Communication Complexity of Efficient Two-Party Computation Protocols

Abstract

We study the communication complexity of secure two party computation protocols. In cut-and-choose protocols for two party computation, we analyze the optimal parameters to keep the probability of undetected cheating minimum. We first study this for a constant number of circuits, and then generalize it to the case of constant bandwidth. More generally, the communication cost of opening a circuit is different from retaining the circuit for evaluation and we analyze the optimal parameters in this case, by fixing the total bits of communication. We then minimize the communication complexity while still keeping the security guarantees, that is, the cheating probability negligible. We achieve a concrete improvement in communication complexity by using optimal parameters in existing cut-and-choose protocols.
Keywords: secure computation, malicious adversaries, cheating probability, communication complexity.

1 Introduction

Secure two party computation. In the setting of two-party computation, two parties with respective private inputs x and y, wish to jointly compute a functionality $f(x,y) = (f_1(x,y), f_2(x,y))$, such that the first party receives $f_1(x,y)$ and the second party receives $f_2(x,y)$. The security requirements are that nothing is learned from the protocol other than the output (privacy), and that the output is distributed according to the prescribed functionality (correctness). The actual definition follows the simulation paradigm and blends the two requirements. Security must be guaranteed even when one of the parties is adversarial. Such an adversary may be semi-honest, in which case it correctly follows the protocol specification, yet attempts to learn additional information by analyzing the transcript of messages received during the execution. In contrast, the adversary may be malicious, in which case it can arbitrarily deviate from the specifications of the protocol. The first general solutions for the problem of secure computation were presented by Yao [Yao86] for the two-party case with security against semi-honest adversaries and Goldreich, Micali and Wigderson [GMW87] for the multi-party case with security even against malicious adversaries. The results of [GMW87] constitute important and powerful feasibility results for secure two-party and multi-party computation.

Yao's Protocol. Yao presented a constant-round protocol for securely computing any functionality in the presence of semi-honest adversaries. Let f be the functionality that the two parties wish to compute, and let x,y be their respective inputs. (for simplicity, assume that both parties wish to receive $f(x,y)$). Yao's protocol works by having one of the parties, say party P_1 first generate a "garbled" (or encrypted) circuit computing $f(x,\cdot)$ and then send it to P_2. The circuit is such that it reveals nothing in its encrypted form and therefore P_2 learns nothing from this stage. However, P_2 can obtain the output $f(x, y)$ by "decrypting" the circuit. In order to ensure that P_2 learns nothing more than the output itself, this decryption must be partial and must reveal $f(x, y)$ only. That is, P_2 should learn the value on the circuit output wire without learning the values on any of the internal wires. This is accomplished by P_2 obtaining a series of keys corresponding to its input y, such that given these keys and the circuit, the output value $f(x, y)$, and only this value, may be obtained. Now, P_2 must somehow receive these keys without revealing anything about y to P_1. This is accomplished by running secure 1-out-of-2 Oblivious Transfer protocol [Rab81]. A detailed description of Yao's protocol, and a proof of security can be found in [LP04]. Yao's generic protocol is highly efficient, and even practical, for functionalities that have relatively small circuits.

Malicious behavior and cut-and-choose. The compiler of GMW [GMW87] converts any protocol that is secure for semi-honest adversaries into one that is secure for malicious adversaries, and as such is a powerful tool for demonstrating feasibility. However, it is based on reducing the statement that needs to be proved (the honesty of the parties' behavior in this case) to an NP-complete problem, and using generic zero-knowledge proofs to prove this statement. The resulting secure protocol therefore runs in polynomial time but is rather inefficient. Lindell and Pinkas gave an efficient protocol secure against malicious adversaries, based on cut-and-choose methodology in [LP07]. P_1 first constructs many garbled circuits and sends them to P_2. Then, P_2 asks P_1 to "open" half of them (namely, reveal the decryption keys corresponding to these circuits). P_1 opens the requested half, and P_2 checks that they were constructed correctly. If they were, then P_2 evaluates the rest of the circuits and derives the output from them. The idea behind this methodology is that if a malicious P_1 constructs the circuits incorrectly, then P_2 will detect this with high probability. Clearly, this solution solves the problem of P_1 constructing the circuit incorrectly. In [LP07], the authors give a cut-and-choose based solution, and a simulation based proof of security.

Our Results. We study the communication efficiency of protocols for two party computation against malicious adversaries. We analyze the optimal parameters to achieve minimum cheating probability and minimum communication complexity of cut-and-choose protocols. Our analysis yields parameters which can be used in any of the existing cut-and-choose protocols and get an improvement in the communication complexity. We first study the optimal fraction of the total number of circuits that should be check circuits, so as to minimize the undetected probability of cheating by P_1. We then generalize this to the case when a constant bandwidth is allowed (as opposed to constant number of total circuits). In this case, the communication cost of a check circuit is cheaper than the cost of an evaluation circuit [GMS08]. We show the optimal number of check circuits to minimize the cheating probability in this setting. Further, we improve the communication complexity of existing protocols, while still keeping the probability of undetected cheating by P_1 negligible. The parameters we obtain from our analysis can be used to improve the number of communication bits in existing cut-and-choose protocols.

2 Preliminaries

2.1 Cut-and-choose protocol

Lindell and Pinkas gave an efficient two party protocol secure against malicious adversaries [LP07]. Their construction is based on applying cut-and-choose techniques to the original Yao's circuit and inputs. Security is proved in the ideal/real simulation paradigm.

A malicious P_1 is forced to construct the garbled circuit correctly so that it indeed computes the desired function. According to the cut-and-choose methodology, P_1 constructs many independent copies of the garbled circuit and sends them to P_2. Party P_2 randomly chooses half of them, and asks P_1 to open the chosen circuits. Now, P_2 checks that the opened circuits are constructed correctly. If they are, then P_2 is convinced that most of the remaining garbled circuits are also constructed correctly. If there are many circuits that are incorrectly constructed, then with high probability, one of those circuits will be in the set that P_2 asks to open. The parties then evaluate the remaining circuits as in the original protocol for semi-honest adversaries, and take the majority output. The protocol also has to make sure that both P_1 and P_2 use the same inputs in each circuit. Such consistency checks are important since if the parties were able to provide different inputs to different copies of the circuit, then they can learn information that is more than just the desired output of the function. This is ensured by having P_1 commit to the garbled circuits and also to the garbled values corresponding to the input wires of the circuits. We give a high-level overview of the protocol here.

Parties P_1 and P_2 have respective inputs x and y, and wish to compute $f(x, y)$.

- The parties decide on a circuit computing f. They then change the circuit by replacing each input wire of P_2 by a gate whose input consists of s new input wires of P_2 and whose output is the exclusive-or of these wires (such an s-bit exclusive-or gate can be implemented using $s-1$ two-bit exclusive-or gates). Consequently, the number of input wires of P_2 increases by a factor of s.

- P_1 commits to s different garbled circuits computing f , where s is a statistical security parameter. P_1 also generates additional commitments to the garbled values corresponding to the input wires of the circuits. These commitments are constructed in a special way in order to enable consistency checks.

- For every input bit of P_2, parties P_1 and P_2 run a 1-out-of-2 oblivious transfer protocol in which P_2 learns the garbled values of input wires corresponding to its input.

- P_1 sends all the commitments to P_2.

- P_1 and P_2 run a coin-tossing protocol in order to choose a random string that defines which commitments and garbled circuits will be opened.

- P_1 opens the garbled circuits and commitments chosen in the previous step. P_2 verifies the correctness and runs the consistency checks based on decommitted input values.

- P_1 sends the garbled values corresponding to P_1's input wires in the unopened garbled circuits, P_2 runs consistency checks on these values.

- If all the checks pass, P_2 evaluates the unopened circuits, and takes the majority value as output.

2.2 Efficient two party computation protocols

In [GMS08], the authors design an efficient multi party computation protocol in the covert adversary model. The techniques used in the two party case generalize to improve the communication efficiency of two party computation protocols secure against standard malicious adversaries. To achieve an improvement in communication complexity, they take a different approach to constructing the garbled circuit. In order to construct a garbled circuit and the commitments for input keys, party P_1 generates a short random seed and feeds it to a pseudorandom generator in order to generate the necessary randomness. He then uses the randomness to construct the garbled circuit and the necessary commitments. When the protocol starts, party P_1 sends to P_2 only a hash of each garbled circuit using a collision-resistant hash function. P_2 chooses half of the circuits at random. In order to expose the secrets of each circuit, P_1 sends the seeds corresponding to the circuit, and not the whole opened circuit. Once the checks go through, P_1 sends the remaining circuits, called the evaluation circuits to P_2.

We briefly give the garbling procedure of [GMS08]. Let G be the description of a pseudorandom generator, *seed* a seed of suitable length, C be the description of the circuit to be garbled. $Garble(G, seed, C, 1^s)$ denotes the garbling procedure where s is the security parameter. The randomness required for constructing the garbled circuits includes, the random keys corresponding to the wires, the random permutation chosen for each gate, and the random string chosen for encryption. A random string of length $O(s|C|)$ is sufficient for garbling. A pseudorandom generator $G : \{0,1\}^{n_1} \rightarrow \{0,1\}^{n_2}$, where n_1 is polynomial in s and $n_2 = O(s|C|)$ is used. The algorithm *Garble* runs G on *seed* to generate the randomness required for the circuit, and then computes the garbled circuit for C as described in [LP04]. The details of [GMS08] is given in Appendix A.1.

3 Optimal number of check circuits

In the cut-and-choose protocol described in 2.1, we have the flexibility of choosing the number of check circuits. This is the subject of this section; we study what is the optimal number of circuits that should be check circuits to minimize the probability of undetected cheating by Party P_1. In section 3.1, we keep the total number of garbled circuits a constant, and minimize the cheating probability. In the next subsection, we generalize this to the case when a constant amount of communication bits is allowed, (and the total number of circuits is not fixed). We minimize the cheating probability in these settings by choosing the optimal number of check circuits, c. In [ShS11], the authors show the optimal number of check circuits, but our techniques to get an approximation for the expression of cheating probability makes the analysis extendible to the more general case of cheaper check circuits. We discuss both the cases, of same cost circuits and cheaper check circuits as the latter is an extension of the computations done for the former case.

3.1 Constant number of total circuits

Let n be the number of garbled circuits constructed by P_1, and let c be the number of circuits checked.
Assume that i out of n circuits are constructed incorrectly by a cheating P_1. P_1's cheating is not caught if all the check circuits are constructed correctly, and P_2 does not abort after evaluation of the remaining circuits.
Now, the probability of P_1's cheating not caught is given by,

$$\frac{\binom{n-i}{c}}{\binom{n}{c}}$$

If $i < \dfrac{n-c}{2}$, then the majority output is correct, and if $i > n-c$, corrupt P_1 is caught in one of the check circuits.

Therefore, the cheating probability is,

$$\begin{aligned} P &= \max_i \frac{\binom{n-i}{c}}{\binom{n}{c}} \\ &= \frac{1}{\binom{n}{c}} \max_i \binom{n-i}{c} \end{aligned}$$

The above is maximum for $i = \dfrac{n-c}{2}$. Thus,

$$\begin{aligned} P &= \frac{\binom{\frac{n+c}{2}}{c}}{\binom{n}{c}} \\ &= \frac{\left(\frac{n+c}{2}\right)!}{c!\left(\frac{n-c}{2}\right)!} \frac{(n-c)!c!}{n!} \\ &= \frac{\left(\frac{n+c}{2}\right)!}{\left(\frac{n-c}{2}\right)!} \frac{(n-c)!}{n!} \end{aligned}$$

By Stirling's approximation,

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

$$P = \frac{(n+c)^{\frac{n+c+1}{2}}(n-c)^{\frac{n-c}{2}}}{2^c\, n^{n+\frac{1}{2}}} \tag{1}$$

To minimize P for a given n, we differentiate partially with respect to c and set the resulting expression to 0.

Taking logarithm of (1), we get,

$$\log P = \left(\frac{n+c+1}{2}\right)\log(n+c) + \frac{n-c}{2}\log(n-c) - c\log 2 - \left(n+\frac{1}{2}\right)\log n$$

Now differentiating,

$$\frac{dP}{dc} = \frac{P}{2}\left(\log\frac{n+c}{4(n-c)} - \frac{n-c}{n-c} + \frac{n+c+1}{n+c}\right)$$

We now have,

$$\frac{dP}{dc} = 0$$

$$\log \frac{n+c}{4(n-c)} - \frac{n-c}{n-c} + \frac{n+c+1}{n+c} = 0$$

$\frac{n+c+1}{n+c} \approx 1$ and cancels out $\frac{n-c}{n-c}$. Therefore, we have,

$$\log \frac{n+c}{4(n-c)} = 0$$

$$\frac{n+c}{4(n-c)} = 1$$

$$\Rightarrow n + c = 4(n-c)$$

This yields,

$$c = \frac{3n}{5}$$

Therefore, for a given number of total circuits, n, minimum cheating probability is achieved when the number of check circuits is 3/5th of n.

Theorem 1. *For a given total number of garbled circuits n, sent by P_1 in the cut-and-choose protocol, P_2 should ask $\frac{3}{5}$th of them to be opened, to minimize the probability of cheating by P_1.*

The above result is also obtained in [ShS11], by counting the number of bad circuits that optimizes the cheating probability. They do not derive an explicit expression for the cheating probability as we do above, which we use in the generalization to cheaper check circuits which is the subject of the next section.

3.2 Cheaper check circuits

In this section, we fix the amount of communication bits allowed. A given number of communication bits does not fix the total number of circuits when the costs of a check circuit and an evaluation circuit are different. Consider the protocol of [GMS08], i.e applying the ideas of [GMS08] to the protocol of the previous section. In this protocol, P_1 uses a short seed and a pseudorandom generator to generate the required randomness for construction of the garbled circuits. A hash of the n garbled circuits are sent to P_2. P_2 asks for a random half of them to be opened, and P_1 sends only the short seeds of the selected half. P_2 verifies that they are constructed correctly. The evaluation circuits are now sent by P_1. The cost of a check circuit is therefore, less than the cost of an evaluation circuit; once the hashes of all n circuits are sent, only a short seed is communicated in case of a check circuit, whereas the whole garbled circuit is the communication cost for an evaluation circuit. In this case, we analyze the optimal number of check circuits for minimum cheating probability. In the previous case when the costs of a check circuit and an evaluation circuit are the same, fixing the communication bits k, also fixes the total number of circuits n. We now study the general case, for a given amount of communication bits when a check circuit is cheaper than an evaluation circuit. Given a constant amount of communication bits, we minimize the cheating probability.

Let k be the number of bits of communication allowed. Let c be the number of check circuits, e the number of evaluation circuits and n the total number of circuits.

Then,

$$c \cdot Cost_{check} + e \cdot Cost_{eval} = k$$

Let q be the ratio of the cost of check circuits to the cost of evaluation circuits.

$$cq + e = s$$

where,

$$q = \frac{Cost_{check}}{Cost_{eval}}$$

and,

$$s = \frac{k}{Cost_{eval}}$$

Using $n = c + e$ and $e = s - cq$ in the cheating probability, we have,

$$\begin{aligned} P &= \frac{(n+c)^{\frac{n+c+1}{2}} \, (n-c)^{\frac{n-c}{2}}}{2^c \, n^{n+\frac{1}{2}}} \\ &= \frac{(2c+e)^{\frac{2c+e+1}{2}} \, e^{\frac{e}{2}}}{2^c \, (e+c)^{e+c+\frac{1}{2}}} \\ &= \frac{(2c+s-cq)^{\frac{2c+s-cq+1}{2}} \, (s-cq)^{\frac{s-cq}{2}}}{2^c \, (s-cq+c)^{s-cq+c+\frac{1}{2}}} \end{aligned} \tag{2}$$

We now differentiate with respect to c and equate the resulting expression to zero.

Taking logarithm of (2) we get,

$$\log P = \frac{(2-q)c+s+1}{2} \log\left((2-q)c+s\right) + \frac{s-cq}{2} \log\left(s-cq\right) - c\log 2 - \left(c(1-q)+s+\frac{1}{2}\right) \log\left(c(1-q)+s\right)$$

Differentiating with respect to c we have,

$$\frac{1}{P}\frac{dP}{dc} = \frac{2-q}{2} \log\left((2-q)c+s\right) - \frac{q}{2} \log(s-qc) - \log 2 - (1-q) \log\left((1-q)c+s\right)$$

Now setting the derivative to zero,

$$\frac{dP}{dc} = 0$$

$$\log\left(\frac{\left((2-q)c+s\right)^{\frac{2-q}{2}}}{2(s-qc)^{\frac{q}{2}} \left((1-q)c+s\right)^{1-q}}\right) = 0$$

$$\frac{((2-q)c+s)^{\frac{2-q}{2}}}{2(s-qc)^{\frac{q}{2}}\,((1-q)c+s)^{1-q}} = 1$$

This implies,

$$\begin{align} ((2-q)c+s)^{\frac{2-q}{2}} &= 2(s-qc)^{\frac{q}{2}}\,((1-q)c+s)^{1-q} \tag{3}\\ (2c+e)^{1-\frac{q}{2}} &= 2\,e^{\frac{q}{2}}\,(c+e)^{1-q} \notag\\ (n+c)^{1-\frac{q}{2}} &= 2\,(n-c)^{\frac{q}{2}}\,n^{1-q} \notag\\ \left(1+\frac{c}{n}\right)^{2-q} &= 4\left(1-\frac{c}{n}\right)^{q} \tag{4} \end{align}$$

Let r be the fraction of the circuits which are check circuits. i.e

$$r = \frac{c}{n}$$

Using this in (4) yields,

$$\begin{align} (1+r)^{2-q} &= 4\;(1-r)^{q} \notag\\ (1+r)^{2} &= 4\;\left(1-r^{2}\right)^{q} \tag{5} \end{align}$$

Given q, the ratio of costs, we can solve the above equation for r.

The total number of circuits to be sent n is then given by,

$$n = \frac{k}{(1-r)\,Cost_{eval} + r\,Cost_{check}}$$

Given that we are allowed a constant k bits of communication, the total number of circuits, and the number of check circuits can be set as in the above analysis to minimize the cheating probability by P_1. If the cost of check circuit is the same as the cost of the evaluation circuit,
i.e when,

$$Cost_{check} = Cost_{eval}, q = 1$$

Setting $q = 1$ in equation (5) yields,

$$r = \frac{3}{5}$$

and this agrees with our earlier conclusion when the costs are same.

Theorem 2. *Let q be the ratio of the communication cost of a check circuit to the cost of an evaluation circuit in a two party cut-and-choose protocol, and k, a constant amount of communication bits allowed. Then, probability of cheating by P_1 is minimized if the ratio of the number of check circuits to the number of evaluation circuits, r is as given by, $(1+r)^2 = 4\;\left(1-r^2\right)^q$, and the number of circuits, $n = \dfrac{k}{(1-r)\,Cost_{eval} + r\,Cost_{check}}$.*

4 Communication Complexity

In this section, we minimize the number of communication bits for a given cheating probability. We show that, for our choice of parameters, the communication complexity of existing cut-and-choose protocols can be improved. The communication complexity of existing protocols are stated in the following theorems.

Theorem 3. *([Woo07]) Let n be the number of circuits, and g the number of gates in the circuit. The protocol of [Woo07] is secure in the malicious model with inverse exponential (in n) probability of undetected cheating. The communication complexity is $O(ng)$.*

Theorem 4. *([MF06]) The equality-checker protocol of [MF06] is secure in the malicious model with probability of undetected cheating ϵ. If g is the number of gates the circuit, and I the number of input bits, the communication complexity of the scheme is $O(ln\left(\frac{1}{\epsilon}\right)g + ln\left(\frac{1}{\epsilon}\right)^2 I)$.*

4.1 Minimize Communication Complexity

We now formulate the problem as minimizing the communication complexity which is a function of two variables, given a cheating probability. Given a cheating probability p, for what relation between the check circuits and the total number of circuits, is p achieved in minimum number of communication bits? Minimize the function,

$$f(c,n) = c \cdot Cost_{check} + (n-c) \cdot Cost_{eval}$$

That is, minimize,

$$k = c + (n-c)Q$$

subject to the constraint that,

$$p = \frac{1}{2^c} \frac{(n+c)^{\frac{n+c+1}{2}} (n-c)^{\frac{n-c}{2}}}{n^{n+\frac{1}{2}}}$$

where,

$$Q = \frac{Cost_{eval}}{Cost_{check}}$$

Since the constraint equation is exponential, we go back to the expression of cheating probability and try to get a more friendly approximation.

$$p \approx \frac{\binom{\frac{n+c}{2}}{c}}{\binom{n}{c}}$$

We know that,

$$\binom{x}{y} \geq \left(\frac{x}{y}\right)^y$$

$$\binom{x}{y} \leq \frac{x^y}{y!} \leq \frac{x^y}{2^{y-1}}$$

Therefore,

$$p \geq \frac{\left(\frac{n+c}{2c}\right)^c}{\frac{n^c}{2^{c-1}}} = \left(\frac{n+c}{cn}\right)^c \frac{2^{c-1}}{2^c}$$

$$p \geq \frac{1}{2}\left(\frac{1}{c}+\frac{1}{n}\right)^{c}$$

$$(2p)^{\frac{1}{c}} \geq \frac{1}{c}+\frac{1}{n}$$

$$\frac{1}{n} \leq (2p)^{\frac{1}{c}} - \frac{1}{c}$$

$$n \geq \frac{c}{c\,(2p)^{\frac{1}{c}} - 1}$$

Since we do not get a closed form for c in terms of n and p, we investigate, of what order c should be in, so that k is minimized while keeping p negligible.

Let $n-c=n^{\epsilon}$, and, $\epsilon = \dfrac{\log\log n}{\log n}$
For this value of ϵ, $n^{\epsilon}=\log n$, and $n-n^{\epsilon}=c=n-\log n$.

The communication bits, $k \approx n^{\epsilon}Q$, and it is easy to verify that for the above value of c, p is still negligible. We now show that the cheating probability p is negligible when $c=n-\log n$.

$$p = \frac{1}{2^{c}}\,\frac{(n+c)^{\frac{n+c+1}{2}}\,(n-c)^{\frac{n-c}{2}}}{n^{n+\frac{1}{2}}}$$

For $c=n-\log n$,

$$p = \frac{1}{2^{n-\log n}}\,\frac{(2n-\log n)^{\frac{2n-\log n+1}{2}}\,(\log n)^{\frac{\log n}{2}}}{n^{n+\frac{1}{2}}}$$

For large n, and $c=n-n^{\epsilon}$, p is still negligible. That is, $p \approx \dfrac{1}{n^{\frac{1}{2}+\frac{n^{\epsilon}}{2}}}$.
We observe that, the cheating probability p remains negligible, for $\epsilon = \dfrac{\log^{*} n}{\log n}$, giving communication bits, $k=O(n^{\epsilon}Q)$. Therefore, we achieve minimum communication complexity, keeping the cheating probability negligible, by choosing the number of check circuits to be, $c=n-n^{\epsilon}$, for $\epsilon = \dfrac{\log^{*} n}{\log n}$.

Thus, the communication cost of existing cut-and-choose protocols can be improved, while still retaining the security guarantees, i.e. keeping the cheating probability negligible. The communication complexity and the cheating probability achieved by the above optimal parameters in a cut-and-choose protocol are stated in the following theorem.

Theorem 5. *Let* $\epsilon = \dfrac{\log\log n}{\log n}$*, and* Q *be the ratio of the cost of an evaluation circuit to the cost of a check circuit. For the total number of circuits* n*, and number of check circuits* $c=n-n^{\epsilon}$*, we achieve a communication cost of* $O(n^{\epsilon}Q)$*. The cheating probability* $p \approx \dfrac{1}{n^{\frac{1}{2}+\frac{n^{\epsilon}}{2}}}$*, remains negligible for the above choice of* ϵ*.*

The improvement by using the above parameters in the setting of [GMS08], is discussed in the following section. Using the above choice of parameters for ϵ, the number of check circuits c and the technique of [GMS08] as applied to cut-and-choose protocol for general secure two party computation, we get an improvement in the communication complexity in concrete terms compared to the protocols of [MF06], [Woo07], [LP11].

4.2 Comparison of communication bits

In the previous section, we analyzed in detail the parameter values such as the number of circuits to be challenged and arrived at optimal values to achieve minimum communication complexity. We now show how the use of these optimal parameters in existing protocols significantly improve the communication complexity. In particular, we sketch how our optimal parameters along with the technique of [GMS08], improve the efficiency of existing cut-and-choose constructions. Informally, in the setting of [GMS08], P_1 sends only a hash of the check circuits, and the entire garbled circuit is sent only for an evaluation circuit. Check circuits are therefore cheaper than evaluation circuits. Let the total number of circuits be n. From the analysis of section 4.1, we choose the number of evaluation circuits to be $\log n$. We summarize our results below.

Let the size of a garbled circuit be, $|GC| = 4g|E|$, where $|E|$ is the size of the ciphertext of the encryption scheme, g the number of gates in the circuit and I the input size. Let $|E|$ also be the output size of the commitment scheme. If t is the number of garbled circuits in [MF06], the number of bits communicated is $t|GC| + t^2I|E| = 4tg|E| + t^2I|E|$ (Theorem 4). The same protocol, by using the parameters of our analysis communicates, roughly, $4g|E|\log n + (\log n)^2I|E|$ bits, where n is the total number of circuits (Theorem 5). Consider a circuit with 32 gates and input wires, $g = I = 32$. The cheating probabilty and the communication complexity achieved by [MF06] and by incorporating the optimal choice from our analysis in [MF06] is shown in the table below.

Table 1:

Scheme	Communication Complexity	Cheating Probability
[MF06]	$128t|E| + 32t^2|E|$	$2.\,(1/2)^{t/4}$
This paper	$128|E|\log n + 32(\log n)^2|E|$	$1/n^{\frac{1}{2}+\frac{\log n}{2}}$

For the purpose of our comparison, we fix the cheating probability. Setting the number of circuits t and n, in the two variants such that, the cheating probability is the same, say, 2^{-50}, we get $t = 196$ and $\log n = 10$. Substituting the values of t and n in the number of communication bits shown above, we see that the communication bits using our optimal parameters is around 250 times less than the communication bits of [MF06]. It is also important to note that this factor of improvement increases as the number of gates and input wires in the circuit increase.

We now consider the protocol of [Woo07]. If t is the number of garbled circuits in [Woo07], the number of bits communicated is roughly $4tg|E| + 2tI|E| + tg + tI|E|$. The number of bits communicated by using our techniques is, roughly, $4g|E|\log n + 3\log nI|E|$ (Theorem 5), where n is the total number of circuits. If we consider a circuit with 32 gates and input wires, $g = I = 32$, and set the parameters t and n such that the cheating probability is the same, $t = \dfrac{\log n}{2} + \dfrac{(\log n)^2}{2}$. Now, for a cheating probability of 2^{-50}, we get $t = 50$ and $\log n = 10$. Substituting the values of t and n in the number of communication bits shown above, we see that the communication bits using our optimal parameters is a factor of 9 less than the scheme of [Woo07].

More recently, in [LP11], Lindell and Pinkas presented a protocol using the cut-and-choose methodology relying on DDH assumption, and significantly improved the efficiency. We briefly compare the communication bits of our techniques with that in [LP11]. The communication complexity of [LP11] is the exchange of $5tI + 14I + 7t + 5$ group elements and t copies of the garbled circuit. The cheating probability is $\frac{1}{2^{t/4}}$. The communication cost is dominated by the garbled circuits. Using the techniques in this paper, the communication cost of sending the garbled circuits is $4 \log n g|E|$, for a cheating probability of $\frac{1}{n^{\frac{1}{2}+\frac{\log n}{2}}}$.

Table 2:

Scheme	Cheating Probability	Communication Complexity
[LP11]	$1/2^{t/4}$	$O(tg)$
This paper	$1/n^{\frac{1}{2}+\frac{\log n}{2}}$	$O(g \log n)$

To achieve the same cheating probability, say 2^{-50} we compute the number of GC's required as $t = 200$. We now set $n^{\frac{1}{2}+\frac{\log n}{2}} = 2^{-50}$, and solve the quadratic equation to get $\log n = 10$. This gives a factor of 20 improvement in the communication cost. Thus, for the same cheating probability, our techniques lead to a factor of 20 less communication between the parties compared to [LP11]. We also remark that the increase in computational complexity is feasible. $\log n = 10$ implies that the total number of GC's the parties need, $n = 2^{10} = 1024$.

References

[GMS08] Vipul Goyal, Payman Mohassel, and Adam Smith. Efficient two party and multi party computation against covert adversaries. In *EUROCRYPT 2008*, pages 289–306, 2008.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. *In proceedings of 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, 1987.

[LP04] Yehuda Lindell and Benny Pinkas. A proof of yao's protocol for secure two-party computation. Cryptology ePrint Archive, Report 2004/175, 2004.

[LP07] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *EUROCRYPT*, pages 52–78, 2007.

[LP11] Yehuda Lindell and Benny Pinkas. Secure two-party computation via cut-and-choose oblivious transfer. In *8th TCC*, pages 329–346, 2011.

[MF06] Payman Mohassel and Mathew Franklin. Efficiency tradeoffs for malicious two party computation. In *Public Key Cryptography Conference*, pages 458–473, 2006.

[Rab81] M. Rabin. How to exhange secrets by oblivious transfer. Technical Memo, TR-81, Aiken computation laboratory, Harvard U., 1981.

[ShS11] Abhi Shelat and Chih hao Shen. Two-output secure computation with malicious adversaries. In *Advances in Cryptology - Eurocrypt 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 386–405. Springer, 2011.

[Woo07] David P. Woodruff. Revisiting the efficiency of malicious two party computation. In *EUROCRYPT 2007*, pages 79–96, 2007.

[Yao86] A. C. Yao. How to generate and exchange secrets. In *FOCS '86: Proceedings of 27th Annual Symposium on Foundations of Computer Science*, pages 162–167, 1986.

A Appendix

A.1 Technique of [GMS08]

We give here an informal sketch of the improvements achieved by the techniques of [GMS08] over the protocol of [MF06]. We give the relevant steps of the equality-checker protocol of [MF06], and the modifications to the protocol using the technique of [GMS08].

Equality-Checker:

- P_1 creates t garbled circuits $C_1, \cdots, C_t$. He sends C_j and commitments to the garbled values for the input wires.
- P_2 chooses a random subset $S \subset \{1, \cdots, t\}$ with $|S| = t/2$ and sends S to P_1.
- P_1 exposes the secrets of C_i for every $i \in S$, and also sends the witnesses to the decommitments. P_2 verifies that the garbled circuits and the commitments are correct.
- P_1 sends the garbled values corresponding to its input wire for the remaining circuits. It also sends witnesses to the commitment to the garbled values.
- P_2 uses the witnesses to verify that P_1's input to all the circuits is the same.

The modifications to Equality-Checker as in [GMS08]:

- P_1 generates t_1 random seeds and computes $GC_1, \cdots, GC_{t_1}$ using the *Garble* algorithm. Let h be a collision resistant hash function. P_1 sends $h(GC_1), \cdots, h(GC_{t_1})$ to P_2. P_1 also generates t_1^2 random seeds and generates witnesses for the decommitments. P_1 computes and sends a hash of the commitment to P_2.
- P_1 chooses a random subset $S \subset \{1, \cdots, t_1\}$ with $|S| = t_1 - t_2$ and sends S to P_1.
- P_1 sends the seeds that expose the secrets of C_i for every $i \in S$. He also sends the seeds that expose the witnesses and P_2 verifies that the garbled circuits and the commitments are correct.
- P_1 now sends the remaining garbled circuits, the garbled values of his input wires and the witnesses corresponding to the garbled values of input wires.
- P_2 uses the witnesses to verify that P_1's input to all the circuits is the same.