# The Noisy Oracle Problem

U. Feige, A. Shamir, M. Tennenholtz
Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel

**Abstract**

We describe a model in which a computationally bounded verifier consults with a computationally unbounded oracle, in the presence of malicious faults on the communication lines. We require a fairness condition which in essence says that some of the oracle's messages arrive uncorrupted. We show that a deterministic polynomial time verifier can test membership in any language in P-space, but cannot test membership in languages not in P-space, even if he is allowed to toss random coins in private. We discuss the zero knowledge aspects of our model, and demonstrate zero knowledge tests of membership for any language in P-space.

## 1 Introduction

The original GMR [7] model of interactive proof systems (IPS) is based on a powerful prover $P$ who wants to convince a polynomial verifier $V$ that a certain assertion Q is true. The model considers two separate scenarios: $P$ can be trustworthy (and then $V$ should accept true assertions), or $P$ can be a cheater (and then $V$ should reject false assertions). By introducing interaction into the proof process, GMR hoped to achieve two goals:

- To increase the class of provable assertions beyond NP.
- To provide zero knowledge proofs.

Recently, Ben-or, Goldwasser, Killian and Wigderson (BGKW [1]) have introduced a general model of multi-prover protocols, and studied a variant which makes it possible to provide perfect zero knowledge proofs, but does not seem to increase the power of the proof system. Motivated and inspired by their work, we consider in this paper a different variant which does extend the class of provable assertions.

In our new Noisy Oracle model, a probabilistic polynomial time verifier $V$ tries to decide whether an input assertion $Q$ is true or false. He is aided by a trusted and infinitely powerful prover $P$, but their communication is disrupted by an adversary

S. Goldwasser (Ed.): Advances in Cryptology - CRYPTO '88, LNCS 403, pp. 284-296, 1990.

$A$ who can block or modify their messages. If $A$ does not exist, $V$ can just believe the one-bit advice provided by $P$. If $A$ can totally block the communication, $P$ cannot help $V$. In our model we consider the interesting case in which $A$ is computationally unbounded but an imperfect jammer in the sense that a non-negligible fraction of the messages exchanged by $P$ and $V$ reach their destination unaltered, in spite of the malicious interference. The problem from $V$'s point of view is that he does not know which messages are authentic, and the only assumption we allow him to make is that occasionally he gets good advice. In particular, if he asks the same question sufficiently many times, at least one of the answers will be correct with overwhelming probability, but he is still left with the problem of deciding which one it is.

An alternative model which has essentially the same properties as the Noisy Oracle model is the following Multi-Oracle model: $V$ interacts with several infinitely powerful oracles $P_1,...,P_n$, and has to decide whether the common input $Q$ is true or false. To make the model non-trivial, we assume that at least one of the oracles is trustworthy and at least one of the oracles is a cheater, but $V$ does not know who is who. The powerful cheaters can break cryptosystems, forge signatures, and test all the possible outcomes of their actions, while the limited verifier has to rely on the (unknown) trustworthy prover to refute incorrect claims.

As a motivating example to the Multi-Oracle model, consider a court of law in which two lawyers (oracles) present their conflicting views of the same case in front of a judge (verifier). Both lawyers have spent much time learning all details of the case. On the other hand, the judge does not have much time to spend. He must use the lawyer's knowledge in order to extract the facts correctly in a short time, and then reach a decision based on the facts he knows. One of the lawyers wants the judge to learn the facts correctly, because then his client is bound to win the case. The other lawyer naturally wants to fool the judge, as otherwise his client will lose the case. The judge does not know which is the truthful lawyer and which is the cheating lawyer. Using our Multi-Oracle model we characterize the facts that the judge can extract efficiently despite the presence of a cheating lawyer.

In the Multi-Oracle presentation of our model, we restrict the cheating oracles by limiting their numbers. In this the model resembles models for distributed computing in the presence of faults (and in particular Byzantine agreement HM [9]). But there is a great difference: we are interested in proving mathematical facts, which the verifier could not prove by himself because of limited computing power. In the Byzantine agreement model, the emphasis is on reaching a consensus about the environment (e. g. the value of a global coin), rather than testing the correctness of mathematical assertions.

In section 2 we present our model. In section 3 we consider verifiers bounded to polynomial time. Our main result is that in the main variant of the Noisy Oracle model, the testable assertions are exactly those which lie in the complexity class P-space. This is in contrast to what is believed to be the case with IPS, where proving even co-NP-complete assertions will lead to the collapse of the polynomial

hierarchy (see GS [8]). In section 4 we define the zero-knowledge aspects of the model, and demonstrate zero-knowledge protocols for all statements in P-space. Section 5 suggests directions for further research, and in particular discusses some recent results on multi oracle protocols with space bounded verifiers.

# 2 The Model

A computationally bounded verifier $V$ interacts with a set of oracles, where the size of the set (denoted by $n$) is fixed, depending on the length of the input. $t$ of the oracles are potential cheaters, where $\frac{n}{2} \leq t < n$. (The case where $t < \frac{n}{2}$ is uninteresting, as $V$ can take a majority vote to determine the correct answer). In the multi-oracle model $V$ receives a set of up to $n$ answers, $t$ of which may be incorrect, but at least one answer must be correct. In the noisy oracle model $V$ receives one answer, where the probability of it being correct is $\frac{n-t}{n}$. Thus in the noisy oracle model, we can only demand that the verifier be convinced with overwhelming probability, as $V$ is not guaranteed to ever receive a correct answer.

In the multi oracle model we may consider different network configurations. If we have a star shaped (or similar) network, with the verifier at the center and the oracles at the leaves, then the oracles become addressable. The verifier can ask each oracle a different question, and can ignore certain oracles once it has determined that they are cheaters. On a broadcast network, the oracles are anonymous, and the verifier cannot associate between messages and their senders. In the configuration of the network we include the ability or inability of one oracle to perform interactions with other oracles, or to perform eavesdropping on lines not his own.

An *interactive test* is a protocol in which the verifier manages to test an assertion, even if the adversary (cheaters) displays his worst case behavior. The assertion is one concerned with the common input, such as: "the following instance $x$ is a legitimate member of language L". In our model, a language L is said to be testable if the following condition holds:

For every input $x$, and for any adversary $A$,
if $x \in L$ then the verifier outputs "$x \in L$" ,
and if $x \notin L$ the verifier outputs "$x \notin L$" .
(In some cases we should add — "with overwhelming probability").

# 3 Polynomial Time Verifiers

First we demonstrate a protocol in which the polynomial time verifier tests an NP-complete assertion. $n$ is assumed to be equal to the length of the input. Note that in this section we do not care about the zero knowledge aspects of the protocols.

NP-protocol:

- V sends the problem instance and asks for a witness.

- P (a truthful prover) sends a correct witness if such exists, and null otherwise.
  C (cheater) sends any message.
- V tests if at least one of the messages was a valid witness to the statement. If so, he concludes that the instance was a yes instance. If not, he concludes it was a no instance.

The correctness of this protocol follows from the fact that $t < n$. V knows that at least one answer is correct, so if he receives no witness, he can conclude that no witness exists. Note that the same protocol is applicable to co-NP statements, if P sends a counter witness whenever it exists.

Next we demonstrate a protocol for a P-space complete problem. In order to simplify the initial demonstration, we start with a simple version of the model: $n = 2, t = 1$, addressable model. That is: we have a star configuration with a verifier interacting separately with two oracles. One of the oracles may be a cheater, but the verifier does not know which one. We shall demonstrate the protocol on any P-space complete game (as can be found in GJ [5]). In such a game we ask whether the first player to move ("white"), or the other player ("black") has a winning strategy. The players alternate in the moves they make, and the game is guaranteed to end after a polynomial number of moves.

Simple P-space protocol:

1. $V$ sends the instance (initial position) of the game to the two oracles, and asks them who wins. If their answers agree $V$ stops.
2. $V$ sends the current position to the oracle who claims that white wins (denote him by $W$) and asks for a winning move for white.
3. If $W$ does not reply with a move, $V$ concludes that he was cheating, and so black is the one who wins. So $W$ replies with a move $w$ for white.
4. $V$ computes the new position generated by applying $w$ to the previous position, sends it to $B$ (the oracle who claims that black wins), and asks for a winning move for black.
5. If $B$ does not reply with a move, $V$ concludes that white wins. So $B$ replies with $b$. $V$ applies $b$ to the current position and returns to step 2.

After not more than a polynomial number of moves, either one of the two oracles is caught not following the protocol, or the game reaches its natural end. In either case $V$ concludes correctly who wins in the initial position, as he knows that the trustful oracle must have chosen the wining side, and must have made the optimal moves.

The above protocol may easily be generalized to a star network with one prover and $n - 1$ cheaters. The verifier asks each oracle for the outcome of the game

instance. If all agree, there is no problem. If V receives conflicting answers, he chooses two oracles who gave conflicting answers, and lets them play one against the other. The loser of the game is marked as a cheating oracle, and is discarded from the network. Now V remains with a network of $n-1$ oracles, at least one of which is truthful. By induction on the size of the network, we see that V determines the correct outcome of the game.

The above protocols cannot be directly applied to the broadcast model or to the noisy oracle model, where V receives unordered sets of answers. The problem is that V does not know which answer corresponds to which oracle. For each position, half the cheaters can actually claim the correct answer, but then try to discredit it with bad moves. If $V$ tries to discard some of the moves, he might discard the alternative suggested by the real prover. If V does not discard moves, this results in an exponential number of variations of the ongoing game that V has to keep track of. Nevertheless, a good protocol can be constructed.

**THEOREM 1:** Let $n$ be polynomial in the length of the input, with $t < n$. Any statement in P-space can be tested even in the broadcast model (where V receives unordered sets of answers).

We shall introduce terminology and a few short lemmas simplifying the proof of the above theorem.

Let $G$ be a two player ($W$ and $B$) game, in which the player to move looses if he has no legal move. The *outcome* of a position of $G$ is $W$ ($B$ respectively) if $W$ ($B$) has a winning strategy. An *n-hyper-position* $HP$ of $G$ is a $k_1 < n$ by $k_2 < n$ matrix in which each entry is a position in $G$, and in all positions the same player is to move. A *hyper-move* is made by choosing a column in $HP$, transposing it to a row, and for each entry in the row — performing a legal move of $G$. An *HP-transposition* is constructed by considering $k_3 < n$ $H$-moves (from now on, $H$ stands for *hyper*) as rows of a new $H$-position $HP^t$. Note that if $W$ is about to move in the entries of $HP$ then $B$ is about to move in the entries of $HP^t$, and vice versa. A *n-hyper-game* is played by *n hyper-players*. For each $H$-position each $H$-player may choose to make one $H$-move, resulting in an $HP$-transposition. The $H$-game ends with one of the two *outcomes* $W$ or $B$ in one of the two ways:

1. Agreement: All $n$ $H$-players choose to make an $H$-move in the same $H$-position. The outcome of the $H$-game is $W$ if $W$ has the move in the entries of $HP$, and $B$ otherwise.

2. Resignation: No player makes an $H$-move. The outcome of the $H$-game is $W$ if $B$ has the move in the entries of $HP$, and $B$ otherwise.

Note that a 2-$H$-game can be viewed as just an ordinary two player game.

$HP$ is *row-dominated* (*column-dominated* respectively) if it has a row (column) in which all entries are positions of $G$ in which the previous (next) player wins. Note that $HP$ cannot be both row-dominated and column-dominated simultaneously.

**Lemma:** If $HP$ is column-dominated then there exists an $H$-move such that $HP^t$ is row-dominated.

**Proof:** Pick the dominating column in $HP$. In all its entries the next player wins. For each entry perform the winning moves in $G$ and transpose the column. This is the desired $H$-move. QED.

An $H$-move as described above shall be called *optimal.*

**Lemma:** If $HP$ contains a dominating row then $HP^t$ contains a dominating column.

**Proof:** Assume row $j$ is dominating $HP$. Column $j$ will necessarily dominate $HP^t$. QED.

An *optimal* $H$-player makes an optimal $H$-move whenever one exists, and no $H$-move otherwise.

**Lemma:** If $HP$ is dominated, and if at least one of the $H$-players is optimal, then if the $H$-game ends, it ends with a outcome equal to the outcome of any of the positions in the entries of the row/column dominating $HP$.

**Proof:** The existence of the optimal $H$-player ensures that row/column domination will alternate in $HP$-transpositions, that resignation will not occur in column-dominated $H$-positions, and that agreement will not occur in row-dominated $H$-positions. So the $H$-game may not end with a outcome different than stated in the lemma. QED.

Now we return to the proof of theorem 1.

**Proof (theorem 1):** It is sufficient to consider any game $G$ complete in P-space, because reductions into P-space complete problems can be done in polynomial time. $V$ can monitor an $n$-$H$-game based on $G$, where the oracles act as $H$-players:

1. The initial $HP$ consists of only one entry: the initial position of $G$. This implies that the initial $HP$ is dominated.

2. $V$ sends the current $HP$ as a challenge to all oracles.

3. $V$ receives a set of $H$-moves, at most one from each oracle.

4. $V$ constructs $HP^t$ from the legal $H$-moves, makes it the current $HP$ and returns to step 2 (unless the $H$-game ended).

The truthful prover plays the role of the optimal $H$-player. By the above lemmas the outcome of the $H$-game is identical to the outcome of $G$. The size of each $n$-$HP$ and the number of $HP$-transpositions are bounded by a polynomial in the length of the input. So $V$ can perform the protocol in polynomial time. QED.

**Corollary:** Any assertion in P-space can be tested with overwhelming probability in a system where the probability of a correct response is non-negligible. (The Noisy Oracle model).

**Proof:** Let $n$ be the length of the input, let $p$ be a polynomial, and let $\frac{1}{p(n)}$ be the probability of a correct response. Then by repeating each question $n \cdot p(n)$ times V receives a set of answers which with overwhelming probability includes at least one correct answer. Because the whole protocol is limited to a polynomial

number of steps, the probability that each of the steps contains at least one correct answer remains overwhelming. Thus the verifier can transform his original noisy-oracle setting to a new multi-oracle setting, but with modified parameters: $n \cdot p(n)$ oracles and one true prover. The proof follows, as V monitors an $n \cdot p(n)$-$H$-game. QED.

We have seen that $V$ can test any P-space assertion even in the most difficult scenario — the noisy oracle model with low probability of correct answers, and no coin tossing allowed. We shall now show that even in a more favorable scenario, the multi-oracle broadcast model with only one cheating oracle and with a secret source of randomness, $V$ cannot test assertions not in P-space.

**THEOREM 2:** In a broadcast network, if in every round the adversary sees the message of the prover before deciding on his own message, a probabilistic polynomial time verifier cannot test assertions not in P-space.

**Proof (sketch):** Suppose a language L is testable in the broadcast model. We demonstrate a P-space algorithm for testing whether an instance $x$ of length $n$ belongs to L. V is assumed to truthfully follow a certain algorithm, and to output either "$x \in L$" or "$x \notin L$" after a number of steps bounded by some polynomial $p(n)$. This implies bounds of $p(n)$ on the number of coin tosses V makes, on the number of messages he expects to receive and on the maximal message length. We construct a game tree of depth $3p(n)$, composed of three types of nodes: $O_1$, $O_2$, $V$. The edges leading out of nodes of $O_1$ ($O_2$, $V$ respectively) correspond to *all* possible messages (whether they make sense or not) of oracles trying to prove $x \in L$ (oracles trying to prove $x \notin L$, verifier respectively). A string $S$ is said to *agree* with the path from the root to a leaf, if V's algorithm, given $S$ on his random tape and assuming $O_1$ and $O_2$ send the messages implied by their edges along the path, would indeed cause V to send the messages implied by his edges. The *value* $R$ of a leaf is the number of length-$p(n)$ random strings which agree with the corresponding path, and which cause V to output "$x \in L$". The value of an inner node of the tree is defined recursively: An $O_1$-node maximizes over its sons (nodes of $O_2$). An $O_2$-node minimizes over its sons (nodes of $V$). A $V$-node sums over its sons (nodes of $O_1$). We shall prove that the value of the root is greater than $2^{p(n)-1}$ iff $x \in L$.

Assume $x \in L$. We want to show that this implies $R > 2^{p(n)-1}$. Consider an adversary who takes the role of $O_2$ in the game tree constructed, and chooses his messages optimally so as to minimize $R$. Because we assume that L is testable, there exists a strategy for the real prover which causes V to output "$x \in L$" with overwhelming probability. In particular, the *optimal* strategy of taking the role of $O_1$ and choosing messages so as to maximize $R$ must work. Note that there exists no better strategy, because we assumed the adversary may choose his messages in each round after viewing the prover's messages. This optimal strategy convinces V with probability $\frac{R}{2^{p(n)}}$, and so $R > 2^{p(n)-1}$.

Assume $x \notin L$. We want to show that this implies $R < 2^{p(n)-1}$. Consider an adversary who takes the role of $O_1$ in the game tree constructed, and chooses his messages optimally so as to maximize $R$. Because we assume that L is testable,

there exists a strategy for the real prover which causes V to output "$x \notin L$" with overwhelming probability. In particular, the *better than optimal* strategy of taking the role of $O_2$ and choosing messages so as to minimize $R$ must work. This strategy is better than optimal because the prover chooses his messages *after* viewing the adversary's messages. This better than optimal strategy convinces V with probability $\frac{2^{p(n)}-R}{2^{p(n)}}$, and so $R < 2^{p(n)-1}$.

Finally, the value of the root can be computed by a Depth First Search traversal algorithm in polynomial space. This is a consequence of the simplicity we maintain in the construction of the game tree. We do not try to consider only "sensible" messages of the oracles, as a P-space algorithm cannot judge what messages are considered sensible by computationally unbounded and possibly cheating oracles. Furthermore, the edges leading out of $V$ nodes are not restricted only to those corresponding to messages $V$ really sends, as this will imply extensive bookkeeping. The only information our algorithm needs to save is both a message number and an $R$-counter for each node along the current path considered in the tree, and this requires $O(p^2(n))$ space. QED.

The assumption that the adversary sees the prover's message before deciding on his own was crucial in the above proof. In particular, if the situation is known to be reversed, the verifier can correctly test any assertion, as the prover just sends a message whose exclusive-or with the adversary's messages gives the correct answer. In addressable models, we assume no oracle knows which messages were sent by other oracles, and so again the proof of theorem 2 does not hold. A major open question is whether addressable models allow probabilistic polynomial time verifiers to test membership in languages not known to be in P-space.

# 4 Knowledge complexity

GMR introduced the notion of knowledge complexity of a language. We extend their definitions so as to apply to our model with a probabilistic polynomial time verifier.

One may view the interaction between the verifier and the noisy oracle as an interaction between two parties, where one of the parties wants to test the value of a predicate through the other party. The adversary (noise) models possible trouble. Allowing for this possibility, the oracle agrees to help the verifier filter out incorrect answers, by demonstrating their incorrectness. In *zero-knowledge* protocols the oracle does not agree to allow a cheating verifier's claim of alleged imperfect communication to cause the oracle to reveal additional information. For example, the oracle may not agree to give the verifier specific witnesses which demonstrate the correctness of the claims made by the oracle, or which demonstrate the incorrectness of claims made by the adversary. We model the above setting by stating that there is no adversary along the communication lines. A cheating verifier interacts with a cautious oracle, and *claims* that he receives conflicting messages. The corresponding definition for the *multi oracle* model allows polynomial time cheating

verifier $V^*$ to manage the behavior of the cheating oracles as best suits him. A protocol is in zero knowledge if $V^*$ can not increase his knowledge beyond the one bit he receives anyway (the truth value). Note that if we do not limit the cheaters to polynomial time, nothing prevents them from revealing information to the verifier.

As an example motivating the zero knowledge concept, we return to our two lawyers making their claims in front of the judge. The defending lawyer claims that the defendant has a perfect alibi for the night of the murder, while the prosecution claims the opposite. The defense is in a delicate position because the alibi is rather embarassing: the defendant has spent the night of the murder with the judge's wife! How can the defense convince the judge that the defendant is not guilty (of murder), without revealing the actual alibi? The answer is simple — do it in zero knowledge.

We assume the reader is familiar with the definitions of zero-knowledge. We just sketch our definition.

**Definition:** In the noisy oracle model, a protocol is said to be *zero knowledge* if the following condition holds: There exists a probabilistic polynomial time algorithm M (which may run $V^*$ as a subroutine), where M is given the truth value of the assertion involved, such that for any non-uniform polynomial time algorithm $V^*$ which manages the behavior of the verifier and of the cheaters, M's output is indistinguishable from $V^*$'s view of the communication with the real prover.

We shall consider the model in which $n$ is equal to the length of the input, $t = n - 1$ and the oracles are not addressable. The adversary $A$ manages the behavior of all the cheaters. In proving that a protocol is in zero-knowledge, $V^*$ manages the behavior of $V$ and of $A$, but is limited to polynomial time. $P$ denotes the truthful prover.

The trivial protocol for NP languages, (sending a witness whenever one exists), reveals no information in the case where $x \notin L$ (co-NP statements), as $P$ sends no message, and M has nothing to simulate. But the protocol as a whole is not zero knowledge, as a polynomial time simulator M is not guaranteed to produce witnesses for NP statements in cases where $x \in L$. So in order to construct zero knowledge protocols in our model, we shall apply techniques used in IPS. We shall demonstrate that care should be taken when doing so.

Following the footsteps of GMW [6], we assume that safe encryption functions exist. We use a protocol proposed by Manuel Blum, and sketch its basic structure in order to make this paper self contained:

1. $V$ asks for an encrypted random permutation on the graph G.
2. $P$ sends the encryption, and $A$ adds whatever messages he wants.
3. For each message he receives, $V$ requests either it's full decryption and the permutation used, or partial decryption revealing a Hamiltonian cycle.
4. $P$ sends the requested information, and $A$ adds whatever messages he wants.

5. $V$ checks that at least in one of the received messages the protocol was followed correctly.

In order to diminish the chance of cheating, this basic structure may be iterated $n$ times (serial version). Alternatively $V$ may ask at step 1 for $n$ encrypted random permutations of G (parallel version). We shall demonstrate that both approaches are not recommended.

1. Serial version:

   Here the adversary has a good chance of cheating in a broadcast network. Suppose G does not have a Hamiltonian cycle. The adversary may convince $V$ to the contrary: when asked for an encrypted permutation of the graph, the adversary sends $\frac{n}{4}$ such encryptions, $\frac{n}{4}$ encryptions of Hamiltonian cycles and $\frac{n}{2}$ null answers. (We assume the prover returns a null answer as well). When $V$ asks for decryptions, $A$ has 0.5 chance to succeed for each message. Because there are $O(n)$ messages, he has exponentially high probability of cheating at least once. Even if the protocol is iterated a polynomial number of times, the adversary has high probability of succeeding in all rounds. In the broadcast model, $V$ does not know if there existed one oracle which succeeded in all iterations, in which case $V$ should accept, or whether each oracle failed at least once, in which case $V$ should reject.

2. Parallel version:

   Here the adversary has a negligible chance of cheating, but now the simulator (M) has an impossible task: he himself needs exponential time in order to simulate a run (assuming he does not know any Hamiltonian cycle).

Since both the serial version and the parallel version are incorrect, we shall use a combined version. $V$ asks for $\log n$ encryptions of different random permutations of G, all in parallel. This basic protocol is iterated $n$ times. In each iteration, the adversary has $\frac{1}{n}$ probability of cheating with any single message, or constant probability of cheating with at least one message. So $A$ has negligible chance of cheating at least once in every iteration of the whole protocol. On the other hand, M can simulate each iteration in $O(n)$ trials, and the whole protocol in $O(n^2)$ steps.

The zero knowledge protocol presented above suits Co-NP statements as well. If each iteration contains at least one good message, the verifier concludes that there exists a witness. Otherwise, he concludes that no witness exists. This is true of protocols in our model in general: they either demonstrate that an assertion is correct, or that it is not correct. In no case does the verifier remain in state of doubt.

Finally, we consider the P-space complete game discussed in Theorem 1. The $H$-game constructed in the proof of this theorem can be played even if the entries

to the $H$-positions are encrypted. The powerful provers play by breaking the corresponding cryptosystems, choosing an $H$-move and encrypting it. Polynomial time V has the task of blindly constructing encrypted $H$-positions from the encrypted $H$-moves he receives. When the encrypted $H$-game ends (it must end in a polynomial number of rounds), the verifier tests in zero knowledge a statement in NP — that the decryption of the whole history of the $H$-game would give the desired outcome.

**Theorem 3:** Under the assumption that encryption functions exist, any statement in P-space can be tested in zero-knowledge (in all models).

A proof can be constructed from the discussion above.

The above theorem can be extended to any protocol in which all V does is shift messages back and forth. For all such protocols we can easily construct zero-knowledge versions. On the other hand, in protocols in which V takes actions which depend upon the messages he receives and upon his coin tosses, the above technique does not give a zero-knowledge protocol, since V does not know which action to take if the messages he receives are encrypted.

# 5 Further Research

In this paper we presented a new model, which has many variants: Addressable versus broadcast communication, time bounded versus space bounded verifiers, etc. There are many open questions, of which we want to point out one in particular: Is there any language not known to be in P-space which is testable by a probabilistic polynomial time verifier? As can be derived from theorem 2, the best model for looking for such a language is the addressable model with private coins.

It may be interesting to mix our model with the multi-oracle model of BGKW [1]. We may assume that at least two oracles are trustworthy, that oracles cannot communicate among themselves, and that the good oracles share a read only common random string. The results of BGKW transform to this model, giving perfect zero knowledge proofs for every language in NP. Furthermore, this automatically gives perfect zero knowledge proofs of all languages in co-NP, as the model is closed under complement. Can these results be pushed further up in the polynomial hierarchy?

A different line of research is to study the structure of our models. In what models are public coins as powerful as private coins? In what ways does the number of rounds of a protocol correspond to the complexity level of the tested language in the polynomial hierarchy?

Interactive protocols with space bounded verifiers received much attention recently (see Condon and Ladner [2], Dwork and Stockmeyer [3], Kilian [10] and others). The fact that if some information is hidden from the players, the outcome of very complex games can be tested in very little space, was demonstrated by Reif [12] and by Peterson and Reif [11]. In a paper now in preperation (FS [4]), the no-

tion of probabilistic space bounded verifiers in addressable multi oracle systems is defined. The following surprising results are proved:

**Theorem 4 (FS [4]):** In our multi oracle model, a log-space verifier can test any P-space assertion in polynomial time.

**Theorem 5 (FS [4]):** In our multi oracle model, the set of elementary recursive languages (that is, languages recognized by a Turing machine in time $2^{\cdots^{2^n}}$, with a fixed number of exponentiations) is strictly contained in the set of languages testable by log-space verifiers.

**Theorem 6 (FS [4]):** In the multi prover model of BGKW [1], the set of elementary recursive languages is strictly contained in the set of languages testable by log-space verifiers. Furthermore, if protocols are not requested to end with probability 1, constant space verifiers can recognize any partial recursive language.

# References


[1] M. Ben-or, S. Goldwasser, J. Killian, A. Wigderson, *Multi Prover Interactive Proofs: How to Remove Intractability* Proc. of 20th STOC 1988, pp. 113-131.

[2] A. Condon, R. Ladner, *Probabilistic Game Automata* Proc. Structure in Complexity, 1986, pp. 144-162.

[3] C. Dwork, L. Stockmeyer, *Zero-Knowledge with Finite State Verifiers* these proceedings.

[4] U. Feige, A. Shamir, *Multi Oracle Interactive Protocols with Space Bounded Verifiers* in preperation.

[5] M. Garey, D. Johnson, *Computers and intractability, A guide to theory of NP-Completeness* Freeman 1979.

[6] O. Goldreich, S. Micali, A. Wigderson, *Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design* Proc. 27th FOCS, 1986, pp. 174-187.

[7] S. Goldwasser, S. Micali, C. Rackoff, *The Knowledge Complexity of Interactive Proofs* Proc. of 17th STOC, 1985, pp. 291-304.

[8] S.Goldwasser, M.Sipser , *Arthur Merlin Games versus Interactive proof systems* 18th STOC, 1986, pp. 59-68.

[9] J. Y. Halpern, Y. Moses, *Knowledge and common knowledge in a distributed environment* Proceedings of the Fourth PODC, 1985, pp. 224-236.

[10] J. Kilian, *Zero-Knowledge with Log-Space Verifiers* to be presented at FOCS 1988.

[11] G. L. Peterson and J. H. Reif, *Multiple-Person Alternation* Proceedings of 20th FOCS, 1979, pp. 348-363.

[12] J. H. Reif, *The complexity of two-player games of incomplete information* JCSS 29, 1984, pp 274-301.