Submitted to the CRYPTO '95 Conference

Committed Oblivious Transfer

Claude Crépeau
École Normale Supérieure *

Alain Tapp
Université de Montréal †

February 21, 1995

Abstract

This paper presents an *efficient* protocol for "Committed Oblivious Transfer" to perform oblivious transfer on committed bits: suppose Alice is committed to b_0 and b_1 and Bob is committed to c, they both want Bob to be committed to b_c without Alice learning c nor Bob learning $b_{\bar{c}}$. Our protocol, based on the properties of error correcting codes, uses simple Bit Commitment and Oblivious Transfer as black boxes. Consequently the protocol may be implement with or without a computational assumption depending on the kind of Bit Commitment and Oblivious Transfer used by the participants. This protocol is used as a base sub-protocol in some Oblivious Circuit Evaluation and Multi-party Distributed Computation protocols. To transfer one bit out of two, our protocol requires $O(n)$ Oblivious transfers and $O(n)$ operations on Bit Commitment (with XOR), for a fixed security parameter n.

1 Introduction

Committed Oblivious Transfer (COT) is the natural reunion of Oblivious Transfer (one-out-of-two OT) [23] and Bit Commitment (BC). In the beginning of the protocol Alice is committed to $\boxed{b_0}$, $\boxed{b_1}$ and Bob to $\boxed{c}$. At the end Bob will be committed to a bit $\boxed{b} = b_c$ and will know nothing about $b_{\bar{c}}$. Alice learns nothing about c in the protocol.

COT is a very powerful tool that can be used to perform general cryptographic tasks such as Oblivious Circuit Evaluation (or Mental Games) [15, 16], and Distributed Computation [1, 17] or more specific tasks such as Oblivious String Comparison [13, 11].

* Laboratoire d'Informatique de l'École Normale Supérieure, C.N.R.S. URA 1327
45 rue d'Ulm, 75230 Paris Cedex 05, France. e-mail: crepeau@dmi.ens.fr.

† Département IRO, Université de Montréal, C.P. 6128, succursale centre-ville,
Montréal (Québec), Canada H3C 3J7. e-mail: tappa@iro.umontreal.ca.

Oblivious Circuit Evaluation based on generic BC and OT was first introduced at STOC '88 by Kilian [20]. Unfortunately his protocol was not only complicated but also completely inefficient. At EUROCRYPT '89 Crépeau [8] introduced COT under the label "Verifiable Oblivious Transfer" and used it to obtain a simpler protocol for OCE based on the work of Goldreich and Vanish [16]. Unfortunately his protocol for COT was still rather inefficient (used $\Omega(n^3)$ OTs). An apparently more efficient protocol (using only $O(n)$ OTs) was presented in [17] under the labels "Preprocess-Oblivious-Transfer" but unfortunately it was very easy to misbehave in that protocol in a way that allows Bob to get both bits with non-negligeable probability and for Alice to learn c with a non-negligeable probability. (Please consult the Appendix for more details about this protocol.) These problems can be fixed easily by straightforward techniques but then the protocol becomes more or less equivalent to that of [8] using $\Omega(n^3)$ OTs. The best that the authors were able to get from protocols in the spirit of [8] and [17] is a protocol using $\Omega(n^2)$ OTs.

The current paper presents an *efficient* COT protocol. This protocol makes no assumption on the type of Bit Commitment and Oblivious Transfer used. For example with OT and BC based on the Quantum Channel [2, 5] we can perform COT and OCE without any computational assumption! Our protocol uses some elements of coding theory and simple Zero-Knowledge sub-protocols. It uses only $O(n)$ OTs, $O(n)$ BCs if they have a special XOR property and $O(n^2)$ BCs otherwise. The global running time is $O(n)$ in the first case and $O(n^2)$ in the second (excluding the time necessary for constructing the code).

2 Preliminaries

2.1 Bit Commitment

Bit Commitment, BC, is a two-party protocol, in which Alice sends a bit to Bob in such a way that she will be able to reveal it later in a UNIQUE way but Bob is NOT ABLE TO FIND its value by himself.

It is a well known fact that from an information theoretical point of view, BC is impossible without making an additional assumption. We now review some of the most important results in that area. Obviously, BC can be implemented with the help of a trusted third party. It can be implemented without a trusted third party under the assumption of the existence of a one-way function (or equivalently a pseudorandom bit generator) by a result of Naor [22]. A result of Brassard, Crépeau, Jozsa and Langlois shows that it may be implemented with a Quantum Channel. The existence of a noisy channel between the two participants is sufficient to implement BC [9]. These last two methods make no computational assumption.

The implementation of Bit Commitment based on the Quadratic Residuosity Assumption [4] provides us with a remarkable property: the creator of a set of commitments may prove that a number of them XOR to 0 or 1 without revealing any of them. We call BCX (Bits Commitment with XOR) the bit commitments with this property. All the bit commitment used in our protocol are assumed of that kind.

This property is useful because our protocol uses simple Zero-Knowledge proofs to show the parity of some set of BCs. Although some BC are in fact already BCX, this is not the case in general. Therefore we present a procedure to transform ordinary BC into BCX. Our technique is simple and uses a linear number of conventional BC. It is a generalization of a technique put forward by Charles H. Bennett and Steven Rudich who was first used in [18].

2.2 Bit Commitment with XOR proofs

In this Section we briefly explain how to create BCXs when only BCs are available. A BCX have properties of ordinary BC, namely binding and hiding, but in addition it is possible to perform copies and parity proofs on it. This simple construction based on ordinary BC was first presented in [19] and is more extensively explained in [1], the interested reader should consult these articles for more details. We only describe here the basic properties that we need for our protocol.

To commit to b with B via a BCX $\boxed{\boxed{b}}$ Alice make n pairs $(\boxed{b_{iL}}, \boxed{b_{iR}})$ of randomly chosen BCs such that $b_{iL} \oplus b_{iR} = b$. To open a BCX $\boxed{\boxed{b}}$ Alice simply opens all the BCs that constitutes it. Bob accepts only if $b_{iL} \oplus b_{iR} = b$, for $1 \leq i \leq n$.

Suppose Bob wants to be convinced by Alice that $\boxed{\boxed{b^1}} \oplus \boxed{\boxed{b^2}} \oplus ... \oplus \boxed{\boxed{b^k}} = c$. Bob challenges Alice n times in such a way that if the statement is false he would have overwhelming probability of catching her. Each challenge is of the following form: Bob chooses at random one unused pair α_i in each BCX and asks Alice to reveal $c_L = b^1_{\alpha_1 L} \oplus b^2_{\alpha_2 L} \oplus ... \oplus b^k_{\alpha_k L}$ (notice that $c_R = c \oplus c_L$ can be computed as well since c is public) without opening any commitment and then Bob chooses at random to ask Alice to open all $\boxed{b^k_{\alpha_i L}}$ or $\boxed{b^k_{\alpha_i R}}$ and verify according to the challenge that $c_L = b^1_{\alpha_1 L} \oplus b^2_{\alpha_2 L} \oplus ... \oplus b^k_{\alpha_k L}$ in the first case, or $c_R = b^1_{\alpha_1 R} \oplus b^2_{\alpha_2 R} \oplus ... \oplus b^k_{\alpha_k R}$, in the second case. Bob accepts if Alice met all the challenges. One can verify that if all BCX are not well constructed or if they do not have the parity c as claimed by Alice, Bob accepts with exponential small probability.

The parity proof procedure has the inconvenient of destroying the BCX involved. This is why we need to make copies of BCX. For Alice to make a copy of a BCX $\boxed{\boxed{b}}$ she simply has to make $3n$ additional pairs $(\boxed{b_{iL}}, \boxed{b_{iR}})$ for which $b_{iL} \oplus b_{iR} = b$. Then Bob randomly separates these $3n$ pairs to form $\boxed{\boxed{b^0}}$, $\boxed{\boxed{b^1}}$ and $\boxed{\boxed{b^2}}$. Bob keeps $\boxed{\boxed{b^1}}$, $\boxed{\boxed{b^2}}$ and asks Alice to prove that $\boxed{\boxed{b}} \oplus \boxed{\boxed{b^0}} = 0$ by the above proof. If the proof succeed the two BCX still intact at the end of the interaction are equal to the original BCX with exponential probabilities.

2.3 One-out-of-two Oblivious Transfer

One-out-of-two Oblivious Transfer, $\binom{1}{2}-OT$, is a well known protocol in which two parties exchange information in an asymmetric manner. In a $\binom{1}{2}-OT$ from Alice to Bob, Alice knows two bits b_0 and b_1, she agrees to let Bob know one of them but not both and Bob does not want Alice to know which bits he learns.

As for BC we have to make some assumption to implement $\binom{1}{2}-OT$. Fortunately in many cases the assumptions under which we can perform BC are sufficient to perform $\binom{1}{2}-OT$. It may be implemented if a noisy channel exists between the participant [10], if a Quantum Channel exists [2] or with a computational assumption [15], and in particular under the Quadratic Residuosity Assumption [6]. In general, it is possible to obtain a Bit Commitment protocol from a $\binom{1}{2}-OT$, but not the other way around.

2.4 Coding Theory

Codes are used to perform COT. A few concepts of coding theory are thus needed to understand the protocol. Unfortunately the proof of security uses Coding theory and Information theory in a more subtle manner. For more information on codes consult [21].

In a COT from Alice to Bob, Bob must choose a code. The code is not used to correct transmission errors (although if necessary it could) but in a more elaborate way for efficiency and security reasons.

Let σ and ϵ be some positive constants. Bob must choose an $[n,k,d]$ code for which $k > (1/2+2\sigma)n$ and $d > \epsilon n$. An efficient decoding algorithm must exist that enable him to correct errors if necessary. The number of errors that the decoding algorithm may correct will affect the efficiency of the protocol but correcting a number of errors (t) proportional to d is sufficient to guarantee the asymptotic security of the protocol.

Although Concatenated codes [14] are most common codes with these properties we recommend using the brand new *Superconcentrator Codes* of Spielman [24] for their remarkable efficiency: they can be coded and decoded in linear time. In our protocol, we need to construct codewords, decode codewords and prove that some words are codewords. All these operations can achieved in time $O(n)$. In particular, to prove that a committed word is a codeword it is enough to prove a linear number of statements each involving only a constant number of committed bits XORed together.

3 Committed Oblivious Transfer

Suppose that Alice is committed to $\boxed{b_0}$, $\boxed{b_1}$ and Bob is committed to $\boxed{c}$. After running $COT(\boxed{b_0}, \boxed{b_1})(\boxed{c})$ Bob will be committed to $\boxed{b} = b_c$. Alice, whatever she does, cannot use the protocol to learn information on c and Bob, whatever he does, cannot use the protocol to learn information on $b_{\bar{c}}$, even if he already knew b_c.

3.1 COT Informal Protocol

Intuitively the $COT(\boxed{b_0}, \boxed{b_1})(\boxed{c})$ is performed by doing an imperfect "One-out-of-two Committed Oblivious String Transfer" of random codewords w_0 and w_1. This COST is imperfect in the sense that even the honest Bob learns information about both words. He learns all the bits of w_c and some bits of $w_{\bar{c}}$. We use this trick because Bob is not only interested in getting b_c and committing to $\boxed{b} = b_c$ but he also wants to know that if he executed the protocol with $\neg c$ it would work as well. This is to prevent Alice from cheating by misbehaving on one of $c = 0, 1$ thus learning c as a result of Bob complaining or not.

In the beginning Alice is committed to $\boxed{b_0}$ and $\boxed{b_1}$, Bob is committed to $\boxed{c}$. Bob chooses a code with suitable properties. Alice commits to $\boxed{w_0}$ and $\boxed{w_1}$, two randomly chose codewords and proves that they are. She Obliviously-Transfers w_0 and w_1 to Bob in such a way that for each pair of bits Bob can choose to learn one from w_0 or w_1, but not both. Bob reads the bits to learn mostly w_c and a small part of $w_{\bar{c}}$. He must gain a little bit of information on $w_{\bar{c}}$ to prevent Alice from learning which word he is interested in (and thus c). A cheating Alice could use very different words w_0, w_1 in the $\binom{1}{2} - OT$ and in the BC. To prevent this, Alice opens a small fraction of the committed bits on both sides, at Bob's choosing, to let him check that what he got through the $\binom{1}{2} - OT$ is the same. If Alice cheats a lot for one of the two words (or both) she will be caught without Bob revealing which one he was interest in. If Alice succeed in the test, only a small number of inconsistencies can remain between the bits Bob received and the bits Alice is committed to. Bob uses the fact that his word w_c should be a codeword to correct these.

Bob commits to $\boxed{w} = w_c$ and proves to Alice that it is a codeword. It is now Bob's turn to convince Alice that he is committed to what he actually received through the $\binom{1}{2} - OT$ (he is not committed no a generic codeword, but actually to w_c). Alice opens a random set of positions and Bob shows that the bits he is committed to are consistent with the bits Alice reveals and $\boxed{c}$. This is enough to convince Alice that Bob is committed to w_c, without learning c.

Although Bob is committed to $\boxed{w} = w_c$, he knows some information on $w_{\bar{c}}$ given by the fact that it is a codeword and by the bits opened by Alice in the checks. To get rid of that information, Alice chooses a random privacy amplification function h [3] such that $b_c = h(w_c)$ and $b_{\bar{c}} = h(w_{\bar{c}})$ and proves it to Bob. Bob commits to $\boxed{b} = h(w_c)$ and proves it to Alice.

3.2 $COT(\boxed{b_0},\boxed{b_1})(\boxed{c})$ Formal protocol

Before starting the protocol, Alice is committed to $\boxed{b_0}$, $\boxed{b_1}$ and Bob to $\boxed{c}$. After running the protocol Bob should be committed to $\boxed{b_c}$. Let σ, ϵ be two positive constants. At any point in the protocol, a failure in a proof or a check leads the participants to abort.

Protocol 3.1 ($COT(\boxed{b_0},\boxed{b_1})(\boxed{c})$)

1: Bob chooses and announces to Alice a decodable $[n,k,d]$ linear code C with $k > (1/2 + 2\sigma)n$ and $d > \epsilon n$, efficiently correcting $t \in \Omega(n)$ errors.

2: Alice randomly picks $w_0, w_1 \in C$, commits to $\boxed{w_0^i}$ and $\boxed{w_1^i}$, for $i \in \{1 \ldots n\}$, and proves that $\boxed{w_0^1}\boxed{w_0^2}\ldots\boxed{w_0^n}$ and $\boxed{w_1^1}\boxed{w_1^2}\ldots\boxed{w_1^n}$ are codewords.

3: Bob randomly picks two subsets $I_0, I_1 \subset \{1...n\}$, with $|I_0| = |I_1| = \sigma n, I_1 \cap I_0 = \emptyset$.

4: $\underset{i=1}{\overset{n}{\mathbf{DO}}}$ Alice runs $\binom{2}{1}$–$OT(w_0^i, w_1^i)(c_i)$ with Bob who gets $v_{c_i}^i$, with $c_i = \begin{cases} \bar{c} & \text{if } i \in I_0 \\ c & \text{if } i \notin I_0 \end{cases}$.

5: Bob announces $I = I_0 \cup I_1$ to Alice who opens $\boxed{w_0^i}$ and $\boxed{w_1^i}$, for each $i \in I$.

6: Bob

- checks that $v_0^i = w_0^i$, for $i \in I_0$ and that $v_1^i = w_1^i$, for $i \in I_1$,
- if necessary corrects v_c using C's decoding algorithm,
- commits to $\boxed{v_c^i}$, for $i \in \{1 \ldots n\}$ and proves that $\boxed{v_c^1}\boxed{v_c^2}\ldots\boxed{v_c^n}$ is a codeword.

7: Alice randomly picks and announces a subset $I_2 \subset \{1...n\}$ with $|I_2| = \sigma n, I_2 \cap I = \phi$ and opens $\boxed{w_0^i}$ and $\boxed{w_1^i}$, for $i \in I_2$.

8: Bob proves that $\boxed{v_c^i} = w^i_{\boxed{c}}$, for $i \in I_2$.

9: Alice randomly picks and announces a privacy amplification function $h : \{0,1\}^n \to \{0,1\}$, such that $b_0 = h(w_0), b_1 = h(w_1)$ and proves $\boxed{b_0} = h\left(\boxed{w_0^1}\boxed{w_0^2}\ldots\boxed{w_0^n}\right)$ and $\boxed{b_1} = h\left(\boxed{w_1^1}\boxed{w_1^2}\ldots\boxed{w_1^n}\right)$.

10: Bob computes $b \leftarrow h(v_c)$, commits to $\boxed{b}$ and proves $\boxed{b} = h\left(\boxed{v_c^1}\boxed{v_c^2}\ldots\boxed{v_c^n}\right)$.

3.3 Zero Knowledge proofs

In the formal protocol, Alice and Bob make a number of zero-knowledge proofs. All those proofs are easily achieved if we are able to perform XOR proofs on committed bits. This is why in the protocol we assume BCX and not ordinary BC.

In Steps 2 and 6 a party must show to the other that a set of committed bits form a codeword. In any linear code, this operation can be achieved by showing that the syndrome of the word is the zero-vector. This takes $O(n^2)$ time in general. In the case of Superconcentrator Codes, it is sufficient to prove that each bit is the XOR of a constant number of (publicly known) other bits of the word. Therefore it takes only $O(n)$ time.

In Step 8, Bob must prove to Alice that $\boxed{v^i} = w^i_{\boxed{c}}$, for $i \in I_2$. First, for each position in which $\boxed{w^i_0} = \boxed{w^i_1}$, Bob can simply open $\boxed{v^i_c}$ for in this case v^i_c contains no information on c. For the other positions, where $\boxed{w^i_0} \neq \boxed{w^i_1}$, it must be that w_0 or w_1 is 0 and the other is 1. When $w_0 = 0$ and $w_1 = 1$ we have $w^i_c = c$, and thus Bob proves $w^i_{\boxed{c}} \oplus \boxed{c} = 0$ and in the opposite situation $w^i_{\boxed{c}} = \neg\boxed{c}$, and thus Bob proves $w^i_{\boxed{c}} \oplus \boxed{c} = 1$. Both of these proofs give no information on c and take a total of $O(n)$ time.

In Steps 9 and 10 a party must show to the other that $b = h\left(\boxed{w^1}\boxed{w^2}\cdots\boxed{w^n}\right)$ for some committed bit b and word w. If h is chosen to be a random linear function specified by a public random subset H of the bits of w it defines a universal hash function [7]. In this case, it suffices to show $b \oplus \bigoplus_{i\in H} w^i = 0$ which takes $O(n)$ time.

4 Validity of the Protocol

There is a number of ways in which Alice and Bob may misbehave when they engage in our protocol. At any point Alice or Bob can decide to stop cooperating, they can commit to values different than those used in the oblivious transfer. The important thing is that Bob cannot pretend he is committed to b_c if not and that even if the protocol abort no unintended information leak.

4.1 Bob cannot commit to $b = \bar{b_c}$

If Alice is honest, Bob cannot commit to $\boxed{b} \neq b_c$. Obviously after Bob learns b_c we cannot force him to commit to $b = b_c$. This is not a problem since even if we could, Bit Commitment are such that if Bob is not cooperating after the protocol this BCX $\boxed{b}$ will be as useless as a BCX to a random bit. So the only important point is that if the protocol went well and Bob is commit to a Bit, it must be the good one.

Let's see informally why this is true. Suppose Bob can commit to $\boxed{b} = \bar{b_c}$. This means that he had proved $\boxed{b} = h(\boxed{v_c})$. Since $\boxed{b} \neq b_c$ and $b_c = h(w_c)$ it must be that $\boxed{v_c} \neq w_c$. Bob has proved to Alice that $\boxed{v_c}$ is a codeword therefore the Hamming distance of $\boxed{v_c}$ and w_c is greater than ϵn or equivalently that they have more than ϵn positions where they are different. But Alice asks Bob to open σn positions chosen at random and for all these positions the bits were equal to its counterpart in w_c. This cannot happen except with a probability exponentially small in n.

4.2 Alice learns nothing about c

It is not hard to be convinced that if Alice performs COT honestly she learns nothing about c. The important thing is that regardless of how Alice cheats the protocol she gains no information about c. A straightforward analysis of each step of the protocol shows that Bob sends actual information to Alice only in Steps 1, 5, 6, 8 and 10. In Steps 1 and 5 the information is independent of c. In Steps 6,8 and 10, by definition of BC and the fact that the proofs are Zero-knowledge grantee that Alice learns nothing about c. The only other way Alice may learn information would be to misbehave in such a way that Bob will have to abort if $c = 0$ (or $c = 1$) in steps 7 or 9. This way she could learn c and Bob would catch her only if $c = 1$ (or $c = 0$).

Bob will always succeed in Step 10 since he is responsible for calculating b himself. If Bob fail the proofs at Step 6 or 8, it must be that $\boxed{v_c^1}\boxed{v_c^2}\ldots\boxed{v_c^n} \neq \boxed{w_c^1}\boxed{w_c^2}\ldots\boxed{w_c^n}$. But since w_c is a codeword (as proven in Step 2) this implies that their Hamming distance is at least t because otherwise the decoding algorithm would have lead to w_c. Since $t \in \Omega(n)$, the check of Step 6 has only an exponentially small probability of success. Thus, if the check of Step 6 succeeds, the proofs of Steps 6 and 8 will fail with exponentially small probability whatever c is. The check of Step 6 has the same probability of success whatever c is, since $|I_0| = |I_1|$.

4.3 Bob learns nothing about $b_{\bar{c}}$

Alice does not want a dishonest Bob to learn $b_{\bar{c}}$. In fact it is not sufficient that Bob learns no information about $b_{\bar{c}}$. We want that for all dishonest Bob, even if he knew b_c to start with, the amount of information he will have about $b_{\bar{c}}$ after interacting with Alice will be the same as in the beginning. This is to ensure that, for instance, Bob does not learn $b_c \oplus b_{\bar{c}}$ or any correlation through our protocol.

Note that all the information on b_0 and b_1 is completely uncorrelated. Because Alice picks w_0 and w_1 at random, no information about $b_{\bar{c}}$ is sent before she reveals h. For Bob to learn h he must pass Alice's test of Step 8 and convince her that he is committed to $\boxed{v_c^1}\boxed{v_c^2}\ldots\boxed{v_c^n} = w_c$. If Bob wishes to have no more than $2^{\sigma n/2}$ candidates for w_c by the time he commits at Step 6, he must have acquired on side c at least $n/2 + 3\sigma n/2$ of the $n + 2\sigma n$ bits available to him through Steps 4 and 5. This is the case because there are $2^{n/2+2\sigma n}$ codewords, and thus after learning $n/2 + 3\sigma n/2$ extra bits, $2^{\sigma n/2}$ of them remaining equally possible. If Bob gets $n/2+3\sigma n/2$ bits on side c that leaves him with no more than $n/2 + \sigma n/2$ bits on side $\bar{c}$ of the $n + 2\sigma n$ available. In that case there are also $2^{\sigma n/2}$ candidates for $w_{\bar{c}}$ even after learning σn extra bits at Step 7.

In conclusion, if Bob gains as few as $n/2 + 3\sigma n/2$ bits on side c he has probability no more than $2^{-\sigma n/2}$ of committing to the right codeword at Step 6, and any wrong codeword will have at least ϵn differences with the correct one. The test of Step 8 would have an exponentially small probability in n of succeeding in that case. On the other hand, if Bob gains as few as $n/2 + \sigma n/2$ bits on side $\bar{c}$ he has probability no more than $2^{-\sigma n/2}$ that he has any information whatsoever about $h(w_{\bar{c}})$ when h is chosen at random, by a theorem of Bennett, Brassard and Robert [3].

5 Complexity

The exact complexity of the protocol depends on the kind of BCs and OTs used by the participants. Therefore we make our analysis based on the number of BCs and OTs that are required to perform the protocol. Because all the OTs and proofs can be done in parallel if the OT and BC used require a constant number of communication rounds the COT protocol will also require a constant number of rounds.

The protocol uses $O(n)$ OTs that are perform when w_0 and w_1 are transfered. It also require $O(n)$ BCX for both Alice and Bob to commit to w_0,w_1 and v_c. The overall complexity is $O(n)$ if OT and BCX operations are unitary. Most of the work consist in showing committed words are codeword.

If the BCX are not available one can uses the trick we suggest in Section 2.2. To perform COT, $O(n^2)$ BC will be used to perform the the $O(n)$ necessary operations on the BCX. The overall complexity in term of OT and BC will also be $O(n^2)$ still for the $O(n)$ operation on BCX which require $O(n)$ operation on BC.

5.1 Remarks

Notice that our protocol can be used to transfer general l-bit words instead of single bits. In that case only Steps 9 and 10 would have to be modified in order to use a universal hash function $h : \{0,1\}^n \to \{0,1\}^l$. Thus, the number of elementary OTs used by the protocol remains the same as well as the number of BCs. Only the time necessary to achieve the proofs of these two Steps will be increased by a factor of l.

References

[1] Anonymous , Anonymous, *Secure Distributed Computation on Private Inputs*, Submitted to Advances in Cryptology: Proceedings of Crypto '95, Springer-Verlag, 1995.

[2] C.H. Bennett, G. Brassard, C. Crépeau, M.-H. Skubiszewska, *Practical Quantum Oblivious Transfer*, In proceedings of CRYPTO'91, Lecture Notes in Computer Science, vol 576, Springer Verlag, Berlin, 1992, pp 351–366.

[3] C.H. Bennett, G. Brassard, J.-M. Robert, *Privacy Amplification by Public Discussion*, SIAM Journal on Computing, Vol. 17, No.2, 1988, pp. 210–229.

[4] B. Brassard, C. Crépeau, *Zero-Knowledge Simulation of Boolean Circuits (Extended abstract)*, Advances in Cryptology: Proceedings of Crypto '86, Springer-Verlag, 1987.

[5] G. Brassard, C. Crépeau, R. Jozsa, D. Langlois, *A Quantum Bit Commitment Scheme Provably Unbreakable by both Parties*, 29^{th} Symp. on Found. of Computer Sci.,IEEE,1993.

[6] , G. Brassard, C. Crépeau, J.–M. Robert, *All-or-Nothing Disclosure of Secrets (Extended abstract)*, ,Advances in Cryptology: Proceedings of Crypto '86, Springer-Verlag, 1987.

[7] J. L. Carter, M. N. Wegman, *New Hash Functions and Their Use in Authentication and Set Equality*, Journal of Computer and System Sciences, Vol. 22, 1981, pp. 265–279.

[8] Crépeau, C., *Verifiable Disclosure of Secrets and Applications*, Advances in Cryptology: Proceedings of EUROCRYPT '89,Springer-Verlag, 1990.

[9] Crépeau, C., *Cryptographic Protocols based on Noisy Channel*, Manuscript, 1995.

[10] C. Crépeau, J. Kilian, *Achieving Oblivious Transfer Using Weakened security Assumptions*, FOCS '88, 1988.

[11] C. Crépeau and L. Salvail *Quantum Oblivious Mutual Identification* Advances in Cryptology: Proceedings of EUROCRYPT '85,Springer-Verlag, 1995.

[12] C. Crépeau and M. Sántha. *Efficient reductions among oblivious transfer protocols based on new self-intersecting codes.* In Sequences II, Methods in Communications, Security, and Computer Science, pp. 360–368. Springer-Verlag, 1991.

[13] R. Fagin, M. Naor, and P. Winkler, *Comparing Secret String without Revealing It*, submitted for publication, CACM, 1994.

[14] Forney, G. D., *Concatenated Codes*, The M.I.T. Press, 1966.

[15] O. Goldreich, S. Micali, and A. Wigderson. *How to play any mental game, or: A completeness theorem for protocols with honest majority.* In Proc. 19th ACM Symposium on Theory of Computing, pages 218–229, ACM, New York City, 1987.

[16] O. Goldreich, R.Vainish, *How to Solve any Protocol Problem - An Efficiency Improvement*, Advances in Cryptology CRYPTO '87 Proceedings, Springer-Verlag, 1988.

[17] S. Goldwasser, L. Levin, *Fair Computation of General Functions in Presence of Immoral Majority* Advances in Cryptology CRYPTO '90 Proceedings, Springer-Verlag, 1991.

[18] J. Kilian, *A note on zero-knowledge proofs and arguments*, 24th ANNUAL ACM STOC,1992.

[19] J. Kilian, *On the complexity of bounded-interaction and non-interactive zero-knowledge proofs* FOCS94.

[20] J. Kilian. *Founding cryptography on oblivious transfer.* In Proc. 20th ACM Symposium on Theory of Computing, pp. 20–31, Chicago, 1988. ACM.

[21] F. J. Mac Williams, N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, 1977.

[22] M. Naor, *Bit Commitment using Pseudo-Randomness*, Advances in Cryptology: Proceedings of Crypto '89, Springer-Verlag,1989.

[23] Rabin, M. O., *How to exchange secrets by oblivious transfer*, Technical Memo TR–81, Aiken Computation Laboratory, Harvard University, 1981.

[24] D. Spielman, *Linear-Time Encodable and Decodable Error-Correcting Codes*, 27th ANNUAL ACM STOC,1995.

Appendix: the Protocol of Goldwasser-Levin

A lot of renaming and simplifications have been necessary to put this protocol out of context. Nevertheless the following COT is in essence the same as Preprocess-O-T of [17].

Protocol 5.1 ($COT(\boxed{b_0},\boxed{b_1})(\boxed{c})$)

1: $\mathbf{DO}_{i=1}^{n}$

1. Alice commits to random bits $\boxed{b_0^i}$, $\boxed{b_1^i}$, Bob commits to a random bit $\boxed{c^i}$,
2. Alice runs $\binom{2}{1}$–$OT(b_0^i, b_1^i)(c^i)$ with Bob who gets v^i and commits to $\boxed{v^i}$.

2: Alice and Bob jointly pick a random subset $I \in \{1, ..., n\}$.

3: $\mathbf{DO}_{i \notin I}$ Alice unveils b_0^i, b_1^i, Bob unveils c^i, v^i, and they abort if $b_{c^i}^i \neq v^i$.

4: Bob reveals a subset $I_1 \subseteq I$ such that $c^i = c$ for $i \in I_1$ and proves that $\boxed{c^i} = \boxed{c}$.

5: Alice reveals a subset $I_2 \subseteq I_1$ such that $b_0^i = b_0, b_1^i = b_1$ for $i \in I_2$ and proves that $\boxed{b_0^i} = \boxed{b_0}$, $\boxed{b_1^i} = \boxed{b_1}$.

6: **IF** for any $i, j \in I_2$, $v^i \neq v^j$ **THEN** Bob aborts.

7: Bob commits to his output $\boxed{v}$ and proves for $i \in I_2$ that $\boxed{v^i} = \boxed{v}$.

Alice can cheat

Alice can cheat this protocol by using an improper pair $b_0^i, \boxed{b_0^i}$ in Step 1.1-1.2. If she commits to $\boxed{b_0^i}$ in Step 1.1 and uses the opposite $\neg b_0^i$ in Step 1.2, with probability 1/2 she will not be caught in Step 3 and Bob will later abort at Step 6 if he used $c^i = c = 0$ in Step 1.2 and will not abort otherwise. In this later case Alice deduces $c = 1$.

Bob can cheat

Bob can cheat this protocol by using an improper pair $c^i, \boxed{c^i}$ in Step 1.1-1.2. If he commits to $\boxed{c^i}$ in Step 1.1 and uses the opposite $\neg c^i$ in Step 1.2 he is able to read bit $b_{\neg c^i}^i$. For his commitment in Step 1.2 he puts a random bit and with probability 1/2 will not be caught in Step 3. If he gets $b_{c^i}^i$ in all the other positions he learns both of Alice's bits.