ON THE COST OF REDUCING THE INTERACTION IN INTERACTIVE PROOF PROTOCOLS

by

Shlomo Moran

Technical Report #437

November 1986
ON THE COST OF REDUCING THE INTERACTION IN
INTERACTIVE PROOF PROTOCOLS

by

Shlomo Moran(l)

Department of Computer Science,
the Technion, Haifa 32000, Israel

ABSTRACT
The following problem is considered: Let $L$ be a language that can be accepted by an interactive proof protocol with $f(n)$-time bounded verifier within $g(n)$ interactions. How powerful, in terms of time complexity, must be a verifier that accepts $L$ by an interactive proof protocol with only one interaction? A pure complexity-theoretic argument implies an that a $2^{O(g(n))}$-time bounded verifier is always sufficiently powerful (independent on the number of interactions, $g(n)$). This bound was improved for slowly increasing $g(n)$ by the results of [B] and [OS], which implies that a $f(n)^{O(2^{g(n)})}$-time bounded verifier can accept $L$ within one interaction. The main result of this paper improves that last upper bound to $f(n)^{O(g(n))}$. Another application of that result is that the number of interactions needed to recognize a language by an interactive proof protocol with a polynomial time bounded verifier is invariant under multiplication by a (positive) constant. For comparison, the results of [B] and [GS] mentioned above implies only that this number is invariant under additive constant.

(l) Part of this work was done while the author was at IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.
1. INTRODUCTION

The notion of interactive proof was introduced by Goldwaser et al. in [GMR], where it was used in the wider context of zero-knowledge interactive proofs. Since then it was studied in few other papers, e.g. [B,BH,F,GMW,FZ], and it appears to be of independent interest, as it provide a randomized extension of the class \( NP \) which is analogue to the randomized extension of the class \( P \) by the class \( BPP \). One way to define an interactive proof is as a game between two players, as follows:

One player, called \( V \) (the verifier), is represented by a polynomial time bounded Turing machine, and the second player, called \( P \) (the prover), is represented by an arbitrary function. Both players receive an input word \( w \) of some length \( n \), and \( V \) receives also a random string \( r \), which is not known to \( P \); the length of \( r \) is \( l = l(n) \), where \( l(n) \) is bounded by a polynomial in \( n \). \( V \) and \( P \) exchange messages of length at most polynomial in \( n \), for at most polynomially many rounds: At round \( i \) \( V \) sends a string \( x_i \) to \( P \) and receives back from \( P \) a string \( y_i \). More formally, \( x_i = V(w,r,x_1,y_1,\ldots,y_{i-1}) \) and \( y_i = P(w,x_1,y_1,\ldots,y_i) \); the pair of messages \((x_i,y_i)\) is denoted as the \( i\)-th round of the protocol. The game is terminated by having \( V \) decide whether to accept or reject the input word, \( w \).

Such a computation is denoted as a \( V^*P \) computation, and is represented by the message stream \( s = (x_1,y_1,\ldots,x_g,y_g,\text{DECIDE}) \), where DECIDE is ACCEPT or REJECT. Note that \( V, P, w \) and \( r \) uniquely determine the message stream \( s \). \( V^*P(w,r) \) denotes the output of a \( V^*P \) computation on input \( w \) and a random string \( r \).

The probability that a \( V^*P \) computation accepts \( w \) is defined by the ratio:

\[
pr(V^*P \text{ accepts } w) = \frac{|\{r : V^*P(w,r) \text{ ACCEPTs} \}|}{2^i},
\]

and the probability that \( V \) accepts \( w \) is defined by:

\[
pr(V \text{ accepts } w) = \max_p \{ pr(V^*P \text{ accepts } w) \}.
\]

A language \( L = L(V) \) is accepted by \( V \) if for each \( w \in L \), \( pr(V \text{ accepts } w) \geq 2/3 \), and for each \( w \notin L \), \( pr(V \text{ accepts } w) \leq 1/3 \). \( L \) is said to be in \( IP[g(n)] \) if \( L = L(V) \) for some verifier \( V \) that never sends more than \( g(n) \) messages on input of length \( n \) (i.e., within at most \( g(n) \) rounds). \( L \) is in \( IP \) if it is in \( IP[g(n)] \) for some polynomial \( g \).

A restricted version of interactive proofs, denoted as Arthur-Merlin games, was defined by Babai in [B]. The difference is in that the verifier is restricted to send messages \( x_i \) such that the concatenation \( x_1x_2\cdots x_g \) is the "secret" random string \( r \). The verifier and prover in such protocols were denoted as Arthur (A) and Merlin (M) respectively, and the protocol is denoted as an \( AM \) protocol. \( AM[g(n)] \) and \( AM \) are defined similarly to \( IP[g(n)] \) and \( IP \). A verifier in an \( AM \) protocol will be denoted as an Arthur-type verifier. We shall assume that the complexity of an Arthur-type verifier is proportional to the length of the random string it generates, which is proportional to
the length of the message stream (a similar assumption is also implicit in [B]).

Clearly, if a language \( L \) is in \( AM \) then it is also in \( IP \). In [GS] it was shown that the implication in the opposite direction also holds, namely that \( IP \{ g(n) \} \subseteq AM \{ g(n)+2 \} \); in particular, their result implies that \( IP = AM \).

One of the most interesting questions related to interactive proofs in general and to the class \( IP \) in particular is whether "one round is enough", i.e. whether \( IP = IP[1] \). In [GMR] it was conjectured that this is not the case, and in fact that for all constants \( k \geq 1, IP[k] \neq IP[k+1] \). In [B] Babai shows that if the number of rounds of an \( AM \) protocol is at least two, then it can be reduced by one, in the cost of squaring the time complexity of \( A \)’s algorithm. This implies that for each constant \( k \geq 1, AM[k] = AM[1] \). By the result of [GS] mentioned above, this last result implies that for each constant \( k \geq 1, IP[k] = IP[1] \). This disproves the \( IP[k] \neq IP[k+1] \) conjecture, but leaves open the more general \( IP = IP[1] \) problem. This paper is motivated by the refinement of that problem given below:

Let \( F \) be a class of non decreasing real valued functions. The definition of interactive proofs naturally extends to \( F \)-time bounded \(^{(2)} \) verifiers in the following way:

Definition 1.1: \( IP[F,g(n)] \) is the set of all languages that can be accepted by interactive proofs with \( F \)-time bounded verifier within at most \( g(n) \) rounds.

By the above definition, \( IP[g(n)] = IP[P, g(n)] \), where \( P \) is the set of polynomials.

Using the above notation, the problem studied in this paper can be described as follows:

Problem: Given \( g(n) \), find a 'best' class \( F \) such that \( IP[g(n)] \subseteq IP[F,1] \).

For a class of functions \( F \), let \( 2^F \) denote the class \( \{ f(n) : f(n) \in F \} \), and let \( EXP = 2^P \). First we note that \( IP \subseteq IP[EXP,1] \), and in fact that \( IP \subseteq PSPACE \): Let \( L = L(V) \) be in \( IP \). Then by [F], an optimal prover for \( V \) can be realized by a PSPACE bounded machine (a prover \( P \) is optimal for \( V \) if for each \( w \), \( pr(V^*P) \) accepts \( w \) is maximized). Hence, a P-space bounded machine that simulates a \( V^*P \) computation with an optimal prover \( P \) on all possible random strings \( r \) accepts \( L(V) \). Thus, a P-space (and EXP-time) bounded verifier can accept \( L \) (with no prover at all) in one round.

So far no general bound which is better than the above complexity-theoretic bound was published. Combining Corollary 4.2 in [B] with the Main Theorem of [GS] implies that \( IP[g(n)] \subseteq IP[P^{2^{o(n)}},1] \). This latter result improves the above bound for \( g(n) = o(\log n) \), but yields a weaker bound for \( g(n) = \omega(\log n) \).

\(^{(2)} \) A function \( h \) is \( F \) bounded if for some \( f \) in \( F \), \( h(n) \leq f(n) \) for all \( n \).
In this paper we show that it is possible to reduce the number of rounds of an AM protocol by a factor of one half, in the price of increasing the verifier's complexity by a constant power. For an \( f(n) \)-bounded verifier, this provides a bound which is better than the complexity-theoretic bound as far as \( g(n) < f(n)^e \) for some absolute constant \( e \). Our main result is the following:

**Main Theorem:** If a language \( L \subseteq \{0,1\}^* \) is accepted by an Arthur-Merlin protocol with \( f(n) \)-time bounded verifier in \( g(n) \) rounds, then \( L \) is accepted by an Arthur-Merlin protocol with an \( f(n)^c \)-time bounded verifier in \( \lceil g(n)/2 \rceil + 1 \) rounds, where \( c \) is some absolute constant that does not depend on \( L \).

The above theorem implies that \( IP[g(n)] \subseteq IP[p^c(n),1] \), where \( p^c(n) = \{ p(n)^c : p \) is a polynomial \}. It also implies that the number of rounds needed to recognize a language by interactive proof protocols with a polynomial time bounded verifier can always be reduced by a constant factor.

The rest of the paper is devoted to the proof of the main theorem as follows. Let \( V \) be an Arthur-type verifier and let \( w \) be an input word. In the next section we reduce the problem "is \( w \) in \( L(V) \)?" to a graph theoretic problem; specifically, we define the concept of lower bound tree, and then show that \( V \) accepts \( w \) in \( g \) rounds iff a certain tree of height \( 2g \) contains a large lower bound sub-tree. In section 3 we present an AM protocol that in \( \lceil g/2 \rceil + 1 \) rounds checks whether \( w \) is in \( L(V) \). In this protocol \( M \) is trying to convince \( A \) that a lower bound tree of the kind described above indeed exists. This is done by using the Carter-Wegman universal hashing functions \([CW]\) in a way that is an extension of the way these functions are used in [GS]. The correctness and complexity of the protocol are given in Section 4, together with some corollaries.

2. REDUCTION TO A GRAPH-THEORETIC PROBLEM.

We start by showing that it is sufficient to consider verifiers of special kind:

**Lemma 2.1:** If a language \( L \) is accepted by an Arthur-type verifier \( V' \) that uses random strings of length \( l' = l'(n) \) within \( g = g(n) \) rounds, then it is also accepted within \( g \) rounds by an Arthur-type verifier \( V \) that uses random strings of length \( l = O(l'g^6 \log (g^{l'})) \), which satisfies assumptions \( (A1) \) and \( (A2) \) below for each input \( w \) of length \( n \) \((l, g \) and \( \mu \) denote \( l(n), g(n) \) and \( \mu(n) \) resp.):

- \( (A1) \) All messages \( x_i \) (sent by \( V \)) and \( y_i \) (sent by \( P \)) are binary words of the same length \( \mu = l/g \).
- \( (A2) \) The error probability \( e \) of \( V \)'s protocol is smaller than \( l'^{-e'} \).

**Proof:** \( (A2) \) can be achieved by executing \( V' \)'s protocol, whose error probability \( \leq 1/3 \), at parallel \( O(g^5 \log (g^{l'})) \) times, and deciding according to the majority of the outcomes (see [B,GS]).
(A1) can be achieved by padding the messages sent by V and P with random bits to make them of equal length. This increases the length of the message stream by a factor of at most \( \mu \). Note that \( \mu = \frac{1}{g} \), since the concatenation of V's messages gives the random string \( r \); the lemma follows.

In the rest of the paper, V denotes a verifier that satisfies (A1) and (A2) above, and \( w \) denotes an input word. \( s = (x_1, y_1, \ldots, x_g, y_g) \) will denote a message stream produced by a \( V^* P (w, r) \) computation for some prover \( P \) and some random string \( r \); \( s(i) \) denotes the prefix of the first \( i \) messages in \( s \) (\( i = 0, \ldots, 2g \)). By definition, \( s(0) \) is the empty sequence \( \Phi \) for all message streams \( s \).

**Definition 2.1:** For a given \( V \) and \( w \), \( D(V, w) = (N, E) \) is a directed tree defined as follows:

- \( N = \{ s(i) : s \) is an accepting message stream, \( 0 \leq i \leq 2g \} \).
- \( E = \{ (s(i) \rightarrow s(i+1)) : s \) is an accepting message stream, \( 0 \leq i < 2g \} \).

The root of \( D(V, w) \) is the empty message stream \( \Phi \), and the leaves are \( 2g \)-tuples \( (x_1, y_1, \ldots, x_g, y_g) \) that correspond to accepting message streams. Informally, \( D(V, w) \) is that part of the game-tree of \( V \) on input \( w \) that contains only the games in which \( P \) wins.

**Definition 2.2:** Let \( V \) and \( w \) be given. A sub-tree \( T \) of \( D(V, w) \) is a **consistent tree** if it is a tree with the following properties:

- a) The root of \( V \) is \( \Phi \) and the leaves of \( T \) are accepting message streams.
- b) for each \( 0 < i \leq g \) and for each accepting message stream \( s \), if \( s(2i-1) = (x_1, y_1, \ldots, x_i) \) is in \( T \), then the out-degree of \( s(2i-1) \) in \( T \) is 1.

**Lemma 2.2:** Let \( B \) be a set of accepting message streams. Then the following are equivalent:

1. There is a unique prover \( P \) such that all the messages in \( B \) are produced by \( V^* P \) computations.
2. \( B \) is the set of leaves of a consistent sub tree \( T \) of \( D(V, w) \).

**Proof:** (2) \( \rightarrow \) (1): We must define a function \( P \) such that the leaves of \( T \) are accepting message streams of \( V^* P \) computations on input \( w \). It is easy to verify that any \( P \) satisfying property (C) below does the job, and that such a \( P \) exists:

- (C) If \( (x_1, y_1, \ldots, x_i, y_i) \) is in \( T \), then \( P(x_1, y_1, \ldots, x_i) = y_i \).

The proof that (1) \( \rightarrow \) (2) is similar, and omitted.

**Definition 2.3:** A directed tree \( U \) of height \( g \) is **semi-uniform** if there exists an integer sequence \( \bar{b}(U) = (b_0, \ldots, b_{g-1}) \) such that for each \( 0 \leq i < g \), each vertex \( v \) at level \( i \) in \( U \) satisfies \( 2^h \leq d_{\text{out}}(v) < 2^{h+1} \). The sequence \( \bar{b}(U) \) is called the **characteristic sequence** of \( U \).
For a tree $T$, let $\lambda(T)$ denote the number of leaves in $T$; for a vertex $u$ in $T$, $\lambda_T(u)$ denotes the number of leaves in the sub-tree of $T$ rooted at $u$. When $T$ will be clear from the text, we shall use $\lambda(u)$ instead of $\lambda_T(u)$.

Note that if $U$ is a semi-uniform tree with characteristic sequence $\vec{b}(U) = (b_0, \ldots, b_{k-1})$, then all the leaves of $U$ are at level $g$, and $\lambda(U) \geq 2^{b_0 \cdot \cdot \cdot b_{k-1}}$.

**Lemma 2.3:** Let $T$ be a tree all whose leaves are at level $g$, and assume that $\lambda(T) < 2^g$. Then $T$ contains a semi-uniform sub tree $U$ with characteristic sequence $\vec{b}(U) = (b_0, \ldots, b_{k-1})$, such that $2^{b_0 \cdot \cdot \cdot b_{k-1}} > \lambda(T)/(2^g)^{-1}$.

**Proof:** Induction on $g$. For $g=1$ take $U=T$ and $\vec{b}(U) = (b_0)$, where $b_0 = \lceil \log \lambda(T) \rceil$.

We now assume that the lemma holds for $g-1$, and prove it for $g$. Let $T$ satisfying the hypothesis of the lemma be given. We have to define a characteristic sequence $(b_0, \ldots, b_{k-1})$ and a corresponding semi-uniform sub-tree $U$:

Partition the vertices at level $g-1$ of $T$ (the leaves' fathers) to at most $l$ sets, $\alpha_0, \ldots, \alpha_{l-1}$, where $\alpha_i$ contains those vertices $v$ satisfying $2^i \leq \lambda(v) < 2^{i+1}$. Let $i_0$ be such that $\sum_{v \in \alpha_i} \lambda(v)$ is maximized. In particular, we have that

$$\sum_{v \in \alpha_i} \lambda(v) \geq \lambda(T)/I, \text{ and hence that:}$$

$$(\&\&) \ 2^{i_0} | \alpha_{i_0} | > \sum_{v \in \alpha_i} \lambda(v) / 2 \geq \lambda(T)/2I.$$  

Set $b_{k-1}$ to $i_0$, and let $T'$ be the tree obtained by deleting from $T$ all the vertices that are not in $\alpha_i$, or ancestors of vertices in $\alpha_i$. Then $\alpha_i$ is the set of leaves of $T'$, and are all at level $g-1$. By the induction hypothesis $T'$ contains a semi-uniform sub-tree of $U'$ with a characteristic sequence $\vec{b}(U') = (b_0, \ldots, b_{k-1})$ that satisfies the lemma. In particular,

$$(\&\&\&) \ 2^{b_0 \cdot \cdot \cdot b_{k-1}} \geq \lambda(T')/(2I)^{g-2} = | \alpha_{i_0} | / (2I)^{g-2}.$$

$U$ is obtained by adding to the leaves of $U'$ their sons in $T$. The lemma follows by noting that:

$$2^{b_0 \cdot \cdot \cdot b_{k-1}} = [2^{b_0 \cdot \cdot \cdot b_{k-1}}] \cdot 2^{i_0} \geq | \alpha_{i_0} | / (2I)^{g-2} \geq \lambda(T)/(2I)^{g-1}.$$  

(The first inequality follows from (\&\&\&), and the last from (\&\&)).

The following definition plays an essential role in our protocol.

**Definition 2.4:** A tree $U$ is a lower bound tree for $V$ and $w$ if it is a semi-uniform, consistent sub tree of $D(V,w)$.

Let $U$ be a lower bound tree for $V$ and $w$. Then, since $U$ is consistent, the out-degrees of all vertices in the odd levels of $U$ are 1. This means that the characteristic sequence $\vec{b}(U) = (b_0,0,b_1,0,\ldots,b_{k-1},0)$ (i.e., for a vertex $v$ at level $2i$, $2^h \leq \deg_v(v) < 2^{h+1}$). For brevity, we will omit the zeroes from $\vec{b}(U)$, and will denote it by $(b_0,b_1,\ldots,b_{k-1})$.

(3) All logarithms in this paper are to base 2.
The following lemma reduces the problem whether \( w \in L(V) \) to two purely graph theoretic problems.

**Lemma 2.4:** Assume that \( l \geq 4 \). Then the following are equivalent for a verifier \( V \) and a word \( w \).

1. \( w \in L(V) \).
2. \( D(V,w) \) contains a lower-bound tree \( U \) with characteristic sequence \( \vec{b}(U)=(b_0, \ldots, b_{g-1}) \), such that
   \[
   2^{b_0+\cdots+b_{g-1}} > 2^{l-1-(x-1)(\log (l+1)+1)} > 2^{l-2g \log l}.
   \]
3. \( D(V,w) \) contains a consistent sub-tree \( T \) with \( \lambda(T) \geq 2^{l-4 \log l} \).

**Proof:** (1) \( \Rightarrow \) (2): Let \( P \) be an optimal prover, and let \( T \) be the consistent subtree of \( D(V,w) \) whose leaves correspond to the accepting computation of \( V^*P \), as follows from Lemma 2.2. Since \( V \) is an Arthur-type verifier, each leaf in \( D(V,w) \) corresponds to a unique random string \( r \) such that \( V^*P(w,r) \) accepts \( w \). It follows from Lemma 2.2 and the definitions that \( 2^{l-1} < 2^{l-4 \log l} \). Let \( T' \) be the tree obtained by ignoring the vertices in the odd levels of \( T \) (that have out-degree one). Then \( \lambda(T')=\lambda(T) < 2^{l+1} \), and all the leaves of \( T' \) are at level \( g \). By substituting in Lemma 2.3, \( T \) has a lower-bound sub-tree satisfying the first inequality. The second inequality follows by the assumption \( l \geq 4 \).

(2) \( \Rightarrow \) (3): Take \( T = U \).

(3) \( \Rightarrow \) (1): We shall show that if (1) does not hold then (3) does not hold too: By (A2), if (1) does not hold then for each prover \( P \), the number of accepting \( V^*P \) computations is smaller than \( 2^{l-4 \log l} \). By Lemma 2.2, this means that every consistent sub-tree of \( D(V,w) \) has less than that number of leaves. \( \square \)

**Lemma 2.5** Let \( \vec{b}=(b_0, \ldots, b_{g-1}) \) be the characteristic sequence of a lower-bound tree \( U \) satisfying Lemma 2.4(2). Then for \( i=0, \ldots, g-1 \) it holds that \( \mu - 2g \log l \leq b_i \leq \mu \).

**Proof:** Let \( u = s(2i) \) be a vertex in \( U \). Then we have that

(a) \( 2^{b_i} \leq d_{out}(u) \) (since \( U \) is a lower bound tree with characteristic sequence \( (b_0, \ldots, b_{g-1}) \)), and

(b) \( d_{out}(u) \leq 2^\mu \) (by (A1) and the fact that \( d_{out}(u) \) is bounded by the number of distinct messages \( x_{i+1} \)).

The right inequality follows from (a) and (b).

To see the left inequality, assume that \( b_{i_0} < \mu - 2g \log l \) for some \( i_0 \). Then

\[
\sum_{i=0}^{g-1} b_i \leq g \mu - 2g \log l \quad (\text{since for all } i \ b_i \leq \mu, \text{ and } b_{i_0} \leq \mu - 2g \log l).
\]

Since, by (A1), \( g \mu = l \), we have that \( \sum_{i=0}^{g-1} b_i < l - 2g \log l \). But this contradicts Lemma 2.4(2). \( \square \)
3. THE PROTOCOL

Before presenting our protocol, it might be useful to describe the protocol in [GS] using our terminology: In that protocol, \( M \) is trying to convince \( A \) that an input word \( w \) is accepted by a given \( V^*P \) protocol within \( g \) rounds. For our sake we may assume that the prover in the \( V^*P \) protocol is an Arthur-type prover. The \( AM \) protocol of [GS] can be described as follows: At round \( i \) \( M \) sends \( A \) a pair of messages \((x_i, y_i)\), and \( A \) uses the Carter-Wegman universal hashing functions to verify that the message \( x_i \) is 'chosen at random' from a set of at least \( 2^{b_0} \) distinct messages, where \((b_0, \ldots, b_{g-1})\) is the characteristic sequence of a lower bound sub-tree of \( D(V, w) \). Thus, the protocol can be viewed as a 'random' walk from the root to a leaf in a supposed lower bound sub tree of \( D(V, w) \). If such a tree indeed exists, then with high probability the walk is terminated at one of its leaves, which is an accepting message stream, and then \( A \) accepts. Otherwise, with high probability the walk is terminated in a rejecting message stream, and \( A \) rejects.

In its general outline, our protocol is similar to the one in [GS]. The crucial difference is that our protocol simulates two rounds of the \( V^*P \) protocol in one round. However, if we just let \( M \) send \( A \) at each round one 'random' quadruple \((x_i, y_i, x_{i+1}, y_{i+1})\), then \( M \) will have an unfair advantage over \( A \), since \( A \) cannot check that all the possible 'random' quadruples corresponds to the same consistent sub tree (i.e., can be produced by the same prover). To overcome this difficulty, we modify the protocol as follows: First, instead of sending a single quadruple, \( M \) sends \( A \) at each round a 'random' set of many quadruples of the form \( \{(x_i, y_i, x_{i+1}, y_{i+1}) \} : j = 1, \ldots, k \) (note that all these quadruples correspond to the same consistent tree). Second, in addition to verifying that this set is taken 'at random' from a large collection of such sets, \( A \) also picks at random one element from this set as the actual quadruple that will be used in the rest of the protocol.

A key tool in the protocol is a lemma based on the Carter-Wegman universal hashing functions [CW], that appeared in [S,GS]. This lemma, with the corresponding definitions, follow.

A random linear function from \( \{0,1\}^k \) to \( \{0,1\}^b \) is a function \( \overline{h} \) defined by a random 0-1 matrix \( D \) of size \( k \times b \), that maps each \( \overline{x} \in \{0,1\}^k \) to \( \overline{h}(\overline{x}) = \overline{x}D \in \{0,1\}^b \). For a linear function \( \overline{h} \) and a subset \( C \) of \( \{0,1\}^k \), \( \overline{h}(C) \) is the image of \( C \) by \( \overline{h} \). For a set \( H \) of linear functions from \( \{0,1\}^k \) to \( \{0,1\}^b \) and a subset \( C \) of \( \{0,1\}^k \), \( H(C) \) denotes the union \( \bigcup_{h \in H} \overline{h}(C) \).

Lemma 3.1 [CW, S, GS]: Let \( H \) be a set of \( l \) random linear functions from \( \{0,1\}^k \) to \( \{0,1\}^b \), let \( C \) be an arbitrary subset of \( \{0,1\}^k \), and let \( Z \) be a random subset of \( \{0,1\}^b \) of cardinality \( l^2 \). Then:

1. If \( b \leq 2 + \lceil \log \frac{1}{\epsilon} \rceil \), then \( \Pr[H(C) \cap Z \neq \emptyset] \geq 1 - 2^{-l/8} \) (here \( \emptyset \) denotes the empty set).
The description of the AM protocol that accepts $L(V)$ follows.

ROUND 0(A): A sends $M$ a dummy string and waits for $M$’s response.

ROUND 0(M): $M$ finds a lower-bound sub-tree $U$ of $D(V,w)$ and submits the corresponding characteristic sequence $\bar{b} = \bar{b}(U) = (b_0, \ldots, b_{k-1})$ to $A$, where $\sum_{i=0}^{k-1} b_i \geq 1 - 2g \log l$ (if $M$ fails to send such a sequence, $w$ is rejected). $M$ also sends $A$ $8l$ copies of the empty message stream $s(0) = \Phi$.

In the rest of the protocol $M$ is trying to convince $A$ that $D(V,w)$ contains a lower-bound sub-tree with characteristic sequence $\bar{b}$. ROUND $i$ is described below ($1 \leq i \leq g/2$). It is assumed that $g$ is even. For brevity, $b$ denotes $b_{2i-1}$ and $c$ denotes $b_{2i}$.

ROUND $i$ (A): $A$ chooses at random a vertex $u = s(4i - 4)$ out of the $8l$ vertices sent by $M$ in ROUND $i - 1$. Then $A$ sends $M$ the vertex $u$, and sets $H_1, H_2, Z_1$ and $Z_2$ as follows:

- $H_1$ is a set of $l$ random linear functions from $\{0,1\}^h$ to $\{0,1\}^{b+2}$.
- $H_2$ is a set of $l$ random linear functions from $\{0,1\}^{8l}$ to $\{0,1\}^{8c+2}$.
- $Z_1$ and $Z_2$ are random subsets of cardinality $l^2$ of $\{0,1\}^{b+2}$ and $\{0,1\}^{8c+2}$, resp.

ROUND $i$ (M): $M$ sends $A$ a pair $(x,y)$ such that $x$ is in $H_1^{-1}(Z_1)$, and $8l$ pairs $(x[1], y[1]), \ldots, (x[8l], y[8l])$ such that the string $x[1]x[2] \cdots x[8l]$ is in $H_2^{-1}(Z_2)$, and the $8l$ vertices $\{(u,x,y,x[j],y[j]) : j = 1, \ldots, 8l\}$ are in the lower-bound sub-tree $U$ computed by $M$ at ROUND 0. $((u,x,y,x[j],y[j])$ denotes the sequence obtained by appending the messages $x$, $y$, $x[j]$ and $y[j]$ to $u$.)

Finally, $A$ chooses at random a vertex $v = s(2g)$ out of the $8l$ messages sent by $M$ at round $g/2$, and it accepts iff $s(2g)$ is an accepting message-stream.

4. PROOF OF CORRECTNESS.

Informal description of the proof: Let $u \in D(V,w)$ be given. We shall say that $u$ is rich if it occurs in a "very large" lower bound sub-tree of $D(V,w)$, and that $u$ is poor if it occurs only in "very small" consistent sub-trees of $D(V,w)$. The proof then proceeds as follows:

First we use Lemma 2.4 to show that the vertex $s(0) = \Phi$, which is chosen by $V$ at ROUND 1(A), is rich (with respect to $V$ and $w$) if $w$ is in $L(V)$, and that it is poor otherwise.
The next step, which is the main part of the proof, consists of showing that if \( A \) picks a rich vertex \( s(4i-4) \) at round \( i \), then it is very likely to pick a rich vertex \( s(4i) \) also at round \( i+1 \). Similarly, if \( A \) picks a poor vertex at round \( i \), then it is very likely to pick a poor vertex at round \( i+1 \).

The proof is completed by showing that if the vertex \( s(2g) \) picked by \( A \) at round \( g/2+1 \) is rich then \( A \) accepts, and that if \( s(2g) \) is poor then \( A \) rejects.

A formal proof: We start with some definitions and notations. Throughout the rest of the discussion, \( \overline{b} = (b_0, \ldots, b_{g-1}) \) is the sequence produced by \( M \) at ROUND 0.

Definition 4.1: Let \( u \) be a prefix of a message stream \( s \). Then for some \( i \) in \([0, g]\), \( u = s(2i) \) or \( u = s(2i-1) \). \( h(u) \) and \( B(u) \) are integers defined by: \( h(u) = g-i \), and \( B(u) = [\text{if } i = g \text{ then } 0, \text{ else } b_i + b_{i+1} + \cdots + b_{g-1}] \).

Definition 4.2: For a prefix of a message stream \( u \), \( \beta(u) \) is defined by:

\[
\beta(u) = \begin{cases} 
\text{If } u \text{ is in } D(V,w) \text{ then } \max\{\lambda_T(u); T \text{ is a consistent sub-tree of } D(V,w)\}, & \text{else } 0.
\end{cases}
\]

The following facts are easily verified for a vertex \( u \) in \( D(V,w) \):

(F1) If \( u = s(2i-1) \) then \( \beta(u) = \max(\beta(v); v \text{ is a son of } u \text{ in } D(V,w)) \) \((1 \leq i \leq g)\).

(F2) If \( u = s(2i) \) then \( \beta(u) = \sum(\beta(v); v \text{ is a son of } u \text{ in } D(V,w)) \) \((0 \leq i < g)\).

Definition 4.3: Let \( u \) be a prefix of a message stream, and let \( h = h(u) \) and \( B = B(u) \). Then \( u \) is rich if it is in a lower bound tree \( U \) with characteristic sequence \( \overline{b} = (b_0, \ldots, b_{g-1}) \). \( u \) is poor if \( \beta(u) \leq 2^{8-h} \log l \). In particular, if \( u \) is a message stream \( (x_1, y_1, \ldots, x_g, y_g) \), then \( u \) is rich if it is an accepting message stream, and it is poor otherwise.

It follows from Lemma 2.4(2) that if \( w \in L(V) \) then the root \( \Phi \) of \( D(V,w) \) is rich, and from Lemma 2.4(3) that if \( w \) is not in \( L(V) \) then \( \Phi \) is poor.

In the following \( i \) is some fixed integer between 1 and \( g/2 \). \( b \) and \( c \) denote, as before, \( b_{2i-1} \) and \( b_{2i} \) resp.

The fact that if \( w \) is in \( L(V) \) then it is likely to be accepted by \( A \) follows from the next theorem:

Theorem 4.1 Assume that the vertex \( u = s(4i-4) \) sent by \( A \) to \( M \) at ROUND \( i \) is rich, then with probability \( \geq(1-2^{-l/8})^2 \) so is the vertex \( s(4i) \) sent by \( A \) at ROUND \( i+1 \).

Proof: Let \( p_0 = 1-2^{-l/8} \). It is sufficient to show that with probability \( \geq p_0^2 \), \( M \) is able to produce \( (x,y) \) and \( (x[1],y[1]), \ldots, (x[8l],y[8l]) \) such that all the vertices \( (u,x,y,x[j],y[j]) \) are in \( U \) \((j = 1, \ldots, 8l)\).

Since \( u \) is in \( U \), there are at least \( 2^8 \) distinct \( x \)-s, for each of which there is a \( y \) such that \( v = (u,x,y) \) is also in \( U \). Thus, by Lemma 3.1(1), \( M \) is able to produce such an \( x \) (and \( y \)) with probability \( \geq p_0 \).

Conditioning on the event that \( M \) produces \( x \) as above, there are at least \( 2^8 x^{-}\)s, for each of which there is a \( y^{-} \) such
that \( v' = (u, x, y, x', y') \) is in \( U \). The number of \( 8l \)-tuples of such \( x' \)'s is, therefore, at least \( 2^{8l} \). The theorem now follows by another application of Lemma 3.1(1).

The fact that if \( w \) is not in \( L(V) \) then it is likely to be rejected by \( A \) follows from the next theorem:

**Theorem 4.2:** Assume that \( g \geq 4 \). Then if the vertex \( u = s(4i - 4) \) sent by \( A \) to \( M \) at round \( i \) is poor, then with probability \( \geq (1 - 1/(8g))(1 - 1/(g^2))^2 \) the vertex \( u' = s(4i) \) sent by \( A \) at round \( i + 1 \) is also poor.

**Proof:** Let \( E_1, E_2 \) and \( E_3 \) be the following events:

- **E1:** For every \( x \) in \( H_1^{-1}(Z_1) \), and for every \( y \), the vertex \( v = (u, x, y) \) (i.e., \( u \) concatenated by \( (x, y) \)) is poor.
- **E2:** For every \( x \) in \( H_1^{-1}(Z_1) \), for every \( x[1] \cdots x[8l] \) in \( H_2^{-1}(Z_2) \), and for all \( y, x[1], \cdots, y[8l] \), at most \( \mu \) out of the \( 8l \) vertices \( u_j = (u, x, y, x[j], y[j]) \) are not poor (\( j = 1, \cdots, 8l \)).
- **E3:** The vertex \( u' = s(4i) = (s, x, y, x', y') \) picked by \( A \) at the beginning of ROUND \( i + 1 \) is poor.

We have to prove that \( \text{pr} [E_3] \geq (1 - 1/(8g))(1 - 1/(g^2))^2 \). Using the inequality:

\[
\text{pr} [E_3 | E_2] \text{pr} [E_2] = \text{pr} [E_3 \cap E_2] \leq \text{pr} [E_3],
\]
and the fact that \( \text{pr} [E_3 | E_2] \geq (8l - \mu)/8l = 1 - 1/(8g) \), we get that \( \text{pr} [E_3] \geq (1 - 1/(8g)) \text{pr} [E_2] \).

Thus, the theorem will follow if we show that \( \text{pr} [E_2] \geq (1 - 1/g^2)^2 \). For this, we shall use the inequality:

\[
\text{pr} [E_2 | E_1] \text{pr} [E_1] = \text{pr} [E_2 \cap E_1] \leq \text{pr} [E_2],
\]
and then show that both \( \text{pr} [E_1] \) and \( \text{pr} [E_2 | E_1] \) are larger than \( 1 - 1/g^2 \).

The proof of Theorem 4.2 proceeds via Lemmas E1 and E2 below. To simplify notations, we shall denote \( h(u) \) by \( h \), and \( B(u) \) by \( B \).

**Lemma E1:** \( \text{pr} [E_1] \geq 1 - 1/g^2 \).

**Proof:** By (F1) and Definition 4.3, the vertex \( (u, x) \) is poor iff for every \( y \) the vertex \( (u, x, y) \) is poor. Hence, \( E_1 \) is equivalent to: For every \( x \) in \( H_1^{-1}(Z_1) \), the vertex \( v = (u, x) \) is poor.

By the assumption that \( u \) is poor, \( \beta(u) \leq 2^{B - h} g^{4 \log l} \). Assume now that for some message \( x \) the vertex \( v = (u, x) \) is not poor. Since \( h(v) = h - 1 \) and \( B(v) = B - b \) we have, by the definition of the term 'poor', that \( \beta(v) \geq 2^{B - b - (h - 1)g^4 \log l} \).

Since, by (F2), \( \beta(u) = \sum_{v = (u, x)} \beta(v) \), we have that the number of possible \( x \) for which \( v = (u, x) \) is not poor is bounded by:

\[
\beta(u)/2^{B - b - (h - 1)g^4 \log l} \leq 2^{B - h - 4 \log l} (B - b - (h - 1)g^4 \log l) = 2^{B - b - 4 \log l}.
\]

By Lemma 3.1(2), the probability that \( H_1^{-1}(Z_1) \) contains such an \( x \) is bounded by \( l^3/2g^4 \log l = 2^{(3 - g^4) \log l} \), which is smaller than \( 1/g^2 \), provided \( g \geq 2 \). The lemma follows.
For the next lemma, we need the following proposition:

**Proposition 4.1:** Let $S$ be a set of cardinality $d$, and $S'$ be a subset of $S$ of cardinality $e$, where $0 < e < \mu d/(16l-2\mu)$. Then the number $\tau$ of $8l$-tuples of elements of $S$ that contain at least $\mu$ elements of $S'$ is less than $2\binom{8l}{\mu} e^{\mu} d^{8l-\mu} < 2(8l)^\mu e^{\mu} d^{8l-\mu}$.

**Proof:** By definition, $\tau = \sum_{i=\mu}^{8l} \binom{8l}{i} e^i d^{8l-i}$. Since $e < \mu d/(16l-2\mu)$, this sum is dominated by the sum of the geometric series with first element $\binom{8l}{\mu} e^{\mu} d^{8l-\mu}$ and ratio 0.5, which implies the proposition.

---

**Lemma E21:** $\Pr [E2 | E1] \geq 1 - 1/2^2$.

**Proof:** By (F1) and Definition 4.3 (see the proof of Lemma E1 above), $E2$ is equivalent to:

For every $x$ in $H_{1-l}(Z_1)$, for every $x[1] \cdots x[8l]$ in $H_{2-1}(Z_2)$, and for every $y$, at most $\mu$ out of the $8l$ vertices $\{(u,x,y,x[j]) : j = 1, \cdots, 8l\}$ are not poor.

Assume that $E1$ holds. To bound the probability that $E2$ does not hold, we must bound the number $N$ of $8l+1$-tuples $(x,x[1], \cdots, x[8l])$ having the following properties:

(i) $x$ is in $H_{1-l}(Z_1)$.

(ii) The concatenation $x[1]x[2] \cdots x[8l]$ is in $H_{2-1}(Z_2)$.

(iii) There is a $y$ such that at least $\mu$ out of the $8l$ message streams $(u,x,y,x[j])$ are not poor ($j = 1, \cdots, 8l$).

The bound on $N$ is derived in three steps, as follows:

**Step 1:** Fix a pair $(x,y)$, such that $x$ is in $H_{1-l}(Z_1)$ and $y$ is an arbitrary message of length $\mu$, and let $e(x,y)$ be the number of distinct messages $x'$ such that the vertex $v = (u,x,y,x')$ is not poor. Since we condition on $E1$, $(u,x)$ is poor. Hence $\beta(u,x,y) \leq \beta(u,x) \leq 2^{\beta-b-c-(h-1)e \log l}$. On the other hand, if $v = (u,x,y,x')$ is not poor then, since $h(v) = h-2$ and $B(v) = B-b-c$, it holds that $\beta(v) \geq 2^{\beta-b-c-(h-2)e \log l}$. By (P2) we have that $\beta(u,x,y) = \sum_{x'} \beta(u,x,y,x')$. Hence

$$e(x,y) \leq \beta(u,x,y)/2^{\beta-b-c-(h-2)e \log l} \leq 2^{c - [(h-1)-(h-2)e \log l]}. $$

**Step 2:** Next we derive an upper bound on the number $\tau(x,y)$ of distinct $8l$-tuples that satisfies (iii) above for a fixed pair $(x,y)$ as defined in Step 1. For this we use Proposition 4.1, where $\tau = \tau(x,y)$, $d = 2^\mu$ is the number of all possible $x$'s of length $\mu$, and $e = e(x,y)$ is given above. By substitution in Proposition 4.1 we get:

$$\tau(x,y) < 2(8l)^\mu 2^{(c-g \log l)} \mu_2 (8l-\mu)^\mu = \tau_{max}. $$

**Step 3:** Finally, by summing $\tau(x,y)$ over all possible pairs $(x,y)$, we get:
\[ N \leq \sum_{(x,y) : x \in H_1^{-1}(Z_1), |y| = \mu} \tau(x,y) \]

\[ < 2^{2\mu \tau_{\max}} \quad \text{(since there are less than } 2^{2\mu} \text{ such pairs } (x,y)), \]

\[ < 2(8l)^{2}(e^{-g \log l})\mu+(8\log -\mu)\mu+2\mu \quad \text{(by substituting for } \tau_{\max}), \]

\[ = 2^{1} + \mu(5 + \log l + g \log l) + c\mu + \mu(8g - 1) \quad \text{(by rearranging terms)}. \]

Using Lemma 3.1(2) again, the probability that \( H_2^{-1}(Z_2) \) contains an 8/-tuple that satisfies (iii) is bounded by \( l^3 N/2^{8c_4 \mu} \), i.e. by

\[ 2^{1 + 3\log l + 5\mu + \mu \log l(1 - 2g^2)} \]

Since, by Lemma 2.5, \( c = b \log 2 \geq \mu - 2g \log l \), we get that \( c \mu(1 - 8g) + \mu^2(8g - 1) \leq 2\mu g \log l(8g - 1) \). Hence, this probability is smaller than

\[ 2^{1 + 3\log l + 5\mu + \mu \log l(1 - 2g^2 - g^4)} \]

which is smaller than \( 1/g^2 \), provided \( g \geq 4 \). This complete the proof of Lemma E21, and of Theorem 4.2.

**Proof of the Main Theorem:** Let \( L \) be accepted by an \( f(n) \)-bounded, Arthur-type verifier \( V' \) within \( g(n) \) rounds.

Then, by Lemma 2.1, \( L \) is also accepted within \( g(n) \) rounds by a verifier \( V \) that satisfies assumptions (A1) and (A2). Assume also that in \( V \)'s protocol, \( \mu(n) > 40 \) and \( g(n) \geq 4 \). First we show that the \( AM \) protocol in section 3 accepts \( L \) with error probability \( \leq 1/3 \). For this, let \( w \) be an input word of length \( n \); as before, \( l, g \) and \( \mu \) denote \( l(n), \) \( g(n) \) and \( \mu(n) \) resp. \( v = s(2g) \) denotes the vertex chosen by \( A \) at the end of the protocol.

If \( w \) is in \( L(V) \), then the root \( \Phi \) of \( D(V,w) \) is rich, and hence, by Theorem 4.1, with probability \( \geq (1 - 2^{-l/8})^{r/2} = 1 - 2^{-l/8} > 2/3 \) the vertex \( v \) is rich, and hence is an accepting message stream.

If \( w \) is not in \( L(V) \), then the root \( \Phi \) is poor, and hence, by Theorem 4.2, with probability \( \geq (1 - 1/8g)(1 - 1/8g)^{r/2} > 2/3 \), the vertex \( v \) is poor, which means that it is a rejecting message stream.

To complete the proof we must show that the time complexity of \( A \) in the \( AM \) protocol given above is \( O(f(n)^c) \) for some constant \( c \). Since the time complexity of an Arthur-type verifier is proportional to the length of the random strings it generates, it is sufficient to prove that the length of the random string used by \( A \) in the \( AM \) protocol is \( O(l'\log l') \), where \( l' \) is the length of the random string used by the verifier \( V' \).

By Lemma 2.1, the length \( l \) of the random string used by \( V \), the verifier obtained by applying Lemma 2.1 to \( V' \), is \( l = O(l' g^{6 \log g(3 \log l')}) < l'^{c_1} \) for some constant \( c_1 \) (\( c_1 < 10 \)).

The length of the random string generated by \( A \) at ROUND \( i \) in the protocol above is proportional to the length of the representation of \( H_1, H_2, Z_1 \) and \( Z_2 \). Since a random linear function from \( \{0,1\}^k \) to \( \{0,1\}^k \) can be encoded in \( kb \) bits, it is easily verified that the length of this representation is \( O(\mu^2/3) \), and hence that the length of
To evaluate the complexity of the verifier in the above protocol we have to evaluate the length of the random algorithm result in an interactive proof protocol having only one round.

Proof: (2): Let \( L \) be in \( \text{IP}[F, g(n)] \), then, by the main theorem of [GS], \( L \) is accepted by an \( F^{O(1)} \)-time bounded verifier \( V \) in \( g(n)+2 \) rounds, where \( V \) is an Arthur-type verifier. By iterating the main theorem on \( V \)'s protocol at most \( \lceil \log g(n) \rceil \) times, the number of rounds is reduced to at most three. Another two iterations of Babai’s algorithm result in an interactive proof protocol having only one round.

To evaluate the complexity of the verifier in the above protocol we have to evaluate the length of the random string it uses. For this, let \( l_0 = l' \) be the length of the random string used by \( V' \), and let \( l_i \) be the length of the random string at the \( i \)-th iteration. Then, by the main theorem, \( l_{i+1} < l_i^{c'} \), for some constant \( c' \) which implies that \( l_i < l_0^{c'} \).

The result now follows by substituting \( i = \lceil \log g \rceil \).

(1) follows by substituting \( i = \text{constant} \) in the bound on \( l_i \) above.

References

[BH] Boppana, R. and Hastad, J., If co-NP has interactive proof systems with constant number of interactions, then the polynomial hierarchy collapses, in preparation.
[F] Feldman, P. The prover in \( \text{IP} \) need not be more powerful than \( \text{PSPACE} \).