Fast Verification of Any Remote Procedure Call: Short Witness-Indistinguishable One-Round Proofs for NP*

W. Aiello[†] S. Bhatt[‡] R. Ostrovsky[§] S. Rajagopalan[¶]

May 9, 2000

Abstract: Under a computational assumption, and assuming that both Prover and Verifier are computationally bounded, we show a one-round (i.e. Verifier speaks and then Prover answers) witness-indistinguishable interactive proof for NP with poly-logarithmic communication complexity.

A major application of our main result, resolving an open problem posed by Micali, is that we show how to check in an efficient manner and without any additional interaction, the correctness of the output of *any* remote procedure call (i.e. any polynomial-time computation) based on the computational assumption. That is, one can ask any polynomial-time *untrusted* machine to execute an arbitrary code of our choice and give back the result. We will ask the untrusted machine to work just a little bit longer than to just execute the code, but we show how the untrusted machine can then produce not only the answer but also a very short and easily verifiable *certificate*. The untrusted machine can send us back both the alleged output of the remote procedure call and the certificate. The guarantee is that we can very easily check the certificate and the output (without re-doing the computation!) and if the output of the remote procedure call is correct, then the certificate will always be accepted. However, no polynomial-time cheating machine can fool us in accepting a certificate for *any incorrect output* with more than negligible error probability.

* *This paper will appear in the Proceedings of the 27th International Colloquium on Automata, Languages and Programming (ICALP 2000), to be held in Geneva on July 9–15, 2000.*

[†] AT&T Labs-Research. aiello@research.att.com

[‡] Akamai Technologies. bhatt@akamai.com

[§] Telcordia Technologies. rafail@research.telcordia.com

[¶] Telcordia Technologies. sraj@research.telcordia.com

1 Introduction

Under a computational assumption, and assuming that both Prover and Verifier are computationally bounded, we show a one-round (i.e. Verifier speaks and then Prover answers) witness-indistinguishable interactive proof for NP with poly-logarithmic communication complexity. More formally, our one-round interactive proof (argument) has the following properties:

- *Perfect Completeness*: On a common input $x \in L$ for any L in NP if Prover is given a witness as a private input, it convinces Verifier (i.e. Verifier accepts) with probability one.
- *Computational Soundness*: If $x \notin L$, no poly-time cheating Prover can make the Verifier accept except with negligible probability.
- *Protocol is Short*: the communication complexity (of both messages) is poly-logarithmic in the length of the proof.
- *Witness-indistinguishability*: The interactive proof is *witness indistinguishable*: if there is more than one witness for $x \in L$ then for any two witnesses no poly-time cheating Verifier can distinguish with non-negligible success probability which witness was given to the Prover as a private input.

Our result improves upon best previous results on short computationally-sound proofs (arguments) of Micali [20] and of Kilian [15, 16] which required either three messages of interaction or the assumption of the existence of a random oracle. Our result also improves the communication-complexity of one-round witness-indistinguishable arguments ("Zaps") of Dwork and Naor [7] which required polynomial communication complexity in the length of the proof. One important difference between our protocol and the one in [7] is that our protocol is private-coin protocol[1] where as the Dwork and Naor protocol is public-coin.

A major corollary of our result is that we show how to check in an efficient manner and without any additional interaction, the correctness of the output of *any* polynomial time remote procedure call. That is, if Alice requests Bob to execute some remote code and give back the result, Alice can provide Bob (along with the code) a very short public-key and receive back from Bob not only the result of the execution

[1] Private-coin protocol is the one where Verifier tosses coins but keeps them *secret* from the prover, whereas public-coin protocols are the ones where Verifier can publish its coin-flips.

but also a very short and easily[2] verifiable *certificate* that the output was computed correctly. We stress that our scheme for verifiable remote procedure call is round-optimal: there is only one message from Alice to Bob (i.e. the procedure call and public key) and only one message back from Bob to Alice (with the output of the procedure call and the certificate of correctness). Our scheme guarantees that if Bob follows the protocol then Alice always accepts the output, but if the claimed output is incorrect Bob can not construct a certificate which Alice will accept, except with negligible probability. The public key and the certificate are only of poly-logarithmic size and can be constructed by Bob in polynomial time and verified by Alice in poly-logarithmic time. This resolves the major open problem posed Micali in the theory of short computationally-sound proofs [20], namely, that they exist with only one round of interaction under a complexity assumption. This also improves the works of [20, 15, 16, 24] and also of Ergün, Kumar and Rubinfeld [12] who consider this problem for a restricted case of languages in NP with additional algebraic properties.

It is interesting to point out that the proof of our main result also shows that the heuristic approach of Biehl, Meyer and Wetzel [4] of combining Probabilistically Checkable Proofs of Arora et al [2] with Single-Database Private Information Retrieval of Kushilevitz and Ostrovsky [18] and of Cachin, Micali and Stadler [6] is valid, though with additional steps both in the protocol and in its proof of security. More specifically, our main result uses a novel combinatorial technique to prove computational soundness of our interactive proof and we believe this proof technique is of independent interest.

Our result also has an interesting connection to the PCP theorem [2]. Recall that, in the PCP setting, the Prover must send to the Verifier (or alternatively "write down") an entire proof. Subsequently, the Verifier needs to read only a few bits of this proof at random in order to check its validity. Notice that the communication complexity in this case is large – polynomial in the running time bound – since the entire PCP proof has to be sent. Furthermore, even though the Verifier is interested in reading only a few bits, she must *count* all the received proof bits in order to decide which bits to read. In our setting, the Verifier asks her question first and the dishonest Prover may make his answers dependent on the Verifier's question. Nevertheless, we show that even if the dishonest but computationally bounded Prover answers adaptively to the Verifier's message, he can not cheat the Verifier with non-negligible probability. Moreover, the length of this answer is dramatically shorter than the PCP proof.

[2]By very short and easily verifiable, we mean that the size of the public-key/certificate as well as generation/verification time is a product of two quantities: a fixed polynomial in the security parameter and a factor polylogarithmic in the running time of the actual remote procedure call.

Our main result has several implications including a different method of reducing the error in one-round two-prover interactive proofs with small communication complexity. Our proof also has implications to generalized PIR protocols where the user asks for many bits and wishes to know if there exists *any* database which is consistent with all the answers. We discuss these and other implications after the proof of our main result.

2 Preliminaries

First, we recall some of the terminology and tools that we will use in our construction. For definitions and further discussion see the references provided. We use standard notions of expected polynomial-time computations, polynomial indistinguishability, negligible fractions and hybrid arguments (see, for example, Yao [25]) and the notion of witness indistinguishability [11].

We will use Single Database Private Information Retrieval (PIR) protocols first proposed by Kushilevitz and Ostrovsky [18] and further improved (to essentially optimal communication complexity) by Cachin, Micali and Stadler [6]. (In the setting of multiple non-communicating copies of the database, PIR was originally defined by Chor et al [5].) Recall that Single Database PIR are private-coin protocols between a user and a database where the database has an n-bit string D and a user has an index $1 \leq i \leq n$. Assuming some computational (hardness) assumption [18, 6, 9], the user forms a PIR question to the database based on index i and a security parameter[3] k which database answers. The answer can be "decrypted" by the user to correctly learn the ith bit of D. The key property is that i remains hidden from the honest but curious database. That is, for any two indices i and j, the PIR encodings of i and j are computationally indistinguishable (can not be distinguished in polynomial time except with negligible probability, denoted as ϵ_{PIR}.) $PIR(cc, u, db)$ denotes any one-round single database PIR protocol with communication complexity $cc(n, k)$, the running time $u(n, k)$ of the user (to both generate the question and decode the answer from the database), and the running time $db(n, k)$ of the database ($db(n, k)$ must be polynomial in n and k.)

We remark that by the standard hybrid argument, polynomially long vectors of PIR queries are also indistinguishable with slight degradation in error probability.

[3] By $k = k(\cdot)$, we denote a sufficiently large security parameter, i.e. $k(\cdot)$ is a function of n such that for sufficiently large n, the error probability is negligible. In PIR implementations ([18, 6]) depending on particular computational assumptions, $k(n)$ is typically assumed to be either $\log^{O(1)}(n)$ or n^{ϵ} for $\epsilon > 0$.

We also point out that both [18] and [6] are one-round protocols, unlike [19]. We will also use Secure-PIR (SPIR) protocols in a single database setting (see Gertner et al [13] and Kushilevitz and Ostrovsky [18]) and one-round SPIR protocol (based on any 1-round PIR protocol and one-round OT) of Naor and Pinkas [21]. The latter result shows how to convert any PIR protocol to a SPIR protocol using one call to PIR and an additive polylogarithmic overhead in communication.

We will use Probabilistically Checkable Proofs (PCP) of Arora et al [2] and 3-query PCP proofs of Håstad [14] as well as holographic PCP proofs of Babai et al [3] and Polischuk and Spielman [22]. Recall that an NP language L is in (ϵ_{PCP}, k) PCP if there is a probabilistic polynomial time Verifier V that reads at most k bits of this proof such that the following two conditions hold: for every string $x \in L$ there is a polynomially long "proof" (which, given a witness, can be constructed in polynomial time) that V will always accept. For every string $x' \notin L$ there is no "proof" string (even among those constructible in exponential time) that V will accept with probability greater than ϵ_{pcp} (probability is over the choice of V's coin flips only). Additionally, we will use the Zero-Knowledge PCP of Kilian et al [17] which guarantees that as long as the verifier does not read more than polylogarithmically many bits in the proof, it remains ZK.

3 Problem statements and Our results

We state our problem in two settings, the setting of proving an NP statement and the setting of efficient verification of remote procedure calls.

In the setting of proving an NP statement, both prover and verifier are probabilistic poly-time machines. They both receive as common input (i.e. on their work-tapes) an input x and a security parameter k. Since x is given on the work-tape of the verifier, we do not charge the length of x as part of communication complexity, nor do we charge the time required to read x, as it is given before the protocol begins. The prover receives (if $x \in L$) an additional private input of witness w that $x \in L$. Wlog, we assume that $|w| \geq |x|$ (the interesting case, of course, is when $|w| >> |x|$).

THEOREM 1 Assuming the existence of a one-round $PIR(cc, db, u)$ scheme there exists a (P,V) one-round proof with perfect completeness and computational soundness such that the communication complexity is $O(\log^{O(1)}|w| \cdot cc(|w|^{O(1)}, k))$, the prover's running time is $O(|w|^{O(1)} + db(|w|^{O(1)}, k))$, and the verifier's running time is $O(\log^{O(1)}|w| \cdot u(|w|^{O(1)}, k))$.

If we use the Cachin et al implementation of PIR which is based on the ϕ-hiding assumption [6], and $k = \log^{O(1)}|x|$ for sufficiently long x, then we can achieve *both*

poly-logarithmic communication complexity and poly-logarithmic verifier computation time:

COROLLARY 1 Assuming that the ϕ-hiding assumption holds, for any $\epsilon > 0$ there exists a (P,V) one-round proof with perfect completeness and computational soundness such that for x sufficiently long, the total communication complexity is $O(\log^{O(1)} |w|)$, the prover's running time is $|w|^{O(1)}$ and the verifier's running time is $O(\log^{O(1)} |w|)$.

If we use the Kushilevitz and Ostrovsky implementation of PIR based on the quadratic residuosity assumption [18], and $k = |x|^{\epsilon}$ for $\epsilon > 0$ and x sufficiently long, then we get the following:

COROLLARY 2 Assuming that the quadratic residuosity assumption holds, for any $\epsilon > 0$ there exists a (P,V) one-round with perfect completeness and computational soundness such that, for x sufficiently long, the total communication complexity is $|w|^{\epsilon}$, the prover's running time is $|w|^{O(1)}$ and verifier's running time is $|w|^{\epsilon}$.

We remark that using *holographic proofs* of [3, 22], we can make the running times of the prover in the above theorem and in both corollaries almost linear.

We also show how we can add the witness-indistinguishability property to our low-communication complexity protocol by combining Zero-Knowledge PCP [17]) with one-round SPIR protocols [13, 18, 21]. More formally we have the following theorem.

THEOREM 2 Assuming the existence of a one-round $PIR(cc, db, u)$ scheme there exists a (P,V) one-round witness-indistinguishable proof with perfect completeness and computational soundness such that the communication complexity is $O(\log^{O(1)} |w| \cdot cc(|w|^{O(1)}, k))$, the prover's running time is $O(|w|^{O(1)} + db(|w|^{O(1)}, k))$, and the verifier's running time is $O(\log^{O(1)} |w| \cdot u(|w|^{O(1)}, k))$.

Our second setting is that of verification of any remote procedure call. Here, Alice has a remote procedure call Π (P is some arbitrary program with an arbitrary input) and a polynomial bound t on the running time. Alice generates a public key-private key pair, keeps the private key to herself, and sends the public-key, Π and t to Bob. Bob executes $y \leftarrow \Pi$ for at most t steps (y is defined as $\perp$ if the program Π did not terminate in t steps). Using the public-key, Bob also computes a string c that certifies that y is correct and sends c and y back to Alice. Alice using her private-key decides to either accept or reject y. We call this a correct verification scheme if two conditions hold: if Bob follows the protocol then Alice always accepts and no

polynomial time-bounded cheating Bob' can find a certificate for any incorrect y' on which Alice will accept, except with negligible probability. Clearly, Alice must take the time to send Π to Bob and to read y. Wlog, we assume $|y| = |\Pi| < t$. However, if $|y| << t$, the question of fast verification of y becomes important. We show that Alice can do this in an efficient manner:

THEOREM 3 Assuming the existence of a one-round $PIR(cc, db, u)$ scheme, there exists a correct verification scheme for any remote procedure call (Π, t) such that the running time for Bob is $O(t^{O(1)} + db(t^{O(1)}, k))$, the sizes of the public key, private key, and certificate are all $O(\log^{O(1)} t \cdot cc(t^{O(1)}, k))$, and the running time of Alice is $O(|y| + \log^{O(1)} t \cdot u(t^{O(1)}, k))$.

We stress that Theorem 3 holds for *any* one-round implementation of PIR protocol. For example, if we use Cachin et al implementation of PIR which is based on the ϕ-hiding assumption [6], and $k = \log^{O(1)} t$ for sufficiently large t, then we achieve poly-logarithmic bounds for the verification of y. We remark again that using the *holographic proofs* of [3, 22], we can make Bob's running time in the above theorem almost linear. Combining, we achieve:

COROLLARY 3 Assuming that the ϕ-hiding assumption holds, for any $\epsilon > 0$, there exists a correct verification scheme for any remote procedure call (Π, t) such that for t sufficiently large, the running time for Bob is $O(t^{1+\epsilon})$, the sizes of the public key, private key, and certificate are all $O(\log^{O(1)} t)$, and the running time for Alice is $O(|y| + \log^{O(1)} t)$.

We can obtain another corollary of Theorem 3 by combining it with single-database PIR implementation based on quadratic residuosity assumption [18]:

COROLLARY 4 Assuming that the quadratic residuosity assumption holds, for any $\epsilon > 0$, there exists a correct verification scheme for any remote procedure call (Π, t) such that, for t sufficiently large, the running time of Bob is $O(t^{1+\epsilon})$, the sizes of the public key, private key, and certificate are all $O(t^{\epsilon})$, and the running time for Alice is $O(|y| + t^{\epsilon})$.

We remark that the above theorems also hold even if Π is a probabilistic computation. Moreover, the coin-flips of Bob can be made *witness-indistinguishable*, similar to Theorem 2.

4 Constructions

First, we describe our construction for Theorem 1. Recall that the Prover is given x and a witness w. Let $|PCP(x,w)| = N$ be Håstad's 3-query PCP proof that $x \in L$ [14]. We remark that the use of of Hastad's version of PCP is not essential in our construction, and we can replace it with any *holographic proof* [3, 22] that achieves negligible error probability, and improve the running time of the prover to be nearly linear. We choose to use Hastad's version of PCP to simplify the presentation of our main theorem. However, we stress that *permuting* PCP queries is essential in our proof.

V: For $j = 1, \ldots, \log^2 N$ do:

- Choose $(i_1, i_2, i_3)_j \in [1, N]$ according to Hastad's PCP Verifier.
- Choose a random permutation σ_j over $\{i_1, i_2, i_3\}$. Let $(i'_1, i'_2, i'_3)_j = \sigma_j(i_1, i_2, i_3)$.
- Compute 3 *independent* PIR encodings of i'_1, i'_2, i'_3 $PIR_j(i'_1)$ $PIR_j(i'_2)$ $PIR_j(i'_3)$ (i.e. 3 queries for retrieval from N-bit database, each having its own PIR independent encoding and decoding keys)

$V \rightarrow P$: For $1 \leq j \leq \log^2 N$ send $PIR_j(i'_1)$ $PIR_j(i'_2)$ $PIR_j(i'_3)$

$P \rightarrow V$: Prover computes Hastad PCP proof on (x, w) treats it as an N-bit database, evaluates $3\log^2 N$ PIR queries received from the verifier and sends the PIR answers back to the Verifier.

V: For $1 \leq j \leq \log^2 N$ PIR-decode the answers to $PIR_j(i'_1)$, $PIR_j(i'_2)$, $PIR_j(i'_3)$, apply σ_j^{-1} to get the answers to $(i_1, i_2, i_3)_j$. If for all $1 \leq j \leq \log^2 N$, Hastad's PCP verifier accepts on answers to PCP queries $(i_1, i_2, i_3)_j$ then accept, else reject.

In the construction of Theorem 3 simply note that Prover can write down the *trace* of the execution of the procedure call, which serves as a *witness* that the output is correct, and the same construction applies. The corollaries 2, 1, and 3 follow from

our construction by simply by plugging in the appropriate implementation of PIR protocols.

Next, we describe the construction of Theorem 2, which also achieves witness-indistinguishability. We shall use Zero-Knowledge PCP and one-round SPIR. Again, we denote $|ZKPCP(w,x)| = N$ (recall that N is polynomial in $|w|$). This time by j we denote $O(\log^{O(1)} N)$ queries needed by ZKPCP to achieve negligible error probability while maintaining super-logarithmic bound on the number of bits needed to be read by the ZKPCP verifier to break the Zero-Knowledge property of the PCP [17].

V:	Choose $i_1, \ldots, i_j \in [1, N]$ according to ZKPCP Verifier; Choose a random permutation σ over $\{i_1, \ldots, i_j\}$; Let $(i'_1, \ldots, i'_j) = \sigma(i_1, \ldots, i_j)$; Compute j *independent* SPIR encodings $SPIR_1(i'_1), \ldots, SPIR_j(i;_j)$ (i.e. j queries for retrieval from N-bit database, each having its own independent PIR encoding and decoding keys);
$V \rightarrow P$:	Send $SPIR_1(i'_1), \ldots, SPIR_j(i'_j)$.
$P \rightarrow V$:	Prover computes ZKPCP proof on (x, w) and treats it as an N-bit database, evaluates j received SPIR queries on it and sends SPIR answers back to the Verifier.
V:	Decode j SPIR answers to $SPIR_1(i'_1), \ldots, SPIR_j(i'_j)$, apply σ^{-1} to get the answers to $i_1, \ldots, i_j$ queries and check if ZKPCP verifier accepts on answers to $i_1, \ldots, i_j$. If so accept, else reject.

5 Proof of Theorem 1

In order to prove Theorem 1, we need to prove completeness and soundness. We will prove them in that order.

The proof of completeness follows from our construction, i.e. there exist algorithms for C and (honest) S such that for every triple f, x, y where f is a polynomial time computation, there is a certificate which is the set of correct answers for every query set from a PCP verifier. The PIR encoding of this query set can be efficiently computed by C. Similarly, the correct PCP answers to the query set and their PIR encoding can also be computed efficiently by S. There are $\log^{O(1)} n$ PIR queries each

of which can be encoded in the [6] construction using $\log^{O(1)} n$ bits giving a total communication complexity of $\log^{O(1)} n$ bits both in the query and the response.

The intuition of the soundness proof is that even though a cheating server may use a different proof string for each query response, we can construct an "emulator" algorithm for this server that flips its coins independent of the server and the PIR encoded queries, selects one "proof" string according to some probability distribution, and responds to queries honestly according to this string. Most of the work in the proof will be in showing that the induced distribution on the query responses from the emulator is close to that of the server and hence the error bounds of the PCP theorem apply with some small slack. Recall that N is the length of the PCP proof string and let $[N]$ represent $\{1, 2, \ldots, N\}$. ϵ_{PCP} is the acceptance error of the PCP verifier using l queries and ϵ_{PIR} is the error probability of the PIR scheme used. Let $I_1, I_2, \ldots, I_l$ be random variables representing the l queries over $[N]$ asked by the verifier V and $B_1, \ldots, B_l$ be the random variables representing the decoded bit responses of the server. Let $P^{i_1 \ldots i_l}_{b_1 \ldots b_l} = Pr[B_1 = b_1 \ldots B_l = b_l | I_1 = i_1, \ldots, I_l = i_l]$ be the probability distribution of the server's responses to queries where the choice is over the server's coin flips and PIR-encodings.

Given the distribution $P^{i_1 \ldots i_l}_{b_1 \ldots b_l}$, we will construct an emulator algorithm which probabilistically (using its own coin flips and independent of the server and queries) chooses a proof string and answers the queries according to this string. Furthermore, the distribution induced on the emulator's responses by this choice of string will be close to $P^{i_1 \ldots i_l}_{b_1 \ldots b_l}$. We will show that the PCP error bound applies to the emulator's responses and thence that the error bound will also apply to the server's responses but with some additional slack. Let the emulator have a probability distribution Q on N-bit strings and let $\tilde{B}_1 \ldots \tilde{B}_l$ be random variables representing the bits of the proof string chosen according to Q. Q induces a probability distribution on the emulator's responses to queries. Denote this induced distribution by $\tilde{P}^{i_1 \ldots i_l}_{b_1 \ldots b_l} = Pr[\tilde{B}_1 = b_1 \ldots \tilde{B}_l = b_l | I_1 = i_1, \ldots, I_l = i_l]$. Define $\epsilon_S := \max_{i_1 \ldots i_l, b_1 \ldots b_l} |P^{i_1 \ldots i_l}_{b_1 \ldots b_l} - \tilde{P}^{i_1 \ldots i_l}_{b_1 \ldots b_l}|$. First, we note that the emulator's probability of acceptance by the PCP verifier is bounded by ϵ_{PCP}.

CLAIM 1 $Pr[V(I_1, \ldots, I_l; \tilde{B}_1 \ldots \tilde{B}_l) = 1] \leq \epsilon_{PCP}$.

Proof: $Pr[V(I_1, \ldots, I_l; \tilde{B}_1 \ldots \tilde{B}_l) = 1] = \sum_{D \in \{0,1\}^N} Pr[V(I_1, \ldots, I_l; d_{I_1} \ldots_{I_l}) = 1 | D = d] Pr[D = d]$ where d_{I_j} denotes the I_j-th bit of d. From the PCP Theorem, for all d, $Pr[V(I_1, \ldots, I_l; d_{I_1} \ldots_{I_l}) = 1] \leq \epsilon_{PCP}$. Applying this to the previous equation we get, $Pr[V(I_1, \ldots, I_l; \tilde{B}_1 \ldots \tilde{B}_l) = 1] \leq \sum_{D \in \{0,1\}^N} \epsilon_{PCP} Pr[D = d] = \epsilon_{PCP}$. ■

Next, we note that the server's probability of acceptance by the Verifier is close to that of the emulator.

LEMMA 1 $Pr[V(I_1, \ldots, I_l; B_1 \ldots B_l) = 1] \leq \epsilon_{PCP} + 2^l \epsilon_S$.

Proof: From the PCP Theorem, $Pr[V(I_1, \ldots, I_l; \tilde{B}_1 \ldots \tilde{B}_l) = 1] \leq \epsilon_{PCP}$. Let $i_1, \ldots, i_l \in [N]$ be query instances and $b_1 \ldots b_l$ be bit strings. Using conditional probabilities we can express the LHS as a sum over all query instances and response bit strings. Using $|\tilde{P}^{i_1 \ldots i_l}_{b_1 \ldots b_l} - P^{i_1 \ldots i_l}_{b_1 \ldots b_l}| \leq \epsilon_S$ and collapsing the sums we get, $Pr[V(I_1, \ldots, I_l; B_1 \ldots B_l) = 1] \leq Pr[V(I_1, \ldots, I_l; \tilde{B}_1 \ldots \tilde{B}_l) = 1] + \epsilon_S \sum_{I_1 \ldots I_l} \sum_{b_1 \ldots b_l} \leq \epsilon_{PCP} + 2^l \epsilon_S$. ■

Next, we show the existence of an emulator whose response distribution is close to the server's. Note mere proof of existence suffices.

LEMMA 2 There exists an "emulator" algorithm such that $\epsilon_S \leq \epsilon_{PIR} \cdot 2^{O(l \log l)}$.

For clarity of exposition, we write the proof using only 3-tuple queries and provide the calculation for $l > 3$ queries at the end. First, we make a simple claim which follows from the fact that the client chooses a random permutation of the 3-query, hence the server strategy has to be oblivious of the order of elements within the 3-tuple. Thus, wlog. we only need to consider the distributions $P^{i,j,k}$ where $1 \leq i < j < k \leq N$.

CLAIM 2 For all permutations σ on $\{1, 2, 3\}$ and all 3-tuples i_1, i_2, i_3, and all decodings b_1, b_2, b_3, $P^{i_1, i_2, i_3}_{b_1, b_2, b_3} = P^{i_{\sigma(1)}, i_{\sigma(2)}, i_{\sigma(3)}}_{b_{\sigma(1)}, b_{\sigma(2)}, b_{\sigma(3)}}$.

We want an emulator whose probability distribution Q over N-bit strings is consistent with the distribution of the server's responses. Formally, if $Q^{[N]|i_1 i_2 i_3}_{b_1 b_2 b_3}$ is the probability distribution induced by Q on indices i_1, i_2, i_3, then we want $\forall i_1, i_2, i_3, b_1, b_2, b_3$ $Q^{[N]|i_1, i_2, i_3}_{b_1 b_2 b_3} = P^{i_1 i_2 i_3}_{b_1 b_2 b_3}$ to be true. What we will actually show that there is a emulator for which this is true with small error, i.e. $max_{i_1 i_2 i_3, b_1 b_2 b_3} |Q^{[N]|i_1 i_2 i_3}_{b_1 b_2 b_3} - P^{i_1 i_2 i_3}_{b_1 b_2 b_3}| \leq \epsilon_S$. To show the existence of such an emulator, we write out the equations that the above equality implies on the actual proof strings that Q is over.

Construct matrix A with $2^3 \binom{N}{3}$ rows and 2^N columns as follows: the rows of A are labeled by the different query-response tuples $r^{ijk}_{b_1, b_2, b_3}$ and the columns are labeled by the 2^N possible N-bit strings. Let B be the vector of $8\binom{N}{3}$ of values $P^{i,j,k}_{b_1, b_2, b_3}$ from the server's strategy. Let x be the probability vector with 2^N entries where x_i is the probability that the i-th N bit string is chosen as the database. To prove Lemma 2, it is enough to show that the system $Ax = B$ can be solved for the probability vector x. Any such solution can be used for the distribution Q.

For clarity, we label the $8\binom{N}{3}$ rows of matrix A in lexicographic order over the tuples and in increasing order of weight over bit strings $r^{123}_{000}, r^{123}_{001}, r^{123}_{010}, r^{123}_{100}, r^{123}_{011} \ldots r^{N-2, N-1, N}_{111}$

and the 2^N columns of matrix A by subsets of $[N]$ enumerated in lexicographic order. That is, let $\emptyset, \{1\}, \{2\}, \ldots, \{N\}, \{1,2\}, \{1,3\}, \ldots, \{N-1, N\}, \ldots, \{1, \ldots, N\}$ represent the labels of the columns of A in order. Now we can describe how the elements of A can be filled: let $I = r^{i,j,k}_{b_1 b_2 b_3}$ be the label of the row in question and let $R^{i,j,k}_{b_1 b_2 b_3}$ be the actual vector. Let $J \subset \{1, \ldots, N\}$ be the label of the column, then set $A[I, J] = 1$ if $D_J[I] = 0$, i.e. if the J-th string has a zero in its I-th position. Clearly, if there is a solution x which is a probability distribution on N-bit strings, then the emulator using this distribution can generate the same distribution as the server on query-response tuples.

In order to prove that a solution exists, we now define a series of row transformations on A in lexicographic order. For the first row R^{123}_{000} there is no change. For the next row, r^{123}_{001}, we add the previous row to this one. That is, the second row with label r^{123}_{001} is now $R^{123}_{000} + R^{123}_{001}$. Define $R^{i,j}_{0,0}$ as the N bit vector which is defined as: $R^{i,j}_{0,0}[l] = 1$ if the l-th column label contains i and j. Note that $R^{i,j,k}_{0.0.0}$ and $R^{i,j,k}_{0,0,1}$ can not both be 1 in any index. Hence, the row with label r^{123}_{001} has the row R^{12}_{00}. Next, for the row with label r^{123}_{010}, we add the row labeled r^{123}_{000} to get the row R^{13}_{00} and similarly, for the row labeled r^{123}_{100}. Finally, for the rows labeled r^{123}_{011} we add the rows labeled r^{123}_{000}, r^{123}_{001} and r^{123}_{010}. Defining R^i_0 as the vector with 1 in all the positions where the i-th bit is 1, we note that the row labeled r^{123}_{011} has the row R^1_0. Analogously, we transform the rows labeled r^{123}_{101} to R^2_0 and r^{123}_{110} to R^3_0. Finally, the row labeled r^{123}_{111} can be transformed to the all 1's vector by adding to it all the previous rows. Next in the lexicographic order are the rows labeled r^{124}_{000} through r^{124}_{111}. We follow the same procedure outlined above for these rows to get the rows $R^{124}_{000}, \ldots, R^4_0, \mathbf{1}$. Since every one of the $8\binom{N}{3}$ rows in the matrix has a label of one of these forms we can carry out this procedure for all the rows of A to give us a new matrix A'.

In order that the solutions to the system $Ax = B$ remain unchanged when we transformed A to A', we have to perform the same set of row operations on B. To start with, consider B^{123}_{001}. To be consistent with A, we add B^{123}_{001} to give $B^{123|12}_{00}$. Similarly, we add B^{123}_{000} to B^{123}_{010}. Let this sum be called $B^{123|13}_{00}$ and analogously for $B^{123|23}_{00}$. Next, for B^{123}_{011}, we add $B^{123}_{000}, B^{123}_{001}$ and B^{123}_{010} to get $B^{123|1}_0$. Similarly, we get $B^{123|2}_0$ and $B^{123|3}_0$. Finally, replace B^{123}_{111} by the sum of all the eight quantities i.e. $\sum_{b_1 b_2 b_3} B^{123}_{b_1 b_2 b_3}$. Follow this procedure for all the values of B to give a new vector B'.

CLAIM 3 The systems $Ax = B$ and $Ax' = B'$ have exactly the same set of solutions.

Proof: We can write the row transformations that we have performed as a matrix T that left multiplies both sides. Since we have performed the same transformations on B as on A, we get $TAx = TB$. Trivially, T is an invertible matrix. ■

Now, we proceed to show that the system $A'x = B'$ is solvable , i.e. there is at least one solution which is a probability. As is obvious from the construction of A', all rows are duplicated many times. Hence A' is not full rank and we have to show that B' is in the column space of A'. We do this by collecting all the unique rows and reordering the rows of matrix A' so that it becomes upper triangular. Since, we want A' to be upper triangular, we must dovetail the row order to the column order that we have already established. Hence, we list the rows in the following order now: the first row will be the vector $R^{\emptyset}_{\emptyset}$ which is the all ones vector. Next, we list in order $R^i_0, i = 1, \ldots, N$ (only one copy each). Consider the elements of the row R^1_0. Recall that the columns of A' (as were the columns of A) are labeled as $\emptyset, \{1\}, \{2\}, \ldots, \{N\}, \{1,2\}, \ldots$. Hence, R^1_0 is of the form $0, 1, \ldots$ whereas R^2_0 has the form $0, 0, 1, \ldots$ and so on. Next, we list in order one copy each of $R^{i,j}_{00}$. Similarly, $R^{1,2}_{0,0}$ will have $N+1$ leading zeros and then a 1 in the column labeled $\{1,2\}$. One can similarly write the elements of the rows of the form $R^{i,j,k}_{0,0,0}$. We have accounted for $l = 1 + \binom{N}{1} + \binom{N}{2} + \binom{N}{3}$ rows and by listing the rows in this order gives an upper triangular part of A'. To preserve equivalence, we have to reorder the elements of B' in the same way. The first element of B' is 1. The second element in B' (corresponding to row R^1_0) is $P^{123|1}_0$. Similarly, the next $N-1$ elements are $P^{123|2}_0$ through $P^{N-2,N-1,N|N}_0$. The next set of elements are $P^{123|12}_{00}$ through $P^{N-2,N-1,N|N-1,N}_{00}$ followed by P^{123}_{000} through $P^{N-2,N-1,N}_{000}$.

The remaining $8\binom{N}{3} - (1 + \binom{N}{1} + \binom{N}{2} + \binom{N}{3})$ rows are all duplicates of the rows in the upper triangle since our transformation from A to A' transformed all the rows. For example, the row R^{ij}_{00} will be repeated $N-2$ times, once for every $k \in [N], k \neq i \neq j$. Similarly, R^i_0 will be repeated once for every pair $j, k \in [N], i \neq j \neq k$. This gives us an easy construction of the left null space $L_{A'}$ of A'. For every i-th row not in the upper triangle, we know that it is a duplicate of some j-th row in the upper triangle. We can construct a vector in $L_{A'}$ which has a 1 in the i-th position and -1 in the j-th position. This construction describes all the vectors in $L_{A'}$. Now, it is sufficient that B' be orthogonal to all the vectors in $L_{A'}$, i.e. $\forall y \in L_{A'}(y, B') = 0$. By our construction, every such y has a -1 in one of the first t elements and a $+1$ in one of the remaining positions. This gives us the following equality constraints between elements of B'. The duplicates of the rows in A' of the form $R^{i,j}_{00}$ give constraints of the form:

$$\forall i, j, k, k' \in [N], i \neq j \neq k \neq k', P^{ijk}_{000} + P^{ijk}_{001} = P^{ijk'}_{000} + P^{ijk'}_{001}.$$

All the duplicate rows in A' of the form R^i_0 give constraints of the form

$$\forall i, j, j', k, k' \in [N], i \neq j \neq j' \neq k \neq k'$$

$$P^{ijk}_{000} + P^{ijk}_{001} + P^{ijk}_{010} + P^{ijk}_{011} = P^{ij'k'}_{000} + P^{ij'k'}_{001} + P^{ij'k'}_{010} + P^{ij'k'}_{011}.$$

Hence, if these constraints are true on B' then B' is in the column space of A'.

Define projections of the probability distributions $P^{i_1 i_2 i_3}$ as follows. Let $P^{i_1 i_2 i_3 | i_1 i_2}_{b_1 b_2} = P^{i_1 i_2 i_3}_{b_1 b_2 0} + P^{i_1 i_2 i_3}_{b_1 b_2 1}$. Similarly, define projections of the form $P^{i_1}_0$. The following claim follows from the assumption of zero error probability for PIR. It says that the server's response on a particular query index cannot depend on the tuple.

CLAIM 4 Let t be any single query or two query indices and let s and s' be any three-query tuples containing t. Then, if $\epsilon_{PIR} = 0$, $P^{s|t} = P^{s'|t}$.

By the claim above, projection probabilities are well defined. Let $P^{i_1,i_2} := P^{s|i_1,i_2}$ for any s which contains i_1 and i_2. Likewise, P^{i_1} is defined analogously.

LEMMA 3 The system $A'x = B'$ yields a valid set x of probabilities on N-bit strings when $\epsilon_{PIR} = 0$.

Proof: We have shown above that the system $A'x = B'$ has a solution when B' satisfies the projection constraints above. Indeed, that these constraints are satisfied follows immediately from the assumption of zero-error PIR. However, we still have to ensure that the solutions we have are probabilities. The first row ensures that all the elements of the solution vector sum up to 1. However, it is not obvious that the elements are non-negative. To see this, we break the solution to the upper triangular system into blocks. An (i, j) block is a subset of rows consisting of the row $R^{i,j}_{00}$ and all rows of the form R^{ijk}_{000}. Note that P^{ijk}_{000} is already non-negative since it is a given probability. The constraints on P^{ij}_{00} imply that $P^{ij}_{00} = P^{ijk}_{000} + P^{ijk}_{001}$ for all $k \neq i \neq j$. Furthermore, the rows R^{ij}_{00} has no common 1 positions with any row except rows of the form R^{ijk}_{00*}. From this we can infer that a non-negative solution exists in this block. We can follow the same argument for all such (i, j) blocks since they are independent. Finally, by an analogous argument, it can be shown that the i blocks can be solved with non-negative solutions as well. ■

It now remains to consider the case of $\epsilon_{PIR} > 0$. First, we note that permutations of a query do not give any advantage to the adversary since we chose a random permutation of any query. However, the projection constraints may not hold exactly, and hence existence of a solution to $A'x = B'$ is not a given. A PIR scheme with $\epsilon_{PIR} > 0$ implies that the projection constraints are satisfied with error bounded by ϵ_{PIR}. That is, for example, $\forall i \neq j \neq k \neq k' \in [N]\ |P^{ijk}_{000} + P^{ijk}_{001} - (P^{ijk'}_{000} + P^{ijk'}_{001})| \leq \epsilon_{PIR}$. The following lemma shows that there is a solution $\tilde{P}$ that satisfies all the projection constraints and is close to the given probabilities P.

LEMMA 4 There exists a probability vector $\tilde{P}^{i_1 \ldots i_l}_{b_1 \ldots b_l}$ such that the system $Ax = \tilde{P}$ is solvable and $max_{i_1 \ldots i_l, b_1 \ldots b_l} |P^{ijk}_{b_i b_j b_k} - \tilde{P}^{ijk}_{b_i b_j b_k}| \leq 2^{O(l \log l)} \epsilon_{PIR}$.

Proof: To create the vector $\tilde{P}$ of probabilities we first compute $\bar{P}^{ij}_{00} := \frac{1}{N-2}\sum_{k\neq i\neq j} \tilde{P}^{ijk|ij}_{00}$ where $\tilde{P}^{ijk|k}_{b_ib_j} := \tilde{P}^{ijk|k}_{b_ib_j0} + \tilde{P}^{ijk|k}_{b_ib_j1}$. Similarly, $\bar{P}^{i}_{0} := \binom{N-1}{2}^{-1}\sum_{k\neq i\neq j} \tilde{P}^{ijk|i}_{0}$ with $\tilde{P}^{ijk|i}_{0}$ defined analogously. We now show how to compute $\tilde{P}^{ijk}_{b_ib_jb_k}$ from these values. There are 8 values to be calculated $\tilde{P}^{ijk}_{000}$ through P^{ijk}_{111}. Start with $P^{ijk}_{000} = P^{ijk}_{000}$. Given the values $\bar{P}^{ij}_{00}$ and $\bar{P}^{i}_{0}$ we can compute all the $\tilde{P}^{ijk}_{b_ib_jb_k}$ such that they satisfy the projection constraints exactly as follows. $\tilde{P}^{ijk}_{001} := \bar{P}^{ij}_{00} - \tilde{P}^{ijk}_{000}$. Similarly, we can compute $\tilde{P}^{ijk}_{010} := \bar{P}^{ik}_{00} - \tilde{P}^{ijk}_{000}$ and $\tilde{P}^{ijk}_{100} := \bar{P}^{jk}_{00} - \tilde{P}^{ijk}_{000}$. Then, $\tilde{P}^{ijk}_{011} := \bar{P}^{i}_{0} - (\tilde{P}^{ijk}_{000} + \tilde{P}^{ijk}_{001} + \tilde{P}^{ijk}_{010})$ and so on. Finally, we have to show that the probabilities $\tilde{P}$ are within some ϵ of P. We can compute the distance by tracing the computation path of these probabilities above. The error bound depends on the number of queries l asked and so we state this last bound in terms of l.

CLAIM 5 For any client making l queries, there exists a probability vector $\tilde{P}$ satisfying the projection constraints such that for all $i_1, \ldots, i_l$ and $b_1 \ldots b_l$, $|P^{i_1\ldots i_l}_{b_1\ldots b_l} - \tilde{P}^{i_1\ldots i_l}_{b_1\ldots b_l}| \leq 2^{O(l \log l)}\epsilon_{PIR}$.

This claim completes the proof of Lemma 2. We omit the simple proof for lack of space.

Proof of Theorem 1: Given a cheating server whose acceptance probability for a language $L \in NP$ is $\geq \epsilon_{PCP}$, using Lemmas 1 and 2, there is an emulator whose acceptance probability is $\geq \epsilon_{PCP} + \epsilon_{PIR} \cdot 2^{O(l \log l)}$. ϵ_{PCP} is already negligible by assumption. The second term can be made negligible by increasing the security parameter in the PIR scheme by a poly-logarithmic factor. ■

6 Extensions and Further results

We briefly sketch the proof of Theorem 2. It is straightforward to see that Theorem 2 carries all the properties of Theorem 1 using essentially the same proof (i.e. showing that one can construct probability distribution on databases which will pass PCP theorem) but converting the PCP proof into a ZKPCP proof and using PIR instead of SPIR. To prove witness-indistinguishability, we now assume that the Prover is honest, and that there is a poly-bounded Verifier who wants to distinguish witness w_1 from witness w_2. SPIR guarantees that the verifier will not be able to read more than allowed by our protocol $\log^{O(1)} N$ bits and by the properties of ZKPCP the bits retrieved are computationally indistinguishable for w_1 and w_2. Hence, if the Verifier can distinguish this violates either SPIR security or the zero-knowledge property of ZKPCP [17]. This sketch will be elaborated in the full version of the paper.

The proof of Theorem 1 also has implications for one-round multi-prover proof systems in the information-theoretic setting. Notice that we show in Theorem 1 how to combine PIR protocols with PCP in one round. In the multi-prover setting, information-theoretic implementations of PIR are known [5, 1] with small communication complexity. Our theorem provides an alternative way to reduce the communication complexity in the multi-prover setting: all provers can write down a PCP proof on their work tapes, and then the communication complexity is simply the cost of retrieving poly-logarithmic number of bits using information-theoretically secure PIR protocols. So far, the best bounds known for retrieving a single bit using info-theoretic PIR with only two provers is $O(N^{\frac{1}{3}})$ [5]. Thus, our approach gives an inferior result to [10]. However, our technique works with any multi-database PIR protocol and thus an improvement in information-theoretic PIR protocols would improve the communication complexity in our approach. We remark as an aside that our results also hold in the setting of universal service provider PIR [8].

Note that our "emulation" technique is not PCP-specific. More specifically, any adversary who does not answer PIR queries honestly can be emulated (with small error) by an algorithm that flips coins independently of the PIR encodings to choose a database which it then uses to answer the queries honestly.

We also wish to point out that the same Verifier's message can be used for multiple proofs to multiple provers, and they *all* can be checked for correctness (with the same error probability) without any additional messages from the verifier. As long as the Verifier does not reveal any additional information about his private key or whether he accepted or rejected each individual input, the error probability holds.

Acknowledgements

We wish to thank Silvio Micali and Yuval Ishai for helpful discussions.

References

[1] A. Ambainis. Upper bound on the communication complexity of private information retrieval. In *Proc. of 24th ICALP*, 1997.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.

[3] L. Babai, L.Fortnow, L. Levin and M. Szegedy Checking computations in polylogarithmic time. In Proc. of the *23rd Annual ACM Symposium on the Theory of Computing*, pp. 21-31, 1991.

[4] I. Biehl and B. Meyer and S. Wetzel. Ensuring the Integrity of Agent-Based Computation by Short Proofs. Mobile Agents '98, Kurt Rothermel and Fritz Hohl, editors, Lecture Notes in Computer Science, Vol. 1477, 1998, Springer-Verlag, pp. 183–194.

[5] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. of 36th FOCS*, pages 41–50, 1995. Journal version to appear in JACM.

[6] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *Advances in Cryptology: Proceedings of EUROCRYPT99*, 1999.

[7] C. Dwork and M. Naor Zaps and Apps talk given at DIMACS Workshop on Cryptography and Intractability March 20 - 22, 2000 DIMACS Center, Rutgers University, Piscataway, NJ.

[8] G. Di Crescenzo, Y. Ishai, and R. Ostrovsky. Universal service-providers for database private information retrieval. In *Proc. of the 17th Annu. ACM Symp. on Principles of Distributed Computing*, pages 91-100, 1998.

[9] G. Di Crescenzo, T. Malkin, and R. Ostrovsky. Single-database private information retrieval implies oblivious transfer. In *Advances in Cryptology - EUROCRYPT 2000*, 2000.

[10] U. Feige and J. Kilian, Two Prover Protocols: low error at affordable rates. In *Proc of Annual ACM Symposium on the Theory of Computing, 1994* pp. 172-183.

[11] U. Feige and A. Shamir, Witness Indistinguishable and Witness Hiding Protocols. In *Proc. of Annual ACM Symposium on the Theory of Computing 1990*, pages 416-426.

[12] F. Ergün, S Kumar, and R. Rubinfeld. Fast pcps for approximations. FOCS 1999.

[13] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin Protecting data privacy in private information retrieval schemes. In Proc. of the *30th Annual ACM Symposium on the Theory of Computing*, pp. 151-160, 1998.

[14] J. Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 1–10, El Paso, Texas, 4–6 May 1997.

[15] J. Kilian Zero-Knowledge with Logspace Verifiers. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, pages 25-35, El Paso, Texas, 4–6 May 1997.

[16] J. Kilian Improved Efficient Arguments In *Proc. of CRYPTO* Springer LNCS 963:311-324, 1995

[17] J. Kilian, E. Petrank, and G. Tardos. Probabilistic Checkable Proofs with Zero Knowledge. In 29th ACM Symp. on Theory of Computation, May 1997.

[18] E. Kushilevitz and R. Ostrovsky Replication is not needed: Single Database, computationally-private information retrieval. In *Proc. of Thirty-Eight Foundations of Computer Science* pp. 364-373, 1997.

[19] E. Kushilevitz and R. Ostrovsky. One-way Trapdoor Permutations are Sufficient for Non-Trivial Single-Database Computationally-Private Information Retrieval. In *Proc. of EUROCRYPT '00*, 2000.

[20] S. Micali. CS proofs (extended abstracts). In *35th Annual Symposium on Foundations of Computer Science*, pages 436–453, Santa Fe, New Mexico, 20–22 1994. IEEE.

[21] M. Naor and B. Pinkas, Oblivious transfer and polynomial evaluation. In Proceedings of the 31st Annual Symposium on Theory of Computing. ACM, pp. 33-43, 1999.

[22] A. Polischuk and D. Spielman Nearly-linear Size Holographic Proofs In Proceedings of the 25th Annual Symposium on Theory of Computing. ACM, pp. 194-203, 1994.

[23] R. Raz A Parallel Repetition Theorem SIAM J. Computing Vol. 27, No. 3, pp. 763-803, June 1998.

[24] B.S. Yee. A Sanctuary For Mobile Agents. Proceedings of the DARPA Workshop on Foundations for Secure Mobile Code. March, 1997.

[25] A.C. Yao. Theory and Applications of Trapdoor Functions *In 23rd Annual Symposium on Foundations of Computer Science* pages 80-91, Chicago, Illinois, 3-5 November 1982. IEEE.