Statistical Zero-Knowledge Languages Can Be Recognized in Two Rounds

William Aiello* Johan Hastad**

Applied Math Department and Laboratory of
Computer Science, MIT

Abstract: Recently, a hierarchy of probabilistic complexity classes generalizing NP has emerged in the work of Babai [B], and Goldwasser, Micali, and Rackoff [GMR1], and Goldwasser and Sipser [GS]. The class IP is defined through the computational model of an interactive prover-verifier pair. Both Turing machines in a pair receive a common input and exchange messages. Every move of the verifier as well as its final determination of whether to accept or reject w are the result of random polynomial time computations on the input and all messages sent so far. The prover has no resource bounds. A language, L, is in IP if there is a prover-verifier pair such that: 1.) when $w \in L$, the verifier accepts with probability at least $1 - 2^{-|w|}$ and, 2.) when $w \notin L$, the verifier interacting with any prover accepts with probability at most $2^{-|w|}$. Such a prover-verifier pair is called an interactive proof for L.

In addition to defining interactive proofs, Goldwasser, Micali, and Rackoff [GMR1] further defined zero-knowledge interactive proofs. Informally, an interacting pair is a zero-knowledge proof for a language, L, if it is an interactive proof for L with the additional constraint that the prover reveals "nothing" (except language membership) to any verifier that the verifier could not have computed itself. There are three formal definitions for "nothing" which lead to three types of zero-knowledge: perfect, statistical, and computational, each more restrictive than the next.

We show that the first and second definitions are very restrictive. Specifically, any language, L, that has a statistical zero-knowledge interactive proof with an unbounded number of interactions has an interactive proof with two interactions. This complements a result by Fortnow [F] who showed that under the same hypothesis, the complement of L has an interactive proof with two interactions.

1. Introduction

The class NP has traditionally been recognized to capture the notion of efficient provability, containing those languages for which there exist polynomial length proofs of membership which can be verified in polynomial time. Recently, a hierarchy of probabilistic complexity classes generalizing this notion of efficient provability has emerged in the work of Babai [B], and Goldwasser, Micali, and Rackoff [GMR1], and Goldwasser and Sipser [GS]. The class IP is defined through the

* Supported by an ONR fellowship and partially supported by NSF grant DCR-8509905.

** Supported by an IBM Post Doctoral Fellowship and partially supported by Air Force Contract AFOSR-86-0078. Currently at the Royal Institute of Technology.

computational model of an interactive prover-verifier pair. Both Turing machines in a pair receive a common input, w, and exchange up to a polynomial in $|w|$ number of messages, each of which has length at most a polynomial in $|w|$. The verifier's moves and its final determination of whether to accept or reject w result from random polynomial time computations on w and all messages sent so far. The prover has no resource bounds. A language, L, is in $IP[f(n)]$ if there exists an interactive prover-verifier pair that on input w exchanges at most $f(|w|)$ messages such that: 1.) when $w \in L$, the verifier interacting with the prover accepts with probability at least $1 - 2^{-|w|}$ and, 2.) when $w \notin L$, the verifier interacting with any prover accepts with probability at most $2^{-|w|}$. Such a prover-verifier pair is called an interactive proof for L. Let $IP = \cup_k IP[n^k]$. Just as in the case when $L \in NP$, when $L \in IP$, membership in L is efficiently verifiable since the verifier runs in polynomial time and determines membership correctly with probability very close to one. However, IP is thought to strictly contain NP since several languages which have resisted all attempts to be placed in NP have been shown to be in IP, for example, Graph Non-Isomorphism [GMW], Matrix Group Non-Membership, and Matrix Group Order [B].

In addition to defining interactive proofs, Goldwasser, Micali, and Rackoff [GMR1] further defined zero-knowledge interactive proofs. The zero-knowledge definition was motivated by cryptographic considerations. (For motivation and applications of zero-knowledge see for example [GMR2] and [GMW].) Informally, a prover is zero-knowledge for a language if the prover reveals no useful information (other than language membership) when interacting with any verifier. Slightly more formally, a prover is zero-knowledge for L if for any verifier there is a probabilistic polynomial time simulator that, on inputs in L, produces conversations with the "same" probability distribution as the prover interacting with that verifier. Actually, three interpretations of "same" lead to three types of zero-knowledge, each more restrictive than the next. When "same" is informally interpreted as: 1.) identical, 2.) almost identical, or 3.) equivalent with respect to probabilistic polynomial time, then the prover is said to be perfect zero-knowledge, statistical zero-knowledge, or computational zero-knowledge for L, respectively. A language, L, is in PZK (SZK, CZK) if there is an interacting prover-verifier pair which is a interactive proof for L with the additional property that the prover is perfect (statistical, computational) zero-knowledge for L.

Requiring that, for inputs in the language, the conversations between the prover and every verifier be accurately reproducible by some random polynomial time machine would seem to be a severe constraint on the power of the prover and hence the power of the zero-knowledge model. Surprisingly, for computational zero-knowledge this is probably not the case. Through the work of [GMW], [BGGHKMR], [IY] it has been shown that, assuming secure encryption exists, any interactive proof can be transformed into a computational zero-knowledge proof, i.e., $CZK = IP$.

Fortnow [F] was the first to provide evidence that the statistical zero-knowledge requirement may restrict the power of the prover. He proved that if a language has a statistical zero-knowledge proof, then the complement of the language has a bounded round interactive proof, i.e., $SZK \subseteq$ *co-*$IP[2]$. From this theorem we can deduce that it is unlikely that SZK contains all of NP since if $NP \subseteq SZK$ then *co-*$NP \subseteq IP[2]$, which further implies that the polynomial time hierarchy collapses to $IP[2]$ by [BHZ]. This is especially interesting because it implies that membership in SZK can be taken as evidence that a language is *not* NP complete. A particularly important example of this is Graph Isomorphism, which was shown to be in SZK by [GMW].

While Fortnow's result did imply that SZK is probably weaker than IP (IP contains NP whereas SZK probably does not), it still left open the possibility that SZK contained languages in IP which required a polynomial number of interactions. In this paper we show that this cannot be the case. We prove that any language which is recognized by an unbounded round statistical zero-knowledge proof can also be recognized by a two round interactive proof, i.e., $SZK \subseteq IP[2]$. Hence, under the assumption that $IP \neq IP[2]$, the statistical zero-knowledge condition on the prover severely restricts the power of unbounded round interactive proofs. Our result does not depend on any unproven cryptographic assumptions.

We should note that, due to the fact that our proof techniques relativize, it is undoubtedly not possible to strengthen our result to get $CZK \subseteq IP[2]$. Such a result would imply that $IP[2] = IP$ if secure encryption functions exist (since $CZK = IP$ under the same assumption, as stated above). However, Aiello, Goldwasser, and Hastad [AGH] have separated $IP[2]$ and IP with an oracle. We strongly believe that their construction can be modified to incorporate the existence of a one-way permutation. Hence, relative to such an oracle, $CZK \not\subseteq IP[2]$ implying that no proof of containment can relativize.

The outline of the paper is as follows: In section 2 we give the basic definitions that are needed in the paper. We give the intuition behind the proof in section 3. In section 4 we recall some facts about estimating sizes of sets using interactive proofs and in section 5 we state and prove the main theorem.

2. Notation and Definitions

In this section we give the formal definitions needed for the paper. Let P denote a prover: any probabilistic Turing machine which has a "communication" tape (for a formal definition of a "communication" tape see [GMR1]). P has no resource bounds. Let V denote a verifier: any probabilistic polynomial time Turing machine with a communication tape. Let $P \leftrightarrow V$ denote an interacting prover-verifier pair: any prover and verifier which share the same input tape and communication tape (initially empty) and interact in rounds in the following way.

(1) The verifier, V, makes a probabilistic polynomial time computation based on the input, the contents of its memory, and all messages thus far received over the communication tape from the prover, P.

(2) V transmits the result of the computation over the communication tape to P. We will denote the message sent by V in round i by x_{2i-1}.

(3) P performs a probabilistic computation based on the input, and all messages thus far received over the communication tape from V.

(4) P transmits the result of the computation over the communication tape to V. We will denote the message sent by P in round i by y_{2i}.

The number of rounds is at most a polynomial in the input length and the interaction is terminated by the verifier accepting or rejecting.

Let $P{\leftrightarrow}V(w)$ denote a transcript of the interaction between the prover and the verifier. This is of course a stochastic variable depending on P's and V's random choices.

Definition: A given $P{\leftrightarrow}V$ is *α-complete* for a language, L, if for all $w \in L$ the probability that V accepts on w is at least α.

Definition: A verifier, V is *β-sound* for a language, L, if for all $P' \leftrightarrow V$ and all $w \notin L$ the probability that V rejects on w is at least β.

[GMR1] defined the class IP as follows. L is in $IP[f(n)]$ if there exists an interacting prover-verifier pair, $P \leftrightarrow V$, that exchanges at most $f(n)$ messages (n being the length of the input) such that:

1.) $P \leftrightarrow V$ is $(1 - 2^{-n})$-complete for L, and

2.) V is $(1 - 2^{-n})$-sound for L.

Call such a $P \leftrightarrow V$ an interactive proof for L. Note that membership in L is still efficiently verifiable since V runs in polynomial time and verifies membership correctly with probability very close to one. Define IP as the union over k of $IP[n^k]$.

We should make several remarks here. First, we can assume without loss of generality that all messages sent by P and V are of the same length, $l(n)$, and all conversations consist of the same number of messages, $d(n)$, where l and d are polynomials and n is the length of the input. Second, any language, L, which has an interactive prover-verifier pair, $P \leftrightarrow V$, such that $P \leftrightarrow V$ is $(\frac{1}{2} + \epsilon)$-complete for L and V is $(\frac{1}{2} + \epsilon)$-sound for L also has an interactive proof, $P' \leftrightarrow V'$, for L. $P' \leftrightarrow V'$ just simulates $P \leftrightarrow V$ a polynomial number of times (depending on ϵ) and V' accepts if V accepted in a majority of the simulations. Third, it is easy to see that replacing the $(1 - 2^{-n})$-soundness condition with a 1-soundness condition collapses IP down to NP. It is not as easy to see that requiring 1-completeness does not affect the power of the model. The proof of this fact was given by Goldreich, Mansour, and Sipser [GMS].

2.1 Zero-Knowledge

In this section we will give the formal definition of a zero-knowledge interactive proof for a language. We will first need some properties of probability distributions on strings.

Let $A(w)$ and $B(w)$ be two parameterized discrete random variables. Let $A[L] = \{A(w) | w \in L\}$ and similarly define $B[L]$. We define three types of equivalence between $A[L]$ and $B[L]$.

1.) $A[L] \equiv_P B[L]$ or $A[L]$ is *perfectly* equivalent to $B[L]$ if for all y, and all $w \in L$,

$$Pr[A(w) = y] \;=\; Pr[B(w) = y].$$

2.) $A[L] \equiv_{S(k,N)} B[L]$ or $A[L]$ is (k, N)-*statistically* equivalent to $B[L]$ if for all $w \in L$, $|w| \geq N$,

$$\sum_y |Pr[A(w) = y] - Pr[B(w) = y]| \leq \frac{1}{|w|^k}.$$

3.) $A[L] \equiv_{C(k,N)} B[L]$ or $A[L]$ is (k, N)-*computationally* equivalent to $B[L]$ if for all circuits, C, of size at most $|w|^k$, and all $w \in L$, $|w| \geq N$,

$$\sum_y |P^C_{A(w),\,y} - P^C_{B(w),\,y}| \leq \frac{1}{|w|^k}$$

where $P^C_{A(w),\,y}$ denotes the probability that C outputs 1 on input y when y is chosen according to $A(w)$, and $P^C_{B(w),\,y}$ is defined similarly with respect to $B(w)$.

We have already seen what it means for an interacting prover-verifier pair to be an interactive proof for a language. In a cryptographic setting, however, we may require more from our protocol than just completeness and soundness. We may want the prover to give nothing to the verifier that the verifier could not have computed itself. To formalize this [GMR1] introduced the important definition of zero-knowledge. We need the weakest form of zero-knowledge so let us start by introducing it.

The key concept for zero-knowledge is that of a simulator. A simulator, M, is a random Turing machine that produces strings, i.e., "conversations," in expected polynomial time. Let $M(w)$ be the random variable associated with M on input w. Recall that $P{\leftrightarrow}V(w)$ is the random variable associated with the conversations produced by $P{\leftrightarrow}V$ on input w. We will say that P is *perfect* zero-knowledge for V on L if there exists a simulator, M, such that $M[L] \equiv_P P{\leftrightarrow}V[L]$. Define the class $TVPZK$, *Trusted Verifier Perfect Zero-Knowledge*, to be those languages, L, for which there exists an interactive prover-verifier pair, $P{\leftrightarrow}V$, such that:

1.) $P{\leftrightarrow}V$ is $(1-2^{-n})$-complete on L,

2.) V is $(1-2^{-n})$-sound on L, and

3.) P is perfect zero-knowledge for V on L.

Call such a $P{\leftrightarrow}V$ a trusted verifier perfect zero-knowledge proof for L.

We can also define Trusted Verifier Statistical Zero-Knowledge, $TVSZK$, and Trusted Verifier Computational Zero-Knowledge, $TVCZK$. P is *statistical* (*computational*) zero-knowledge for V on L if for all k there exists a simulator, M, and an integer, N, such that $M[L]$ is (k,N)-statistically ((k,N)-computationally) equivalent to $P{\leftrightarrow}V[L]$. Define the class $TVSZK$ ($TVCZK$) to be those languages, L, for which there exists an interactive prover-verifier pair, $P{\leftrightarrow}V$, such that:

1.) $P{\leftrightarrow}V$ is $(1-2^{-n})$-complete on L,

2.) V is $(1-2^{-n})$-sound on L, and

3.) P is statistical (computational) zero-knowledge for V on L.

Call such a $P{\leftrightarrow}V$ a trusted verifier statistical (computational) zero-knowledge proof for L. Although we have no results for computation zero-knowledge, we give that definition for the sake of completeness so the reader can compare the different definitions.

Besides requiring that a pair, $P{\leftrightarrow}V$, be a trusted verifier zero-knowledge proof for a language, in a cryptographic setting we may require even more. For example, P cannot be sure that it is interacting with V, the verifier of the interactive proof. P may be interacting with a another verifier, V', which by deviating from the protocol can extract additional "knowledge" from P. This leads to the normal, more restrictive, definition of zero-knowledge where no verifier should be able to gain any additional "knowledge" from P. However since our proof already works for the less restrictive notion of zero-knowledge we will not need the more restrictive definitions and hence we refer the interested reader to [GMR1] and to [GMR2], [O] and [TW] for a discussion of the definitions.

Fortnow [F] proved that if L admits a polynomial round proof which is perfect or statistical zero-knowledge for a trusted verifier then the complement of L is in $IP[2]$. Our main result is that under the same assumption, L itself is in $IP[2]$.

We will prove the main theorem in section 5. The proof is quite technical and hence we will give an intuitive overview of the proof in the next section.

3. Outline of Proof; Structure of Simulator

For the time being let us consider languages in $TVPZK$. Let L be such a language, let $P \leftrightarrow V$ be the interactive proof recognizing L, and let M be the perfect zero-knowledge simulator for $P \leftrightarrow V$, i.e., $P \leftrightarrow V[L] \equiv_P M[L]$. Further assume that M runs in polynomial time rather than expected polynomial time. Our goal is to show that L can also be recognized by a bounded round interactive proof. Alice, or A, will be the prover in the bounded round proof and Bob, or B, will be the verifier. In conversations produced by the simulator, M, we will say that the prover-moves in the conversation are produced by a *virtual* prover, P', and the verifier-moves are produced by a *virtual* verifier, V'. We will often write this as $M = P' \leftrightarrow V'$ where the $'$ indicates that $P' \leftrightarrow V'$ is not a true interactive prover-verifier pair. To summarize, L is recognized by $P \leftrightarrow V$; M is the zero-knowledge simulator which we think of as $P' \leftrightarrow V'$; and our goal is to construct a bounded round proof, $A \leftrightarrow B$, that also recognizes L.

Recall that V's moves are labeled x and P's moves are labeled y. Let s_k be the string produced by the first k interactions of the $P \leftrightarrow V$ protocol, $s_k = x_1 y_2 \ldots q_k$ where $q_k = x_k$ for k odd or y_k for k even. Without loss of generality assume that V sends its coins to P on the very last move. Let $d(n) + 1$ be the total number of interactions where $d(n)$ is even. We will abbreviate the entire dialogue, $s_{d(n)}, r$, as s, r. Recall that $P \leftrightarrow V(w)$ is a random variable. We will often be interested in the event that the entire conversation is s, r, i.e., that $P \leftrightarrow V(w) = s, r$. In addition, we will also be interested in the event that the first k interactions are s_k and V has coins r. This event is written as $P \leftrightarrow V(w) = s_k *, r$. Finally, we will also be interested in the event $P \leftrightarrow V(w) = s_k *, *$, i.e., the first k interactions yield s_k. We will use similar notation to signify the same events for $M(w)$.

Recall that the verifier's $j + 1$st move is a function of the input, its random bits and the previous $2j$ interactions. We denote this formally as $V(w, r, s_{2j}) = x_{2j+1}$ for $0 \leq 2j < d(n)$.

Definition: A conversation, s, r, output by the simulator is *valid* if the moves of V' are the same as the the moves that the real V would make if it were playing with coins r. That is s, r is valid if $V(w, r, s_{2j}) = x_{2j+1}$ for $0 \leq 2j < d(n)$.

Let us look at the behavior of M on input w. By definition of perfect zero-knowledge, when $w \in L$ the conversations of $P \leftrightarrow V$ and $P' \leftrightarrow V'$ have the same distribution. Hence, M will output valid conversations with probability one and output accepting conversations with probability at least $1 - 2^{-n}$. Also, the moves of P' must be made with the same probabilities as the moves of P. This will be made more formal later.

When $w \notin L$ there are two cases. Either M outputs valid accepting conversations with high probability or it does not. The latter case immediately implies that $w \notin L$. Let us look at the former case more closely. By definition of an interactive proof, if $w \notin L$ then for all provers, Q, the probability that $Q \leftrightarrow V$ accepts is at most 2^{-n}. But P' convinces V' to accept with high probability. P' can gain such an advantage over any real prover, Q, since P' can "see" r of V' (since $P' \leftrightarrow V'$ is actually one machine, M, by definition) whereas Q can only infer certain properties of V's coins given the conversation so far.

Having discerned the gross behavior of M on input w we can give a very broad outline of the interactive proof between Alice and Bob, $A \leftrightarrow B$. On input w

(1) Bob will try to convince himself that $M(w)$ outputs valid accepting conversations with high probability, and

(2) Alice will try to convince Bob that P' is not taking advantage of the fact that P' "sees" the coins of V'.

Bob will accept w only if he is convinced of both (1) and (2).

A broad outline of the proof that $A \leftrightarrow B$ recognizes L is as follows. When $w \in L$, $M(w)$ does output valid accepting conversations with high probability and so Bob will succeed in convincing himself in (1). Furthermore, P' makes moves with the same distribution as the real P and so Alice will succeed in convincing Bob in (2). When $w \notin L$, if M does not output valid accepting conversations with high probability then Bob will not succeed in (1). If M does output valid accepting conversations with high probability then by the above discussion P' must be taking advantage of the fact that it "sees" r of V'. Hence, A will not succeed in (2).

Let us make the above discussion more formal. In order to simplify notation let us state once and for all that we will implicitely consider only nonempty sets or events with positive probability. It follows that all conditional probabilities are well defined as are the logarithms of the size of sets.

Definition: Let α_{s_k} be the set of all the verifier's coins that are *consistent* with the partial conversation s_k. That is, α_{s_k} is the set of all r such that $V(w, r, s_{2i}) = x_{2i+1}$ for $0 \leq 2i < k$.

Note that if the last move of s_k is a verifier-move, i.e., k is odd, then $\alpha_{s_k} = \alpha_{s_k y}$ for all y. This is due to the fact that the prover's move y does not affect which coins are consistent with the verifier's moves in the string s_k.

Observe that since V's coins are uniformly distributed, for any partial conversation, s_{2k-1}, all r that are consistent with s_{2k-1} are equally likely. That is,

$$Pr[P \leftrightarrow V(w) = s_{2k-1}*, r \mid P \leftrightarrow V(w) = s_{2k-1}*, *] = \frac{1}{|\alpha_{s_{2k-1}}|}$$

for all $r \in \alpha_{s_{2k-1}}$ and the probability is zero otherwise. Recall from above that $\alpha_{s_{2k-1}} = \alpha_{s_{2k-1}y}$. Hence

$$Pr[P \leftrightarrow V(w) = s_{2k-1}y*, r \mid P \leftrightarrow V(w) = s_{2k-1}y*, *] = \frac{1}{|\alpha_{s_{2k-1}}|}$$

for all $r \in \alpha_{s_{2k-1}}$ and the probability is zero otherwise. The above identities imply that for all moves y of the prover

$$Pr[P \leftrightarrow V(w) = s_{2k-1}y*, r \mid P \leftrightarrow V(w) = s_{2k-1}*, r] =$$

$$Pr[P \leftrightarrow V(w) = s_{2k-1}y*, * \mid P \leftrightarrow V(w) = s_{2k-1}*, *]$$

for all $r \in \alpha_{s_{2k-1}}$. This motivates the following definition:

Definition: Given partial conversation s_{2k-1} a move y by P' is *honest* if

$$Pr[P' \leftrightarrow V'(w) = s_{2k-1}y*, r \mid P' \leftrightarrow V'(w) = s_{2k-1}*, r] =$$

$$Pr[P' \leftrightarrow V'(w) = s_{2k-1}y*, * \mid P' \leftrightarrow V'(w) = s_{2k-1}*, *]$$

for all $r \in \alpha_{s_{2k-1}}$.

In words, a move y is honest if for all r the probability that P' plays y given that s_{2k-1} has been played so far and V' has coins r is equal to the probability that P''s move is y given only that the conversation thus far is s_{2k-1}. That is, P''s move is based only on the conversation so far and not upon additional information about r. By definition of PZK, $Pr[P' \leftrightarrow V' = s, r] = Pr[P \leftrightarrow V = s, r]$ for all s, r when $w \in L$. It follows that all of P''s moves are honest when $w \in L$.

We will say that P' cheats on $s_{2k-1}y*, r$ if the first conditional probability is much greater than the second. That is, P' cheats on move y if it does use additional information about r. For the sake of the analysis we will actually take the logarithms of the probabilities.

Definition: P' *c-cheats* on $s_{2k-1}y*, r$ if

$$\log\left(Pr[P' \leftrightarrow V'(w) = s_{2k-1}y*, r \mid P' \leftrightarrow V'(w) = s_{2k-1}*, r]\right)$$

$$\geq \log\left(Pr[P' \leftrightarrow V'(w) = s_{2k-1}y*, * \mid P' \leftrightarrow V'(w) = s_{2k-1}*, *]\right) + c.$$

When $w \notin L$ we will prove that if $M(w)$ outputs valid accepting conversations with high probability then P' cheats a great deal on average. This will be made precise in Lemma 5.3.

The goal of the interactive proof between Alice and Bob is to distinguish between P' playing honestly everywhere and P' cheating a great deal on average. With Alice's help Bob will recognize when P' is honest and will accept. However, Alice will not be able to fool Bob into accepting very often when P' cheats a great deal on average.

Remark: It is interesting to note that reason that the proof does not work for computational zero-knowledge is the the virtual prover cheats also in the case when $w \in L$.

The probabilities used in the definition of cheating are actually ratios of the sizes of certain sets, which we now define. Let R be the simulator's coin, $|R| = q(n)$ where $q(n)$ is a polynomial. Let β_{s_k} and $\beta_{s_k*,r}$ be the sets of R for which $M(w, R) = s_k*, *$ or s_k*, r respectively. With these definitions, P' c-cheats on $s_{2k-1}y*, r$ if

$$\left(\log|\beta_{s_{2k-1}y*,r}| - \log|\beta_{s_{2k-1}*,r}|\right) - \left(\log|\beta_{s_{2k-1}y}| - \log|\beta_{s_{2k-1}}|\right) \geq c.$$

The following section deals with the aforementioned subprotocols which prove upper and lower bounds on the size of sets.

4. Upper and Lower Bound Protocols

Let us first consider a subprotocol for proving lower bounds on the size of sets. It is based on a lemma of Sipser's [S] and uses universal hashing [CW].

Suppose $C \subseteq \Sigma^k$ where membership in C is testable in polynomial time. (The protocol can easily be modified to work when membership in C is testable in nondeterministic polynomial time but this will not concern us here.) Let H be a $k \times b$ Boolean matrix and let $h: \Sigma^k \to \Sigma^b$ be defined by matrix multiplication modulo 2, $h(x) = xH$. The protocol "P proves $|C| \geq 2^b$" is as follows:

1. V picks a random $k \times b$ matrix H and a random element z of Σ^b. V sends H and z to P.
2. P responds with $c \in \Sigma^k$.
3. V accepts iff $c \in C$ and $h(c) = z$.

Lemma 4.1: *If P plays optimally then*

(1) $Pr[V \text{ accepts}] \geq 1 - \frac{2^b}{|C|}$.

(2) $Pr[V \text{ accepts}] \leq \frac{|C|}{2^b}$.

Proof: Let us first prove (1). Let S be the number of elements that map onto the randomly chosen z. S is a random variable and $\mu(S) = |C|2^{-b}$. Since if $c_1 \neq c_2$ the probability that $h(c_1) = h(c_2) = z$ is 2^{-2b} we have that $\sigma^2(S) = |C|(2^{-b} - 2^{-2b})$. Using Chebychev's inequality this implies that $Pr[S = 0] \leq \frac{|C|(2^{-b}-2^{-2b})}{(|C|2^{-b})^2} \leq \frac{2^b}{|C|}$. This implies (1) since whenever $S \neq 0$ P can make V accept. (2) follows from the fact that for a fixed element $w \in \Sigma^k$ the probability that $h(w) = z$ for a fixed h and a random z is $1/2^b$. $\square$

Next we present a protocol developed by Fortnow [F] for proving upper bounds on the size of sets. Suppose the verifier has a random element, c, of the set $C \subseteq \Sigma^k$. It is crucial that P has no information about c other than the fact that it lies in C. Define "P proves $|C| \leq 2^b$" as follows.

1. V picks a random $k \times b$ matrix H and calculates $h(c) = z$. V sends H and z to P.
2. P sends a to V.
3. V accepts iff $a = c$.

Lemma 4.2: *If P plays optimally then*

(1) $Pr[V \text{ accepts }] \geq 1 - \frac{|C|-1}{2^b}$.

(2) $Pr[V \text{ accepts }] \leq \frac{g2^b}{|C|-1}$ where $g = 3 + \sqrt{5}$.

Proof: The proof of (1) is as follows. P will certainly be able to answer with an a equal to c whenever none of the elements in $C - \{c\}$ collide with $h(c) = z$. This event occurs with probability at least $1 - (|C| - 1)/2^b$ since for a fixed c and w in Σ^k, the probability that $h(w) = h(c)$ is $1/2^b$.

The proof of (2) is as follows. Let S be the number of elements of C which map to z. Then by a simple argument

$$Pr[V\ accepts] \leq \Sigma_{j=1}^{|C|} \frac{1}{j} Pr[S = j] \leq Pr[S \leq i] + \frac{1}{i}$$

for all i. Again we will apply Chebychev's inequality. Note that $\mu(S) = 1 + \frac{|C|-1}{2^b}$ and $\sigma^2 \leq \mu$. So, for $i < \mu$

$$Pr[S \leq i] \leq Pr[\mu - S \geq \mu - i] \leq \frac{\mu}{(\mu - i)^2}.$$

For $i = \frac{2}{3+\sqrt{5}}\mu$ we get

$$Pr[V\ accepts] \leq \frac{(3 + \sqrt{5})2^b}{|C| - 1}. \quad \square$$

5. Proof of Main Theorem

In this section we give the proof of our main theorem. Initially we will assume that the simulator is polynomial time and the protocol is perfect zero-knowledge. In the end of the section we will take care of the complications which arise when the protocol is statistical zero-knowledge and the simulator is expected polynomial time. Let us start by stating the theorem.

Theorem 5.1: *If L is recognized by an interactive proof which is statistical zero-knowledge for a trusted verifier where the simulator runs in expected polynomial time then L can be recognized by a two round interactive proof.*

Call the prover and verifier recognizing the language P_0 and V_0 and call the simulator M_0. Recall that the probability of error in the protocol is at most 2^{-n} and the number of rounds is $d(n)$ where n is the length of the input. Run this protocol $d(n)$ times in parallel and make the verifier accept if it accepts in a majority of the subprotocols. Call this new protocol $P \leftrightarrow V$. Note that it is still perfect zero-knowledge for trusted verifier: the new simulator, M, just runs the old simulator $d(n)$ times and takes majority. It is easy to show that the probability of error for the new protocol is at most $2^{-cnd(n)}$ for some $c > 0$. We will keep this value of c fixed from now on. We denote the number of coins that V uses by $l(n)$ and the number of coins M uses by $q(n)$.

Observe here that it is crucial that we are working with trusted verifier simulations since it is probably not true in general that running several zero-knowledge protocols in parallel will give a zero-knowledge protocol.

For the sake of the following two lemmas, assume that M always produces valid accepting conversations. Later we will discuss how to modify the lemmas when this assumption is removed. Let $Q(s)$ denote the probability that S appears as partial conversation in the conversation output by the simulator. Here S takes any of the values $(s_{2k-1}), (s_{2k-1}y), (s_{2k-1}*, r)$ and $s_{2k-1}y*, r)$. Using the notation of section 3 define

$$F_0 = E(\log|\beta_{*,r}| - q(n))$$

and

$$\begin{aligned} F_k &= E(\log|\beta_{s_{2k-1}y*,r}| - \log|\beta_{s_{2k-1}*,r}|) \\ &= \sum_{s_{2k-1}y*,r} Q(s_{2k-1}y*, r)(\log|\beta_{s_{2k-1}y*,r}| - \log|\beta_{s_{2k-1}*,r}|). \end{aligned}$$

where E is the expectation over a random partial conversation output by the simulator. Similarly define

$$G_0 = -l(n)$$

and

$$G_k = E(\log|\beta_{s_{2k-1}y}| - \log|\beta_{s_{2k-1}}|).$$

Also let $G = \sum_{k=0}^{d(n)/2} G_k$ and $F = \sum_{k=0}^{d(n)/2} F_k$.

The intuition behind these definitions is that $F - G$ is the expectation of the total amount of cheating by P' on all its moves in a random conversation of the simulator. With these definitions we can characterize the behavior of the simulator with the following two lemmas.

Lemma 5.2: *If $w \in L$ then $G = F$.*

Lemma 5.3: *If $w \notin L$ then $F - G \geq cnd(n)$.*

Proofs: Lemma 5.2 follows clearly from the discussion in section 3 since when $w \in L$, P' does not cheat and $G_k = F_k$ for all k. To prove Lemma 5.3 we will first establish the two facts described below.

Fact 5.4: $F = \sum_{s,r} Q(s,r) \log Q(s,r)$ where, according to our previous conventions, the sum is only over conversations which have a positive probability of being produced by the simulator.

This is a consequence of the definitions since

$$\begin{aligned} F &= \sum_{k=0}^{d(n)/2} F_k \\ &= \sum_{k=1}^{d(n)/2} (\sum_{s,r} Q(s,r)(\log|\beta_{s_{2k-1}y_{2k}*,r}| - \log|\beta_{s_{2k-1}*,r}|)) + \sum_{s,r} Q(s,r)(\log|\beta_{*,r}| - q(n)) \\ &= \sum_{s,r} Q(s,r) \left(\log|\beta_{*,r}| - q(n) + \sum_{k=1}^{d(n)/2} (\log|\beta_{s_{2k-1}y_{2k}*,r}| - \log|\beta_{s_{2k-1}*,r}|) \right). \qquad (*) \end{aligned}$$

Now since we have assumed that M always makes valid moves, it follows that $|\beta_{s_{2k-1}y_{2k}*,r}| = |\beta_{s_{2k-1}y_{2k}x*,r}|$ where x is the unique move that V would make with coins r after the partial conversation $s_{2k-1}y_{2k}$. Using this fact the above sum telescopes to $\log|\beta_{s,r}| - q(n)$ and this is precisely $\log Q(s,r)$.

To get a similar formula for G let us examine the behavior of a new prover $\hat{P}$ which uses the output of the simulator to play against the verifier V. On partial conversation s_{2k-1} $\hat{P}$ makes the move y_{2k} with probability $|\beta_{s_{2k-1}y_{2k}}|/|\beta_{s_{2k-1}}|$ which it can easily do since it is all powerful. If $Pr[\hat{P} \leftrightarrow V = s, r]$ is nonzero then it can be expanded as follows:

$$Pr[\ V \text{ has coins } r\ ] \times \prod_{k=1}^{d(n)/2} Pr[\ \hat{P} \text{ plays } y_{2k} \mid \hat{P} \leftrightarrow V = s_{2k-1} \text{ and } V \text{ has coins } r\ ].$$

By our previous comments the probability that $\hat{P}$ plays y_{2k} given that s_{2k-1} has been played so far is the same as the probability that $\hat{P}$ plays y_{2k} given that s_{2k-1} has been played so far and V has coins r for any r consistent with s_{2k-1}. Let $R(s,r) = Pr[\hat{P} \leftrightarrow V = s, r]$. It follows that $\log R(s,r)$ is equal to

$$-l(n) + \sum_{k=1}^{d(n)/2} (\log|\beta_{s_{2k-1}y_{2k}}| - \log|\beta_{s_{2k-1}}|).$$

This establishes the following fact.

Fact 5.5: $G = \sum_{s,r} Q(s,r) \log R(s,r)$. Again, the sum is only over s, r such that $Q(s,r) > 0$.

Note here that whenever $Q(s,r) > 0$ we also have $R(s,r) > 0$ and hence the given sum is finite.

Observe that when $w \notin L$ the probability that V accepts when interacting with $\hat{P}$ is at most $2^{-cnd(n)}$. Hence, the $\sum R(s,r)$ over s, r with $Q(s,r) > 0$ is at most $2^{-cnd(n)}$. Since $F - G = \sum Q(s,r) \log(Q(s,r)/R(s,r))$ (over the appropriate s, r) Lemma 5.3 follows from Lemma 5.6 below.

Lemma 5.6: *Let* $\sum_{i=1}^{m} p_i = 1$ *and* $\sum_{i=1}^{m} q_i = q$ *where* $p_i > 0$ *and* $q_i > 0$ *for all* i, *then* $\sum_{i=1}^{m} p_i \log \frac{p_i}{q_i} \geq \log \frac{1}{q}$.

Proof: (Lemma 5.6) Fix q_i for all i. Let us minimize the expression over all values of p_i satisfying $\sum_{i=1}^{m} p_i = 1$. The ith component of the gradient is $\log e(\ln \frac{p_i}{q_i} + 1)$ and thus the only interior extreme point is found for $p_i = q_i/q$ for all i. This is easily checked to be the global minimum. $\Box$

Before we continue, let us get rid of the assumption that M always produces valid accepting conversations. We will handle this by making B reject whenever he sees a conversation which is not valid or not accepting. We have to analyze what effect this has. Say that a conversation is OK if it is valid and accepting.

Fix t to be a polynomial. Suppose B picks $|w|t(|w|)$ random R and runs M. If $w \in L$ then with probability $\geq 1 - 2^{-|w|}|w|t(|w|)$ all these conversations will be accepting. Also all conversation will be valid and that is enough to prove Lemma 5.2, and thus in this case there is no major difference.

When $w \notin L$ there are two cases, either the fraction of OK conversations is at least $1 - \frac{1}{t(|w|)}$, or it is not. In the latter case B will except with exponentially small probability see a conversation which is not OK. In the former case the previous analysis is almost correct we just need a few minor changes.

Fact 5.4 is no longer exactly true and we need a slight modification. In the sum $(*)$ let us distinguish two kinds of terms, namely the ones corresponding to s, r which are OK conversations and the ones which are not OK. The first kind of terms gives rise to a sum like the one in Fact 5.4 where the summation is limited to OK conversations. The other kind of terms gives a contribution which is at most $O\left(\frac{q(n)}{t(n)}\right)$.

Fact 5.5 remains valid but here it is also convenient to split the sum into terms corresponding to OK s, r and conversations which not OK. This second sum is as before bounded by $O\left(\frac{q(n)}{t(n)}\right)$. On the other hand when we are summing over OK conversations we can use Lemma 5.6. Here we need the modification that the sum of $Q(s,r)$ over all OK conversations is slightly less than 1, but this again is an error term of size bounded by the same bound. Thus we obtain a weaker version of Lemma 5.3 with $cnd(n)$ replaced by $cnd(n) - O\left(\frac{q(n)}{t(n)}\right)$. For $t(n) \gg q(n)$ we can ignore the error term. Since B will see many more than $q(n)$ conversations during the protocol and he will reject if he sees a conversation which is not OK we can assume throughout that Lemma 5.3 holds.

Now we are ready to define the protocol for recognizing L. It will consist of subprotocols estimating upper bounds for F_k and lower bounds for G_k. On input w, $|w| = n$.

1) For $i = 1, \ldots 4q(n)^2$, B picks a random R_i and runs M to get an r_i. B sends this r_i to A and A responds with b_i and proves that $|\beta_{*,r_i}| \leq 2^{b_i}$. B computes $\hat{F}_0 = \frac{1}{4q(n)^2} \sum_{i=1}^{4q(n)^2} b_i - q(n)$.

2) For $i = 1, \ldots 4q(n)^2$, and $k = 1, \ldots d(n)/2$, B picks a random $R_i^{(k)}$ and runs M to get an s_{2k-1}, y_{2k} and r. B sends s_{2k-1} to A. A responds with $b_i^{(k)}$ and proves that $|\beta_{s_{2k-1}}| \leq 2^{b_i^{(k)}}$. Then B also sends y_{2k} to A and A responds with $c_i^{(k)}$ and proves that $|\beta_{s_{2k-1}y-2k}| \geq 2^{c_i^{(k)}}$. B computes $\hat{G}_k = \frac{1}{4q(n)^2} \sum_{i=1}^{4q(n)^2} (c_i^{(k)} - b_i^{(k)})$.

3) For $i = 1, \ldots, 4q(n)^2$, and $k = 1, \ldots, d(n)/2$, B picks a random $R_i^{(k)}$ and runs M to get an s_{2k-1}, y_{2k} and r. B sends s_{2k-1}, y_{2k} and r to A. A responds with $b_i^{(k)}$ and $c_i^{(k)}$ and proves that

$|\beta_{s_{2k-1}y_{2k}*,r}| \leq 2^{b_i^{(k)}}$ and $|\beta_{s_{2k-1}*,r}| \geq 2^{c_i^{(k)}}$. B computes $\hat{F}_k = \frac{1}{4q(n)^2}\sum_{i=1}^{4q(n)^2}(b_i^{(k)} - c_i^{(k)})$.

4) B computes $\hat{F} = \sum_{k=0}^{d(n)/2} \hat{F}_k$ and $\hat{G} = -l(n) + \sum_{k=1}^{d(n)/2} \hat{G}_k$ and accepts iff $\hat{F} - \hat{G} \leq \frac{c}{2}nd(n)$, A has been successful in all subprotocols and all the R that B has picked corresponds to valid accepting conversations.

To establish that the protocol recognizes the language L, we will often use a standard Chernoff bound. Let us prove it for the sake of completeness.

Lemma 5.7 *Let X be a random variable $0 \leq X \leq K$ and let $E(X) = \mu$. Then given s independent observations $X_i, i = 1 \ldots s$ of X*

$$Pr[|\frac{1}{s}\sum_{i=1}^{s} X_i - \mu| \geq T] \leq 2e^{-\frac{sT^2}{4K^2}}.$$

Proof: To estimate the above probability we look at exponential moments.

$$E(e^{\lambda(s\mu - \sum_{i=1}^{s} X_i)}) = \prod_{i=1}^{s} E(e^{\lambda(\mu - X_i)})$$

To estimate this quantity we use

Lemma 5.8: *If a random variable Y satisfies $E(Y) = 0$ and $|Y| \leq c \leq 1$ then $E(e^Y) \leq e^{c^2}$.*

Proof: (Lemma 5.8) We use $e^Y \leq 1 + Y + Y^2$ which is valid for $|Y| \leq 1$ and $e^x \geq 1 + x$ valid for any x. This gives

$$E(e^Y) \leq 1 + E(Y^2) \leq 1 + c^2 \leq e^{c^2}. \quad \square$$

Now we can prove Lemma 5.7. For $T \geq K$ the lemma is clear and otherwise we put $\lambda = \frac{T}{2K^2}$. In the calculation of the exponential moment we can apply Lemma 5.8 with $Y = \lambda(\mu - X_i)$ and $c = \lambda K$ and we get $E\left(e^{\lambda(s\mu - \sum_{i=1}^{s} X_i)}\right) \leq e^{s\lambda^2 K^2}$. This implies that $Pr[s\mu - \sum_{i=1}^{s} X_i \geq sT] \leq e^{s\lambda^2 K^2 - \lambda s T} = e^{-\frac{sT^2}{4K^2}}$. Using $\lambda = -\frac{T}{2K^2}$ one gets the same bound for $\sum_{i=1}^{s} X_i$ being unusually large and Lemma 5.7 is proved. $\square$

After these preliminaries let us establish that the protocol is correct when $w \in L$.

Lemma 5.9: *If $w \in L$ then for sufficiently large n the probability that A can make B accept is $\geq \frac{8}{10}$.*

Proof: First observe that the probability that B sees a conversation which is not OK is exponentially small and hence for sufficiently large n this is less than $\frac{1}{10}$.

We present a strategy for A that, given that B never sees a conversation which is not OK, will make B accept 9 times out of 10. Whenever A is supposed to prove an upper bound for the size of a set whose true size is T, A states that the size of the set is at most $240q(n)^2d(n)T$ and the proceeds to follow the protocol given in section 4. Similarly when A is asked for a lower bound she states the bound $\max(1, \frac{T}{240q(n)^2d(n)})$ and then follows the protocol of section 4.

Given that A follows this strategy let us analyze the probability that B will accept. Since A proves bounds on $8q(n)^2d(n) + 4q(n)^2 \leq 12q(n)^2d(n)$ sets, the probability that she will ever fail in one of the protocols is by Lemmas 4.1 and 4.2 bounded by $\frac{1}{240q(n)^2d(n)} \cdot 12q(n)^2d(n) = \frac{1}{20}$. Whenever Alice is successful in all these protocols B will get a value of $\hat{F}_k$ which is within $2\log(240q(n)^2d(n))$ of $\frac{1}{4q(n)^2}\sum_{i=1}^{4q(n)^2}\left(\log|\beta_{s_{2k-1}^{(i)}y*,r}| - \log|\beta_{s_{2k-1}^{(i)}*,r}|\right)$. By Lemma 5.7 this implies that

$$Pr[(\hat{F}_k - F_k) \geq 2\log(240q(n)^2d(n)) + 4\log d(n)] \leq 2d(n)^{-2}.$$

By a similar argument we get that

$$Pr[(G_k - \hat{G}_k) \geq 2\log(240q(n)^2d(n)) + 4\log d(n)] \leq 2d(n)^{-2}.$$

Thus with probability at least $1 - \frac{4}{d(n)}$ the reverse inequalities hold for all k and since $F = G$ we get by summing that $\hat{F} - \hat{G} \leq O(d(n)\log q(n)d(n))$ with this probability. Since $\log(q(n)d(n)) = o(n)$ this completes the proof of Lemma 5.9. $\square$

Next we establish that the protocol is correct also for $w \notin L$.

Lemma 5.10: *If $w \notin L$ then for sufficiently large n the probability that A can make B accept is $\leq \frac{1}{10}$.*

Proof: As noted in the discussion before we defined the protocol we can assume that most conversations are valid and accepting and hence that Lemma 5.3 is true.

For A to succeed in convincing B to accept with high probability either $R_i^{(k)}$ must be favorable to A or A must claim lower (upper) bounds which are much larger (smaller) than the actual size of the sets. The former happens with very small probability and the latter will cause B to reject the lower (upper) bound protocol with high probability.

Suppose that A during the protocol claims a bound which is a factor 128 better than the true bound (this refers to either upper or lower bounds). Then the probability that A will get caught in trying to prove this bound is by Lemmas 4.1 and 4.2 at least $\frac{19}{20}$. Thus if A tries to make $\hat{F}_k$ smaller than $\frac{1}{4q(n)^2}\sum_{i=1}^{4q(n)^2}\left(\log|\beta_{s_{2k-1}^{(i)}y*,r}| - \log|\beta_{s_{2k-1}^{(i)}*,r}|\right) - 14$ she will fail with probability $\frac{19}{20}$. On the other hand by Lemma 5.7

$$Pr[\frac{1}{4q(n)^2}\sum_{i=1}^{4q(n)^2}\left(\log|\beta_{s_{2k-1}^{(i)}y*,r}| - \log|\beta_{s_{2k-1}^{(i)}*,r}|\right) - F_k \geq 4\log d(n)] \leq 2d(n)^{-2}.$$

Thus if A has probability at least $\frac{1}{20}$ of succeeding in her upper and lower bounds proofs we have $\hat{F}_k \geq F_k - 14 - 4\log d(n)$ with probability $1 - d(n)^{-2}$. Thus $\hat{F} \geq F - O(d(n)\log d(n))$ with probability at least $1 - \frac{1}{d(n)}$. Similarly we get $\hat{G} \leq G + O(d(n)\log d(n))$ with the same probability. If both these inequalities hold $\hat{F} - \hat{G} \geq F - G - O(d(n)\log d(n))$ and now Lemma 5.3 implies Lemma 5.10. $\square$

Lemmas 5.9 and 5.10 imply Theorem 5.1 in the case of perfect zero-knowledge and M being polynomial time. To see this we have only to verify that the protocol can be implemented in two rounds. However, by the result of Babai [B] it is sufficient to show that one can implement the protocol in a constant number of rounds. But this is clear since we run all the subprotocols for different i and k in parallel.

Remark: Our constant round protocol for L and subsequent analysis are much more complicated than Fortnow's protocol for $\bar{L}$. This is for a fundamental reason. In order for a prover, A, to convince a verifier, B, that $x \notin L$, A need only show that the simulator occasionally cheats. A can simply show B a few partial conversations and prove that the simulator cheats on these. However, in order for A to convince B that $x \in L$, A must convince B that the simulator is behaving honestly almost everywhere.

Not surprisingly, the present framework is general enough to prove Fortnow's result as well. We modify our current protocol for recognizing L to get an inefficient protocol for recognizing $\bar{L}$ as follows. Change every lower (upper) bound subprotocol to a upper (lower) bound subprotocol. The goal of the new prover is to show that $\hat{F} - \hat{G}$ is large. The new verifier will accept when it is convinced that this is the case. The analysis for this new protocol is nearly identical to that described above.

This finishes the analysis for the cases where the simulator is polynomial time and perfect zero-knowledge. Let us see how to take care of the complications that arise when the simulator is statistical zero-knowledge.

Statistical zero-knowledge. For statistical zero-knowledge the same protocol will work. Even the analysis for $w \notin L$ is the same since the zero-knowledge constraint on the behavior of the simulator only applies to $w \in L$. However, when $w \in L$ the analysis does need a slight modification. It is no longer true that $G = F$ since P' no longer behaves exactly like P. However, the difference is small and can be taken care of as follows. Let $T(S)$ denote the probability that S appears as a partial conversation in a conversation between V and P. Define $\bar{F}_k = E(\log(T(s_{2k-1}y*, r)) - \log(T(s_{2k-1}*, r)))$ and $\bar{G}_k = E(\log(T(s_{2k-1}y)) - \log(T(s_{2k-1})))$ Where the expected value is taken with respect to random s_{2k-1}, y and r produced by P and V. It is clear that $\bar{F}_k = \bar{G}_k$.

Writing everything explicitly we have

$$F_k = 2^{-q(n)} \sum_{s_{2k-1}, y, r} |\beta_{s_{2k-1}y*,r}| (\log|\beta_{s_{2k-1}y*,r}| - \log|\beta_{s_{2k-1}*,r}|)$$

and

$$\bar{F}_k = \sum_{s_{2k-1}, y, r} T(s_{2k-1}y*, r) \left(\log(T(s_{2k-1}y*, r)) - \log(T(s_{2k-1}*, r))\right).$$

We know that $\sum_{s_{2k-1}, y, r} |2^{-q(n)}|\beta_{s_{2k-1}y*,r}| - T(s_{2k-1}y*, r)| \leq \left(\frac{1}{t(n)}\right)$ for any polynomial t. Since the derivative of $p \log p$ is $\log p + \log e$ this implies that $|F_k - \bar{F}_k| \leq o\left(\frac{q(n)}{t(n)}\right)$ for any polynomial t. We get the same result for $|G_k - \bar{G}_k|$ and thus $|G - F| \leq o\left(\frac{1}{t(n)}\right)$ for any polynomial t.

Expected polynomial time simulator. Suppose M runs in expected time $q(n)$. The number of coins that M uses is potentially unlimited. Hence, we do not have well defined finite sets of the simulator's coins on which to run our upper and lower bound subprotocols. We can take care of this problem as follows.

Redefine F_k to be $E(\log Pr[y_{2k} = y | s_{2k-1}, r])$ and similarly let $G_k = E(\log Pr[y_{2k} = y | s_{2k-1}])$. These definitions agree with the old definitions in the case when M is polynomial time. For any partial conversation S redefine β_S to be the set of coins of length $10q(n)$ such that given these coins M halts within time $10q(n)$ and produces the partial conversation S. Observe that with this definition the new definitions of F_k and G_k no longer agree with the old definitions.

The protocol is now the same as before having A prove bounds for the β-sets with the new definitions. When B picks a random R there is the possibility that M will not halt. In this case B will just pick another R and try again. If B fails n times consecutively it gives up and rejects the input.

B will reject for this reason with exponentially small probability since the probability of M halting within time $10q(n)$ is at least $9/10$. Thus this will not change the performance of the protocol.

The intuition behind the protocol also working when using the modified sets is that $\frac{|\beta_S|}{2^{10q(n)}}$ is a fairly good approximation for the probability that the partial conversation S appears. Before we analyzed all other aspects of the protocol when $2^{-10q(n)}\beta_S$ is equal to this probability; we need now just analyze the additional problems that appear from this intuition not being exactly correct.

Let $Q(S)$ be the probability that M outputs the partial conversation S. Thus the intuition we need to verify is that this is close to $2^{-10q(n)}|\beta_S|$. Observe that we always have $Q(S) \geq 2^{-10q(n)}|\beta_S|$. We need to prove that A can do almost as well as before when $w \in L$ and not too much better than before when $w \notin L$. Let us start by the former.

Alice will not do as well when she cannot prove good upper bounds on $Q(s_{2k-1}y*, r)$ and $Q(s_{2k-1})$ or when she cannot prove good lower bounds on $Q(s_{2k-1}*, r)$ and $Q(s_{2k-1}y)$. Since the sizes of the β sets can only be smaller than they should be, Alice will always do as well in proving upper bounds as she did before. For the lower bounds we claim that with probability at most $\frac{10}{9D}$ B will pick an R which gives rise to $s_{2k-1}y$ which satisfies

$$|\beta_{s_{2k-1}y}| \leq D^{-1}2^{10q(n)}Q(s_{2k-1}y) \qquad (**).$$

The reason for this is that if we let $\sum'$ denote the sum over all s_{2k-1}, y satisfying $(**)$ then

$$\Sigma'|\beta_{s_{2k-1}y}| \leq D^{-1}2^{10q(n)}\Sigma'Q(s_{2k-1}y) \leq D^{-1}2^{10q(n)}.$$

Now the above claim follows from the fact that the probability that M halts within time $10q(n)$ is at least $9/10$. The same statement is of course true for $s_{2k-1}y$. Now if we choose $D = 240q(n)^2d(n)$ we know that with probability $\frac{19}{20}$ B will never choose a $R_i^{(k)}$ satisfying $(**)$. But in such a case A will be able to get $\hat{G}$ and $\hat{F}$ within $O(d(n)\log(q(n)d(n)))$ of the values she can get when M is polynomial time. Thus Lemma 5.9 also follows in the case of expected polynomial time.

Now consider the case when $w \notin L$. We have to establish that A cannot do too much better than before. This would happen if she could prove better upper bounds for $Q(s_{2k-1}y*, r)$ and

$Q(s_{2k-1})$ or if she could prove better lower bounds for $Q(s_{2k-1}*, r)$ and $Q(s_{2k-1}y)$. The second possibility can never occur and for the first possibility we get the same analysis as above.

Acknowledgments: We would like to thank Lance Fortnow for valuable comments.

References

[AGH] Aiello, W., S. Goldwasser, and J. Hastad, "On the Power of Interaction," to appear in *Combinatorica.* A preliminary version appeared in *Proc. of 27th Symposium on Foundations of Computer Science*, pp 368–379, Toronto, 1986.

[B] Babai L., "Trading Group Theory for Randomness," *Proc. of 17th Symposium on Theory of Computing*, pp 421–429, Providence, 1985.

[BM] Babai L. and S. Moran, "Arthur Merlin Games: a Randomized Proof System and a Hierarchy of Complexity Classes," *JCSS*, Vol. 36 (1988), No. 2, pp 254–276.

[BHZ] Boppana R., J. Hastad and S. Zachos "Does *co-NP* Have Short Interactive Proofs", *Information Processing Letters*, Vol 25 (1987), No. 2, pp 127–132.

[BGGHKMR] Ben Or M., O. Goldreich , S. Goldwasser, J. Hastad, J. Kilian, S. Micali, and P. Rogaway, "Everything Provable is Provable in Zero-Knowledge", *Proc. of CRYPTO '88*, Santa Barabara, 1988.

[CW] Carter J.L. and N.M. Wegman."Universal classes of hash functions", *JCSS*, Vol. 18 (1979), No. 2, pp 143–154.

[F] Fortnow L., "The Complexity of Perfect Zero-Knowledge," *Proc. of 19th Symposium on Theory of Computing*, pp 204–209, New York, 1987.

[GMR1] Goldwasser, S., S. Micali, and C. Rackoff, "The Knowledge Complexity of Interactive Proofs," *Proc. of 17th Symposium on Theory of Computing*, pp 291–305, Providence, 1985.

[GMR2] Goldwasser, S., S. Micali, and C. Rackoff, "Proofs, Knowledge, and Computation," *SIAM Journal on Computing*, Vol. 18 (1989), No. 1, pp 186–208.

[GMS] Goldreich, O., Y. Mansour, and M. Sipser, "Interactive Proof Systems: Provers That Never Fail and Random Selection," *Proc. of 28th Symposium on Foundations of Computer Science*, pp 449–461, Los Angeles, 1987.

[GMW] Goldreich, O., S. Micali, and A. Wigderson, "Proofs that Yield Nothing but their Validity and a Methodology of Cryptographic Protocol Design," *Proc. of 27th Symposium on Foundations of Computer Science*, pp 174–187, Toronto, 1986.

[GS] Goldwasser, S., and M. Sipser, "Private Coins Versus Public Coins in Interactive Proof Systems," *Proc. of 18th Symposium on Theory of Computing*, pp 59–68, Berkeley, 1986.

[IY] Impagliazzo, R., and M. Yung, "Direct Minimum-Knowledge Computations," *Proc. of CRYPTO '87*, pp 40–51, Santa Barbara, 1987.

[O] Oren, Y., "On the Cunning Powers of Cheating Verifiers: Some Observations about Zero-Knowledge Proofs," *Proc. of 28th Symposium on Foundations of Computer Science*, pp 462–471, Los Angeles, 1987.

[TW] Tompa, M., and H. Woll,"Random Self-Reducibility and Zero-Knowledge Interactive Proofs of Possession of Information," *Proc. of 28th Symposium on Foundations of Computer Science*, pp 472–482, Los Angeles, 1987.

[S] Sipser M., "A Complexity Theoretic Approach to Randomness," *Proc. of 15th Symposium on Theory of Computing*, pp 330–335, Boston, 1983.