On the Complexity of Space Bounded Interactive Proofs

Anne Condon *
Computer Sciences Department
University of Wisconsin-Madison
1210 West Dayton Street
Madison, WI 53706
condon@cs wisc.edu

Richard J. Lipton †
Computer Science Department
Princeton University
Princeton, NJ 08540

rjl@cs.princeton.edu

November 29, 2005

Abstract

We prove two results on interactive proof systems with 2-way probabilistic finite state verifiers. The first is a lower bound on the power of such proof systems, if they are not required to halt with high probability on rejected inputs: we show that they can accept any recursively enumerable language. The second is an upper bound on the power of interactive proof systems that halt with high probability on all inputs: any language they accept is in ATIME($2^{2^{O(n)}}$). Our results generalize to other space bounds. The proof techniques we develop have other interesting applications. The proof method for the lower bound also shows that the emptiness problem for 1-way probabilistic finite state machines is undecidable. In proving the upper bound, we 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 study the class of *interactive proof systems* (IPS) introduced by Goldwasser, Micali and Rackoff [13]. Roughly, a IPS for a language L is a 2-party protocol between two computational processes, a "prover" P and a "verifier" V. The prover must convince the verifier, which has limited resources, that inputs in L are actually in L. We study the case where the verifier is space bounded, in particular when the verifier is a 2-way probabilistic finite state machine (2pfa), with the ability to flip "private" coins. Prior to this work, no upper bounds on the complexity of this model were known.

Two different definitions of language acceptance for space bounded IPS's have been proposed. In the weaker definition (Condon and Ladner [5]), a prover-verifier pair (P, V) is an IPS 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

*Work supported by NSF grant numbers DCR-8402565 and CCR-9257241 and by a matching grant from AT&T Bell Labs.

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

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

Note that we do not insist that these systems halt with high probability on inputs not in L. Following the notation of Dwork and Stockmeyer [8], the class of languages accepted by IPS's satisfying these conditions is denoted by IP_w(2pfa) when the verifier is a 2pfa. We are able to show that such systems are very powerful.

Theorem (3.1) *Any recursively enumerable language is in IP_w(2pfa).*

The proof of this theorem requires us to extend a result of Frievalds [9]. He shows that the emptiness problem for 2-way probabilistic finite state machines is undecidable. We can use our method to prove:

Theorem (3.2) *The emptiness problem for 1-way probabilistic finite state machines is undecidable.*

The emptiness problem appears to have first been raised in the late 1960's. See for example, Paz [17]. The key insight into this generalization is that it is possible to "simulate" the 2-way input tape by a 1-way input tape. The 1-way tape repeats over and over the contents of the 2-way tape. The critical point is that the 1-way tape can "cheat", i.e. it need not repeat exactly the same string each time. Thus, the 1-way machine must be a kind of verifier and check that no cheating occurs. In this way ideas from IPS's play an essential role in the proof of Theorem 3.2. Since we often measure the importance of a new area by how well it solves existing problems, Theorem 3.2 is evidence for the importance of space bounded IPS's.

A stronger definition of language acceptance for IPS's (Dwork and Stockmeyer [8]) is obtained by replacing the condition 2 above by the following condition.

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

We denote this class by IP(2pfa). The best known lower bound, shown by Dwork and Stockmeyer [8], is that IP(2pfa) contains any language in DTIME($2^{O(n)}$). Since the stronger definition requires that the IPS halt with high probability on all inputs, it is not hard to show that all languages in IP(2pfa) are recursive. Our next result provides the first non-trivial upper bound for this class.

Theorem (6.2) *IP(2pfa) $\subseteq$ ATIME($2^{2^{O(n)}}$).*

In proving Theorem 6.2, we obtain some results of independent interest on the rate of convergence of discrete time-varying Markov chains to their halting (absorbing) states. We consider the family $\mathcal{M}$ of all n-state time-varying Markov chains, all with the same initial state, such that the transition matrices of every chain in the family are from some fixed finite set, say $\{A, B\}$. We assume that all entries of A and B are of the form p/q where p and q are integers, $p \leq q \leq 2^n$.

Theorem (4.1) *Let $\mathcal{M}$ be a family of n-state time-varying Markov chains. Suppose that for all chains M in $\mathcal{M}$, n is a halting state which is eventually reached from the initial state with probability p. Then the probability that M halts in $2^{2^{O(n)}}$ steps is at least $p - o(1)$. Moreover, this bound is the best possible.*

For comparison sake, we mention a well known result for stationary n-state Markov chains under similar conditions: if a halting state is reachable with probability p, then the expected time to reach that state with probability $p - o(1)$ is $2^{O(n)}$.

Theorem 4.1 is related to some previous work on weak ergodicity of families of time-varying Markov chains. Seneta [19] devotes two chapters to the following question: Given an infinite sequence of $n \times n$ stochastic matrices, $M_1, \ldots, M_j, \ldots$, do the rows of the matrix $M^{(j)} = \prod_{i=1}^{j} M_i$ tend to equality as $j \to \infty$? If so, the sequence is weakly-ergodic. In the case that all matrices in the sequence $M_1, \ldots, M_j, \ldots$

are in the set $\{A, B\}$ as above, clearly convergence to a halting state is a special case of weak ergodicity. Motivated by problems from coding theory on finite state channels, Blackwell *et al.* [3], Paz [16] and Wolfowitz [20] studied what conditions on A, B imply weak ergodicity of all infinite sequences over $\{A, B\}$. Also, building on the work we present in this paper, bounds on the rate of convergence to ergodicity for various sets $\{A, B\}$ were obtained in [6].

The rest of the paper is organized as follows. In Section 2, the interactive proof system model is defined precisely. Section 3 contains the lower bound result for $\mathrm{IP}_w(2\mathrm{pfa})$; and the undecidability result for 1-way probabilistic finite state automata. Our results on time-varying Markov chains are presented in Section 4. In order to obtain our results on space bounded IPS's, we extend our results on time-varying Markov chains to non-Markov chains, which we call *feedback* chains in Section 5. These results are used to obtain our upper bound on IP(2pfa) in Section 6. Conclusions and open problems are presented in Section 7.

2 Definitions

The following definition of an interactive proof system is similar to that used by Dwork and Stockmeyer [8]. 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 state of the verifier, based on the symbol under the tape heads, the current 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 current 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 Δ and that the input alphabet is Σ. Thus the prover's transition function is a mapping from $\Sigma^* \times \{0, 1\}^*$ to Δ. (Unlike the definition of Dwork and Stockmeyer, the prover's transition function is deterministic, not probabilistic. This does not weaken the model; see Condon [4]).

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

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

We consider two possible definitions of language acceptance for IPS's. We say that the prover-verifier pair (P, V) is a weak interactive proof for 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) accepts x with probability at most ϵ.

Note that we do not insist that these systems halt with high probability on inputs not in L. A stronger definition of language acceptance for IPS's (Dwork and Stockmeyer [8]) is obtained by replacing the condition 2 above by the following condition.

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

If for some language L, the prover-verifier pair (P, V) satisfies properties 1 and $2'$, we say that (P, V) is an interactive proof system for L with error probability $\epsilon < 1/2$. An interactive proof system (P, V) is $s(n)$ space bounded if for all provers P^*, the number of work tape cells read or written by the verifier is at most $s(n)$, 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, or 2pfa. We denote the class of languages accepted by interactive proof systems that are $O(s(n))$ space bounded by IP(space($s(n)$)). We denote the class of languages accepted by weak interactive proof systems that are $O(s(n))$ space bounded by IP_w(space($s(n)$)). To be consistent with the notation of Dwork and Stockmeyer, we denote IP(space($O(1)$)) and IP_w(space($O(1)$)) by IP(2pfa) and IP_w(2pfa), respectively.

In the following sections, 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 cell 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. We say "V receives a string" from a prover if V receives the symbols of the string, one at a time, from the prover.

3 Lower Bound Results

In this section we show that any recursively enumerable language can be accepted by some IPS with only a 2pfa as a verifier. We first need a technical lemma about conditional probability.

Lemma 3.1 *Let A,B be independent events with* $0 < kPr[B] = Pr[A] < 1$, *where* $k \geq 1$. *Then,*

$$Pr[A|A\bar{B} \vee \bar{A}B] \geq 1 - \frac{1}{k+1}.$$

Proof: Let $\alpha = Pr[A]$ and $\beta = Pr[B]$. Then, $0 < k\beta = \alpha < 1$. Now, since A and B are independent events,

$$\begin{aligned} Pr[A|A\bar{B} \vee \bar{A}B] &= \frac{Pr[A\bar{B}]}{Pr[A\bar{B}] \vee Pr[\bar{A}B]} \\ &= \frac{\alpha(1-\beta)}{\alpha(1-\beta) + (1-\alpha)\beta} \\ &= \frac{1}{1+\delta} \end{aligned}$$

where $\delta = \frac{(1-\alpha)\beta}{(1-\beta)\alpha}$. Since $\alpha = k\beta$ and $k \geq 1$ it follows that

$$1 + \delta = 1 + \frac{(1-k\beta)}{(1-\beta)k} \leq 1 + \frac{1}{k}$$

$$\frac{1}{1+\delta} \geq \frac{1}{1+1/k} \geq 1 - \frac{1}{k+1}.$$

Therefore, $Pr[A|A\bar{B} \vee \bar{A}B] \geq 1 - \frac{1}{k+1}$ as required. □

We will later describe how to construct an IPS with finite state verifier for a recursively enumerable language. In that IPS, on input x, the prover will repeatedly send to the verifier a description of the computation of a counter machine on input x. We now consider a kind of game that our finite state verifiers will use to check the prover. This game is based on a method of Frievalds [9]. We assume that the verifier receives in the communication cell the sequence of symbols in the string $a^i b^j$, from left to right. Fix a large number p. Using a fair coin, the verifier performs five independent computations:

1. For each letter a, the verifier flips two coins.
2. For each letter b, the verifier flips two coins.
3. For each letter a and b, the verifier flips one coin.
4. For each letter a and b, the verifier flips one coin.
5. The verifier checks that $i \equiv j \bmod p$.

The outcome of the game is determined as follows: if step (5) discovers that i and j are not equal modulo p, then the game outcome is *not-equal*. Therefore, assume that this is not the case. Let A be true if either step (1) or step (2) gets only heads; let B be true if either step (3) or step (4) gets only heads. Then say that the outcome of the game is a *tie* if both A and B are false or both are true. Say that the outcome is *double wins* if A is true and B is false; say that the outcome is *sum wins* if B is true and A is false. Say that the outcome is a *win* provided either double or sum wins.

Lemma 3.2 *For any fixed p, the outcome of a game can be computed by a probabilistic finite state machine.*

Proof: Clearly, each of the five subcomputations can be done with a finite number of states. This follows since for each computation, only one bit about the sequence of coin flips needs to be stored, namely whether all the coins are heads. Moreover, the decision on the game's outcome is also finite state, so the claim follows. □

It is convenient to have one more definition concerning games. Say that a game is *fair* if the string is $a^i b^j$ where $i = j$; say it is *unfair* otherwise. The next lemma states that in any fair game, double or sum are equally likely to win; in any unfair game, double is much more likely to win than sum.

Lemma 3.3 *Consider the action of the game on the string $a^i b^j$ with number p and $i \equiv j \bmod p$.*

1. *If $i = j$, then $Pr[\text{double wins}|\text{win}] = Pr[\text{sum wins}|\text{win}]$.*
2. *If $i \neq j$, then $Pr[\text{double wins}|\text{win}] \geq 1 - \frac{1}{2^{p-1}+1}$.*
3. *If $i \neq j$, then $Pr[\text{sum wins}|\text{win}] \leq \frac{1}{2^{p-1}+1}$.*

Proof: First, suppose that $i = j$. Then each of the computations flips exactly the same number of independent coins, i.e. $2i = 2j = i + j$. Thus, by symmetry it follows that (1) is true. Now assume that $i < j$. Now let A denote the event that double wins; also let B denote the event that sum wins. Then,

$$Pr[A] \geq Pr[2i \text{ heads in a row}] = 1/2^{2i}.$$

Also,

$$Pr[B] \leq 2Pr[i + j \text{ heads in a row}] = 1/2^{i+j-1}.$$

Thus, $Pr[A] \geq kPr[B]$ where k is 2^{j-i-1}. Since i is congruent to j modulo p, $j - i \geq p$. So $k \geq 2^{p-1}$. Lemma 3.1 then shows that $Pr[A|A\bar{B} \vee \bar{A}B]$ is at least $1 - \frac{1}{2^{p-1}+1}$. Now assume that $i > j$. The argument follows in exactly the same manner replacing i by j. This proves (2). Finally, (3) follows directly from (2) since either double or sum must win each game that has a winner. □

We are now ready to show that if L is a recursively enumerable language, then L is in $\text{IP}_w(2\text{pfa})$. It is well known that there is a counter machine with two counters that accepts exactly L (see Hopcroft and Ullman [1]). Our counter machine model is standard; a counter machine operates as follows. At each step the machine reads the contents of the read-only input tape and tests whether or not each of its two counters is zero or not. Then based on its current control state it may move the input head left or right; increment a counter; decrement a counter provided it is nonzero; and finally enter its next state. We encode the instantaneous descriptions (ID) of such a machine as follows: $\#uc^k d^l \#$ where u is a symbol that encodes the control state, k is the value of the first counter and l is the value of the second counter.

Theorem 3.1 *For any recursively enumerable language L, L is in $IP_w(2pfa)$.*

Proof: Fix a counter machine that accepts L. We will show that there is an IPS (P, V) that accepts exactly language L. In fact, the IPS is one-way. The idea is that on input x, V repeatedly receives from P the sequence of ID's that correspond to a computation of the counter machine; and V checks the ID's for correctness.

Let the computation be $\#u_1 c^{k_1} d^{l_1} \#, \ldots, \#u_t c^{k_t} d^{l_t} \#$ and denote it by $\mathcal{C}$. Without loss of generality we assume that t is even. The prover P is defined so that V repeatedly receives a copy of this computation. V checks each computation $\mathcal{C}$ as follows. First, V checks that it is "locally" correct. This includes: (i) checking that the initial and final states are correct; (ii) checking that the correct state transitions are taken. All of these checks can be done in a straightforward manner with a deterministic finite state control. If any of these fail, then V rejects.

Second, V checks that the computation $\mathcal{C}$ is "globally" correct. This means that each of the two counters is modeled correctly. Thus, it must be the case that for each $i = 1, \ldots, t - 1$,

$$k_{i+1} = k_i + r_i \quad (1)$$

$$l_{i+1} = l_i + s_i \quad (2)$$

where r_i (resp. s_i) is the action performed on the first (resp. second) counter, i.e. r_i (resp. s_i) is $-1, 0, 1$ provided the first (resp. second) counter is decremented, left alone, or incremented at the i step of the computation. The probabilistic nature of the verifier V must be used to check these conditions.

Fix a large number p and another large constant q. It is convenient to select their exact values later on; they determine the error probability ϵ of the IPS. V has two "counters" which we call D and S: each is initially 0 and can take values from 0 to q. V performs four computations as it receives the

computation $\mathcal{C}$ from the prover. We denote these by A_i and B_i where i is 0 or 1. We describe A_0. This computation first performs a finite state transduction on the computation $\mathcal{C}$ mapping it to

$$a^{k_2+r_2}b^{k_3}, a^{k_4+r_4}b^{k_5}, \ldots, a^{k_{2m-2}+r_{2m-2}}b^{k_{2m-1}}$$

where $m = t/2$. Since r_i is easy to compute given the ID's, this transduction is computable with only a finite state control. Now V considers each of the pairs $a^{k_{2i}+r_{2i}}b^{k_{2i+1}}$ as a game. It plays these games in the order as they appear. If some game's outcome is not-equal, then it immediately rejects. Otherwise, it records the following information: whether or not *all* the games were won by double or whether or not *all* were won by sum. Again, this can be done with its finite state control. Essentially, A_0 is checking (1) for even i's. In the same way A_1 checks (1) for odd i's, B_0 checks (2) for even i's, and B_1 checks (2) for odd i's.

V updates its two counters D and S as follows. If double (resp. sum) wins *all* the games played by A_0, A_1, B_0, B_1, then increase D (resp. S) by 1. V then considers the values of the counters D and S.

1. If D and S are both less than q, then it gets another computation $\mathcal{C}$ from the prover and repeats the four computations A_0, A_1, B_0, B_1 as before.
2. If D equals q, then accept if $S \neq 0$, else reject.
3. If S equals q, then accept.

It remains to show that this protocol is correct. There are two cases. First, suppose that the input x is in the language L. Then there is an accepting computation of the counter machine, and V repeatedly receives this computation from the prover P. Since the computation is correct, every game played by the A_0, A_1, B_0, B_1 is fair. Thus, the probability that all are won by double is exactly the same as the probability that all are won by sum. Now the only way that V can reject is for D to equal q while S is still 0. This happens with probability 2^{-q}. Thus, if $2^{-q} < \epsilon$, in this case V accepts with probability at least $1 - \epsilon$.

Second, assume that x is not in L. Then, there is no accepting computation of the counter machine on x. If V ever receives a computation $\mathcal{C}$ that violates any condition except (1) and (2), V immediately rejects. Thus, consider the case that in a computation V receives, only (1) or (2) is violated. We can also assume that no game played ever ends in the outcome *not-equal*. Otherwise, V immediately rejects. The key insight is that for each computation $\mathcal{C}$, at least one game played by A_0, A_1, B_0, B_1 must be unfair. Therefore, if either double or sum wins all the games played by A_0, A_1, B_0, B_1 on $\mathcal{C}$, the probability that double wins all these games is always much higher than the probability that sum wins all these games. In fact, from Lemma 3.3, the probability that doubles wins all the games, given that either sum or doubles wins all the games, is at least $1 - 1/(2^{p-1} + 1)$. Now, consider the probability that V rejects, that is, that D equals q while S is still equal to 0. The probability that this occurs is at least $(1 - 1/(2^{p-1} + 1))^q$. Choose p so that this quantity is at least $1 - \epsilon$ (recall that q is already fixed, so p may depend on q). Then, the probability that V rejects when $x \notin L$ is at least $1 - \epsilon$. This completes the proof of the theorem. □

The emptiness problem for 1-way probabilistic finite state machines (1pfa's) is the problem of determining whether or not there is some input that the machine accepts with probability at least $1 - \epsilon$. Our model of 1pfa is that transition probabilities are rational and the input head never moves left.

Theorem 3.2 *The emptiness problem for 1-way probabilistic finite state machines is undecidable.*

Proof: We describe a computable reduction from the halting problem for counter machines on empty input to the emptiness problem for 1pfa's. Since the halting problem on empty input is undecidable (see [1]), this is sufficient to prove the theorem.

Let M be a counter machine. From M, we construct a 1pfa in essentially the same way that the verifier is constructed in the proof of Theorem 3.1. There are the following differences in the computation of the verifier and the 1pfa. The first difference is that the 1pfa does not have a communication cell. Instead, whenever the 1pfa enters a communication state, the next symbol of the input is read. This input symbol determines the transition of the 1pfa in the same way that the symbol in the communication cell determines the transition of the verifier. Thus informally, the role of the prover is now played by the 1-way input tape. Another difference is that whereas the verifier reads the input (in order to check that the correct state transitions are taken by C), the 1pfa does not do this; instead, the 1pfa proceeds as the verifier would on an empty input tape. Finally, if the 1pfa reaches the end of its input tape before a halting state is reached, the 1pfa halts in a rejecting state. By this construction, the probability that the 1pfa accepts an input is equal to the probability that the verifier accepts on empty input, when the string that the verifier receives from the prover equals the input of the 1pfa.

Suppose that M accepts on empty input. Then we claim that there is some input on which the 1pfa accepts with probability at least $1 - \epsilon$. Namely, this is the input which is the concatenation of many copies of the accepting computation for M; the number of copies is chosen to be large enough that the 1pfa accepts with probability at least $1 - \epsilon$ after processing these copies. However, if M does not accept on empty input, then on all inputs, the 1pfa accepts with probability less than ϵ. □

4 Time-varying Markov Chains

In this section, we derive new bounds on the halting properties of time varying Markov chains and feedback chains. In Section 6, we discuss the relationship between these chains and IPS's. A time-varying Markov chain (see Seneta [19]) 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. Furthermore, we assume that 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 $p \leq q \leq 2^n$. Second, the nth row of both A and B has 0's everywhere, except at the $(n, n)th$ position, which is 1. We let $\mathcal{M}$ denote the *family of n-state time-varying Markov chains* with initial state 1. State 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 $\bar{x}_j$. Let $\alpha^{(k)}$ denote the matrix product $\prod_{i=1}^{k} \alpha_i$. Then $\alpha_{ij}^{(k)}$ 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 $\alpha_{in}^{(k)} > 0$. 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, the sequence $\{\alpha_{in}^{(k)}\}, k = 1, 2, \ldots$, is non-decreasing. Also every element of the sequence is bounded above by 1. Hence $\lim_{k\to\infty} \alpha_{in}^{(k)}$ exists. We call this limit the probability that M_α halts, that is, reaches n, from i. When $i = 1$ 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 in Theorem 4.1 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 of Theorem 4.1. 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 $[i]$ denote the set $\{1, \ldots, i\}$. 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. 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 some 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 is proved in Lemma 4.1. 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 4.3. Also in Lemma 4.3, 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 4.4 to get an upper bound on the rate of convergence of $p_{\alpha,i}$ to p and to complete the proof.

Lemma 4.1 *Let $\alpha \in \{A, B\}^\omega$. Suppose that with probability at least q, M_α reaches a state of set $S \subseteq [n-1]$ at time t and that the halting state n is reachable from every $j \in S$ in k steps. Then the probability that M_α halts between steps $t+1$ and $t+k$ is at least $q/(2^n)^k$.*

Proof: Let α' be the same as the sequence α with the first t symbols removed. By definition, $(\alpha')_{jn}^{(k)}$ is the probability of reaching n from j in k steps on sequence α'. 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 $(\alpha')^{(k)}$ is at least $1/(2^n)^k$. This can be proved easily by induction on k. Hence if $j \in S$, $(\alpha')_{jn}^{(k)} \geq 1/(2^n)^k$.

The probability of halting between steps $t+1$ and $t+k$ is at least the probability of reaching a state j of S in exactly t steps of M_α, times the probability of reaching state n from j in a further k steps. The first of these probabilities is at least q, by the statement of the lemma, and the second is at least $1/(2^n)^k$, from the last paragraph. Thus, the probability that M_α halts between steps $t+1$ and $t+k$ is at least $q/(2^n)^k$. □

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 4.2 we prove that any safe set is 2^n-safe.

Lemma 4.2 *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. $\square$

Consider each string $\alpha \in \{A, B\}^\omega$ as composed of substrings of length 2^n. Let $p_{\alpha,i}$ be the probability that M_α halts, that is, reaches state n from state 1, in $i2^n$ steps. Thus $p_{\alpha,i} = \alpha^{(i2^n)}{}_{1n}$. 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 4.3 *Let p be the halting probability of $\mathcal{M}$ 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 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,

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

Also let $\bar{S}$ be the set of states that are reachable from state 1 in $i2^n$ steps of α, but from which the halting state n cannot be reached in a further 2^n steps. Thus,

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

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 4.2, $\bar{S}$ 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 probability that M_α is in a state of $\bar{S}$ after exactly $i2^n$ steps is less than $1 - p$. Otherwise, with probability at least $1 - p$, M_α would never halt on the sequence obtained by concatenating the first $i2^n$ symbols of α with α''. This would contradict the fact that p is the halting probability of $\mathcal{M}$.

The probability that M_α is in a state of S after exactly $i2^n$ steps is exactly 1$-$ (the probability that M_α has already halted) $-$ (the probability that M_α is in a state of $\bar{S}$). Hence, the probability that M_α is in a state of S after exactly $i2^n$ steps is at least $1 - p_{\alpha,i} - (1 - p) = p - p_{\alpha,i}$.

From Lemma 4.1 it follows that the probability that M_α halts between steps $i2^n + 1$ and $(i + 1)2^n$, that is, $p_{\alpha,i+1} - p_{\alpha,i}$, is at least $(p - p_{\alpha,i})/(2^n)2^n$, completing the proof of the lemma. $\square$

Lemma 4.4 *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

$$\begin{aligned} p - p_{\alpha,i+1} &= (p - p_{\alpha,i}) - (p_{\alpha,i+1} - p_{\alpha,i}) \\ &\leq (p - p_{\alpha,i}) - (p - p_{\alpha,i})/\mu \qquad \text{from Lemma 4.3} \\ &= (p - p_{\alpha,i})(1 - 1/\mu) \\ &\leq (1 - 1/\mu)^{i+1} \qquad \text{(from the induction hypothesis).} \end{aligned}$$

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 4.4 is fairly tight, as the next lemma shows.

Lemma 4.5 *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 Section 6 because it is best described by constructing an interactive proof that runs in expected time $2^{2^{\Omega(n)}}$.

The following theorem combines Lemmas 4.4 and 4.5 to obtain our main result on the rate of halting of time-varying Markov chains.

Theorem 4.1 *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.*

5 Feedback Chains

In order to obtain our upper bounds on interactive proof systems, we need to generalize the definitions and results of Section 4 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*(η). 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*(η).

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 $b_1 \ldots b_k \in \{0, 1\}^*$, $P_{b_1 \ldots b_k} = \textit{matrix}(\eta)$, where $\textit{feedback}(\eta) = b_1 \ldots b_k$.

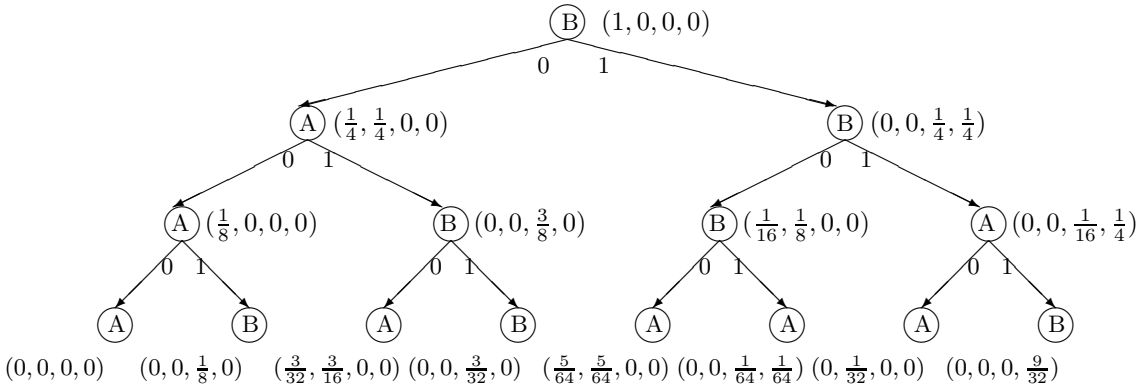
For each i and each node η of tree α, we define a vector $\bar{x}(\eta, i)$ with the following property. Suppose that η is at level k of tree α. Then $(\bar{x}(\eta, i))_j$ is the probability that the $(k+1)$st random variable in the chain corresponding to α is j *and* that the feedback from the first k steps is *feedback*(η), 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) the n-vector with 0's everywhere, except from the ith entry, which is 1. Let F_0 be the matrix such that entry $[i, i] = 1$ 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 $(\bar{x}(\eta_0, i))_j = (\bar{x}(\eta, i)\textit{matrix}(\eta)F_0)_j$ and $(\bar{x}(\eta_1, i))_j = (\bar{x}(\eta, i)\textit{matrix}(\eta)F_1)_j$.

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 $\alpha_1^{(k)}$ is the sum of the vectors at level k. Hence for example, $\alpha_1^{(3)} = (\frac{11}{64}, \frac{19}{64}, \frac{15}{64}, \frac{19}{64})$. Thus the probability of being in state 1 in three steps of M_α is 11/64.

The definitions of reachability for feedback chains is a natural extension of the definition for time-



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

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 $\bar{x}(\eta, 1)$.*

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 $\alpha_{in}^{(k)} > 0$; 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 state i, the sequence $\{\alpha_{in}^{(k)}\}, k = 0, 1, 2, \ldots$ is non-decreasing. Also every element of the sequence is bounded above by 1. Hence $\lim_{k\to\infty} \alpha_{in}^{(k)}$ exists. We call this limit the probability of reaching n from i on the chain M_α. When $i = 1$ 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 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 4.

Lemma 5.1 *Let $\alpha \in T(A, B)$. Suppose that on some fixed feedback sequence of length t, with probability at least q, M_α reaches a state of set $S \subseteq [n-1]$ at time t and that the halting state n is reachable from every $j \in S$ in k steps. Then the probability that, M_α halts between steps $t+1$ and $t+k$, given the fixed feedback sequence up to time t, is at least $q/(2^n)^k$.*

Proof: The proof is very similar to Lemma 4.1. □

We define a set $S \subseteq [n-1]$ is 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 5.2 *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*(η) 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*(η) 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*(η). 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*(α) be any subtree of α with the following properties: (i) if η and η' are internal nodes of *prune*(α) then *reachable*$(\eta) \neq$ *reachable*(η'); (ii) if η is a leaf of *prune*(α) then *reachable*$(\eta) =$ *reachable*(η'') for some internal node η'' of *prune*(α); and (iii) *prune*(α) is a full tree, that is, each node is either a leaf or has two children. A tree *prune*(α) 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*(η_j) by the pigeon-hole principle.

Now define α' as follows. Let *matrix*(root(α')) be *matrix*(root(α)). For any node η' in α' suppose that η is the unique internal node of *prune*(α) such that *reachable*$(\eta) =$ *reachable*(η'). 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*(η_0) and *matrix*$(\eta'_1) =$ *matrix*(η_1). From the definition of *reachable*(), *reachable*(η'_l) = *reachable*(η_l) for $l = 0, 1$. By the construction of *prune*(α), there are internal nodes η_i, η_j in *prune*(α) such that *reachable*$(\eta'_0) =$ *reachable*(η_i) and *reachable*$(\eta'_1) =$ *reachable*(η_j). From this it follows that α' is well defined.

Also, for all nodes η' in tree α', $n \notin$*reachable*(η'). 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 *reachable*(η). 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 $\alpha^{(i2^n)}{}_{1n}$. 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 5.3 *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 4.3, 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 $1 \leq k \leq m$ let α'_k be the tree rooted at η_k. Also let S^k be the set of states that are reachable at η_k from state 1 in $i2^n$ steps of α and which reach the

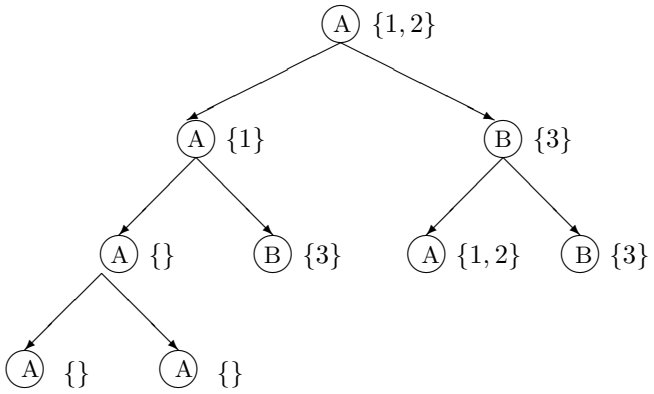


Figure 2: *Template that can be used to construct an infinite tree α' such that state 4 is not reachable from either of states 1,2 on α'.*

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

$$S^k = \{j \in [n-1] \mid (\bar{x}(\eta_k, 1))_j > 0 \text{ and } ((\alpha'_k)^{(2^n)})_{jn} > 0\}.$$

Let $\bar{S}^k = \{j \in [n-1] \mid (\bar{x}(\eta_k, 1))_j > 0 \text{ and } ((\alpha'_k)^{(2^n)})_{jn} = 0\}$. Then $\bar{S}^k$ is not 2^n-safe. This is because on tree α'_k, no state in $\bar{S}^k$ reaches the halting state in 2^n steps. Then from Lemma 5.2, $\bar{S}^k$ is not safe. Therefore, for each $k, 1 \leq k \leq m$, since $\bar{S}^k$ is not safe, there must exist a tree α''_k such that n is not reachable from any state in $\bar{S}^k$ on α''_k.

In the rest of this proof, let $\bar{x}^k$ denote $\bar{x}(\eta_k, 1)$. We claim that with probability less than $1-p$, M_α reaches node η_k in a state of $\bar{S}^k$ for some k, $1 \leq k \leq m$. Otherwise, with probability at least $1-p$, M_α would never halt on the tree obtained from α by replacing the tree rooted at η_k by α''_k, $1 \leq k \leq m$. This would contradict the fact that p is the halting probability of $\mathcal{M}$.

Therefore the probability that M_α reaches node η_k in an a state of S^k for some $k, 1 \leq k \leq m$, after exactly $i2^n$ steps, is exactly 1− (the probability that M_α has already halted) − (the probability that M_α reaches node η_k in a state of $\bar{S}_k$ for some k). This is at least $1 - p_{\alpha,i} - (1-p) = p - p_{\alpha,i}$.

From Lemma 5.1 it follows that the probability that M_α halts between steps $i2^n + 1$ and $(i+1)2^n$, that is, $p_{\alpha,i+1} - p_{\alpha,i}$, is at least $(p - p_{\alpha,i})/(2^n)2^n$, completing the proof of the lemma. □

Lemma 5.4 . *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 4.4. □

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

Theorem 5.1 *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.*

6 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.

We need the following notation. Just as for Turing machines, a configuration of an interactive proof system for a fixed input is a tuple containing binary encodings 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 $\log n + O(s(n))$. (All logs are to base 2.) 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 $2^{\log n + O(s(n))}$. We call a configuration of (P, V) that contains a communication state or reading state a communication configuration or reading configuration, respectively.

For convenience we make the following assumptions about interactive proof systems, without loss of generality. First, we assume that the prover's alphabet is $\{a, b\}$. Also, 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 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. We number the communication configurations $\{1, \ldots, m-1\}$, and assume that 1 is the initial configuration. Finally, we assume that there is a unique rejecting configuration.

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

Lemma 6.1 *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 one being the rejecting state, equals the probability that M^* reaches its halting state in k steps.

Proof: Given (P, V), let m be as in the assumptions above. 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 [12]) 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 6.1, 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 6.1 and our results on feedback chains of Section 5 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 6.1 *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 6.1, 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 5.4 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 6.1, 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 6.2 *IP(2pfa) ⊆ ATIME($2^{2^{O(n)}}$).*

Proof: Let (P, V) be an interactive proof for language L with error probability $\epsilon < 1/2$. Let $t(n) = 2^{2^{O(n)}}$ be the function of Theorem 6.1 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 $k^2t(n)$. From this the result follows easily, by applying a result of Condon and Ladner [5], 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 $k^2t(n)$ steps without ever halting, V' halts the computation and accepts. Otherwise, if the computation halts within $k^2t(n)$ 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 6.1, 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 6.2.

Theorem 6.3 *For any space constructible function $s(n) \geq \log n$, IP(space(s(n)) ⊆ 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 6.2 for one-way proofs.

Theorem 6.4 *One-way-IP(2pfa) ⊆ NTIME($2^{2^{O(n)}}$).*

Proof: Let (P, V) be a one-way IPS with a 2pfa verifier that accepts language L with error probability ϵ. We describe a nondeterministic Turing machine M' that accepts L.

Fix an input x of length n and let A and B be the matrices defined in Lemma 6.1 where $m = O(n)$. From the transition function of V, M' first computes A and B. Let $t(n) = 2^{2^{O(n)}}$ be the function of Theorem 6.1 and let k be such that $1 - \epsilon - 1/2^k - 1/k > 1/2$. M' guesses a string $\alpha_1, \ldots \alpha_{t(n)k^2}$ where

each $\alpha_i \in \{A, B\}$. Then M' computes the product $\alpha^{(t(n)k^2)}$. If the entry of $\alpha^{(t(n)k^2)}$ numbered $[1, \textit{reject}] < 1/2$, where *reject* is the number of the rejecting configuration, then 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 6.1, for all strings α of length $k^2t(n)$, the probability of reaching the reject state in $k^2t(n)$ 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 $k^2t(n)$, 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 length of α. The $\alpha^{(t(n)k^2)}$ is the product of $k^2t(n)$ 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 matrix can be represented as the quotient of two binary numbers of length $2^{2^{O(n)}}$, and the matrix can be computed in time $2^{2^{O(n)}}$. Hence the total running time of M' is $2^{2^{O(n)}}$. $\square$

To complete this section, we show that the bounds of Theorem 6.1 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 [18] and in a multi-prover setting by Feige and Shamir [10]. The interactive proof we construct is one-way.

Theorem 6.5 *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.

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. (Here, 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. □

7 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. Our results extend to other types of space bounded verifiers. In proving these results, we also obtained bounds on the rate of convergence of time-varying Markov chains and feedback chains.

It is interesting to contrast the power of single-prover and multi-prover IPS's. Multi-prover systems were introduced in Ben-Or et al. [2]; space bounded multi-prover systems have been investigated independently by Feige and Shamir [10] and Condon and Lipton [7]. Informally, in a k-prover IPS, 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. Feige and Shamir [10] showed that any language accepted by a multi-prover IPS is recursive. Also, any recursive language has a 2-prover interactive proof with a 2pfa verifier that halts with probability 1.

From this result and Theorem 6.2, is is clear that for 2pfa verifiers, a 2-prover IPS is much more powerful than a 1-prover system; but three or more provers is no more powerful than two.

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.

Acknowledgement. We thank Larry Stockmeyer for pointing out an improvement to our original upper bound of Theorem 6.2. Thanks also to Joan Feigenbaum for several helpful comments.

References

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

[2] M. Ben-Or, S. Goldwasser, J. Kilian and A. Wigderson, *Multi-Prover Interactive Proofs: How to Remove Intractability*, Proc. 20th Symposium on the Theory of Computing, ACM, 1988, pp. 113-131.

[3] D. Blackwell, L. Breiman and A. J. Thomasian. *Proof of Shannon's transmission theorem for finite-state indecomposable channels.* Annals of Math. Stat. 29 (1958), pp. 1209-1220.

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

[5] A. Condon and R. Ladner, *Probabilistic Game Automata*, J. Comput. System Sci., 36:3 (1988), pp. 452-489.

[6] A. Condon and D. Hernek, *Random Walks on Colored Graphs* Random Structures and Algorithms, 5:1, 1994.

[7] A. Condon and R. J. Lipton, *Upper Bounds on the Complexity of Space Bounded Interactive Proof Systems*, Technical Report #841, Computer Sciences Department, University of Wisconsin-Madison, 1989.

[8] C. Dwork and L. Stockmeyer, *Finite State Verifiers I: The Power of Interaction*, J. ACM, 39:4 (1992), pp. 800-828.

[9] R. Frievalds, *Probabilistic two-way machines*, Proc. International Symposium on Mathematical Foundations of Computer Science, Springer-Verlag Lecture Notes in Computer Science, 118, New York, 1981, pp. 33-45.

[10] U. Feige and A. Shamir, *Multi-Oracle Interactive Protocols with Space Bounded Verifiers*, Proc. Fourth Annual Conference on Structure in Complexity Theory, IEEE, June 1989, pp. 158-164.

[11] L. Fortnow, J. Rompel and M. Sipser, *On the Power of Multi-Prover Interactive Protocols*, Proc. Third Annual Conference on Structure in Complexity Theory, IEEE, 1988, pp. 156-161.

[12] J. Gill, *Computational Complexity of Probabilistic Turing Machines*, SIAM J. Comput., 6:4 (1977), pp. 675-695.

[13] S. Goldwasser, S. Micali and C. Rackoff, *The knowledge complexity of interactive protocols*, Proc. 17th Symposium on the Theory of Computing, ACM, 1985, pp. 291-304.

[14] J. Kilian, *Zero Knowledge with Log-Space Verifiers*, Proc. 29th Symposium on Foundations of Computer Science, IEEE, 1988, pp. 25-35.

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

[16] A. Paz, *Definite and quasi-definite sets of stochastic matrices*, AMS Proceedings, 16(4):634-641, 1965.

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

[18] G. L. Peterson and J. H. Reif, *Multiple-Person Alternation*, Proc. 20th Symposium on Foundations of Computer Science, IEEE, 1979, pp. 348-363.

[19] E. Seneta, *Non-negative Matrices and Markov Chains (2nd Edition)*, Springer-Verlag, New York, 1981.

[20] J. Wolfowitz, *Products of indecomposable, aperiodic, stochastic matrices*, AMS Proceedings 14 (1963), pp. 733-737.