Linear Zero-Knowledge - A Note on Efficient Zero-Knowledge Proofs and Arguments

Ivan Damgård, Aarhus University, BRICS (ivan@daimi.aau.dk) and
Ronald Cramer, CWI (cramer@cwi.nl)

Mar. 12, 1996

Abstract

We present a zero-knowledge proof system [19] for any NP language L, which allows showing that $x \in L$ with error probability less than 2^{-k} using communication corresponding to $O(|x|^c) + k$ *bit commitments*, where c is a constant depending only on L. The proof can be based on any bit commitment scheme with a particular set of properties. We suggest an efficient implementation based on factoring.

We also present a 4-move perfect zero-knowledge interactive argument for any NP-language L. On input $x \in L$, the communication complexity is $O(|x|^c) \cdot max(k, l)$ *bits*, where l is the security parameter for the prover [1]. Again, the protocol can be based on any bit commitment scheme with a particular set of properties. We suggest efficient implementations based on discrete logarithms or factoring.

We present an application of our techniques to multiparty computations, allowing for example t committed oblivious transfers with error probability 2^{-k} to be done simultaneously using $O(t+k)$ commitments. Results for general computations follow from this.

As a function of the security parameters, our protocols have the smallest known asymptotic communication complexity among general proofs or arguments for NP. Moreover, the constants involved are small enough for the protocols to be practical in a realistic situation: both protocols are based on a Boolean formula Φ containing and-, or- and not-operators which verifies an NP-witness of membership in L. Let n be the number of times this formula reads an input variable. Then the communication complexity of the protocols when using our concrete commitment schemes can be more precisely stated as at most $4n+k+1$

[1] The meaning of l is that if the prover is unable to solve an instance of a hard problem of size l before the protocol is finished, he can cheat with probability at most 2^{-k}

commitments for the interactive proof and at most $5nl+5l$ bits for the argument (assuming $k \leq l$). Thus, if we use $k = n$, the number of commitments required for the proof is linear in n.

Both protocols are also proofs of knowledge of an NP-witness of membership in the language involved.

1 Introduction

Most known zero-knowledge interactive proofs or arguments for a general NP language (such as [12] and [3]) are built from a basic step allowing the prover to cheat with some constant probability, e.g. 1/2. In order to achieve a smaller error probability, the obvious method is to iterate this basic step k times to get error probability at most 2^{-k}. If one adopts the often used convention of setting k equal to the size of the input, such a method would require $\Omega(n^2)$ commitments to show e.g. satisfiability of a Boolean circuit of size n.

Several methods have been suggested for achieving the security amplification more efficiently than by the naive method. Boyar et al.[1] was the first to find a "sub-quadratic" zero-knowledge protocol for circuit satisfiability, and their results were later extended by Killian [15]. Killian obtained, using the probabilistically checkable proofs (PCP) of [4], a zero-knowledge interactive proof that x is in NP language L using $O(|x|^{c_1}) + O(\log^{c_2}(|x|)k$ ideal bit commitments and having error probability 2^{-k}.

Results were also given on interactive arguments, that used a similar technique, plus a collision intractable hash function. This result was further improved in [16], resulting in an interactive argument with communication complexity $O(l \log(l)k)$ bits. Here, and in the following, l is the security parameter for the prover, i.e. in order to cheat with probability larger than 2^{-k}, the prover must solve an instance of size l of a hard computational problem, such as finding a discrete logarithm modulo an l-bit prime[2].

In this work, we show that these communication complexities can be further improved, if one uses bit commitments with particular properties. Of course, any bit commitment scheme must commit the prover, unconditionally or not, to a particular bit, and it must be impossible for the verifier, unconditionally or not, to find the bit committed to from a commitment. The additional properties we need can be informally described as follows:

[2] More precisely, one can show that if a prover can argue a false statement with success probability $\epsilon > 2^{-k}$, then he can solve the hard problem in time $O(1/(\epsilon - 2^{-k}))$

- Given any commitment C containing the bit b, the verifier can (on his own) compute a commitment C' that contains the bit $1 - b$.
- Given a commitment C that contains a 1-bit, the prover must be able to convince the verifier that this is the case, using an honest verifier zero-knowledge protocol. This protocol must be a 3-move Arthur-Merlin game, it must have exponentially small error pobability, and have communication complexity corresponding to a constant number of commitments.

As detailed later, these conditions can be met by commitments based on the factoring problem, or the discrete logarithm problem in a group of prime order.

The second property may seem unnecessary: if the prover wants to demonstrate that C contains a 1, why not just open the commitment? However, since the above involves a zero-knowledge protocol, it can allow simulation of a "proof" that C contains a 1, even if it actually contains a 0. This technicality will be of crucial importance later (more precisely, in the proof of Theorem 3.1).

We do not use PCP's to build our protocols, in stead we use a new proof technique that may be of independent interest: We start from any Boolean formula Φ for checking an NP-witness for the language in question, and reduce the problem of showing that Φ is satisfiable to showing that a monotone formula constructed from Φ is satisfied by inputs contained in a given set of commitments. We then apply a technique derived from the "proofs of partial knowledge" introduced by Cramer et al. [7] (and independently in [18]).

This results in zero-knowledge interactive proofs for NP with error probability 2^{-k} and communication complexity corresponding to $O(|x|^c) + k$ commitments; and in interactive arguments for NP with communication complexity $O(|x|^c) \cdot max(k, l)$ bits (we count commitments for the proof and bits for the argument to facilitate comparison with [15, 16]).

Comparing this to [15], [16] which were the best results so far, we see that for interactive proofs, the term depending on k has been reduced from $O(\log^c |x|)k$ to k. For arguments, our result is inferior to [16] when viewed as a function of $|x|$, but superior as a function of the security parameters k and l. Note that our interactive argument has no need for a collision-intractable hash function, we only need commitments with the right properties. Hence our cryptographic assumption is potentially weaker than the ones needed in

[16] [3].

The constants involved in our communication complexities are small enough for the protocols to be practical in a realistic situation: let n be the number of times the formula Φ reads an input variable. Then the communication complexity of the protocols when using our concrete commitment schemes can be more precisely stated as at most $4n+k+1$ commitments for the interactive proof and at most $5nl + 5l$ bits for the argument (assuming $k \leq l$). By contrast, the PCP-based methods of [15], [16] hardly have any practical relevance because of the elaborate reductions needed to build a PCP.

Given a circuit C of size m, one can easily build a formula of size $O(m)$ that is satisfiable precisely if C is. Hence our result implies an interactive proof that proves satisfiability of a circuit of size m with error probability 2^{-m} using $O(m)$ commitments, adopting again the convention of setting the security parameter equal to the input size. Even if an extremely small PCP would exist, the protocol in [15] would use $\Omega(m \log^c m)$ commitments to solve the same problem. To the best of our knowledge, our protocol is the first to acheive "linear zero-knowledge" in this sense. For arguments, we get $O(m^2)$ bits using $l = k = m$, where [16] would be $O(m^2 \log m)$.

As a further application of our techniques, we show a connection to multiparty computations, allowing for example t committed oblivious transfers with error probability 2^{-k} to be done simultaneously using $O(t + k)$ commitments. Improved results for the communication complexity of general computations follow from this.

Remark

For the case of interactive proofs, we have, like [15], ignored in the statement of results the communication needed to set up the commitment scheme [4]. This is reasonable, as the same commitment scheme can be reused in many proofs. For arguments, however, an attractive point is that cheating is only possible if the intractability assumption used is broken *while the protocol is running* [5]. This, however, is only true if a new instance of the commitment

[3] although no example is currently known that would support our needs, and not simultaneously imply a collision intractable hash function

[4] In any real implementation, the verifier needs to receive some public parameters of the commitment scheme, and possibly a zero-knowledge proof that they were chosen correctly

[5] in contrast to the situation for proofs, where breaking the assumption at any later time can cause problems

scheme is chosen in every run of the protocol. Our communication complexity for arguments therefore includes communication for setting up the commitment scheme.

We also remark that in all our protocols, the verifier only sends random bits, so they can be made non-interactive using the Fiat-Shamir heuristic.

2 Notation and Properties for Bit Commitment Schemes

This section introduces some notation for bit commitments and states a little more precisely the properties we need. In this extended abstract, there will not be enough space for a full formal statement. For definitions of interactive poof systems and zero-knowledge, please refer to [19].

We will think of a bit commitment scheme as defined by a probabilistic polynomial time algorithm G called a *key generator*. It takes as input 1^l, where l is a security parameter. It produces a description of two functions $c : \{0,1\}^{l_r} \times \{0,1\} \to \{0,1\}^l$ and $v : \{0,1\}^l \times \{0,1\}^{l_r} \times \{0,1\} \to \{accept, reject\}$. Here, l_r polynomially bounded in l. We refer to these functions as the *public key* of the commitment scheme.

To commit to a bit b, the prover (P) chooses r at random and sends $c(r,b)$ to the verifier (V). To open a commitment, P sends r, b to V, who computes $v(c(r,b), r, b)$ and accepts or rejects the opening, depending on the outcome.

This is certainly not the most general description possible of a commitment scheme, but it will cover all the cases we consider here.

For interactive proofs, we will need commitments that are *unconditionally binding:* b is uniquely determined from $c(r,b)$; and *computationally hiding:* the distributions of $c(r,1)$ and of $c(r,0)$, where r is random, are computationally indistinguishable (refer to [19] for details). For such a commitment, P will run G, send the result to V, and possibly convince V that the result was computed correctly.

For interactive arguments we need the dual poperties, namely that commitments are *unconditionally hiding:* the distributions of $c(r,1)$ and of $c(r,0)$, where r is random, are equal; and *computationally binding:* consider any probabilistic polynomial time algorithm that receives c, v as generated by G on input 1^l and produces x, r_0, r_1 as output. Then the probability that $v(x, r_0, 0) = v(x, r_1, 1) = accept$ is superpolynomially small in l. For such a commitment, V will run G, send the result to P, and possibly convince P

that the result was computed correctly.

Unconditionally hiding commitment may in addition be *trapdoor*, or chameleon [3]. For a trapdoor commitment, the generator G outputs in addition a string T called the *trapdoor information*. Given the trapdoor, one can cheat the commitment scheme, i.e. there is a polynomial time algorithm that on input T will produce pairs r_0, r_1 such that $c(r_0, 0) = c(r_1, 1) = C$, $v(C, r_0, 0) = v(C, r_1, 1) = accept$, and the distribution of C is the same as that of $c(r, b)$ for random r. We will assume that, on the other hand, given C and any pair r_0, r_1 such that $v(C, r_0, 0) = v(C, r_1, 1) = accept$, it is easy to compute T.

2.1 Special Properties Needed

We now explain three extra properties that we will need our bit commitment to possess (two of them suffice for unconditionally binding commitments).

Definition 2.1 Let a bit commitment scheme be given. We say that commitments *can be negated* if, when given a commitment $C = c(r, b)$, V can (on his own) compute efficiently a commitment C', such that there is an r' for which $C' = c(r', 1 - b)$; moreover P can efficiently compute r' from r, and vice versa. □

The second special property we need is that the scheme has an efficient proof of contents, i.e. a protocol of a special form, that P can use to convince V that a commitment contains a 1.

Definition 2.2 A bit commitment scheme has an *efficient proof of contents* if there is a pair of probabilistic polynomial time interactive Turing machines (A, B), that receive a commitment C as common input and have the following properties:

- (A, B) is a 3-move Arthur-Merlin game, i.e. conversations have the form (a, e, z), where a, z are generated by A, and e is chosen by B at random in $\{0,1\}^t$, for some t depending on l. At the end of the protocol, B evaluates a predicate ϕ on input C, a, e, z and accepts if and only if $\phi(C, a, e, z) = 1$. B always accepts if A know to open C as a 1.
- Some technical conditions are needed for our main results: The communication complexity of (A, B) is $O(l)$ bits and t must be linear in l;

for unconditionally binding schemes we will always choose l such that $t \geq k$, where 2^{-k} is the error probability we want for the interactive proof in which the scheme will be used.

- Given two accepting conversations of form (a, e, z), (a, e', z') (where $e \neq e'$), one can efficiently compute r, such that $v(C, r, 1) = accept$. That is, (A, B) is a proof of knowledge that A knows how to open C as a 1 (satisfying a particularly strong version of knowledge soundness).

- (A, B) is honest verifier perfect zero-knowledge, with a simulator that on input C, e produces a conversation (a, e, z) such that $\phi(C, a, e, z) = 1$. We assume for simplicity that the simulator will always produce an accepting conversation, even on input a commitment containing a 0 (this is true of our concrete examples) [6].

□

Finally, for an unconditionally hiding trapdoor commmitment scheme, we will need that the the scheme has *an efficient proof of knowledge of the trapdoor*, i.e. there is a constant round witness hiding proof of knowledge (see [10]), with communication complexity linear in l, that V can use to demonstrate that he knows the trapdoor of the commitment scheme. This will be necessary to prove that our argument is perfect zero-knowledge.

Examples of commitments that have these properties, based on either factoring or the discrete log problem can be found in Section 6.

3 A General Framework

In this section, we give a general method for demonstrating that a word x is in a language $L \in NP$. The same high level method works for both proofs and arguments, the only difference being the type of bit commitment scheme used.

The resulting protocol will only be honest verifier zero-knowledge. Then in the following two sections, we show how to obtain zero-knowledge in general for interactive proof, resp. arguments.

We assume in this section that a bit commitment scheme satisfying Definitions 2.1 and 2.2 is already set up, so that we have functions c, v for committing and verifying. By Cook's theorem, there exists a Boolean formula Φ

[6] Note that since commitments containing 0's are assumed to be indistinguishable from 1-commitments, the simulator would in general, except with negligible probability, produce an accepting conversation on input a random 0-commitment

of size polynomial in $|x|$ that can verify an NP-witness of membership of x in L. In fact, any formula that does the verification will do - and usually a formula constructed ad hoc can be much smaller than one constructed through the machinery of Cook's theorem. Without loss of generality, assume that Φ contains only and-, or- and not operators, and that all negations occur at the inputs.

Let m be the number of different input variables to Φ, and n be the number of times Φ reads an input variable. Thus $m \leq n$. Let Φ' denote the monotone formula obtained from Φ by removing all the negations and renaming the input variables, so that all n references to the input refer to different variables. For example, if $\Phi = (a \wedge b) \vee (\neg a \wedge \neg b)$, then we would have $\Phi' = (a \wedge b) \vee (c \wedge d)$.

Let Ψ be a monotone formula on n input variables, and let a set of commitments $D_1, .., D_n$ be given. Then we say that the set of strings $r_1, ..., r_n$ *Ψ-opens* $D_1, ..., D_n$ if $\Psi(\gamma_1, ..., \gamma_n) = 1$, where $\gamma_i = 1$ if and only if $v(D_i, r_i, 1) = accept$.

To describe the general method, we will need the following theorem:

Theorem 3.1 *Suppose we are given a bit commitment scheme which has an efficient proof of contents (A, B) according to Definition 2.2. Let commitments $D_1, ..., D_n$ and a monotone Boolean formula Ψ on n inputs be given, where Ψ uses each input bit only once. Then there exists a protocol (A', B') with the following properties: (A', B') is a 3-move Arthur-Merlin game, is honest verifier perfect zero-knowledge, and has communication complexity t bits plus n times that of (A, B). Furthermore, from 2 conversations of (A', B') of form $(a, e, z), (a, e', z')$, where $e \neq e'$ one can efficiently compute a set of strings that Ψ-opens $D_1, ..., D_n$.*

Very roughly speaking, what (A', B') does is to execute (A, B) n times in parallel, using D_i as input to the i'th instance. Here, A' is to allowed to choose himself the challenge to be answered in each instance, *however* the challenge values must obey some constraints defined in terms of Ψ and a random value chosen by the verifier. The full proof of the theorem uses methods derived from [7] and can be found in Appendix A. This leads to the following protocol for showing that Φ is satisfiable, with error probability at most 2^{-k}:

Protocol (P, V)

1. Let $b_1, .., b_m$ be a set of input bits that satisfy Φ. For $i = 1..m$, P now makes a commitment C_i to b_i, and sends them to V.

2. For $i = 1...m$, V computes from C_i a commitment C'_i containing $1 - b_i$.

3. Number the positions in Φ where an input bit is used from 1 through n. For $j = 1..n$, let $D_j = C_i$, if the bit b_i is used at this position, and let $D_j = C'_i$ if the bit $1 - b_i$ is used.

4. Using the protocol (A', B') guaranteed by Theorem 3.1, P now convinces V that $D_1, ..., D_n$ can be Φ'-opened, i.e. that the bits contained in $D_1, ..., D_n$ satisfy the monotone formula Φ'. The protocol (A', B') is repeated w times in parallel, where w is minimal so that $wt \geq k$.

The protocol is sound and complete
Completeness is trivial. For soundness, observe that if V accepts with probability $> 2^{-k}$ then the prover must be able to answer at least two different challenges in some instance of (A', B'). By Theorem 3.1, this gives us a way to Φ'-open $D_1, ..., D_n$. Let S be the set of commitments we can open. Note that each D_j is either C_i or C'_i for some i. By assumption on the commitment scheme, we can then open all the C_i's for which C_i or C'_i is in S. In fact, if Φ reads the same variable several times, we may have several values for each bit in C_i and its complement. However, depending on the type of bit commitment, inconsistencies are either impossible or allow breaking the intractability assumption. But if there are no inconsistencies, we can make a string of bits $b_1, ..., b_m$ that will satisfy Φ by choosing b_i to be the bit contained in C_i if C_i or C'_i is in S, and use an arbitrary value otherwise. This works since by choice of the b_i's corresponding to elements in S, we have ensured that Φ' will receive enough 1 bits to be satisfied, and hence by monotonicity of Φ', the arbitrary choice of the rest of the bits does not affect the output.
The protocol is honest verifier zero-knowledge
To simulate, we construct the C_i's as a set of all-0 commitments, and compute the C'_i's from this. We then invoke w times the honest verifier simulator of (A', B'). This simulation is perfect for unconditionally hiding commitments, and is computationally indistinguishable for unconditionally binding commitments.

4 Interactive Proofs for NP

To make a zero-knowledge interactive proof from (P, V), we use an unconditionally binding commitment scheme. The problem that (P, V) is only honest verifier zero-knowledge can be solved using a method due to Okamoto

[9], namely to let the verifier's challenge be determined by a two party coin-flipping protocol:

1. The prover sends the first message of protocol (P, V).
2. For $i = 1$ to k do: the prover commits to a bit p_i, the verifier chooses a random bit v_i, and the prover opens the commitment to p_i.
3. The prover sends P's final message in protocol (P, V), taking the verifier's challenge to be $p_1 \oplus v_1, ..., p_k \oplus v_k$.

When the verifier is honest, the challenge is still uniformly chosen, so the error probability is still 2^{-k}. Moreover, the protocol can now be simulated against a general verifier since by rewinding the verifier, the simulator can force the challenge to be a particular value of its choice, and hence the honest verifier simulator we already had is enough to simulate the rest of the conversation. The coinflipping step costs k commitments. Note also that we have set up the bit commitments such that the parameter t is equal to k, and hence only 1 iteration of the protocol (A', B') is needed. Now by simple inspection of (P, V), we get

Theorem 4.1 *Suppose there exists an unconditionally binding bit commitment scheme, in which commitments can be negated, and which has an efficient proof of contents. Then any $L \in NP$ has a zero-knowledge interactive proof system that proves $x \in L$ with error probability at most 2^{-k} using a total of $O(|x|^c) + k$ commitments, for some constant c depending only on L.*

5 Interactive Arguments for NP

To build a zero-knowledge interactive argument from (P, V), we use an unconditionally hiding trapdoor bit commitment scheme:

1. The verifier runs the key generator G, sends the resulting public key for the commitment scheme to the prover, and keeps the trapdoor private.
2. The verifier gives a witness hiding proof of knowledge of the trapdoor.
3. Protocol (P, V) is executed using the commitment scheme instance just generated.

The idea is taken from [11]. The protocol as shown here has 6 moves, but this can be condensed to 4 moves in the same way as in [11]. The prover must compute the trapdoor information in order to cheat the commitment scheme, and the verifier's witness hiding proof does not help him to do that. Hence the soundness of (P, V) is preserved. Furthermore, the protocol is now zero-knowledge, since the simulator can use the knowledge extractor for the verifier's proof of knowledge to get the trapdoor information [7]. Given the trapdoor, simulation of the rest of the protocol is trivial. The witness hiding proof costs, by assumption on the commitment scheme, a communication complexity that is $O(l)$ bits. We then get the following by inspection of (P, V):

Theorem 5.1 *Suppose there exists an unconditionally hiding trapdoor bit commitment scheme, in which commitments can be negated, and which has an efficient proof of contents and an efficient witness hiding proof of knowledge of the trapdoor. Then any $L \in NP$ has a perfect zero-knowledge interactive argument that $x \in L$ with communication complexity $O(|x|^c) \cdot max(k, l)$ bits, where c is a constant depending only on L, and if the prover can cheat with probability $\epsilon > 2^{-k}$, the prover can find the trapdoor of an instance of the commitment scheme of size l in time $O(1/(\epsilon - 2^{-k}))$.*

6 Practical Implementation and Optimizations

This section contains material related to practical implementation of our protocols. We present examples of commitments with the properties we need and compute the concrete communication complexities that result.

6.1 An Unconditionally Binding Bit Commitment Scheme

This scheme is based on the factoring problem, and is derived from the identification scheme of Goulliou and Quisquater [14].

The key generator for this scheme chooses an l bit integer n as the product of two primes p_1, p_2, such that there is an odd prime q that divides $p_1 - 1$, but not $p_2 - 1$ (easily done by choosing q, p_2 first and then looking for a prime of form $2jq + 1$ for some j). Then a constant w is chosen as a

[7] Although this by itself may not produce the trapdoor with absolute certainty, the simulator can run an exhaustive search for the trapdoor in parallel with the extractor. This will then produce the trapdoor in those exponentially few cases where the extractor fails, while the expected running time remains polynomial

random number in Z_n^* which is *not* a q'th power modulo n. The public key is now n, q, w. The functions c and v are defined by

$$c(r,b) = r^q w^{\delta(b)} \bmod n$$

where r is random in Z_n^*; and

$$v(C,r,b) = \mathit{accept} \text{ if and only if } C = r^q w^{\delta(b)} \bmod n,$$

where δ is defined as in the previous subsection.

Uisng the fact that 2 and q are relatively prime, it is not hard to show that we can have $wr^q = w^{-1}r'^q \bmod n$ if and only if w is a q'th power, and the scheme is therefore *unconditionally binding*. After choosing the public key, P should convince V in zero-knowledge that w does not have a q'th root. This is easily done using a variant of the protocol from [19] for quadratic non-residuosity - note that it is easy for P to test if a number y has a q'th root by testing if $y^{\phi(n)/q} \bmod n = 1$. Finding the bit contained in a commitment C is precisely the problem of distinguishing cosets of the subgroup of q'th powers in Z_n^*, a variant of the quadratic residuosity problem. This is a well-known problem, believed to be hard, if factoring is hard.

Hence we conjecture that the scheme is *computationally hiding* under the *q'th residuosity assumption*: the distributions of $r^q w \bmod n$ and $r^q w^{-1} \bmod n$ are computationally indistinguishable, when r is random in Z_n^* and the length of q is linear in the length of n.

Commitments can be *negated*: if $C = c(r,b)$, then anyone can easily compute $C' = C^{-1} \bmod n = c(r^{-1}, 1-b)$.

Finally, the following protocol (A,B) is an *efficient proof of contents*:

1. Let $y = C/w \bmod n$, where $C = wr^q \bmod n$ is the input commitment. Now A chooses f at random in Z_n^* and sends $a = f^q \bmod n$ to B.
2. B chooses $e \in \{0,1\}^t$ at random (where t is maximal, such that $2^t < q$) and sends it to A.
3. A sends to B $z = f \cdot r^e \bmod n$
4. B checks that $z^q = a \cdot y^e \bmod n$

6.2 Two Unconditionally Hiding Bit Commitment Schemes

As a first example, we observe that the scheme from the previous section can trivially be turned into an unconditionally hiding scheme, by letting in

stead the verifier run the key generator, but choose w to be a q'th power. Then any commitment is a random q'th power, and the trapdoor is a q'th root of w.

The second example is based on the discrete logarithm problem in a group of prime order q. For concreteness, we think here of this group as a subgroup of Z_p^*, where p is a prime, and q divides $p-1$. But any group of order q would do, e.g. on an elliptic curve. The scheme is derived from the identification scheme of Okamoto [17].

To generate the public key of the scheme, choose an l-bit prime p, such that the prime q divides $p-1$ (this is easily done by picking q first of length slightly smaller than l and then searching for p). Also find two elements $g_1, g_2 \in Z_p^*$ of order q. Then choose s_1, s_2 at random modulo q and let $w = g_1^{s_1} g_2^{s_2} \bmod p$. The public information is now p, q, g_1, g_2, w.

Let the function δ be defined by $\delta(0) = -1$ and $\delta(1) = 1$. Then the functions c and v are defined by

$$c((r_1, r_2), b) = g_1^{r_1} g_2^{r_2} w^{\delta(b)} \bmod p$$

where r_1, r_2 are random in Z_q; and

$$v(C, (r_1, r_2), b) = \mathit{accept} \text{ if and only if } C = g_1^{r_1} g_2^{r_2} w^{\delta(b)} \bmod p.$$

This scheme is *unconditionally hiding*, since any commitment is a random element in the group of order q, independently of b. It is *computationally binding* if discrete logs in the subgroup of order q are hard to compute, since being able to open a commitment both ways trivially implies that you can compute u_1, u_2 such that $w = g_1^{u_1} g_2^{u_2} \bmod p$, and solving this problem for given p, q, g_1, g_2, w is well known to be equivalent to finding discrete logs in the group generated by g_i.

It is also clearly a *trapdoor scheme*, where the trapdoor can be any pair u_1, u_2 such that $w = g_1^{u_1} g_2^{u_2} \bmod p$.

Commitments can be *negated*: given $C = c((r_1, r_2), b)$, then $C' = C^{-1} \bmod p = c((-r_1, -r_2), 1-b)$ contains $1-b$.

The scheme has the following *efficient proof of contents* (A, B):

1. Let $y = C \cdot w^{-1}$, where $C = g_1^{r_1} g_2^{r_2} w \bmod p$ is the input commitment. Now A chooses f_1, f_2 at random in Z_q and sends $a = g_1^{f_1} g_2^{f_2} \bmod p$ to B.

2. B chooses $e \in \{0,1\}^t$ at random (where t is maximal, such that $2^t < q$) and sends it to A.

3. A sends to B: $z_1 = f_1 + er_1 \bmod q$ and $z_2 = f_2 + er_2 \bmod q$

4. B checks that $g_1^{z_1} g_2^{z_2} = ay^e \bmod p$

It is trivial to verify that this protocol has the required properties. Note that the protocol is actually a proof of knowledge that A knows how to express y as a product of powers of g_1, g_2. For this problem, there are many witnesses (i.e. pairs of exponents for g_1, g_2), but it is easy to see that the protocol is witness indistinguishable. And since computing from one witness any other one is as hard as computing the discrete log of g_1 base g_2, it follows from [10] that the protocol is witness hiding, if discrete log is hard. The same protocol can therefore be used to prove knowledge of the trapdoor - as required above, it is constant round and witness hiding.

6.3 Concrete Communication Complexities

Due to space limitations we state only the result (details are in Appendix B):

Proposition 6.1 *Suppose the protocols from Theorem 4.1, resp. 5.1 are executed using the commitment schemes from Section 6.1, resp. 6.2, then assuming that $l > k$, the communication complexities will be at most $4n + k + 1$ commitments, resp. $5nl + 5l$ bits, where n is the number of times a Boolean formula for verifying an NP witness for L reads an input variable.*

7 Applying our Technique to Multiparty Computations

Loosely speaking, the multiparty computation problem is defined by a function f with p arguments and p participants such that the i'th participant owns a value x_i of the i'th argument to f. The goal is to design a protocol such that all participants learn the value of $f(x_1, ..., x_p)$, but no coalition of participants can - even by deviating from the protocol - learn more about the inputs than what is implied by their *own* inputs and the result.

The classical protocols for solving this problem in the broadcast model (where only broadcast and no private channels are available) can be found in [13, 20], with efficiency improvements e.g. in [6]. A much more substantial improvement can be obtained by using the commitment scheme from Section 6.1 to implement directly a fundamental primitive for multiparty

computation known as *committed oblivious transfer*. Committed oblivious transfer is sufficient to implement general multiparty computations, a concrete reduction can be found in [8]. We have the following:

Proposition 7.1 *Under the q'th residuosity assumption, there is a protocol for executing t committed oblivious transfers with error probability* 2^{-k} *using communication corresponding to* $O(t+k)$ *commitments.*

A committed oblivious transfer takes part between two parties A and B. Initially A has made two commitments C_0, C_1 to bits a_0, a_1, and B has made a commitment D to bit b. The purpose of the protocol is that B should end up making a commitment T to a_b. This must be done under the conditions that A does not learn b, and that B does not learn a_{1-b}.

Due to space limitations in this extended abstract, we sketch here only our protocol for a single transfer without any formal proof.

1. Let n be the modulus used in A's commitments. Then B chooses $s \in Z_n^*$ and $\epsilon = +1$ or -1 at random. Then sends $S = C_b^{\epsilon} s^q \bmod n$ to A.

2. B proves in zero-knowledge the following statement: (D contains 0 AND B knows a q'th root of $C_0 S$ OR $C_0 S^{-1}$) OR (D contains 1 AND B knows a q'th root of $C_1 S$ OR $C_1 S^{-1}$). This can be done by a straightforward variant of the protocol (P, V) presented earlier.

3. If the above proof was successful, A opens the commitment S to reveal the bit c.

4. B makes the commitment $T = c(r, a_b)$ using that if $\epsilon = 1$, then $c = a_b$, else $c = 1 - a_b$. B proves in zero-knowledge that the formula $((b = 1) \wedge (a_b = a_1)) \vee ((b = 0) \wedge (a_b = a_0))$ holds, where a_0, a_1, b, a_b refer to the bits contained in the commitments C_0, C_1, D, T. Again, it is straigtforward to devise a protocol for this based on the (P, V)-protocol.

It is clear that if B plays honestly, the protocol conveys no information about b since S is distributed independently from C_0, C_1; and B will never accept an incorrect value of a_b since commitments are unconditionally binding. On the other hand, if A plays honestly then except with probability negligible in k, B learns the value of a_b and nothing about a_{1-b} (by the proof in step 2) and T contains the correct value (by the proof in step 4). The

communication complexity amounts to $O(1)$ commitments for the proofs themselves and $O(k)$ commitments to generate the challenges for the proofs mutually at random. But several transfers done in parallel can all use the same challenge values, and the proposition above follows.

If one demands that the probability that a protocol for multiparty computation produces an incorrect result be exponentially small in the parameter k, earlier solutions need communication corresponding to $\Omega(np^2k)$ commitments, where n is the size of a Boolean circuit computing f. Using the method above together with e.g. the framework from [6], we get

Proposition 7.2 *Under the q'th residuosity assumption, there is a protocol for doing secure p-party computation in a circuit of size n that requires communication corresponding to $O(np^2) + O(pk)$ commitments.*

In this result, the $O(np^2)$ contribution accounts for the total size of the proofs that must be given, while the $O(pk)$ contribution accounts for the commitments needed to generate mutually random challenges for the interactive proofs.

References

[1] J.Boyar, G.Brassard and R.Peralta: *Subquadratic Zero-Knowledge*, Journal of the ACM, November 1995.

[2] J.Boyar and R.Peralta: *Short Discreet Proofs*, Proc. of EuroCrypt 96.

[3] G.Brassard, D.Chaum and C.Crépeau: *Minimum Disclosure Proofs of Knowledge*, JCSS, vol.37, 1988.

[4] L.Babai, L.Fortnow, L.Levin and M.Szegedi: *Checking Computations in Poly-logarithmic Time*, Proc. of STOC 91.

[5] J. Benaloh and J. Leichter: *Generalized Secret Sharing and Monotone Functions*, Proc. of Crypto 88, Springer Verlag LNCS series, 25–35.

[6] D.Chaum, I.Damgård and J. van de Graaf: *Multiparty Computations ensuring Privacy of each Party's Input and Correctness of the Result*, Proc. of Crypto 87.

[7] R.Cramer, I.Damgaard, and B.Schoenmakers: *Proofs of partial knowledge and simplified design of witness hiding protocols*, proc. of Crypto 94.

[8] C.Crépeau, J. van de Graaf and A. Tapp: *Committed Oblivious Transfer and Private Multiparty Computation*, Proc. of Crypto 95.

[9] I.Damgård, O.Goldreich, T.Okamoto and A.Wigderson: *Honest Verifier vs. Dishonest Verifier in Public Coin Zero-Knowledge Proofs*, Proc. of Crypto 95.

[10] U. Feige and A. Shamir: *Witness Indistinguishable and Witness Hiding Protocols*, Proc. of STOC 90.

[11] U. Feige and A. Shamir: *Zero-Knowledg Proofs of Knowledge in Two Rounds*, Proc. of Crypto 89.

[12] O.Goldreich, S.Micali and A.Wigderson: *Proofs that yield Nothing but their Validity and a Methodology of Cryptographic Protocol Design*, Proc. of FOCS 86.

[13] O.Goldreich, S.Micali and A.Wigderson: *How to play any mental game*, Proc. of FOCS 87.

[14] L.Guillou and J.J.Quisquater: *A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing both Transmission and Memory*, Proc. of EuroCrypt 88.

[15] J.Killian: *A note on Efficient Proofs and Arguments*, Proc. of STOC 92.

[16] J.Killian: *Efficient Interactive Arguments*, Proc. of Crypto 95.

[17] T.Okamoto: *Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes*, Proc. of Crypto 92.

[18] A.De Santis, G Di Crescenzo, G. Persiano and M.Yung: *On Monotone Formula Closure of SKZ*, Proc. of FOCS 94.

[19] S.Goldwasser, S.Micali and C.Rackoff: *The Knowledge Complexity of Interactive Proof Systems*, SIAM J.Computing, Vol.18, pp.186-208, 1989.

[20] A. Yao: *How to generate and exchange secrets*, Proc. of FOCS 86.

A Proof of Theorem 3.1

For convenience, we restate the result proved in this appendix:

Theorem A.1 *Suppose we are given a bit commitment scheme which has an efficient proof of contents (A, B) according to Definition 2.2. Let commitments $D_1, ..., D_n$ and a monotone Boolean formula Ψ on n inputs be given, where Ψ uses each input bit only once. Then there exists a protocol (A', B') with the following properties: (A', B') is a 3-move Arthur-Merlin game, is honest verifier perfect zero-knowledge, and has communication complexity t bits plus n times that of (A, B). Furthermore, from 2 conversations of (A', B') of form $(a, e, z), (a, e', z')$, where $e \neq e'$ one can efficiently compute a set of strings that Ψ-opens $D_1, ..., D_n$.*

For the proof, we need a result of Benaloh and Leichter [5]. They showed how to construct a *perfect secret sharing scheme* based on any monotone Boolean formula Ψ. The scheme can be thought of as a probabilistic algorithm which we will call BL. It takes as input any monotone formula and any bitstring s (the secret) of finite length, say t. It outputs a set of bit strings, *shares* $s_1, ..., s_n$, where n is the number of variables in Ψ. In our case, where Ψ reads every variable once, the length of each share is t.

If I is a set of indices between 1 and n, we can define a bit string $\beta_1, \ldots, \beta_n$ in a natural way by $\beta_i = 1$ iff $i \in I$. Define $\Psi(I) = \Psi(\beta_1, \ldots, \beta_n)$. We can then describe the properties of the output distribution of BL:

- Given $\{s_i |\ i \in I\}$ where $\Psi(I) = 1$, it is easy to compute s.
- The distribution of any set $\{s_i |\ i \in I\}$ where $\Psi(I) = 0$, is independent of s.

If $\Psi(I) = 0$, we let D_I denote the distribution of $\{s_i |\ i \in I\}$ (it can depend only on I). It can easily be proved that given a secret s, a set of shares $\{s_i |\ i \in I\}$, distributed according to D_I, can always efficiently completed to a full set of shares consistent with s, i.e. distributed according to the output distribution of DL on input Ψ, s.

In the following, let $\mathcal{S}$ denote the honest verifier simulator for the proof of contents. Initially, the prover is given n commitments $D_i = c(\rho_i, \beta_i)$ to bits β_i, where ρ_i and β_i are private input to the prover. Furthermore, $\rho_1, \ldots, \rho_n$ Ψ-opens $D_1, \ldots, D_n$. Let W be the set of indices for which $\beta_i = 1$, and let W' be the complement of W.

The following protocol has the properties claimed by the theorem.

1. A' chooses a random bitstring $s \in \{0,1\}^t$ and runs BL on input Ψ^* and s [8]. Let $s_1, \ldots, s_n$ denote the resulting shares.

   For $i \in W$, A' discards the share s_i (and also the string s. He now computes for each D_i a first message a_i following the algorithm of A.

   For $i \in W'$, A' runs the simulator $\mathcal{S}$ on input of D_i and s_i. This results in accepting conversations a_i, s_i, z_i, i.e. $\phi(D_i, a_i, s_i, z_i) = 1$.

   A' sends all a_i's to B'.

2. The verifier sends a random challenge $e \in \{0,1\}^t$ to the prover.

3. The prover interprets e as a secret in the sharing scheme and completes shares s_i with $i \in W'$ to a full set of shares $s_1, \ldots, s_n$ consistent with e. Next, for $i \in W$ he completes the conversation by computing z_i such that $\phi(D_i, a_i, s_i, z_i) = 1$. Note that here the shares s_i are now interpreted as challenges. Finally, the prover sends $s_1, \ldots, s_n$ and $z_1, \ldots, z_n$ to the B', who accepts iff the set of shares is consistent with c and if all n conversations a_i, s_i, z_i are accepting, i.e., $\phi(D_i, a_i, s_i, z_i) = 1$.

Completeness:
Trivial by the assumptions on the proof of contents and by the properties of the secret sharing scheme BL.

Soundness:
Suppose we are given two conversations $\{a_i\}, e, (\{s_i\}, \{z_i\})$ and $\{a_i\}, c', (\{s_i'\}, \{z_i'\})$ $(i = 1 \ldots n)$ with $e \neq e'$. Let S denote the set of i such that $s_i \neq s_i'$, and S' the complement. Then for $i \in S$ we can compute r_i such that $v(D_i, r_i, 1) = accept$. This follows from the assumptions on (A, B) (Definition 2.2). By monotonicity of Ψ, it is now sufficient to prove that $\Psi(S) = 1$. Note that if we had $\Psi^*(S') = 1$, then the $\{s_i\}$ and the $\{s_i'\}$ would be determine the same secret (challenge value), which contradicts $e \neq e'$ (recall that we are working with a secret sharing scheme for Ψ^*). Therefore $\Psi(S) = 1$, by definition of the dual.

Honest verifier zero-knowledge:
Given commitments $D_1, \ldots, D_n$, the simulator runs as follows.

1. Choose c at random from $\{0,1\}^t$ and run BL on input Ψ^* and c. Let $s_1, \ldots, s_n$ denote the output.

[8] Ψ^* denotes the dual of Ψ, which is a monotone Boolean formula as well. By definition, for $x \in \{0,1\}^n$, $\Psi^*(x) = 1$ iff $\Psi(x \oplus \mathbf{1}) = 0$ and $\Psi^*(x) = 0$ otherwise.

2. For $i = 1 \ldots n$, run the simulator $\mathcal{S}$ on input D_i and s_i. Output the conversations a_i, s_i, z_i.

First note that the s_i are distributed exactly as in the protocol. For $i \in W'$, we are following exactly A''s algorithm for generating the rest of the conversations a_i, s_i, z_i. For $i \in W$, note that we have assumed that the simulator $\mathcal{S}$ can sample perfectly conversations with a given challenge value (Definition 2.2), and so also in this case the conversations a_i, s_i, z_i are distributed exactly as in the protocol.

Communication Complexity:
By inspection of the protocol.

B Concrete Communication Complexities

Assume we use one of the two example commitment schemes shown above to implement our protocols. To evaluate their practical potential, we compute the exact communication complexities that result. In general we can say that in the worst case, the formula Φ uses every input bit and its complement exactly once, so that no reuse of commitments is possible. Hence, in the (A', B') protocol, for each input bit we use at most the commitment containing the bit, and the proof of contents done w.r.t. this bit.

For both commitments schemes, we have to choose the parameter l such that either factoring or discrete log is infeasible, which means that l should be $700 - 1000$. To ensure a chance of at most 1 in a billion of cheating it suffices that $k = 30$, so we reasonably assume that $k \leq l$ in any practical situation.

From the description of the scheme from Section 6.1 based on factoring, we see that the proof of contents requires communication equivalent to two commitments, plus a challenge value that will allways be of length at most 1 commitment (in fact usually much shorter). One proof of contents therefore needs communication corresponding to at most 3 commitments.

From the description of the scheme from Section 6.2 based on discrete log, we see that the proof of contents communicates numbers corresponding to at most 5l bits (3 numbers plus 1 challenge).

For both the commitment schemes, it is clear that (P, V) will only need 1 iteration of (A', B'). Thus we have:

Proposition B.1 *Suppose the protocols from Theorem 4.1, resp. 5.1 are executed using the commitment schemes from Section 6.1, resp. 6.2, then*

assuming that $l \geq k$, the communication complexities will be at most $4n + k + 1$ commitments, resp. $5nl + 5l$ bits, where n is the number of times a Boolean formula for verifying an NP witness for L reads an input variable.

A simple computation shows that with for example about 3 Mbyte of communication, and using $k = 50, l = 768$, n can be up to about 10.000. This might be enough to prove e.g. that you know a DES key encrypting a given cleartext block to a given ciphertext block.

In a similar case, the short discreet proofs of Boyar and Peralta [2] use significantly less communication, about 0.3 Mbyte, but much more computation - a factor of very roughly $k/10 \log n$ more than our protocol[9]. However, it is currently an open problem to prove anything about the soundness and "discreteness" of the proofs from [2], even assuming hardness of factoring or discrete log. Thus the factor 10 in communication seems to be the price we currently have to pay for provable security.

We note that in general, our protocol may be significantly optimized by building ad hoc as small a formula Φ as possible for the problem. Furthermore it may be possible to find a smaller monotone formula computing the same function as the one constructed directly from Φ.

For the interactive proofs, k "extra" commitments are needed to make the proof simulatable against a general verifier. Some of these can be saved by comitting to a logarithmic number of bits in one commitment, thus doing the two-party coinflip on a logarithmic number of bits simultaneously.

[9] This is due to the fact that many randomly generated commitments must be used for every gate. These commitments can be generated from a short random seed, and do not have to be communicated. They do have to be computed on by both parties, however.