computation and accepts. Otherwise, if the computation halts within k^2t steps, V' halts and accepts if and only if V does.

Clearly (P, V') runs in time $2^{2^{O(n)}}$. We now show that (P, V') and (P, V) accept the same language. Fix an input x. If x is in L, then the probability (P, V') accepts x is at least the probability that (P, V) accepts x, since V' always accepts if it halts the computation before V halts. Hence (P, V') accepts x with probability at least $1 - \epsilon$, If x is not in L, then from Theorem 4.1, with probability $\epsilon - 1/2^k - 1/k$ V rejects within k^2t steps on all provers P^*, and hence so does V'. Hence (P, V') accepts L with error probability $\epsilon - 1/2^k - 1/k$, as required. □

The following theorem states the upper bound for more general space bounded interactive proof systems. The proof of this is identical to the proof of Theorem 4.2.

Theorem 4.3 *For any space constructible function* $s(n) \geq \log n$, *IP(space(s(n))* $\subseteq$ ATIME($2^{2^{S(n)}}$), *where* $S(n) = 2^{O(s(n))}$.

We can use the fact that one-way interactive proof systems correspond to time-varying Markov chains to strengthen Theorem 4.2 for one-way proofs.

Theorem 4.4 *One-way-IP(2pfa)* $\subseteq$ NTIME($2^{2^{O(n)}}$).

Proof: We describe a nondeterministic Turing machine M' that accepts the same language as (P, V).

Fix an input x of length n and let A and B be the matrices defined in Lemma 4.1 where $m = O(n)$. From the transition function of V, M' first computes A and B. Let $t = 2^{2^{O(n)}}$ be the constant of Theorem 4.1. M' guesses a string $\alpha_1, \ldots \alpha_{tk}$ where each $\alpha_i \in \{A, B\}$. Then M' computes the product $\bar{e}^1\alpha^{(tk)}$. If the component of $\bar{e}^1\alpha^{(tk)}$ labeled reject is $< 1/2$, M' accepts, else it rejects.

We first argue that M' and M accept the same language. If x is rejected by (P, V) then from Theorem 4.1, for all strings α of length kt, the probability of reaching the reject state in kt steps is $> 1/2$. Hence, for all guesses of M', M' rejects. On the other hand, if x is accepted by (P, V) then on some string

α, the reject state is reached with probability $< 1/2$. If α' is the prefix of this string of length kt, M' accepts when it guesses α'.

It remains to show that M' runs in time $2^{2^{O(n)}}$. The entries of the matrices A and B can be written as the quotient of two integers of value $\leq 2^{O(n)}$ and can be constructed in time $2^{O(n)}$. The time required by M' to guess the string α is $2^{2^{O(n)}}$ since this is the size of the tree. The vector $\bar{e}^1\alpha^{(tk)}$ is the product of kt matrices from the set $\{A, B\}$, whose entries can be represented in binary as the quotient of two numbers of length $O(n)$. Hence the entries of the vector can be represented as the quotient of two binary numbers of length $2^{2^{O(n)}}$, and the vector can be computed in time $2^{2^{O(n)}}$. Hence the total running time of M' is $2^{2^{O(n)}}$. □

To complete this section, we show that the bounds of Theorem 4.1 are fairly tight, by constructing an interactive proof system with a *2pfa* verifier (P, V) that accepts the empty set but runs in expected time $2^{2^{\Omega(n)}}$ on an input of length n. Moreover, the interactive proof *halts* with probability 1; that is, on all inputs x and all provers P^* on x, the probability that (P^*, V) reaches either the accept or the reject state is 1. The protocol followed by the prover and verifier which causes the game to run for this time is a counting protocol. Similar protocols have been used previously by Peterson and Reif [11] and in a multi-prover setting by Feige and Shamir [5]. The interactive proof we construct is one-way.

Theorem 4.5 *There is a one-way interactive proof system with a 2pfa verifier that halts with probability 1 and whose expected running time on an input of length n is $2^{2^{\Omega(n)}}$.*

Proof: We describe an interactive proof system (P, V) that accepts the empty set. Informally on an input of length n, the pair (P, V) repeatedly executes a protocol which we call the *interactive count* protocol. The idea is that in each iteration of this protocol, the verifier tosses 2^n coins and halts in the reject state at the end of that iteration if all coin tosses are heads. Thus, the expected time for the interactive proof to halt is double exponential in n.

In describing the interactive count protocol and other protocols of Section 5, we use the expression "V requests a symbol" from a prover to mean that V enters a communication state and writes a symbol in the communication cell. (In the case of one-way proofs, the same symbol is written in the communication cells every time the verifier enters a communication state.) Similarly we use the expression "V receives a symbol" from a prover to mean that the prover writes the symbol in the communication cell. Let P be the prover defined by the string $(\$N_1\$N_2\$\ldots\$N_{2^n}\$)^\omega$, where N_i is the number i in binary, padded with 0's on the left to be of length exactly n. That is, the kth symbol V receives from P is the kth symbol of this string. (To simplify the description of the protocol, we assume that the prover's alphabet has three symbols).

While receiving the prover's string via the communication cell, the verifier performs some checks that the prover's string is of the form $(\$N_1\$N_2\$\ldots\$N_{2^n}\$)^\omega$. If any of the checks reveal an error, the verifier halts and rejects. In this way we can prove that for all provers P^*, the probability of eventually halting is 1. The checks performed at random times by the verifier are as follows:

- *check-length*, which is started just after the prover sends a \$ and, using the input as a ruler, checks that the next \$ occurs after exactly n symbols from $\{0, 1\}$ have been sent by the prover. If not, the verifier halts in a reject state.
- *check-bit*, which selects a random bit between two \$'s, say the jth bit of the binary number after the ith \$ symbol and checks that the jth bit of the number after the $(i+1)$st \$ is correct. To choose a bit randomly, the verifier tosses a coin for every bit sent by the prover from the time the

check-bit procedure is called, until the coin-toss is heads at some bit or the prover sends a \$. In the latter case the verifier checks bit n of N_{i+1}. To check the correctness of the bit chosen, the verifier does the following.

– V stores the value of the jth bit of N_i on its work tape. Call this bit b_j.

– V initializes a binary value B to 0 and changes it to 1 if any bit $k < j$ of N_i is 0. This can be done simply by observing all the bits sent by the prover before the next \$, since the bits of lower order than j are sent after j by the prover.

– Using the input as a ruler, V counts until the prover has sent $n+1$ bits and stores the $(n+1)$st bit on its work tape. Let this bit be b'_j. V also checks that a \$ appears during that time. If not, the verifier halts in a reject state.

– Otherwise, V checks that if $B = 1$ then $b'_j = b_j$ and if $B = 0$ that b'_j equals the complement of b_j. If not, the verifier halts in a reject state.

While executing the interactive count protocol, the verifier tosses a coin each time a \$ is sent by the prover. When the prover sends a string of all 1's between two \$'s, the verifier terminates that iteration of the interactive count protocol. If all coins tossed during that iteration are heads, the verifier halts and rejects; otherwise the verifier starts another iteration of the above protocol. Since both checks use the input as a ruler, only one can be performed at any time. Therefore the verifier initially chooses one of the two checks at random. Whenever it completes a check successfully, it randomly chooses one of the two checks again and repeats this until it detects an error or the protocol ends.

We claim that the expected running time of pair (P, V) is at least 2^{2^n}. This is true because the only way (P, V) can halt is if it tosses 2^n heads in one round of the protocol and the chance of this occurring is $1/2^{2^n}$.

Secondly, we need to show that for all provers P^*, (P^*, V) eventually halts with probability 1. Define a string $\$x_1\$x_2\$\ldots\$x_i\$\ldots$ to be *valid* if all the x_i's are of length n and $x_{i+1} = x_i + 1 (\bmod 2^n)$. Similarly, define a string $\$x_1\$x_2\$\ldots\$x_i\$\ldots$ to be *invalid at* i if all the x_i's are of length n and $x_{i+1} \neq x_i + 1 (\bmod 2^n)$ for all i.

For any prover P^*, the string sent by P^* to V must satisfy one of four properties. For each property, we argue that (P^*, V) eventually halts. (i) If a finite prefix is removed, the remaining string is valid. In this case the verifier will halt in expected time double exponential in n from the time the prover starts to send the correct part of the string. (ii) If a finite prefix is removed, the remaining string contains no \$'s; in this case, both *check-length* and *check-bit* will cause the verifier to halt. (iii) Infinitely many x_i's are not of length n. In this case, the *check-length* procedure will cause the verifier to halt. (iv) The string is invalid at infinitely many i. In this case, the *check-bit* procedure will eventually cause the verifier to halt. □

5 Multi-Prover Interactive Proof Systems

In this section, we contrast the power of single-prover and multi-prover interactive proof systems. We show that the class of 2-prover interactive proof systems with *2pfa* verifiers accept exactly the set of recursive languages.

The definition of multi-prover interactive proof systems is a straightforward generalization of single-prover interactive proof systems. Briefly, a multi-prover interactive proof system consists of a finite

number of provers, $P_1, P_2, \ldots, P_k$ and a verifier V. The verifier is just as in the single-prover model, except that it has k communication cells, one per prover. Also the communication states of the verifier are partitioned into k groups. Whenever the verifier's state is a communication state in the ith group, the next configuration is determined by communicating with the ith prover as in the single-prover model.

Each prover P_i is specified by a prover transition function that determines what communication symbol is written by P_i in response to a symbol of the verifier. The symbol written by P_i at the ith step is based on the input and the sequence of all past communication symbols written by the verifier in the ith communication cell. Thus the transition function of each prover is a mapping from $\Sigma^* \times \{0,1\}^*$ to $\{a, b\}$. The probability that a multi-prover interactive proof system reaches the accept or reject state and the probability that it halts are defined just as for single-prover interactive proof systems.

The system $(P_1, \ldots, P_k, V)$ is an interactive proof for L with error probability $\epsilon < 1/2$ if

- for all $x \in L$, the probability that $(P_1, \ldots, P_k, V)$ accepts x is $> 1 - \epsilon$,
- for all $x \notin L$ and all $P_1^*, \ldots, P_k^*$, the probability that $(P_1^*, \ldots, P_k^*, V)$ rejects x is $> 1 - \epsilon$.

We first show that any language accepted by a multi-prover interactive proof system is recursive.

Theorem 5.1 *Any language accepted by a multi-prover interactive proof system is recursive.*

Proof: Let L be accepted by a k-prover interactive proof system $(P_1, \ldots P_k, V)$ with error probability $\epsilon < 1/2$. We describe a nondeterministic Turing machine, M, that accepts L and has the property that for all $x \notin L$, all possible computations of M on x are finite. From this it follows that L is recursive.

The construction of M is as follows. On any input x, M nondeterministically simulates some interactive proof system $(P_1^*, P_2^*, \ldots, P_k^*, V)$ on x. The simulation proceeds in rounds. At the tth round, M nondeterministically writes down the transition function of each prover on strings of length t. Then M computes the probability that V reaches the accept and reject states in exactly t steps with these provers. This probability can be computed just as in the single-prover case. If V reaches the accept state on x with probability $> 1/2$ in t steps, then M halts in an accept state. Similarly, if V reaches the reject state on x with probability $> 1/2$, then M halts in a reject state. Otherwise M continues to round $t + 1$.

We first show that if $x \in L$ then on some computation of M on x, the accept state is reached. By definition, if $x \in L$ then the limit as $t \to \infty$ of the probability that $(P_1, \ldots P_k, V)$ reaches an accept state on x in t steps is $> 1/2$. Hence for some t, with probability $> 1/2$ $(P_1, \ldots P_k, V)$ reaches an accept state on x in t steps. Furthermore, since V never leaves a reject state once it reaches it, there cannot exist a t' such that the probability that $(P_1, \ldots P_k, V)$ reaches a reject state on x in t' steps is $> 1/2$. Hence on the computation of M that simulates $(P_1, \ldots P_k, V)$ on x, M halts in an accept state at the tth round and so M accepts x.

Next suppose that $x \notin L$ and let $P_1^*, \ldots, P_k^*$ be any provers. Then the limit as $t \to \infty$ of the probability that $(P_1^*, \ldots P_k^*, V)$ reaches a reject state on x in t steps is $> 1/2$. Hence there is some t such that with probability $> 1/2$ $(P_1^*, \ldots P_k^*, V)$ reaches a reject state on x in t steps. Since in all computations of M, M simulates some provers $P_1^*, \ldots, P_k^*$, M halts in a reject state in a finite number of steps on all computations. □

In Theorem 5.2, we prove that any recursive language is accepted by some multi-prover interactive proof system. We use the following lemma, due to Feige and Shamir [5]. Our proof of this is modified slightly from theirs.

Lemma 5.1 *For any recursive language L, there is a 2-prover interactive proof with a* 2pfa *verifier (P_1, P_2, V) with the following properties.*

(i) For all $x \in L$, (P_1, P_2, V) accepts x with probability 1.

(ii) For all $x \notin L$, all (P_1^*, P_2^*, V) reject x with probability $\geq 1/2$.

Proof: Let L be a recursive language and let M be a 1-tape deterministic Turing machine accepting L. For any input x of M, we represent the computation of M on x by a string $\$C_1\$C_2\$\ldots\$C_m\$$. Each C_i represents a configuration of M and is of the form $v_i q_i w_i$ where q_i is the state of M at the ith step on x, $v_i w_i$ is the contents of the work tape and the head is positioned on the leftmost symbol of w_i. We also assume that the initial configuration is of length $|x|+1$ and that the ith configuration is of length $|x|+i+1$, by padding the right end of the configuration with blanks if necessary. Thus $C_1 = q_0 x$, where q_0 is the initial state of M. We write $C \rightarrow C'$ if there is a transition of M from C to C'.

We describe a 2-prover interactive proof system (P_1, P_2, V) that accepts L. The proof system we define is one-way; thus for a fixed input x each prover can be fully specified by a string. On input x, we define provers P_1 and P_2 to correspond to the string $\$C_1\$C_2\$\ldots\$C_m\$$ representing the computation of M on x. In the rest of the proof we equate a prover with the string corresponding to it.

Next we describe the protocol of the verifier V. Fix an input x. The verifier performs one of two checks, to verify that both provers' strings on x equal the computation of M on x. Initially V tosses a coin to decide which check to perform; thus each check is equally likely. The first check is simply that the strings of both provers are equal. A finite state verifier can easily check this by simply requesting symbols from the provers in lock step and comparing the symbols. If both strings are equal, V halts in an accept state; otherwise V halts in a reject state.

In the second check performed by the verifier on input x, if $\$C_1\$C_2\ldots\$C_k\$$ is the computation of M on x then V checks the following conditions.

1. The string of the first prover starts with the string $\$C_1\$$.
2. The last state of the string of the first prover is an accept state.
3. For all i, if $C_i^{(2)}$ is the string after the ith \$ of the second prover's string and before the $(i+1)$st \$ (if any) and $C_{i+1}^{(1)}$ is the string after the $(i+1)$st \$ of the first prover's string and before the $(i+2)$nd \$ (if any) then C_i and C_{i+1} are of the form vqw where q is a state of M and $v, w \in \Sigma^*$. Also, $C_i^{(2)} \rightarrow C_{i+1}^{(1)}$ and $|C_{i+1}^{(2)}| = |C_i^{(1)}| + 1$.

Using the input tape, the verifier can easily check condition (1) – that the string sent by the first prover starts with the string $\$C_1\$ = \$q_0 x\$$, where q_0 is the initial state of M. Also a 2pfa verifier can easily check that condition (2) holds while receiving the symbols from the provers. To check condition (3), V requests symbols from the provers in the following order. V first requests the string $\$C_1\$$ from the first prover. Then V iterates the following. It requests a symbol from the second prover and verifies that it is the \$ symbol. Next, V repeatedly requests one symbol from the first prover and one from the second prover until it receives a \$ from the second prover. Finally it requests a symbol from the first prover and checks that it is the \$ symbol. In this way, V receives the ith configuration of the second prover while it receives the $(i+1)$st configuration of the first prover and can perform check (3).

We now argue that (P_1, P_2, V) satisfies the lemma. Fix an input x and let $\$C_1\$C_2\$\ldots\$C_m\$$ be the computation of M on x. First, suppose that $x \in L$. Then P_1 and P_2 both send the computation of M

on x to V. On either check, V reaches an accept state after a finite number of steps and so V accepts x with probability 1.

Next suppose that $x \notin L$ and let P_1^*, P_2^* be any pair of provers. We must show that (P_1^*, P_2^*, V) reaches a rejecting state with probability at least $1/2$ on x. If the strings corresponding to P_1^* and P_2^* are not equal then this is surely true, since with probability $1/2$, V checks that the provers' strings are equal. Also if P_1's string is $\$C_1\$C_2\$\ldots\$C_m\$$ then V halts in a rejecting state when performing the second check, condition (2). Again V performs this check with probability $1/2$ and so rejects with probability $1/2$. Finally we consider the case when the strings of P_1^* and P_2^* are equal but do not equal the computation of M on x. Let the first symbol of P_1^*'s string that differs from the string $\$C_1\$C_2\$\ldots\$C_m\$$ be after the ith \$ symbol and before the $(i+1)$st \$ (if any). If $i = 1$ then when V performs check (1), it halts in reject state. If $i > 1$ then the string after the $(i-1)$st \$ of P_2's string must start with $C_{i-1}\$$, since P_2's string equals P_1's string and P_1's string is consistent with $\$C_1\$\ldots\$C_m\$$, up to the ith \$ symbol. Thus when V receives the $(i-1)$st configuration of P_2, it halts in a rejecting state when executing the second check, condition (3). Since the second check is performed with probability $1/2$, again V halts in a reject state with probability $1/2$. □

We now show that 2-prover interactive proof systems accept the recursive languages.

Theorem 5.2 *Let L be any recursive language. Then there is a 2-prover interactive proof system with a* 2pfa *verifier that accepts L and halts with probability* 1.

Proof: Informally, we construct an interactive proof (P_1, P_2, V), that repeatedly simulates the protocol of Lemma 5.1, in order to reduce the error probability, and dovetails this with a simple procedure of to guarantee that the interactive proof eventually halts.

On any input x, the verifier V repeatedly simulates some or all of the protocol of Lemma 5.1, which is called the simulation protocol in this proof. V maintains a count, *num-accepts*, of the number of times the simulation protocol reaches the accept state. Initially, *num-accepts* is set to 0. If this count ever reaches k, a constant that depends on the desired error probability, V halts in an accept state. If on any iteration of the simulation protocol, a reject state is reached, V halts in a reject state. We assume that V writes the symbol '0' in the communication cell when executing the simulation protocol, when it requests another symbol from a prover.

To ensure that the protocol eventually halts, V does the following. Each time V requests a symbol from the first prover, it flips a coin. If the outcome of the coin toss is heads, then the verifier restarts the simulation protocol. To do this, V writes the symbol '1' in the communication cell indicating that the simulation protocol is to be restarted. Otherwise the outcome of the coin toss it tails and V continues the execution of the simulation protocol by writing a '0' in the communication cell.

Next we define the provers P_1 and P_2. Both provers are identical, and map the pair $(x, v10^k)$ to the kth symbol of $\$C_1\$\ldots C_m\$$. Thus, each time the verifier restarts the simulation protocol, by writing a 1 on the communication tape, the provers send the symbols of string $\$C_1\$\ldots C_m\$$ until they receive another '1' symbol from the verifier.

Let x be any input. We show that for any provers P_1^* and P_2^*, (P_1^*, P_2^*, V) halts with probability 1 on x. We call any part of the computation of (P_1^*, P_2^*, V) between two steps where V writes the '1' symbol in a communication cell a *round* of the computation. We say that V *makes progress* in a round if the simulation protocol either reaches the accept or the reject state in that round. The length of a round is the number of symbols V requests from the first prover in that round. When the input is x we

say a prover's string is invalid if it is not equal to the string $\$C_1\$ \ldots C_m\$$. Let the length of this string be t.

Clearly the expected length of any round is finite – in fact the expected length is at most 2 steps. Also the probability that a round has length at least t is 2^{t-1}. On a round of length at least t, the probability that V makes progress is $\geq 1/2$ since the string V receives from the provers on such a round must either be invalid or end in an accept or a reject state and in all of these cases, V reaches the accept or reject state with probability at least $1/2$. If V makes progress at most k times it always halts. From these observations it follows that V halts with probability 1 on any input.

When $x \in L$, (P_1, P_2, V) never reaches a reject state since the strings of both provers are valid; therefore (P_1, P_2, V) must reach the accept state with probability 1. Finally, we show that when $x \notin L$, for any P_1^* and P_2^*, the probability that (P_1^*, P_2^*, V) accepts x is at most $1/2^k$. This follows from the fact that on any round, if P^* sends to V a string that contains an accept state, then the string must be invalid, since $x \notin L$. Each time P_1^* sends such a string to V, with probability at least $1/2$ V halts in a reject state. Hence the probability that P_1 sends k such strings to V without V detecting an error is at most $1/2^k$. □

Conclusions and Open Problems

We considered interactive proof systems with a *2pfa* verifier and obtained new upper bounds on the power of single-prover interactive proof systems. We have completely characterized the class of languages accepted by multi-prover systems and have shown them to be much more powerful than the single-prover systems. Our results extend to other types of space bounded verifiers. In proving these results, we also obtained bounds on the rate of halting of time-varying Markov chains and feedback chains.

Some interesting questions remain unresolved. For example, there is a large gap between the best known upper and lower bounds for *IP(2pfa)*; the best known lower bound being deterministic exponential time and the best known upper bound being alternating double exponential time. How can either of these bounds be improved? Also, we do not know the power of *one-way* multi-prover interactive proofs with a *2pfa* verifier. In our proof of Theorem 5.2, which shows that multi-prover systems accept exactly the recursive languages, the multi-prover system we construct is not one-way since the verifier communicates with the prover to tell it when to restart the simulation protocol. Are one-way multi-prover systems less powerful than two-way systems? In another direction, we would like to examine the rate of halting of special types of time-varying Markov chains. We are currently investigating chains that can be represented by an undirected graph; we conjecture that the rate of halting of such chains is at least an exponential faster than in the general case.

References

[1] M. Ben-Or, S. Goldwasser, J. Killian and A. Wigderson, Multi-Prover Interactive Proofs: How to Remove Intractability, Proceedings of the 20th Symposium on the Theory of Computing (STOC), May, 1988, pp 113-131.

[2] A. Condon, Computational models of games, MIT Press, July 1989.

[3] A. Condon and R. Ladner, Probabilistic Game Automata, Journal of Computer and System Sciences, Volume 36, No. 3, June, 1988, pp 452-489.

[4] C. Dwork and L. Stockmeyer, Interactive Proof Systems with Finite State Verifiers, Tech. Report RJ 6262, IBM Research Division, Almaden Research Center, San Jose, CA, 1988.

[5] U. Feige and A. Shamir, Multi-Oracle Interactive Protocols with Space Bounded Verifiers. Proceedings of the conference on Structure in Complexity Theory, June 1989, pp 158-164.

[6] L. Fortnow, J. Rompel and M. Sipser, On the Power of Multi-Prover Interactive Protocols, Proceedings of the conference on Structure in Complexity Theory, 1988, pp 156-161.

[7] J. Gill, Computational Complexity of Probabilistic Turing Machines, SIAM Journal on Computing, Vol. 6, No. 4, 1977, pps 675-695.

[8] S. Goldwasser, S. Micali and C. Rackoff, The knowledge complexity of interactive protocols, Proceedings of 17th Symposium of the Theory of Computing (STOC), 1985, pp 291-304.

[9] J. Killian, Zero Knowledge with Log-Space Verifiers, Proceedings of 29th Symposium on Foundations of Computer Science (FOCS), 1988, pp 25-35.

[10] R. J. Lipton, Recursively Enumerable Languages have Finite State Interactive Proofs, Technical Report Number CS-TR-213-89, Princeton University, 1989.

[11] G. L. Peterson and J. H. Reif, Multiple-Person Alternation, Proceedings of 20th Symposium on Foundations of Computer Science (FOCS), 1979, pp 348-363.