Finite State Verifiers II: Zero Knowledge

Cynthia Dwork and Larry Stockmeyer

IBM Almaden Research Center
San Jose, California

Abstract. The zero knowledge properties of interactive proof systems (IPS's) are studied in the case that the verifier is a 2-way probabilistic finite state automaton (2pfa). The following results are proved:

1. There is a language L such that L has an IPS with 2pfa verifiers but L has no zero knowledge IPS with 2pfa verifiers.
2. Consider the class of 2pfa's that are sweeping and that halt in polynomial expected time. There is a language L such that L has a zero knowledge IPS with respect to this class of verifiers, and L cannot be recognized by any verifier in the class on its own.

A new definition of zero knowledge is introduced. This definition captures a concept of "zero knowledge" for IPS's which are used for language recognition.

This paper appeared in the *Journal of the ACM*, Vol. 39, No. 4, 1992, pp. 829–858.

1 Introduction

In this paper we continue the study, begun in [7], of interactive proof systems where the verifier is a 2-way probabilisitic finite state automaton (2pfa). We focus here on zero knowledge interactive proof systems. Informally, an *Interactive Proof System* (IPS) for membership in a language L is a two-party protocol whereby a "prover" convinces a "verifier" that elements $x \in L$ are actually in L. The IPS is zero knowledge if *no* verifier can extract any information from the interactive proof that $x \in L$ except for the fact that $x \in L$, i.e., nothing extra is learned. Zero knowledge interactive proof systems were first defined by Goldwasser, Micali, and Rackoff [14] for the case where the verifier is a probabilistic Turing machine (ptm) which runs in polynomial time. Prior to the work reported here, all work on zero knowledge IPS's has dealt with this class of verifiers. Zero knowledge interactive proof systems are powerful tools for cryptography, security systems, and distributed computing; see, for example, [8, 11, 12, 13].

Due to the present lack of understanding of the power of polynomial time computation, many previous results are based on unproven assumptions, typically that a certain problem is not in P or that a certain cryptosystem cannot be broken in polynomial time. If the given assumptions are false, then either the proof becomes invalid or the result becomes trivial. For example, the important and powerful result of Goldreich, Micali, and Wigderson [12], that any language in NP has a zero knowledge IPS, would become unproven if secure probabilistic encryption schemes do not exist and would become vacuous if P = NP. On the other hand, proving that the class ZKIP of languages accepted by zero knowledge IPS's is not equal to the class IP of languages accepted by general IPS's would prove that P $\neq$ PSPACE. This is true because

$$\mathrm{P} \subseteq \mathrm{ZKIP} \subseteq \mathrm{IP} \subseteq \mathrm{PSPACE}.$$

The first two inclusions are trivial, and the third is proved by Condon and Ladner [5] and P. Feldman (personal communication in [15]). Similarly, if there is a language $L \in$ ZKIP such that L cannot be recognized by a polynomial-time Turing machine (without help from a prover), then P $\neq$ PSPACE.

In order to better understand the concept of zero knowledge without using unproved assumptions and without being forced to deal with such hard problems as P vs. PSPACE, we restrict the verifiers to a class where we have some hope of proving lower bounds, namely, the class of 2pfa's. We summarize our results for 2pfa verifiers. As in previous work on IPS's, the prover is assumed to be arbitrarily powerful. We also give corresponding known results for polynomial-time probabilistic Turing machine verifiers.

1. There are *no* zero knowledge IPS's for the palindrome problem or for the graph isomorphism problem, although there are (not zero knowledge) IPS's for these problems.

   Polynomial time ptm: There is no example of a problem which has an IPS but has no zero knowledge IPS. Graph isomorphism has a zero knowledge IPS [12]. Under certain cryptographic assumptions, if a problem has an IPS then it has a zero knowledge IPS [1, 17].

2. With further restriction on the verifier, namely that it is a sweeping automaton which runs in polynomial expected time, the language $\{\, 0^n 1^n \mid n \geq 1 \,\}$ has a zero knowledge IPS and, moreover, no verifier so restricted can recognize this language alone. The latter fact, proved

by Greenberg and Weiss [16], is crucial; if a verifier could recognize the language alone, the existence of a zero knowledge IPS is trivial since the prover doesn't have to say anything.

Polynomial time ptm: Zero knowledge IPS's are known only for problems which have not been proved to be unsolvable by a verifier alone.

The second result is proved in terms of a new definition of "zero knowledge" which is somewhat weaker than definitions considered previously. Result 1 holds for the previous definitions as well as the new definition. We discuss this further in §3.

We now outline the remainder of the paper. In Section 2 we review the definition of an IPS with a 2pfa verifier and give some related definitions. In Section 3 we review the previous definitions of zero knowledge and give our new definition. In Section 4 we prove result 1 above and in Section 5 we prove result 2. In Section 6 we suggest some directions for future research.

Although we occasionally use results from [7], the present paper is largely independent of [7] and can be read separately. The only exception is that the reader might want to consult Section 2 of [7] for the precise definition of an IPS with a 2pfa verifier. Although we review this definition in the next section, we do not reproduce the complete formal definition given in [7].

Finally, we mention two papers which describe variations on the standard IPS model and which show, without using unproved assumptions, that every language having an IPS also has a zero knowledge IPS. Ben-Or, Goldwasser, Kilian, and Wigderson [2] consider an IPS model where there are two provers which cannot communicate with each other; the verifier is a polynomial-time ptm. Kilian [18], adopting a definition of zero knowledge based on the one introduced here, considers the single prover model where the verifier is a polynomial-time log-space ptm.

2 Definitions

First we review, informally, the definition of an IPS with a 2pfa verifier. A complete formal definition is given in Section 2 of [7]. An *interactive proof system* (*IPS*) consists of a prover P and a verifier V. A verifier consists of a probabilisitic finite state control with a single 2-way head which reads the input word. The verifier can also write symbols into and read symbols from a *communication cell* which is used to communicate with the prover and which can hold a single symbol from some finite communication alphabet. We assume that the states of V are partitioned into *reading states*, *communication states*, and *halting states*. Whenever the machine is in a reading state, based on the current state and the input symbol being scanned, V probabilistically chooses an action; each action consists of a new state and a direction of movement of the input head (left, right, or stationary). Whenever the machine is in a communication state, this state determines a symbol which V writes in the communication cell. This causes the prover to write some symbol in the cell, and based on this symbol, V probabilistically chooses a new state; the head does not move in this case. There are two halting states, an *accepting state* and a *rejecting state*. The prover is arbitrarily powerful. The actions of the prover can be chosen probabilistically, and each response of the prover can depend on the input and on the entire history of symbols which have been written in the cell up to the time of this response. However, the prover's view of the verifier's computation is restricted to the view seen through the communication cell. In general, the prover does not know the exact configuration of the verifier, i.e., its state and head position. All probabilistic choices are made by flipping a fair coin. In particular, each transition probability of the verifier is either 0, 1/2, or 1. (Our results do not change if transition probabilities are allowed to be arbitrary rational numbers.)

Let Σ be the input alphabet. The computation of (P, V) on a particular input $x \in \Sigma^*$ starts with the string $\cent x \cent$ written on V's input tape, where $\cent \notin \Sigma$ is an endmarker symbol. V is in a designated initial state scanning the leftmost symbol of x. The computation is then governed (probabilistically) by the transition functions of P and V until the verifier enters a halting state.

The prover-verifier pair (P, V) is an *interactive proof system for the language* $L \subseteq \Sigma^*$ *with error probability* ε if

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

The probabilities here are taken over all the random choices of the verifier and the prover. Let IP(2pfa) be the class of languages L such that L has an interactive proof system with error probability ε for some $\varepsilon < 1/2$.

We note that if L has an IPS (P, V) with error probability $\varepsilon < 1/2$ and if, for all x and all provers P^*, the computation $(P^*, V)(x)$ halts with probability 1, then for any $\varepsilon' > 0$, L has an IPS with error probability ε'. The new IPS is obtained by running (P, V) a sufficiently large constant number of times and taking the majority answer. However, we do not know in general how to obtain error probabilities as small as $2^{-|x|}$, as can be done when verifiers are polynomial-time probabilistic Turing machines.

The *time complexity* of the verifier is the number of steps taken by the verifier before halting. There are two cases: the *expected time*, averaged over all random choices of the verifier and the prover; and the *worst-case time*, maximized over all such choices. The verifier *runs in expected* (*resp., worst-case*) *time* $T(n)$ if, for any input x of length n and any prover P^*, the verifier halts in expected (resp., worst-case) time $T(n)$ in the computation $(P^*, V)(x)$. By attaching the prefix "ptime" to a class of machines, we restrict the class to those machines that run in polynomial expected time.

We defer the definition of "zero knowledge" interactive proof systems to §3.

A *sweeping 2pfa* is a 2pfa restricted so that the input head can switch its direction of movement only when scanning an endmarker $\cent$. In any computation of a sweeping 2pfa, the head alternately sweeps from the left endmarker to the right endmarker, then back from right to left, and so on. The head can remain stationary at certain steps during a sweep; in particular, the head does not move when the verifier is communicating with the prover. Deterministic and nondeterministic sweeping automata have been studied in another context by Sipser [21].

We let $|S|$ denote the cardinality of the set S, and we let $|w|$ denote the length of the word w.

Let M be an IPS or a 2pfa, and let $A, B \subseteq \Sigma^*$ with $A \cap B = \emptyset$. Then M *separates* A *and* B if there is some constant $\varepsilon < 1/2$ (the error probability) such that, for all $x \in A$, $M(x)$ accepts with probability at least $1 - \varepsilon$, and for all $x \in B$, $M(x)$ rejects with probability at least $1 - \varepsilon$. We do not care about the behavior of M on inputs not in A or B.

We shall need the following result which gives conditions under which no 2pfa can separate A and B; see [7, Theorem 3.3].

Theorem 2.1 *Let* $A, B \subseteq \Sigma^*$ *with* $A \cap B = \emptyset$*. Suppose there is an infinite set* I *of positive integers and, for each* $m \in I$*, a set* $W_m \subseteq \Sigma^*$ *such that*

1. $|w| \leq m$ *for all* $w \in W_m$,

2. *for every integer* k *there is an* m_k *such that* $|W_m| \geq m^k$ *for all* $m \in I$ *with* $m \geq m_k$, *and*

3. *for every* $m \in I$ *and every* $w, w' \in W_m$ *with* $w \neq w'$, *there are words* $u, v \in \Sigma^*$ *such that either* $uwv \in A$ *and* $uw'v \in B$, *or* $uwv \in B$ *and* $uw'v \in A$.

Then no 2pfa separates A *and* B.

2.1 An Example

If x is a string, let x^R be x written backwards. Define the set of palindromes:

$$Pal = \{\, x \in \{0,1\}^* \mid x = x^R \,\}.$$

We review the IPS (P, V) for *Pal* described in [7] since this IPS provides an important example in the next section. Fix any error probability $\varepsilon > 0$. Assuming that x is a palindrome, the interaction involves k iterations, where $k = \lceil \log_2(1/\varepsilon) \rceil$. On each iteration, the prover P sends x to the verifier one symbol at a time as these symbols are requested by the verifier. At the start of each iteration, the verifier V (privately) tosses a fair coin. Letting w denote the string received from the prover during this iteration, if the outcome of the coin toss is "heads" then V checks that $w = x$ and rejects if not. If the outcome is "tails" then V checks that $w = x^R$ and rejects if not. If the check succeeds for all k iterations, then V accepts.

If x is a palindrome, then $(P, V)(x)$ accepts with probability 1. If x is not a palindrome, then V rejects with probability at least $1/2$ at each iteration, so the overall error probability is at most ε. This establishes the following.

Theorem 2.2 *Pal* $\in$ *IP(2pfa). Moreover, for any error probability* $\varepsilon > 0$, *there is an IPS for Pal where the verifier is a sweeping 2pfa which runs in worst-case time* $O(n)$.

3 Old and New Definitions of Zero Knowledge

Informally, an interactive proof system (P, V) for a language L is *zero knowledge* if for any input $x \in L$ and any verifier V^*, the only information which V^* can get from P during their interaction is the single bit of information that x belongs to L. Previous papers, e.g. [14], considered zero knowledge only in the case where verifiers are polynomial-time probabilistic Turing machines. For our purposes, it is useful to generalize the definition to an arbitrary class of verifiers as follows. Fix some class $\mathcal{V}$ of verifier machines, for example, ptime-ptm's or 2pfa's. Let $(\emptyset, \mathcal{V})$ be the subclass of machines in $\mathcal{V}$ that do not communicate with the prover (the symbol $\emptyset$ in this notation should be a reminder that the prover is empty). The interactive computation of $(P, V^*)(x)$ defines a distribution of conversations between P and V^*, where a *conversation* is the string of symbols written in the communication cell by both the verifier and the prover. The IPS (P, V) is *zero knowledge* if for any verifier $V^* \in \mathcal{V}$ there is an $M_{V^*} \in (\emptyset, \mathcal{V})$ such that, for all $x \in L$, $M_{V^*}(x)$ produces a distribution of conversations which is "close" to the distribution produced by $(P, V^*)(x)$. Here, "close" could mean either that the two distributions cannot be distinguished by any polynomial-size family of circuits (polynomial in the maximum of the length of x and the length of the conversation), or that the two distributions are actually identical; the former case is called *computational zero knowledge*, and the latter case is called *perfect zero knowledge*. Intuitively, the verifier V^* gets no extra information from P since there is an M_{V^*} which does not interact with P and which can produce nearly the same distribution of conversations that V^* would see in its interaction with P. (Further discussion and definitions of distiguishing distributions by families of circuits can be found in [14].)

At first glance, it would appear that the IPS (P, V) for palindromes described in Section 2.1 is perfect zero knowledge according to this definition. On input x, the conversation consists of the prover sending x to the verifier several times, and obviously a 2pfa can produce this conversation alone. On an intuitive level, however, this IPS is clearly not zero knowledge for the following reason. Let A be the set of "double palindromes", i.e., the set of palindromes of the form ww^R where w is itself a palindrome, and let B be the set of palindromes not in A. Then there is a 2pfa V^* such that (P, V^*) separates A and B (this is true even if P sends x to the verifier only once). On input x, V^* first checks that $|x|$ is even and rejects if not. Then starting from the left endmarker, V^* moves its head two to the right for every symbol sent to it by the prover until the right endmarker is reached. At this point, P has finished sending w and is ready to send w^R to V^*, where $x = ww^R$. So V^* is now in a position to compare w with w^R. Using Theorem 2.1 it is easy to show that no 2pfa separates A and B. (A similar argument will be given shortly in the proof of Theorem 4.1.) Therefore, it is clear that P is giving V^* some extra information which it cannot get by itself.

We should mention that the difficulties in this example arise essentially because the verifier does not have space to remember an entire conversation. The example also hinges on the assumption that the verifier does not write its answer (whether x is a double palindrome or not) in the communication cell. For if the verifier is allowed to write its answer, then no $M_{V^*} \in (\emptyset, \mathcal{V})$ can produce the same conversation. Thus, seemingly innocuous changes in the definition of "conversation" can result in very different definitions of "zero knowledge," when verifiers are space-bounded.

The example suggests the following new definition of zero knowledge which is not based on conversations but which captures an intuitive notion of the verifier getting no extra information from the prover. We call the new definition "recognition zero knowledge" to distinguish it from previous definitions.

Definition. Let $\mathcal{V}$ be a class of verifier machines. Let (P, V) be an IPS for the language L where $V \in \mathcal{V}$. Then (P, V) is a *recognition zero knowledge IPS for L with $\mathcal{V}$ verifiers* if, for any $V^* \in \mathcal{V}$ and any $A, B \subseteq L$ with $A \cap B = \emptyset$ such that (P, V^*) separates A and B, there is an $M_{V^*} \in (\emptyset, \mathcal{V})$ such that M_{V^*} separates A and B.

In order to compare recognition zero knowledge with previous definitions, we follow the approach, taken by Tompa and Woll [22], of considering verifiers which produce output. This approach seems to be a natural one for handling verifiers with restricted space (in particular, 2pfa's). In this approach, verifier automata have *output states*, and each output state is associated with some output symbol which is produced whenever the state is entered. In this way, an IPS $(P, V)(x)$ or a machine $M(x)$ produces a distribution of output strings. The definitions of perfect zero knowledge and computational zero knowledge are as above, but with distributions of output strings in place of distributions of conversations.

It is obvious that if (P, V) is perfect zero knowledge then (P, V) is computational zero knowledge. The following very simple fact shows that recognition zero knowledge is yet a weaker notion of zero knowledge.

Proposition 3.1 *If (P, V) is a computational zero knowledge IPS for L with $\mathcal{V}$ verifiers, then (P, V) is a recognition zero knowledge IPS for L with $\mathcal{V}$ verifiers.*

Proof. Suppose there is a $V^* \in \mathcal{V}$ and sets $A, B \subseteq L$ such that (P, V^*) separates A and B with error probability ε. We argue that the hypothesis that (P, V) is computational zero knowledge implies that there exists an $M \in (\emptyset, \mathcal{V})$ such that M separates A and B. Modify V^* so that it produces output 1 (resp., 0) whenever it accepts (resp., rejects). In this case, the output distribution is a distribution on $\{0, 1\}$. If $x \in A$, then $\Pr[(P, V^*)(x) = 1] \geq 1 - \varepsilon$; and if $x \in B$, then $\Pr[(P, V^*)(x) = 0] \geq 1 - \varepsilon$. Since (P, V) is computational zero knowledge, there is an $M = M_{V^*} \in (\emptyset, \mathcal{V})$ such that the output distribution of M is indistinguishable from the output distribution of (P, V^*). In particular, this means that for every real $\alpha > 0$ there is an n_α such that, for all $x \in L$ with $|x| \geq n_\alpha$,

$$\begin{aligned} |\Pr[(P, V^*)(x) = 1] - \Pr[M(x) = 1]| &\leq \alpha, \text{ and} \\ |\Pr[(P, V^*)(x) = 0] - \Pr[M(x) = 0]| &\leq \alpha. \end{aligned}$$

Choosing $\alpha = (1/2 - \varepsilon)/2$ and $\varepsilon' = \varepsilon + \alpha$ (so $\varepsilon' < 1/2$), for all x with $|x| \geq n_\alpha$ we have $\Pr[M(x) = 1] \geq 1 - \varepsilon'$ for all $x \in A$, and $\Pr[M(x) = 0] \geq 1 - \varepsilon'$ for all $x \in B$. Building a table of answers into M for all the x with $|x| < n_\alpha$ gives an $M \in (\emptyset, \mathcal{V})$ which separates A and B. Therefore, (P, V) is recognition zero knowledge. $\square$

As a consequence of this fact, if we prove that a certain language L does not have a recognition zero knowledge IPS, it follows that L does not have a computational or perfect zero knowledge IPS. In other words, using a weak concept of zero knowledge gives strong impossibility results. For example, in Section 4 we prove that the palindrome and graph isomorphism problems do not have recognition zero knowledge IPS's with 2pfa verifiers.

Recognition zero knowledge could also be useful in proving that certain languages do have zero knowledge IPS's, in cases where proving a stronger perfect or computational zero knowledge property might be very difficult. For example, in Section 5 we consider a restricted class $\mathcal{R}$ of 2pfa verifiers and show that a certain language L has a recognition zero knowledge IPS with $\mathcal{R}$ verifiers; moreover, L cannot be recognized by any machine in $(\emptyset, \mathcal{R})$, so this result is not vacuous. Recognition zero knowledge has also been used by Kilian [18].

In addition, we think that recognition zero knowledge is a simple and intuitively appealing definition of zero knowledge for language recognition problems.

4 Languages Having No Zero Knowledge IPS

We first consider the palindrome language *Pal* defined in §2.1.

Theorem 4.1 *There is no recognition zero knowledge IPS for Pal with 2pfa verifiers. This remains true with 2pfa verifiers which are restricted to be sweeping automata, or restricted to run in either polynomial worst-case time or polynomial expected time, or both.*

Proof. The proof is identical for general 2pfa verifiers and for the various types of restricted 2pfa's mentioned in the statement of the theorem. Let U_{Pal} be the set of words in Pal which have exactly one occurrence of the subword 11. Let

$$A_{Pal} = U_{Pal} \cap \{\, x \mid x = ww^R \text{ and } w = w^R \,\},$$

$$B_{Pal} = Pal - A_{Pal}.$$

Let (P, V) be any IPS for Pal with error probability $\varepsilon < 1/2$. We describe a 2pfa verifier V^* such that (P, V^*) separates A_{Pal} and B_{Pal} with error probability ε.

Let x be a palindrome input to the IPS. V^* first checks whether $x \in U_{Pal}$ and rejects if not. If $x \in U_{Pal}$, then V^* begins an interactive computation with P. In this computation, V^* on input $x = ww^R$ simulates V on input $x' = ww$. This is possible because $x \in U_{Pal}$. It is easy to see that any $x \in U_{Pal}$ has the form $x = ww^R$ where w ends with 1 and has no occurrence of the subword 11. So V^* can use the unique occurrence of 11 to identify the center of x. For example, V^* acts just like V as long as the simulation of V on ww has the head positioned on the first copy of w. When the simulation moves the head into the second copy of w, V^* moves its head to the leftmost symbol of x and carries on the simulation from there, scanning w instead of w^R. V^* remembers in its finite state control whether the simulation of V on ww has the head positioned on the first or second copy of w.

We now argue that (P, V^*) separates A_{Pal} and B_{Pal}. We let $P(\alpha) \leftrightarrow V(\beta)$ denote the interactive computation between P and V where P has input α and V has input β. In this notation, $(P, V^*)(x)$ is simulating $P(x) \leftrightarrow V(x')$. There are two cases to consider.

First, suppose that $x \in A_{Pal}$, so $x = ww^R$ where $w = w^R$. Therefore $x = ww^R = ww = x'$. Since x is a palindrome, the interactive computation $P(x) \leftrightarrow V(x)$ accepts with probability at least $1-\varepsilon$. Since $x = x'$ it follows that $P(x) \leftrightarrow V(x')$ accepts with probability at least $1-\varepsilon$, so $(P, V^*)(x)$ accepts with probability at least $1 - \varepsilon$.

Second, suppose that $x \in B_{Pal}$. Since $w \neq w^R$ it follows that $x' = ww$ is not a palindrome. We can view the computation $P(x) \leftrightarrow V(x')$ as the computation $P^*(x') \leftrightarrow V(x')$ where P^* is a prover which replaces its input x' with x and then acts just like P. Since (P, V) is an IPS for Pal and x' is not a palindrome, it follows from the definition of an IPS that $P^*(x') \leftrightarrow V(x')$ rejects with probability at least $1 - \varepsilon$. Therefore $(P, V^*)(x)$ rejects with probability at least $1 - \varepsilon$.

To complete the proof of the theorem, we apply Theorem 2.1 to show that no 2pfa can separate A_{Pal} and B_{Pal}. Let I be the set of positive integers divisible by 4, and for each $m \in I$ let W_m be the set of binary words of the form yy^R where $|y| = m/2$ and where y begins with 0, ends with 1, and contains no occurrence of 11. It is easy to see that $|W_m| \geq 2^{m/4-2}$. If $yy^R, zz^R \in W_m$ with $y \neq z$, and if we take $u = y^R$ and $v = y$, then $uyy^Rv \in A_{Pal}$ while $uzz^Rv \in B_{Pal}$, so all of the conditions of Theorem 2.1 are satisfied. □

By a similar proof, we now show that the graph isomorphism problem has no recognition zero knowledge IPS with 2pfa verifiers. Since graph isomorphism can be checked in deterministic exponential time, it follows from Theorem 3.12 in [7] that the graph isomorphism problem does have an IPS with a 2pfa verifier.

If G is an undirected graph with k vertices, the *adjacency representation* of G is the word

$$a_{12}a_{13}\ldots a_{1k}2a_{23}a_{24}\ldots a_{2k}2\ldots 2a_{k-1,k}$$

where $a_{ij} = 1$ if vertex i and vertex j are adjacent (i.e., connected by an edge), or $a_{ij} = 0$ otherwise. We refer to the subword $a_{i,i+1} \ldots a_{i,k}$ as the ith *row*. We write $G \sim H$ to mean that the graphs G and H are isomorphic. We define *Iso* to be the set of words of the form $A_G\#A_H$ where A_G and A_H are adjacency representations of graphs G and H, respectively, and $G \sim H$. We abbreviate such an input as $G\#H$.

Theorem 4.2 *There is no recognition zero knowledge IPS for Iso with 2pfa verifiers.*

Proof. The proof is very similar to the proof of Theorem 4.1. Let U_{Iso} be the set of words $G\#H \in \mathit{Iso}$ such that G and H both have the following form (see Figure 1):

1. the graph has vertices $1, 2, \ldots, k$, where $k = 2l$ for some even integer l,
2. vertices l and $l + 1$ are adjacent to all other vertices,
3. vertex $l - 1$ is adjacent to all even numbered vertices $i < l - 1$ and to vertices l, $l + 1$, and $l + 2$, but to no others,
4. vertex $l + 2$ is adjacent to all even numbered vertices $j > l + 2$ and to vertices l, $l + 1$, and $l - 1$, but to no others,
5. odd numbered vertices other than vertices $l-1$ and $l+1$ are adjacent to no vertex other than l and $l + 1$, and
6. no vertex $i < l - 1$ is adjacent to any vertex $j > l + 2$.

In G, the even numbered vertices less than $l - 1$ induce a subgraph G_1 and the even numbered vertices greater than $l + 2$ induce a subgraph G_2, where G_1 and G_2 both have $t = k/4 - 1$ vertices. So G consists of the subgraphs G_1 and G_2, the vertices l and $l + 1$ which are adjacent to all other vertices, the vertex $l - 1$ which is adjacent to all vertices of G_1, the vertex $l + 2$ which is adjacent to all vertices of G_2, and the odd numbered vertices other than $l - 1$ and $l + 1$ which are adjacent only to l and $l + 1$. We abbreviate this as $G = G_1\$G_2$ (similarly, $H = H_1\$H_2$), and we denote a word in U_{Iso} as $G_1\$G_2\#H_1\$H_2$. Since vertices $l - 1$ and $l + 2$ are the only vertices of degree $t + 3$ in G and H, since vertex $l - 1$ is adjacent to G_1 (H_1) but not to G_2 (H_2), and since vertex $l + 2$ is adjacent to G_2 (H_2) but not to G_1 (H_1), it follows that

$$G_1\$G_2 \sim H_1\$H_2 \;\text{ iff }\; [\,G_1 \sim H_1 \text{ and } G_2 \sim H_2\,] \text{ or } [\,G_1 \sim H_2 \text{ and } G_2 \sim H_1\,].$$

Let

$$A_{\mathit{Iso}} = U_{\mathit{Iso}} \cap \{\, x \mid x = G_1\$G_2\#H_1\$H_2 \text{ and } H_1 = H_2 \,\}$$
$$B_{\mathit{Iso}} = U_{\mathit{Iso}} \cap \{\, x \mid x = G_1\$G_2\#H_1\$H_2 \text{ and } H_1 \not\sim H_2 \,\}.$$

Let (P, V) be any IPS for *Iso* with error probability ε. We describe a 2pfa verifier V^* such that (P, V^*) separates A_{Iso} and B_{Iso} with error probability ε. Assuming $x \in U_{\mathit{Iso}}$, V^* on input $x = G_1\$G_2\#H_1\$H_2$ simulates V on input $x' = G_1\$G_2\#H_1\$H_1$. In the notation of the previous proof, $(P, V^*)(x)$ is simulating $P(x) \leftrightarrow V(x')$. Using $x \in U_{\mathit{Iso}}$, it is not difficult to see how V^* can carry out this simulation. For example, note that, for all even i with $1 \le i \le l - 2$, the ith row

See http://www.geocities.com/stockmeyer@sbcglobal.net/ipsfigs.pdf (page 1)

Figure 1: A graph of the form $G_1\$G_2$ used in the proof of Theorem 4.2, where $k = 16$. G_1 is the subgraph induced by vertices $\{2, 4, 6\}$ and G_2 is the subgraph induced by vertices $\{12, 14, 16\}$. To simplify the example, the edges of G_1 and G_2 are not shown.

of the adjacency representation of $H_1\$H_2$ contains a unique occurrence of 111; the even numbered positions before the marker 111 in the even numbered rows constitute the adjacency representation of H_1. Rows l and $l+1$ are the only rows which consist entirely of 1's; the rows after row $(l+2)$ contain the adjacency representation of H_2, again restricted to even numbered positions of even numbered rows. To keep track of its place, V^* uses the two rows which contain all 1's and the markers 111 within rows. Further details of the simulation are left to the reader.

We now argue that (P, V^*) separates A_{Iso} and B_{Iso}. Let

$$x = G_1\$G_2\#H_1\$H_2 \in U_{Iso}$$

$$x' = G_1\$G_2\#H_1\$H_1.$$

First, if $x \in A_{Iso}$, then $H_1 = H_2$, so $x = x'$ and the argument is the same as before. Second, if $x \in B_{Iso}$, then $H_1 \not\sim H_2$. We first argue that $G_1\$G_2 \not\sim H_1\$H_1$, so $x' \notin Iso$. Assuming for contradiction that $G_1\$G_2 \sim H_1\$H_1$, we would have $G_1 \sim H_1$ and $G_2 \sim H_1$, so $G_1 \sim G_2$. Since $x \in Iso$, we have $G_1\$G_2 \sim H_1\$H_2$, and together with $G_1 \sim G_2$ this implies that $H_1 \sim H_2$, contradicting the assumption that $x \in B_{Iso}$. Viewing $P(x) \leftrightarrow V(x')$ as $P^*(x') \leftrightarrow V(x')$ as before and using $x' \notin Iso$, it follows that $P(x) \leftrightarrow V(x')$ rejects with probability at least $1 - \varepsilon$.

The proof that no 2pfa can separate A_{Iso} and B_{Iso} again uses Theorem 2.1. Note that in the adjacency representation of a graph $G = G_1\$G_2$, the bits which specify the edges in G_1 all precede the bits which specify the edges in G_2. For any k divisible by 4, let $m = k^2$, $l = k/2$, and $t = k/4-1$. Let K, G_1, and H_2 be graphs with t vertices. Let $\$K\#K\$$ denote that subword of $G_1\$K\#K\$H_2$ which begins with the first bit of row $(l+1)$ of the adjacency representation of $G_1\$K$ and ends with the last bit of row l of the adjacency representation of $K\$H_2$. Note that $\$K\#K\$$ does not depend on G_1 or H_2. By choice of m and t, it is easy to see that $|\$K\#K\$| \leq m$. Let $\mathcal{G}$ be a maximal set of pairwise nonisomorphic graphs with t vertices. $|\mathcal{G}|$ grows faster than any polynomial in m. Let

$$W_m = \{\, \$K\#K\$ \mid K \in \mathcal{G} \,\}.$$

For any $K, K' \in \mathcal{G}$ with $K \neq K'$, we have

$$K\$K\#K\$K \in A_{Iso} \quad \text{and} \quad K\$K'\#K'\$K \in B_{Iso}.$$

By Theorem 2.1, no 2pfa separates A_{Iso} and B_{Iso}. $\square$

Although not needed in the proof above, it is worth mentioning that a 2pfa can separate U_{Iso} and $Iso - U_{Iso}$. This can be proved using methods of Freivalds [10]. Therefore, V^* can recognize instances of Iso for which it can get extra information from the prover.

5 A Language With a Recognition Zero Knowledge IPS

That the graph isomorphism problem has no (recognition) zero knowledge IPS with 2pfa verifiers suggests that techniques which have been used to obtain zero knowledge IPS's with polynomial-time probabilistic Turing machine verifiers will not extend directly to 2pfa verifiers. In fact, we have no example of a language L which has a recognition zero knowledge IPS with 2pfa verifiers, but which cannot be recognized by any 2pfa alone, i.e., by any machine in $(\emptyset, \text{2pfa})$. With two restrictions on 2pfa verifiers, however, we do have such an example. The first restriction is that the verifier halt in polynomial expected time. The second restriction is that the verifier be a sweeping 2pfa (recall that this means that the input head can switch its direction of movement only when scanning an endmarker $\not{c}$). Let $\mathcal{R}$ denote the class of sweeping 2pfa verifiers that halt in polynomial expected time.

Theorem 4.1 shows that there is no recognition zero knowledge IPS for palindromes with $\mathcal{R}$ verifiers. In contrast, we now show that the unary version of palindromes has a recognition zero knowledge IPS with $\mathcal{R}$ verifiers. The unary version of palindromes is the language

$$Upal = \{\, 0^n 1^n \mid n \geq 1 \,\}.$$

Greenberg and Weiss [16] show that *Upal* cannot be recognized by any 2pfa which runs in polynomial expected time; in particular, *Upal* is recognized by no machine in $(\emptyset, \mathcal{R})$.

Consider the following IPS $(P_=, V_=)$ for *Upal*. For a desired error probability ε, let k be the smallest power of 2 such that $k \geq 1/\varepsilon$. On input $0^n 1^n$, $P_=$ sends 1^n to the verifier; that is, the ith time that the verifier requests a symbol from the prover, the prover responds with the symbol 1 if $1 \leq i \leq n$ or with the symbol $\not{c}$ if $i > n$. On input x, $V_=$ first checks that $x = 0^m 1^n$ for some m and n, and rejects if not. Assuming $x = 0^m 1^n$, $V_=$ chooses a random integer l with $1 \leq l \leq k$. Letting 1^b be the string received from the prover, $V_=$ checks whether

$$kb = lm + (k - l)n \tag{1}$$

and accepts (resp., rejects) if the answer is equal (resp., unequal). It is easy to see that this can be done by a verifier in $\mathcal{R}$. ($V_=$ keeps a counter C which can hold any integer in the interval $[-k, k]$. If C is not positive, $V_=$ requests a symbol from the prover and increments C by k if the prover's response is 1. If C is positive, $V_=$ reads a symbol of the input and decrements C by l (resp., $k - l$) if the symbol is 0 (resp., 1).)

If $x = 0^n 1^n$, then we have $b = m = n$, so (1) holds for all l, and $(P_=, V_=)(x)$ accepts with probability 1. If $x = 0^m 1^n$ with $m \neq n$, then no matter which string 1^b is received from the prover P^*, there is at most one l which satisfies (1), namely, $l = k(b - n)/(m - n)$. So $(P^*, V_=)(x)$ rejects with probability at least $1 - 1/k \geq 1 - \varepsilon$.

The difficult part is to show that this IPS is recognition zero knowledge. As some indication that this is not obvious, we remark that if the prover sends 1^n to the verifier $\lceil \log n \rceil$ times on input $0^n 1^n$, there is a verifier V^* in $\mathcal{R}$ which can determine whether or not n is a power of two, and no machine in $(\emptyset, \mathcal{R})$ can determine this alone. The proof of the first fact is easy. (The proof of Theorem 4.1 in [7] uses a similar procedure.) The second fact follows easily from results in [16] which imply that if $L \subseteq \{1\}^*$ and L is recognized by a machine in $(\emptyset, \mathcal{R})$, then L is regular. (The key fact here is the Perron-Frobenius Lemma which we state below.)

Theorem 5.1 *There is a recognition zero knowledge IPS for Upal with $\mathcal{R}$ verifiers.*

The remainder of this section is dedicated entirely to proving Theorem 5.1 by proving that $(P_=, V_=)$ satisfies the recognition zero knowledge property with $\mathcal{R}$ verifiers. Specifically, we prove the following theorem.

Theorem 5.2 *$\forall V^* \in \mathcal{R}$, $\forall \alpha > 0$, $\exists M_{V^*} \in (\emptyset, \mathcal{R})$, $\forall n$,*

$$|\Pr[(P_=, V^*)(0^n 1^n) \text{ accepts}] - \Pr[M_{V^*}(0^n 1^n) \text{ accepts}]| \leq \alpha.$$

It follows easily from this that $(P_=, V_=)$ is recognition zero knowledge. For let V^* be such that $(P_=, V^*)$ separates A and B with error probability $\varepsilon < 1/2$. Using Theorem 5.2 with $\alpha = (1/2 - \varepsilon)/2$, we obtain an $M_{V^*} \in (\emptyset, \mathcal{R})$ such that M_{V^*} separates A and B with error probability at most $\varepsilon + \alpha < 1/2$.

Remark. Theorem 5.2 and the Perron-Frobenius Lemma together show that the acceptance behavior of $(P_=, V^*)$ is essentially ultimately periodic in the following sense: $\forall V^* \in \mathcal{R}$, $\exists d = d(V^*)$, $\forall \alpha > 0$, $\exists n_\alpha$, $\forall n \geq n_\alpha$, $\forall k \geq 0$,

$$|\Pr[(P_=, V^*)(0^n 1^n) \text{ accepts}] - \Pr[(P_=, V^*)(0^{n+kd} 1^{n+kd}) \text{ accepts}]| \leq \alpha.$$

In the preliminary version of this paper [6], we give a direct proof of this.

To prove Theorem 5.2, we begin by slightly changing the problem. Since we are concerned only with inputs of the form $0^n 1^n$, and since the only information in such an input is the value n, it is convenient to replace the input by 1^n. Formally, let P be the prover which sends $1^n \cent$ to the verifier on input 1^n. That is, P's behavior on 1^n is identical to the behavior of $P_=$ on $0^n 1^n$. Now, for every $V^* \in \mathcal{R}$ there is a $U \in \mathcal{R}$ such that, for all n,

$$\Pr[(P_=, V^*)(0^n 1^n) \text{ accepts}] = \Pr[(P, U)(1^n) \text{ accepts}].$$

U on input 1^n simulates V^* on input $0^n 1^n$ with U keeping track in its finite control of whether V^* is scanning 0 or 1. Similarly, for every $M \in (\emptyset, \mathcal{R})$ with input alphabet $\{1\}$ there is an $M' \in (\emptyset, \mathcal{R})$ with input alphabet $\{0, 1\}$ such that

$$\Pr[M'(0^n 1^n) \text{ accepts}] = \Pr[M(1^n) \text{ accepts}].$$

It therefore suffices to prove the following.

Theorem 5.3 *$\forall U \in \mathcal{R}$, $\forall \alpha > 0$, $\exists M \in (\emptyset, \mathcal{R})$, $\forall n$,*

$$|\Pr[(P, U)(1^n) \text{ accepts}] - \Pr[M(1^n) \text{ accepts}]| \leq \alpha.$$

Before proving the theorem we must give several definitions and preliminary results.

First, it will be convenient to describe M as a sweeping 2pfa with rational transition probabilities, possibly other than 0, 1/2, and 1. It is easy to see, however, that general rational transition probabilities can be simulated by fair coin tosses while increasing the expected running time of M by at most a constant factor.

Next we review the basic definition of finite state Markov chains since these play an important part in the rest of the proof. Further details of Markov chain theory will be introduced as needed. A (*finite state, discrete time*) *Markov chain* consists of a finite set of *states*, say $\{s_1, s_2, \ldots, s_c\}$, and a (*one-step*) *transition probability* h_{ij} for each pair of states s_i and s_j such that, for each fixed i, $\sum_j h_{ij} = 1$. We view a Markov chain as describing a finite state discrete time process. If the process is in state s_i at time t, then the process moves to state s_j at time $t+1$ with probability h_{ij}. The $c \times c$ matrix of transition probabilities $H = \{h_{ij}\}_{i,j=1}^{c}$ is the *transition matrix* of the Markov chain. We sometimes identify a Markov chain with its transition matrix. With each Markov chain H, we associate a labelled directed graph $G(H)$ as follows. The vertices of $G(H)$ correspond to the states of H. There is an edge directed from s_i to s_j iff $h_{ij} > 0$; if there is such an edge, it is labelled h_{ij}. The Markov chain H is *irreducible* if $G(H)$ is strongly connected (i.e., for every s_i and s_j, there is a directed path from s_i to s_j).

The following lemma for discrete time finite state Markov chains, which follows from a theorem of Perron and Frobenius (see Seneta [19], Chapters 1 and 4), plays a crucial role in the proof. The result says that the m-step behavior of a Markov chain H depends only on the congruence of m modulo a certain number $\pi(H)$, to within an error term which decreases exponentially with m.

Perron-Frobenius Lemma. *Let H be the transition matrix of a c-state Markov chain having rational transition probabilities. Then there exist a positive integer $\pi = \pi(H)$, a positive real number $\rho = \rho(H) < 1$, and rational numbers $p_k(i,j)$ for $1 \le i, j \le c$ and $0 \le k < \pi$, such that for all i, j, and k, and all $m \ge 1$ with $m \equiv k \bmod \pi$,*

$$H^m(i,j) = p_k(i,j) \pm O(\rho^m).$$

The constant $\pi(H)$ given by the Perron-Frobenius Lemma is called the *periodicity constant of H*. We will call the probabilities $p_k(i,j)$ the *k-periodic stationary* transition probabilities of H.

It will be useful to employ three different views of the verifier. We may think of the verifier as a machine, whose behavior can be described. For example, we may say things like "the verifier moves its head to the right end of the input tape." The verifier is also specified by a transition diagram, which is a labelled directed graph where vertices correspond to states and where each edge, denoting a state transition, is labelled with a symbol (either 1 or ¢) and a transition probability in the usual fashion. Finally, we may think of the verifier as a *Markov chain aggregate*: two or more discrete time Markov chains superimposed on the same state space. Each Markov chain in the aggregate describes state transitions on a given symbol. For the purposes of proving our theorem, each verifier U can be described by the Markov chain aggregate $H^U = \{H_1^U, H_{¢}^U\}$, where H_1^U describes U's transitions on the symbol 1 and $H_{¢}^U$ describes U's transitions on the symbol ¢.

When the prover P sends a symbol to the verifier, we say the verifier *consumes* the symbol. Recall that on input 1^n, P will send exactly n 1's to the verifier, followed by the endmarker, so it will make sense to speak about whether or not the verifier has consumed the endmarker, or what fraction of the string 1^n has been consumed. In this terminology, a *consuming state* is precisely a communication state.

See http://www.geocities.com/stockmeyer@sbcglobal.net/ipsfigs.pdf (page 2)

Figure 2: The transition diagram of U is partitioned into the Before Machine $\mathcal{B}$ (the computation before the endmarker is consumed) and the After Machine $\mathcal{A}$ (the computation after the endmarker is consumed). Reading states are labeled R and consuming states are labeled C. Every endmarker-transition from a consuming state in $\mathcal{B}$ goes to a state in $\mathcal{A}$. There are no consuming states in $\mathcal{A}$.

Next we adopt three simplifying assumptions about the structure of the verifier U. An arbitrary verifier can be easily modified to conform to these assumptions. First, we assume that the input head of U only moves from left to right. Whenever the head scans the right endmarker $\not{c}$, the head is immediately moved to the left endmarker $\not{c}$ so that the machine is ready for another left-to-right sweep. Such a machine is easily constructed from an arbitrary one by a simulation argument of the same genre made above.

Second, we assume that all transitions from every reading state advance the head one symbol to the right. The modifications to an arbitrary verifier are straightforward, although lengthy to give in detail. Briefly, the modification can be done in two steps. In the first step, the verifier U is transformed so that, for every reading state q, either all transitions from q are right-moving or all are stationary. The second step is to remove all reading states q of the latter type, adjusting the transition probabilities as necessary. (A similar state removal process is described below.) Even though the original verifier makes probabilistic choices by flipping a fair coin, the modified verifier could have rational transition probabilities other than 0, 1/2, and 1. Recall that the input head does not move during a transition from a communication state. Therefore, the head moves during a transition from state q iff q is a reading state.

The third simplifying assumption is that the transition diagram of U is partitioned into two disjoint subsets, called $\mathcal{B}$ and $\mathcal{A}$ (for Before and After), so that U begins the computation in $\mathcal{B}$ and moves to $\mathcal{A}$ upon consumption of the endmarker (see Figure 2). Again without loss of generality, we assume that the accepting and rejecting states are in $\mathcal{A}$ (so with probability 1 the verifier enters $\mathcal{A}$). We also assume that U can enter the accepting or rejecting state only via a $\not{c}$-transition from a reading state. In particular, this implies that after entering $\mathcal{A}$ the verifier completes at least one sweep before halting.

Let $\mathcal{B}_1$ denote the subgraph consisting of the vertices in $\mathcal{B}$ and the transitions on reading or consuming 1. $\mathcal{B}_1$ may be disconnected. A *1-closed subset* is a set of states S in $\mathcal{B}_1$ having the properties that (i) the subgraph of $\mathcal{B}_1$ induced by S is strongly connected, and (ii) there is no 1-transition (i.e., edge of $\mathcal{B}_1$) from a state in S to a state not in S. These conditions imply that the subgraph of $\mathcal{B}_1$ induced by S defines an irreducible Markov chain. Since $\mathcal{B}_1$ is fixed, we refer to this Markov chain simply as S. States that are not contained in any 1-closed subset are said to be *1-transient*, or simply *transient*.

In order to construct a machine $M \in (\emptyset, \mathcal{R})$ which simulates the behavior of (P, U) (to within some small error) it will be important to have, for each 1-closed subset S, information about the distribution of the number of 1's consumed in S given that some number m of 1's are read in S. In particular, we show that each 1-closed subset has an associated *consuming ratio*, defined as

$$\mu = \lim_{m\to\infty} \frac{\mathbf{E}[\ \#\ 1\text{'s consumed} \mid m\ 1\text{'s read}]}{m}$$

If a 1-closed subset has no consuming states, then its consuming ratio is 0. Such a 1-closed subset is said to be *lean*. By convention, a 1-closed subset with no reading states has consuming ratio ∞.

Consider an irreducible Markov chain H with state set S and rational transition probabilities, and let $s_0 \in S$. For integer $t \geq 1$, let X_t be the random variable giving the state of H at time t, given that H is started in state s_0 at time 1. Let $S = R \cup C$, where $R \cap C = \emptyset$. (Think of R and C as the set of reading and consuming states, respectively, of S.) For all subsets $T \subseteq S$ and for all $s \in S$ we let

$$1_T(s) = \begin{cases} 1 & \text{if } s \in T \\ 0 & \text{if } s \notin T\ . \end{cases}$$

Let

$$\mathcal{R}(t; s_0) = \sum_{i=1}^{t} 1_R(X_i) \quad \text{and} \quad \mathcal{C}(t; s_0) = \sum_{i=1}^{t} 1_C(X_i)\ .$$

Let $T_m^R(s_0)$ denote time of the mth entry into R, given that H is started in state s_0 at time 1; that is,

$$T_m^R(s_0) = \min\{\, t \mid \mathcal{R}(t; s_0) = m\,\}.$$

Clearly $T_m^R(s_0) \geq m$. By definition, $\mathcal{C}(T_m^R(s_0); s_0)$ is the total amount of time spent in states in C up to and including time $T_m^R(s_0)$. Abbreviate $\mathcal{C}(T_m^R(s_0); s_0) = \mathcal{C}_m(s_0)$. Note also that $\mathcal{C}_m(s_0) = T_m^R(s_0) - m$, since S is partitioned into C and R.

Notation. We let $\mathcal{N}(\mu, \sigma^2)$ denote a random variable having the normal distribution with mean μ and variance σ^2. We let $\Rightarrow$ denote convergence in distribution: if $Y_1, Y_2, \ldots$ is a sequence of random variables and Y is a random variable, then $Y_m \Rightarrow Y$ if, for all real y, $\lim_{m\to\infty} \Pr[Y_m \leq y] = \Pr[Y \leq y]$.

Central Limit Theorem. *If $R \neq \emptyset$, then there exist nonnegative constants $\mu = \mu(H)$, $\sigma = \sigma(H)$, and $\xi = \xi(H)$, with μ rational, such that for any $s_0 \in S$,*

1. *if $\sigma = 0$ then, for all m,*

$$\Pr[\,|\mathcal{C}_m(s_0) - \mu m| \leq \xi\,] = 1.$$

2. *if $\sigma > 0$ then*

$$\frac{\mathcal{C}_m(s_0) - \mu m}{\sigma\sqrt{m}} \Rightarrow \mathcal{N}(0, 1).$$

In other words, in the first case, the distribution of $\mathcal{C}_m(s_0)$ is tightly concentrated about its mean, within some constant independent of m. In the second case, $\mathcal{C}_m(s_0)$ is approximately normally distributed with mean μm and standard deviation $\sigma\sqrt{m}$. In our applications of the Central Limit Theorem, the Markov chain H will be the one induced by S in $\mathcal{B}_1$, so we will speak in terms of S

See http://www.geocities.com/stockmeyer@sbcglobal.net/ipsfigs.pdf (page 3)

Figure 3: An example of $\mathcal{B}$. Reading states are labeled R and consuming states are labeled C. A transition labeled with a number p means that this transition is taken with probability p upon reading or consuming a 1. To simplify the example, a transition labeled with the endmarker from a reading state is taken with probability 1 upon reading the endmarker. Endmarker-transitions from consuming states are not shown since they all go to $\mathcal{A}$. Endmarker-transitions from transient reading states are not shown since, with high probability, such transitions are never taken. This example has four transient states and four 1-closed subsets (each 1-closed subset is enclosed by a dashed line). 1-closed subsets S_1 and S_4 are lean. The 1-closed subset S_2 has consuming ratio $\mu = 1/2$ and variance $\sigma > 0$. The 1-closed subset S_3 has consuming ratio $\mu = 1/2$ and $\sigma = 0$. The two reading states in S_4 are bad. (In this example, the bad states do not depend on n.)

rather than H, for example, writing $\mu(S)$ in place of $\mu(H)$. The different types of 1-closed subsets are illustrated in Figure 3. Since the consequences of the Central Limit Theorem do not depend on the starting state s_0, the starting state is elided in the sequel.

The Central Limit Theorem follows from a related central limit theorem in the literature, and its proof is postponed to the end of this section.

Consider a sweep in which the verifier U enters a non-lean 1-closed subset S with $m-1$ 1's left to read in the sweep. S is partitioned into reading states (R) and consuming states (C). Then T_m^R is the time at which U reads the endmarker, and $\mathcal{C}_m = T_m^R - m$ is the number of of 1's consumed from the time S is entered through the end of the sweep. The constant $\mu(S)$ is actually the consuming ratio of S. Thus, if $R = \emptyset$ we adopt the convention that $\mu(S) = \infty$ and $\sigma(S) = 0$.

The Central Limit Theorem tells us something about the "deviation" in each sweep: how the actual number consumed differs from the number expected to be consumed. Our last technical lemma is used to analyze the cumulative "deviation" over some number z of sweeps.

Summation Lemma. *Let z be fixed. For each $1 \le i \le z$, let $Y^i = \{\, Y_m^i \mid m \ge 1 \,\}$ be a sequence of random variables satisfying*

$$Y_m^i \Rightarrow \mathcal{N}(0, \sigma_i^2) \ .$$

Assume further that, for each m, the random variables $Y_m^1, \ldots, Y_m^z$ are mutually independent. Let us define a new sequence of random variables $\{\, Z_m \mid m \ge 1 \,\}$ by

$$Z_m = \sum_{i=1}^{z} Y_m^i \, .$$

Then

$$Z_m \Rightarrow \mathcal{N}(0, \sum_{i=1}^{z} \sigma_i^2) \ .$$

Proof. We use the Continuous Mapping Theorem (see, e.g., Shedler [20, Appendix 1]): If $Y_m \Rightarrow Y$ and h is continuous, then $h(Y_m) \Rightarrow h(Y)$.

Abbreviate $N^i = \mathcal{N}(0, \sigma_i^2)$. By independence of the Y_m^i and the assumption that $Y_m^i \Rightarrow N^i$ for all i, we have, for all real $y^1, \ldots, y^z$, that

$$\begin{aligned} & \lim_{m\to\infty} \Pr[(Y_m^1, \ldots, Y_m^z) \leq (y^1, \ldots, y^z)] \\ = \; & \lim_{m\to\infty} \Pr[Y_m^1 \leq y^1] \Pr[Y_m^2 \leq y^2] \cdots \Pr[Y_m^z \leq y^z] \\ = \; & \prod_i \Pr[N^i \leq y^i]\,. \end{aligned}$$

Thus,

$$(Y_m^1, \ldots, Y_m^z) \Rightarrow (N^1, \ldots, N^z),$$

whence, by the Continuous Mapping Theorem,

$$Z_m = \sum_{i=1}^{z} Y_m^i \Rightarrow Z = \sum_{i=1}^{z} N^i.$$

Now, Z itself is normally distributed with mean zero (Billingsley [3], page 235, problem 20.19). Since the variance of a sum of independent random variables is the sum of the variances, the variance of Z is $\sum_i \sigma_i^2$. □

The verifier consumes nothing when in a lean closed subset, so it may make several sweeps without consuming, or consuming only a constant expected number of 1's in each sweep. In particular, we identify certain "bad states." Intuitively, a state is bad if it is a reading state and if, once the verifier has entered this state at the beginning of a sweep, all future communication with the prover occurs in 1-transient states, with probability approaching 1 as $n \to \infty$.

To define the bad states, we begin by defining a graph $G_{U,n}$. The vertices of $G_{U,n}$ correspond to the reading states in the before machine $\mathcal{B}$. Intuitively, we draw an edge from state s_i to state s_j if, when U starts a sweep in state s_i on input 1^n, then with non-negligible probability U will be in state s_j the next time that U starts a sweep, assuming that the computation does not leave $\mathcal{B}$ and that n is sufficiently large. More formally, let P' be the prover which writes 1 in the communication cell at every interaction with U. Let the reading states in $\mathcal{B}$ be $\{s_1, \ldots, s_c\}$. Let $r(i,j,n)$ be the probability that if we start U in state s_i with the head scanning the leftmost 1 of 1^n then, in the computation $(P', U)(1^n)$, U will be in state s_j the next time U enters a reading state with the head scanning the leftmost 1. By applying the Perron-Frobenius Lemma, there is a positive integer J, a positive number $\rho < 1$, and numbers $p_k(i,j)$ for $1 \leq i,j \leq c$ and $0 \leq k < J$ such that, for all $n \equiv k \bmod J$, $r(i,j,n) = p_k(i,j) \pm O(\rho^n)$. Draw an edge from s_i to s_j in $G_{U,n}$ iff $p_k(i,j) > 0$ where $k = n \bmod J$.

We can now define the good and bad states of $\mathcal{B}$ on input 1^n. The set of *good* states is the smallest set of reading states satisfying the following two properties: (i) if there is a path of (zero or more) 1-transitions in $\mathcal{B}$ from s to a non-lean 1-closed subset then s is *good*; and (ii) if there is a path in $G_{U,n}$ from s to a good state then s is *good*. A state is *bad* if it is a reading state which is not good. Note that while the property of being good depends on n, it depends only on $n \bmod J$. In the example of Figure 3, the two states in S_4 are bad (for all $n \geq 1$).

If the verifier ever starts a sweep in a bad state, then with overwhelming probability (as $n \to \infty$) it will consume only in 1-transient states in the future. On the other hand, if the verifier starts a sweep in a good reading state, then with at least some constant positive probability, the verifier will enter a non-lean 1-closed subset within some constant number of sweeps in the future, where both constants depend only on U (assuming that n is so large that the $O(\rho^n)$ term in the Perron-Frobenius Lemma is negligible compared to the smallest nonzero $p_k(i,j)$). We first prove Theorem 5.3 assuming that U has no bad states. We then show how to convert an arbitrary U to one which has no bad states.

Let U be a verifier in $\mathcal{R}$ satisfying the assumptions about its behavior and structure stated above. In particular, U has no bad states. Let $\alpha > 0$. We describe a machine $M \in (\emptyset, \mathcal{R})$ such that, for all sufficiently large n, the probability that $(P,U)(1^n)$ accepts (rejects) is within α of the probability that $M(1^n)$ accepts (rejects). To make this condition hold for all n, we can then build a table of answers into M for all the n that are not sufficiently large. The basic strategy is to have M on input 1^n simulate the computation of (P,U) on input 1^n, making all state transitions U would make with exactly the same transition probabilities, with the input head of M simulating the input head of U. The only event which M cannot simulate directly is the consumption of the endmarker. During the simulation, M will collect information (such as the sum of the consuming ratios of the 1-closed subsets entered) which will allow it to make a good guess about when to simulate consumption of the endmarker. The Central Limit Theorem is used to guess the sweep during which the endmarker will be consumed, and the Perron-Frobenius Lemma is used in simulating consumption of the endmarker during this sweep. When describing the simulation, we will assume that certain "fortuitous" events occur. If the fortuitous event does not occur, then the simulation can make an error (e.g., reject when it should accept). Therefore, we must keep track the sum of the probabilities that fortuitous events do not occur to make sure that the sum is strictly less than α. For certain of these fortuitous events, the probability of the event approaches 1 as n approaches infinity. We say that such an event occurs *with high probability.* During any simulation, we will make assumptions about fortuitous events at most a fixed constant (independent of n) number of times. Therefore, for events which happen with high probability, we do not have to keep track of the error probability explicitly, since the error probability can be made negligible by making n large enough.

We begin with four simple observations about the probabilities of certain "non-fortuitous" events.

Four Observations:

$\forall \beta > 0, \exists \psi_\beta, \exists \tau_\beta, \exists n_\beta, \forall n \geq n_\beta,$

1. On input 1^n, the probability that U (interacting with P) makes more than ψ_β sweeps before entering $\mathcal{A}$ is less than β.

2. On input 1^n, for any fixed sweep, the probability that U is in a transient state of $\mathcal{B}$ after τ_β steps from the beginning of the sweep is less than β.

3. Let χ_m be a sequence of random variables such that

$$\frac{\chi_m - \mu m}{\sigma\sqrt{m}} \Rightarrow \mathcal{N}(0,1)$$

for some $\mu > 0$ and $\sigma > 0$. Then

$$\lim_{m\to\infty} \Pr[\,|\chi_m - \mu m| > m^{2/3}\,] = 0.$$

4. Let χ_m be as in 3. Then

$$\lim_{m\to\infty} \Pr[\,|\chi_m - \mu m| < m^{1/3}\,] = 0.$$

The intuition for the first observation is that, in each sweep in which a non-lean 1-closed subset is entered, a constant fraction of the n 1's are consumed. Moreover, since U has no bad states, we expect at least some fixed positive fraction of the sweeps to enter a non-lean 1-closed subset. Thus, after an expected constant $\overline{\psi}$ number of sweeps, the verifier consumes the endmarker and enters $\mathcal{A}$. By Markov's inequality, the probability that the number of sweeps exceeds $\overline{\psi}/\beta$ is at most β. The second observation is true because, for any given sweep, the expected time to enter a 1-closed subset is bounded above by some constant. The third and fourth observations follow from the definition of the normal distribution.

Choose the constant $\psi = \psi_\beta$ where $\beta = \alpha/4$. Choose $\tau = \tau_\beta$ where $\beta = \alpha/(4\psi)$. If, during the simulation of (P, U) by M, the simulation runs for more than ψ sweeps without entering $\mathcal{A}$ or if the simulation stays in transient states of $\mathcal{B}$ for more than τ steps after the beginning of any sweep, then M immediately rejects. By choice of ψ and τ, this introduces an error of at most $\alpha/2$.

Before continuing to describe the simulation, it is useful to make another observation about the total number of 1's consumed in several sweeps. This observation is easier to state if we assume that U is interacting with the prover P' which writes 1 in the communication cell at every interaction with U; thus, with this (faulty) prover it is possible that more than n 1's are consumed. Consider a computation of $(P', U)(1^n)$ involving z sweeps where $z \le \psi$, and assume that the 1-closed subset S_i is entered during the ith sweep for $1 \le i \le z$. Assume further that $\mu(S_i) \ne \infty$ for all i, and that at most τ steps are spent in transient states during each sweep. Let χ_n be the random variable giving the total number of 1's consumed during these z sweeps. Let $\delta = \sum_{i=1}^{z} \mu(S_i)$. Then there is a constant D depending only on U such that:

If $\sigma(S_i) = 0$ for all i then, for all n,

$$\Pr[\,|\chi_n - \delta n| \le D\,] = 1. \tag{2}$$

If there is an i with $\sigma(S_i) > 0$ then there exists a $\sigma > 0$ such that

$$\frac{\chi_n - \delta n}{\sigma\sqrt{n}} \Rightarrow \mathcal{N}(0,1). \tag{3}$$

This fact follows easily from the Central Limit Theorem and the Summation Lemma. Let χ_n^i be the number of 1's consumed in the ith sweep. Let k_i be the number of steps spent in transient states during the ith sweep. By assumption, $k_i \le \tau$. In the case $\sigma(S_i) = 0$, the Central Limit Theorem implies that, with probability 1,

$$\mu(S_i)(n + 1 - k_i) - \xi(S_i) \le \chi_n^i \le k_i + \mu(S_i)(n+1) + \xi(S_i).$$

By simple manipulation this implies

$$\mu(S_i)n - D' \leq \chi_n^i \leq \mu(S_i)n + D'$$

where D' is the maximum of $\mu(S) + \xi(S) + \tau$ and $\mu(S)\tau + \xi(S)$ taken over all 1-closed S with $\mu(S) \neq \infty$. Since there are at most ψ sweeps, taking $D = \psi D'$ proves (2). Two technical points should be made concerning the application of the Summation Lemma to prove (3). First, the random variables χ_n^i, for n fixed and $1 \leq i \leq z$, are not independent. For example, the state s_0 of S_2 first entered during the second sweep can depend on the number of 1's consumed in the first sweep. But since the consequences of the Central Limit Theorem do not depend on the starting state s_0, and since we only need that χ_n is approximately normally distributed, we can model the χ_n^i by independent random variables. The second technical point concerns the handling of (i) the "deviations" $\chi_n^i - \mu(S_i)n$ during sweeps i with $\sigma(S_i) = 0$, and (ii) the 1's consumed in transient states during sweeps i with $\sigma(S_i) > 0$. As argued above, these cases contribute to $(\chi_n - \delta n)/\sqrt{n}$ an additive term of absolute value at most $D/\sqrt{n}$; this term goes to zero as $n \to \infty$.

We now describe the extra information which M keeps during the simulation. Let $\mathcal{A}_1$ be the subgraph of $\mathcal{A}$ consisting of the vertices of $\mathcal{A}$ and the transitions on reading 1. Recall that $\mathcal{A}$ has no consuming states. $\mathcal{A}_1$ defines a Markov chain. Let $\pi(\mathcal{A}_1)$ be the periodicity constant of this Markov chain. M maintains integers n_R and n_C, a rational number δ, and a bit v. At any step of the simulation, n_R is the number of 1's remaining to be read in the current sweep, computed modulo $\pi(\mathcal{A}_1)$. At the beginning of each sweep, n_R is set to $n \bmod \pi(\mathcal{A}_1)$. (This number can be computed once and stored at the beginning of the simulation.) Thereafter, n_R is decremented by 1 (mod $\pi(\mathcal{A}_1)$) every time a transition from a reading state is made. Similarly, n_C is the number of 1's remaining to be consumed (now in the entire computation, not just during the current sweep), computed modulo K, where the integer $K = K(U)$ will be chosen below. The number δ is maintained so that the expected number of 1's that have been consumed is approximately δn. Initially, $\delta = 0$. Whenever, a 1-closed subset S is entered, δ is updated to $\delta + \mu(S)$. To see that δ can be maintained by a finite state machine, note that if U has N_U 1-closed subsets, then δ can take on at most $(N_U)^\psi$ different values, independent of the input. The bit v is initially 0. Whenever a 1-closed subset S is entered, v is set to 1 if $\sigma(S) > 0$. Therefore, at any stage of the simulation, $v = 0$ iff the simulation has entered only 1-closed subsets S with $\sigma(S) = 0$.

During the simulation of each sweep of $(P, U)(1^n)$, M will decide whether to simulate consumption of the endmarker at this sweep. In most cases, this decision will be made at the point when a state of a 1-closed subset is first entered. We consider four cases. In each case, we assume that the simulation has just entered a 1-closed subset S, and that δ and v have already been updated.

Case 1. $\delta < 1$.

By either (2) (in the case $v = 0$) or (3) together with Observation 3 (in the case $v = 1$) we have, with high probability, that at most $\delta n + n^{2/3}$ 1's will have been consumed at the end of the current sweep. Since this quantity is less than n for sufficiently large n, the endmarker will not be consumed during the current sweep, with high probability. Therefore, M completes the simulation of this sweep without simulating consumption of the endmarker.

In the remaining three cases, we consider the first time that an updating of δ results in $\delta \geq 1$. Some additional notation will be useful. Let $\mu = \mu(S)$. Let δ' be the previous value of δ (before updating), so $\delta' + \mu = \delta$, $\delta' < 1$, and $\mu > 0$. Also define two random variables N_C and N_R with respect to the computation $(P, U)(1^n)$. Let N_C be the number of 1's remaining to be consumed at the point when S is first entered in the current sweep. Let N_R be the number of 1's remaining to be read in the current sweep at the point when the endmarker is consumed.

Case 2. $\delta > 1$.

In this case we have, with high probability, that the endmarker will be consumed during the current sweep. Since $\delta' < 1$, we can choose a c such that $0 < c < (1 - \delta')$. Arguing as in Case 1, we have $N_C \geq n - (\delta' n + n^{2/3}) \geq cn$ with high probability for all sufficiently large n. To bound N_R, let m_0 be the number of 1's which have been read in the current sweep, at the point when the endmarker is consumed. If $\mu = \infty$, then $N_R \geq n - \tau$. If $\mu \neq \infty$, then roughly we expect $\delta' n + \mu m_0 = n$, or $m_0 = ((1 - \delta')/\mu)n$. Since $\delta' + \mu = \delta > 1$ gives $\mu > 1 - \delta'$, we can write this as $m_0 = r'n$ where $r' < 1$. Since $N_R = n - m_0$, by choosing $0 < r < 1 - r'$ we have $N_R \geq rn$ with high probability. The bounds $N_C \geq cn$ and $N_R \geq rn$ (with high probability) allow us to use the Perron-Frobenius Lemma to simulate consumption of the endmarker. This is done as follows.

Let W be the Markov chain induced in $\mathcal{B}_1$ by S. We transform W to a new Markov chain $C(S)$ with the property that t steps of $C(S)$ correspond to t consuming steps within S; although $C(S)$ contains no reading states, the information n_R is retained. The transformation is done in two steps. In the first step, let W' be obtained by augmenting W to keep track of n_R, the number of 1's remaining to be read, computed modulo $\pi(\mathcal{A}_1)$ where $\pi(\mathcal{A}_1)$ is the periodicity constant of $\mathcal{A}_1$. Thus every state of W' has the form $[s, a]$, where s is a state of W and $0 \leq a < \pi(\mathcal{A}_1)$. If s is a reading (resp., consuming) state of W and there is a transition with probability p from s to s' in W, then there is a transition with probability p from $[s, a]$ to $[s', a - 1 \bmod \pi(\mathcal{A}_1)]$ (resp., $[s', a]$) for every a. Let W'' be obtained from W' by suppressing the reading states. Specifically, to suppress a reading state s requires two steps. If $w'_{s,s} > 0$, this "self-loop" is first eliminated. (Initially there may be self-loops only if $\pi(\mathcal{A}_1) = 1$, while if $\pi(\mathcal{A}_1) \neq 1$ self-loops may be introduced later in the state-suppression procedure.) For each $i \neq s$, replace the transition probability $w'_{s,i}$ by $w'_{s,i}(1 - w'_{s,s})^{-1}$, and set $w'_{s,s} = 0$. (Since W is irreducible, we always have $w'_{s,s} \neq 1$.) This eliminates the self-loop. Now for each $i, j \neq s$, define the entries of W'' by

$$w''_{i,j} = w'_{i,j} + w'_{i,s} w'_{s,j}.$$

W'' has no row or column for s. To delete all reading states we continue in this fashion, suppressing the next reading state by working on the new matrix, until all reading states are suppressed. The resulting Markov chain is $C(S)$. Let $K(S)$ denote the periodicity constant of $C(S)$. Assume K has been chosen so that $K(S)$ divides K for all 1-closed S.

Let s_1 be the state of S first entered. To simulate consumption of the endmarker, we would like to run $C(S)$ for N_C steps, starting in state $[s_1, n_R]$ and reaching some state $[s_2, a_2]$. By the Perron-Frobenuis Lemma, we can get a very good approximation by choosing $[s_2, a_2]$ according to the periodic stationary transition probabilities of $C(S)$. This requires knowing only $N_C \bmod K(S)$, rather than N_C itself. Having chosen $[s_2, a_2]$ and simulated endmarker consumption by taking a $\cent$-transition from s_2 (reaching some state s_3), we can get a very good aproximation to the state s_4 of U, after all 1's have been read in the current sweep, by choosing s_4 according to the periodic stationary transition probabilities of $\mathcal{A}_1$; this can be done since we know $a_2 = N_R \bmod \pi(\mathcal{A}_1)$. Recall that n_C is the number of 1's remaining to be consumed, computed modulo K, and n_R is the number of 1's remaining to be read in the current sweep, computed modulo $\pi(\mathcal{A}_1)$. Consuming of the endmarker is simulated by the following procedure.

Consume Endmarker. When the procedure is called, the simulation has just entered a 1-closed subset S. Let s_1 be the state of S just entered, and let $a_1 = n_R$. Let $k = n_C \bmod K(S)$; since $K(S)$ divides K and $n_C = N_C \bmod K$, we have $k = N_C \bmod K(S)$. Let $p_k(\cdot, \cdot)$ be the k-periodic stationary transition probabilities of $C(S)$. Choose a pair $[s_2, a_2]$, where $[s_2, a_2]$ is chosen with probability $p_k([s_1, a_1], [s_2, a_2])$. Simulate a $\not{c}$-transition of U from the consuming state s_2 (this is the simulation of endmarker consumption), to reach some state s_3 of $\mathcal{A}$. Let $l = a_2$, and let $q_l(\cdot, \cdot)$ be the l-periodic stationary transition probabilities of $\mathcal{A}_1$. Choose a state s_4 of $\mathcal{A}$, where s_4 is chosen with probability $q_l(s_3, s_4)$. Simulate a $\not{c}$-transition from s_4 (this simulates reading of the endmarker at the end of the current sweep). The simulation is now in some state s_5 of $\mathcal{A}$ with the head scanning the leftmost 1 of 1^n. This completes the description of the procedure.

By our bounds on N_C and N_R, this procedure introduces an error of at most $O(\rho^{cn} + \rho^{rn})$ for some $\rho < 1$ and $c, r > 0$. Therefore, the error goes to zero as $n \to \infty$. This completes the description of Case 2.

Case 3. $\delta = 1$ and $v = 1$.

In this case, we show that the endmarker is consumed during the current sweep with probability approaching 1/2. Moreover, in the case that the endmarker is consumed during the current sweep, for any constant $\varepsilon > 0$ we argue that N_R (the distance of the reading head from the right endmarker at the time of endmarker consumption) is $\Omega(n^{1/2})$ with probability at least $1 - \varepsilon$.

For a fixed real number $\lambda \geq 0$, let $\chi_n(\lambda)$ be the number of times consuming states have been entered (during the entire computation) at the time when $n - \lceil \lambda\sqrt{n} \rceil$ 1's have been read in the current sweep. Since $v = 1$, by the Central Limit Theorem and the Summation Lemma, we have (as in (3))

$$\frac{\chi_n(\lambda) - (\delta' n + \mu(n - \lambda\sqrt{n}))}{\sigma\sqrt{n}} \Rightarrow \mathcal{N}(0, 1) \tag{4}$$

for some $\sigma > 0$. For a real number y, let $N(y) = \Pr[\mathcal{N}(0,1) \leq y]$. Using $\delta' + \mu = 1$ and the definition of convergence in distribution, (4) implies, for every fixed y,

$$\lim_{n\to\infty} \Pr[\,\chi_n(\lambda) \leq y\sigma\sqrt{n} + n - \mu\lambda\sqrt{n}\,] = N(y).$$

Setting $y = \mu\lambda/\sigma$,

$$\lim_{n\to\infty} \Pr[\,\chi_n(\lambda) \leq n\,] = N(\mu\lambda/\sigma). \tag{5}$$

But $\Pr[\,\chi_n(\lambda) \leq n\,]$ is just the probability that the endmarker has *not* been consumed at the time when $n - \lceil \lambda\sqrt{n} \rceil$ 1's have been read in the current sweep. Setting $\lambda = 0$ in (5) proves that the endmarker is consumed during the current sweep with probability approaching $1 - N(0) = 1/2$ as $n \to \infty$. (We can assume, with high probability, that the endmarker is not consumed when the head is scanning the endmarker.)

To bound N_R, first note that $\Pr[\,N_R < \lambda\sqrt{n}\,]$ is the probability that the endmarker has not been consumed when $n - \lceil \lambda\sqrt{n} \rceil$ 1's have been read (i.e., $\chi_n(\lambda) \leq n$) and the endmarker has been consumed when n 1's have been read (i.e., $\chi_n(0) > n$). Note also that, for any $\lambda \geq 0$, the event "$\chi_n(0) \leq n$" implies the event "$\chi_n(\lambda) \leq n$". Therefore,

$$\begin{aligned} \lim_{n\to\infty} \Pr[\,N_R < \lambda\sqrt{n}\,] &= \lim_{n\to\infty} \Pr[\,\chi_n(\lambda) \leq n \text{ and } \chi_n(0) > n\,] \\ &= N(\mu\lambda/\sigma) - N(0). \end{aligned}$$

But for any $\varepsilon > 0$ there is a $\lambda > 0$ such that $N(\mu\lambda/\sigma) - N(0) \leq \varepsilon$.

Therefore, M decides whether or not to simulate endmarker consumption during the current sweep by flipping a fair coin. If the outcome is heads, then M calls Consume Endmarker. To bound the error probability we first note that, with high probability, $N_C = \Omega(n)$; the reasoning is exactly as in Case 2. Using the bound on N_R derived above with $\varepsilon = \alpha/8$, the error approaches $O(\rho^{\Omega(n)} + \rho^{\Omega(\sqrt{n})}) + \alpha/8$.

If the outcome of the coin toss is tails, then M completes the current sweep without simulating endmarker consumption. Given that the endmarker is not consumed in this sweep, (3) together with Observations 3 and 4 imply that $n^{1/3} \leq N_C \leq n^{2/3}$ with high probability at the end of this sweep. M calls Consume Endmarker the next time the simulation enters a 1-closed subset S' with $\mu(S') > 0$. In this case, we have $N_C = \Omega(n^{1/3})$ and $N_R = \Omega(n)$ with high probability.

Case 4. $\delta = 1$ and $v = 0$.

In this case we cannot use the Perron-Frobenius Lemma to simulate endmarker consumption, since we can have $N_R = O(1)$ with probability 1. But since the simulation has entered only 1-closed subsets with $\sigma = 0$, M will be able to simulate endmarker consumption exactly. Let $\gamma_C(m)$ be the number of 1's remaining to be consumed, given that m 1's remain to be read in the current sweep. Arguing as in the proof of (2) we have, with probability 1,

$$|\,(n - \gamma_C(m)) - (\delta' n + \mu(n - m))\,| \leq D$$

for some positive constant D. Rearranging terms and using $\delta' + \mu = 1$,

$$\frac{\gamma_C(m)}{\mu} - \frac{D}{\mu} \leq m \leq \frac{\gamma_C(m)}{\mu} + \frac{D}{\mu}. \tag{6}$$

$\gamma_C(m) = 0$ at the point when the endmarker is consumed. Setting $\gamma_C(m) = 0$ in (6) gives $m \leq D/\mu$.

Assume K has been chosen so that $K \geq 2D + 1$. M simulates a $\cancel{c}$-transition (i.e., consumption of the endmarker) the first time that (i) the simulation is in a consuming state of U, (ii) $n_C = 0$, and (iii) at most D/μ 1's remain to be read in the current sweep. (The implementation of this requires some simple finite-state programming since M cannot check condition (iii) immediately. Whenever conditions (i) and (ii) are satisfied, the simulation splits into two parts which are run in parallel: the first part simulates a $\cancel{c}$-transition and continues the simulation in $\mathcal{A}$, while the second part simulates a 1-transition and starts counting the number of 1's read to check (iii). If (iii) checks true, then the second part is abandoned; otherwise, the first part is abandoned.) To argue that this is correct, suppose that M incorrectly consumes the endmarker at a time when some positive number of 1's remain to be consumed. Let m be the number of 1's that remain to be read in the current sweep at this time. Since $\gamma_C(m) \equiv n_C \equiv 0 \bmod K$ at this point and $\gamma_C(m) > 0$ by assumption, we have $\gamma_C(m) \geq K \geq 2D + 1$. Now (6) gives

$$m \geq \frac{2D+1}{\mu} - \frac{D}{\mu} > \frac{D}{\mu}.$$

But $m > D/\mu$ contradicts condition (iii).

If conditions (i), (ii), and (iii) are never met simultaneously in the current sweep, M continues to the next sweep and consumes the endmarker the first time that (i) and (ii) are met.

This completes the description of the simulation up to the point when a state of $\mathcal{A}$ is entered. The rest of the simulation is relatively straightforward since $\mathcal{A}$ has no consuming states. It is not clear, however, that M can directly simulate $\mathcal{A}$; since the previous part of the simulation has altered the probabilities with which states of $\mathcal{A}$ are entered, it is not immediately obvious that the direct simulation runs in polynomial expected time. Therefore, M will use $q_l(\cdot,\cdot)$, the periodic stationary transition probabilities of $\mathcal{A}_1$, to do an "indirect" simulation. (A similar simulation is used shortly to eliminate bad states.) Whenever M is in state s of $\mathcal{A}$ scanning the leftmost 1 of 1^n, it chooses s' with probability $q_l(s,s')$ where $l = n \bmod \pi(\mathcal{A}_1)$. M then simulates a $\cent$-transition from s' to reach some state s'', sets $s = s''$, and continues. (The input head is not needed.) This procedure is modeled by a Markov chain with states $[s,i]$ for $s \in \mathcal{A}$ and $i = 1,2$; the transition probability from $[s,1]$ to $[s',2]$ is $q_l(s,s')$; the transition probability from $[s',2]$ to $[s'',1]$ is the probability of a $\cent$-transition from s' to s''; and other transition probabilities are zero. If the simulation ever enters a state s such that neither the accepting state nor the rejecting state is reachable from s in the associated graph of this Markov chain, then M rejects. It follows that the simulation takes constant expected time (depending only on U).

To bound the additional error introduced, let A_1 ($A_{\cent}$) be the matrix H_1^U ($H_{\cent}^U$) of 1-transitions ($\cent$-transitions) restricted to states in $\mathcal{A}$. Letting a be the number of states of $\mathcal{A}$, A_1 and $A_{\cent}$ are $a \times a$. Recall that $l = n \bmod \pi(\mathcal{A}_1)$ and let Q be the matrix with entries $q_l(i,j)$ for $1 \le i,j \le a$. For a matrix E, let $\|E\|$ be the maximum absolute value of an entry of E. By the Perron-Frobenius Lemma, we can write $A_1^n = Q + E$ for some matrix E with $\|E\| = O(\rho^n)$ where $\rho < 1$ is constant. Note that if T is any $a \times a$ transition probability matrix (i.e., each row sum of T equals 1) and W is any $a \times a$ matrix, then $\|TW\| \le \|W\|$ and $\|WT\| \le a\|W\|$ Using this fact, it is easy to show by induction on t that

$$(A_1^n A_{\cent})^t = ((Q+E)A_{\cent})^t = (QA_{\cent})^t + E_t,$$
$$\text{where } \|E_t\| \le at\|E\| = O(t\rho^n);$$

details are left to the reader. But $(A_1^n A_{\cent})^t$ models the "direct" simulation of $\mathcal{A}$ for t sweeps, whereas $(QA_{\cent})^t$ models the indirect simulation using the Perron-Frobenius Lemma as described above. Therefore, the error introduced by the indirect simulation is $O(t\rho^n)$. Since U runs in polynomial expected time, t is polynomial in n and the additional error goes to zero as $n \to \infty$.

We now consider the case in which $\mathcal{B}$ has bad states. Our approach in this case is to modify U to obtain a new verifier $\tilde{U}$ such that $\tilde{U}$ has no bad states and the accepting probability of $(P,\tilde{U})(1^n)$ approaches the accepting probability of $(P,U)(1^n)$ as $n \to \infty$. The basic idea is that, once a bad state is entered, the many-step transition behavior of lean 1-closed subsets will be simulated using periodic stationary transition probabilities. Let L be the periodicity constant of H_1^U. Let $q_l(\cdot,\cdot)$ denote the l-periodic stationary transition probabilities of H_1^U for $0 \le l < L$.

On input 1^n, $\tilde{U}$ first computes and stores $n \bmod J$ where J is the constant in the definition of "bad state" such that the property of being bad depends only on $n \bmod J$. This permits $\tilde{U}$ to identify bad states. $\tilde{U}$ then computes and stores $n \bmod L$. $\tilde{U}$ then simulates U directly, i.e., making exactly the same state transitions U would make, as long as U does not enter a bad reading state at the start of a sweep. If U does enter a bad reading state s_1 at the start of a sweep, $\tilde{U}$ executes the following procedure *Transient*(s_1). (When this occurs, the input head is not used further.)

Transient(s_1). Set $n_R = n \bmod L$. The simulation continues in a direct fashion, except that transitions from reading states do not cause head movement; instead, $\tilde{U}$ simulates each such transition as though the symbol 1 were read, and n_R is decremented by 1 mod L as before. This direct simulation continues until either a state of $\mathcal{A}$ is entered or a 1-closed subset S is entered. In the first case, $\tilde{U}$ then simulates $\mathcal{A}$ indirectly, using the periodic stationary transition probabilities of $\mathcal{A}$ as described above. In the second case, if S is not lean (S contains a consuming state), then $\tilde{U}$ rejects; by the definition of bad states, this occurs with probability $O(\rho^n)$ for some constant $\rho < 1$. If S is lean (contains only reading states), let s_2 be the state of S first entered, and execute *Indirect*(s_2).

Indirect(s_2). Set $l = n_R$ and choose a state $s_3 \in S$ with probability $q_l(s_2, s_3)$. Simulate a $\not{c}$-transition from (the reading state) s_3 to reach a state s_4. (The simulated configuration is now the one where U is in state s_4 with the head scanning the leftmost 1.) Execute *Transient*(s_4).

The machine $\tilde{U}$ has no bad states, but has one or more "new" (i.e., not present in U) 1-closed subsets with consuming ratio ∞.

The error introduced by this simulation can be bounded in a way very similar to the bound obtained above in the indirect simulation of $\mathcal{A}$. One minor difference is that, since U can run for a polynomial number of sweeps before entering $\mathcal{A}$, we can no longer argue that U stays in transient states for at most $O(1)$ steps of any sweep. However, it is still true that, on input 1^n, U stays in transient states for at most $n/2$ steps of each sweep, with probability at least $1 - \rho^n$ for some constant $\rho < 1$. It then follows as above that

$$| \Pr[\,(P, U)(1^n) \text{ accepts}\,] - \Pr[\,(P, \tilde{U})(1^n) \text{ accepts}\,] \,| = O(p(n)\, \rho^{n/2})$$

for some polynomial $p(n)$ and some constant $\rho < 1$.

This completes the proof of Theorem 5.3. $\square$

Proof of Central Limit Theorem.

The Central Limit Theorem stated above was originally proved in [6]. We give here a slightly simpler proof. Let H be an irreducible Markov chain with rational transition probabilities and state set S partitioned into sets R and C with R nonempty. We use the following theorem which follows from Theorem 1 and its proof in Chung [4, §I.16]. The theorem gives information about the distribution of $\mathcal{R}(t; s_0)$, the number of times states in R are entered, given that H is started in state s_0 and run for t steps.

Theorem 5.4 *There exist nonnegative constants $\mu' = \mu'(H)$, $B = B(H)$, and $\xi' = \xi'(H)$, with $0 < \mu' \leq 1$ and μ' rational, such that for any starting state s_0,*

1. *if $B = 0$ then, for all t,*
$$\Pr[\,|\mathcal{R}(t; s_0) - \mu' t| \leq \xi'\,] = 1.$$

2. *if $B > 0$, then*
$$\frac{\mathcal{R}(t; s_0) - \mu' t}{B\sqrt{t}} \Rightarrow \mathcal{N}(0, 1).$$

$\square$

Since the starting state s_0 is irrelevant, abbreviate $\mathcal{R}_t = \mathcal{R}(t; s_0)$ and $\mathcal{T}_m = T_m^R(s_0)$. Recall that $\mathcal{T}_m$ is the time of the mth entry into R. We use Theorem 5.4 in "reverse" to obtain information about the distribution of $\mathcal{T}_m$, and then use the identity $\mathcal{C}_m = \mathcal{T}_m - m$. The two cases of the Central Limit Theorem correspond to the two cases of Theorem 5.4. In both cases, set $\mu = (1 - \mu')/\mu'$, so $\mu' = (1 + \mu)^{-1}$.

1. $B = 0$. Set $\sigma = 0$ and $\xi = \xi'/\mu'$. By Theorem 5.4, for all m,

$$\Pr[\,|m - \mu'\mathcal{T}_m| \leq \xi'\,] = 1.$$

Subtituting $\mu' = (1 + \mu)^{-1}$, $\xi' = \mu'\xi$, and $\mathcal{T}_m = \mathcal{C}_m + m$, this is equivalent to

$$\Pr[\,|\mu m - \mathcal{C}_m| \leq \xi\,] = 1.$$

2. $B > 0$. In this case, we follow a calculation from Feller [9, §XIII.6] to show that $\mathcal{T}_m$ is approximately normally distributed with mean m/μ' and standard deviation $\sigma\sqrt{m}$ for some positive σ. By definition, for all positive integers t and m,

$$\Pr[\,\mathcal{R}_t < m\,] = \Pr[\,\mathcal{T}_m > t\,].$$

Define $N(x) = \Pr[\mathcal{N}(0, 1) < x\,]$. Fix an arbitrary real number x. Let $m \to \infty$ and $t = t(m) \to \infty$ so that

$$\frac{m - \mu' t}{B\sqrt{t}} \to x, \tag{7}$$

where $\to$ denotes limiting behavior as $m, t \to \infty$. Note that (7) implies $m/t - \mu' \to xBt^{-1/2} \to 0$, so $m/t \to \mu'$. Now

$$\begin{aligned}
\Pr[\,\mathcal{T}_m > t\,] &= \Pr[\,\mathcal{R}_t < m\,] \\
&\to \Pr[\,\mathcal{R}_t < \mu' t + xB\sqrt{t}\,] \quad \text{by (7)} \\
&= \Pr\left[\frac{\mathcal{R}_t - \mu' t}{B\sqrt{t}} < x\right] \\
&\to N(x) \quad \text{by Theorem 5.4.}
\end{aligned}$$

Define $\mathcal{T}_m^*$ by $\mathcal{T}_m^* = (\mu'\mathcal{T}_m - m)\sqrt{\mu'/(mB^2)}$. By definition,

$$\mathcal{T}_m > t \ \text{ iff } \ \mathcal{T}_m^* > \frac{\mu' t - m}{B\sqrt{t}}\sqrt{\frac{\mu' t}{m}}. \tag{8}$$

By (7), and since $m/t \to \mu'$, the expression in (8) approaches $-x$. Since $\Pr[\,\mathcal{T}_m > t\,] \to N(x)$, we have $\Pr[\,\mathcal{T}_m^* > -x\,] \to N(x)$. Setting $y = -x$,

$$\Pr[\,\mathcal{T}_m^* \leq y\,] \to 1 - N(-y) = N(y). \tag{9}$$

Let $\sigma = B(\mu')^{-3/2}$. Substituting the definition of $\mathcal{T}_m^*$ into (9) we have

$$\Pr\left[\frac{\mathcal{T}_m - m/\mu'}{\sigma\sqrt{m}} \leq y\right] \to N(y).$$

Finally, substituting $\mathcal{T}_m = \mathcal{C}_m + m$ gives

$$\Pr\left[\frac{\mathcal{C}_m - \mu m}{\sigma\sqrt{m}} \leq y\right] \to N(y).$$

□

6 Open Questions

1. Is there a language L which has a recognition zero knowledge IPS with 2pfa verifiers but which cannot be recognized by any machine in $(\emptyset, \text{2pfa})$?

2. Investigate perfect zero knowledge and computational zero knowledge with 2pfa verifiers.

As noted in §3, if (P, V) is perfect zero knowledge then (P, V) is recognition zero knowledge. The converse is not true, at least for verifiers in the class $\mathcal{R}$ considered in §5. The IPS $(P_=, V_=)$ for *Upal*, shown to be recognition zero knowledge in Theorem 5.2, is not perfect zero knowledge. Specifically, let $b(n, k)$ be the probability that, in n tosses of a fair coin, there are k heads. It is easy to construct a V^* in $\mathcal{R}$ with output alphabet $\{1\}$ such that, for all even n and all $d \geq 1$,

$$\Pr[\ (P_=, V^*)(0^n 1^n) \text{ outputs } 1^d\ ] = b(n, d + n/2).$$

Using techniques as in the proof of Theorem 5.2, it is not hard to show that no $M_{V^*} \in (\emptyset, \mathcal{R})$ can produce this output distribution alone.

This could be interpreted as evidence that perfect zero knowledge is too strong a notion when space-bounded IPS's are used for language recognition (as opposed to cryptographic applications). Even though V^*, with help from $P_=$, can produce an output distribution which no machine in $(\emptyset, \mathcal{R})$ can produce, Theorem 5.2 shows that V^* cannot use this distribution to solve any separation problem that machines in $(\emptyset, \mathcal{R})$ cannot solve.

3. Investigate recognition zero knowledge for other classes of verifiers. In particular, is there an L which has a recognition zero knowledge IPS with ptime-ptm verifiers but no perfect zero knowledge IPS with ptime-ptm verifiers?

Acknowledgement. We are deeply grateful to Gerry Shedler for patiently steering us through the literature on probability and Markov chains. His assistance has been invaluable. We also thank Peter Hall and Iain Johnstone for helpful conversations.

References

[1] Ben-Or, M., Goldreich, O., Goldwasser, S., Håstad, J., Kilian, J., Micali, S., and Rogaway, P. Everything provable is provable in zero-knowledge. *Proceedings of Advances in Cryptology — CRYPTO '88*, *Lecture Notes in Computer Science, Vol. 403*, Springer-Verlag, New York, 1990, pp. 37–56.

[2] Ben-Or, M., Goldwasser, S., Kilian, J., and Wigderson, A. Multi-prover interactive proofs: how to remove intractability assumptions. In *Proceedings of the 20th ACM Symposium on Theory of Computing*. ACM, New York, 1988, pp. 113–131.

[3] Billingsley, P. *Probability and Measure*. John Wiley and Sons, New York, 1979.

[4] Chung, K. L. *Markov Chains with Stationary Transition Probabilities*. Springer-Verlag, Berlin, 1960.

[5] Condon, A., and Ladner, R. Probabilistic game automata. *J. Comput. Syst. Sci. 36* (1988), 452–489. Preliminary version in *Proceedings of the Conference on Structure in Complexity Theory, Lecture Notes in Computer Science, Vol. 223*, Springer-Verlag, New York, 1986, pp. 144–162.

[6] Dwork, C., and Stockmeyer, L. Interactive proof systems with finite state verifiers (preliminary report). *Research Report RJ6262*, IBM Research Division, San Jose, CA, May 1988.

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

[8] Feige, U., Fiat, A., and Shamir, A. Zero knowledge proofs of identity. In *Proceedings of the 19th ACM Symposium on Theory of Computing.* ACM, New York, 1987, pp. 210–217.

[9] Feller, W. *An Introduction to Probability Theory and Its Applications, Vol. I, 3rd Ed.* John Wiley and Sons, New York, 1968.

[10] Freivalds, R. Probabilistic two-way machines. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, Vol. 118*, Springer-Verlag, New York, 1981, pp. 33–45.

[11] Galil, Z., Haber, S., and Yung, M. A private interactive test of a Boolean predicate and minimum-knowledge public-key cryptosystems. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science.* IEEE, New York, 1985, pp. 360–371.

[12] Goldreich, O., Micali, S., and Wigderson, A. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science.* IEEE, New York, 1986, pp. 174–187.

[13] Goldreich, O., Micali, S., and Wigderson, A. How to play any mental game. In *Proceedings of the 19th ACM Symposium on Theory of Computing.* ACM, New York, 1987, pp. 218–229.

[14] Goldwasser, S., Micali, S., and Rackoff, C. The knowledge complexity of interactive proof systems. *SIAM J. Comput. 18* (1989), 186–208. Preliminary version in *Proceedings of the 17th ACM Symposium on Theory of Computing.* ACM, New York, 1985, pp. 291–304.

[15] Goldwasser, S., and Sipser, M. Private coins versus public coins in interactive proof systems. In *Proceedings of the 18th ACM Symposium on Theory of Computing.* ACM, New York, 1986, pp. 59–68.

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

[17] Impagliazzo, R., and Yung, M. Direct minimum-knowledge computations. *Proceedings of Advances in Cryptology — CRYPTO '87, Lecture Notes in Computer Science, Vol. 293*, Springer-Verlag, New York, 1987, pp. 40–51.

[18] Kilian, J. Zero-knowledge with log-space verifiers. In *Proceedings of the 29th IEEE Symposium on Foundations of Computer Science.* IEEE, New York, 1988, pp. 25–35.

[19] Seneta, E. *Non-Negative Matrices and Markov Chains, 2nd Ed.* Springer-Verlag, New York, 1981.

[20] Shedler, G. S. *Regeneration and Networks of Queues.* Applied Probability Series, Springer-Verlag, New York, 1987.

[21] Sipser, M. Lower bounds on the size of sweeping automata. *J. Comput. Syst. Sci. 21* (1980), 195–202.

[22] Tompa, M., and Woll, H. Random self-reducibility and zero knowledge interactive proofs of possession of information. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science.* IEEE, New York, 1987, pp. 472–482.