A Universal Problem in Secure and Verifiable Distributed Computation

Ming-Deh A. Huang[*] Shang-Hua Teng[†]

Department of Computer Science
University of Southern California
Los Angeles, California 90089

Abstract

A notion of **reduction** among multi-party distributed computing problems is introduced and formally defined. Here the reduction from one multi-party distributed computing problem to another means, roughly speaking, a secure and verifiable protocol for the first problem can be constructed solely from a secure and verifiable protocol of the second. A **universal** or **complete** multi-party distributed computing problem is defined to be one to which the whole class of multiparty problems is reducible. One is interested in finding a simple and natural multi-party problem which is universal. The *distributed sum problem*, of summing secret inputs from N parties, is shown to be such a universal problem. The reduction yields an efficient systematic method for the automatic generation of secure and verifiable protocols for all multi-party distributed computing problems. Incorporating the result from [14], it also yields an alternative proof to the completeness theorem of [9] that assuming honest majority and the existence of a trap-door function, for all multi-party problems, there is a secure and verifiable protocol.

[*]Supported in part by National Science Foundation through grant CCR-8701541 & USC Faculty Research Award.

[†]Supported in part by National Science Foundation through grant CCR-87-13489. Currently visiting Computer Science Department, Carnegie Mellon University, Pittsburg, PA 15213-3890

1 Introduction

We are concerned with the problem of computing correctly and securely in a distributed environment. This problem, raised by Goldreich, Micali, and Wigderson, was called the *multi-party protocol problem* [9]. Informally, the multi-party protocol problem can be stated as: given a description of a game with incomplete information of any number of players, produce a protocol for playing the game that leaks no partial information, provided that the majority of the players is honest. Such protocols are called *secure and verifiable protocol* they simultaneously guarantee correctness of the corresponding games and privacy of all players.

In [9], Goldreich, Micali, and Wigderson presented the first solution to the multi-party protocol problem and derived a completeness theorem for the class of distributed protocol problems with honest majority, namely, if any trap-door function exists, then for all games, there is secure and verifiable protocol provided that more than half of the players are honest. Ben-Or, Goldwasser and Wigderson [2], Chaum, Crepeau and Damgdra [5] independently prove a completeness result for multi-party protocol problem in a non-cryptographic setting.

In this paper, the relationship among the multi-party problems is studied. We formalize the notion of **reduction** among multi-party problems. Roughly speaking, a multi-party problem $\mathcal{P}$ is **reducible** to a set $\mathcal{S}$ of multi-party problems if a secure and verifiable protocol for $\mathcal{P}$ can be constructed solely from the combination of secure and verifiable protocols for problems in $\mathcal{S}$. From the notion of reduction, the concept of **universal set** and **universal multi-party problem** is defined. A set $\mathcal{S}$ of multi-party problems is a *universal set* if all multi-party problems are reducible to $\mathcal{S}$. In other words, secure and verifiable protocols for a universal set can be used as fundamental building block for constructing secure and verifiable protocols for all multi-party problems. A multi-party problem $\mathcal{P}$ is **universal** if itself forms a universal set.

We are interested in finding a simple and natural multi-party problem that is universal for the whole class of multi-party problems. The *distributed sum problem*, of summing secret inputs from N parties, is shown to be such a universal problem. Besides being a universal problem, the distributed sum problem itself is also an important problem. For example, the well-known election problem [6,14,13,4,7,18] is the distributed sum problem when the secret inputs are restricted to 0 and 1.

We prove that, assuming honest majority, designing a secure and verifiable protocol for any N-player multi-party problem is reducible to the design of secure and verifiable problem for distributed sum problem over N players. This reduction demonstrates that the distributed sum problem is universal, and gives an efficient systematic method for the automatic generation of secure and verifiable protocol for all multi-party problems.

Incorporating the result from [14][1], it yields an alternative proof to the completeness theorem of Goldreich, Micali, and Wigderson [9].

2 Preliminary

The computation model used for multi-party problems is a complete synchronous network of N nodes. Each node (node i) has a probabilistic Turing machine ($\mathcal{U}_i$), called a *user*, with its own private read-only input tape, write-only output tape, and work tape. There is a common read-only tape, a common write-only tape, and a global clock shared by all machines.

Various models can be defined according to the different means of communication among the machines [9,2,6].

- **Private Channel Model**: There are $\frac{N(N-1)}{2}$ perfectly secure private communication tapes. The i^{th} machine communicates with the j^{th} machine, and vice verse, via tape $i \leftrightarrow j$. No other machines can read the message on the $i \leftrightarrow j$ tape.
- **Common Tape Model**: There is only one communication tape. Each machine can read the message from the tape and write message on the tape.
- **Bulletin Board Model**: There are N publicly readable tape, $\mathcal{BB}_1, \ldots, \mathcal{BB}_N$, called *Bulletin Boards*, where $\mathcal{BB}_i$ is writable only by the i^{th} machine.

Throughout this paper, the bulletin board model is assumed. Note that using digital signatures [16] to authenticate the sender, protocols designed on the bulletin board model can be implemented on the common tape model. Also, using Byzantine agreement [15], all machines can agree on what message machine i has sent to machine j at certain time. Hence, protocols designed on the bulletin board model can be implemented on the private channel model.

A distributed protocol $\mathcal{DP}$ consists of a set of probabilistic algorithms $\{\mathcal{A}_i : 1 \leq i \leq N\}$ to be run on a distributed system of N parties $\mathcal{U}_1, \ldots, \mathcal{U}_N$. The algorithm $\mathcal{A}_i$ runs on $\mathcal{U}_i$. The initial content of the shared input tape is the *common input*, and the initial content of the private input tape of $\mathcal{U}_i$ is the *secret input* to $\mathcal{U}_i$. The common input typically consists of the agreed upon *verifiability and security parameters* denoted by V_N and K_N respectively. The final content on the shared output tape is the *public output* of $\mathcal{D}$, and the secret outputs of $\mathcal{U}_i$ appear on the private output tape of $\mathcal{U}_i$.

[1]It was proven in [14] that there is an optimally secure and verifiable protocol for the distributed sum problem.

Let $\mathcal{DP} = \{\mathcal{A}_i : 1 \leq i \leq N\}$ be a distributed protocol of N parties $\mathcal{U}_1,\ldots,\mathcal{U}_N$. A party $\mathcal{U}_i$ is *honest* if it runs its preassigned algorithm $\mathcal{A}_i$ faithfully and only runs $\mathcal{A}_i$, and is called *dishonest* otherwise. We allow the possibility of sharing information among the dishonest parties. A dishonest party can be either *passive* or *malicious* in the sense of [9]. We also allow each party to become dishonest in a dynamic fashion during the execution of the protocol.

A *conspiracy* $\mathcal{C}$ among s dishonest parties is a set of probabilistic polynomial time algorithms $\{\mathcal{C}_i : 1 \leq i \leq N\}$ and a dishonest parties $\mathcal{U}_a$, where $\mathcal{C}_i = \mathcal{A}_i$ if $\mathcal{U}_i$ is honest. The common input of $\mathcal{C}$ and the secret input to the honest $\mathcal{U}_i$ are the same as those in $\mathcal{DP}$. The output of $\mathcal{C}$ is defined to be the private output of $\mathcal{U}_a$, and is either one or zero.

For the ease of understanding, we restrict our consideration to a special subclass of multi-party problems, *distributed transformation problem.* The result achieved for this subclass can be generalized to the general multi-party problems [9,2,5].

The N-party *distributed transformation problem* is stated as: given a $2N$-ary formula[2] $\mathcal{CF}(x_1,\ldots,x_N,y_1,\ldots,y_N)$, design a protocol $\mathcal{P}$ such that on each *tuple of secret inputs* $(s_1,\ldots,s_N)$, the application of the protocol outputs a tuple of secret outputs $(z_1,\ldots,z_N)$, such that:

- **Verifiable Correctness:** $\mathcal{CF}(s_1,\ldots,s_N,z_1,\ldots,z_N) = 1$.
- **Privacy:** No subset of less than $\lceil N/2 \rceil$ parties can extract any more information about s_i's and z_i's from execution of $\mathcal{P}$ than it is already contained in the formula $\mathcal{CF}(s_1,\ldots,s_N,z_1,\ldots,z_N) = 1$ and their shared secret inputs.

where a *tuple of secret inputs* $(s_1,\ldots,s_N)$ means that s_i is the secret input of $\mathcal{U}_i$, and a *tuple of secret outputs* $(z_1,\ldots,z_N)$ means that z_i is the secret output of $\mathcal{U}_i$.

The distributed transformation problem can be interpreted as: at the beginning of the execution, the i^{th} party owns a private database $\mathcal{DB}_i$, the application of the protocol transforms the i^{th} database securely into a new database $\mathcal{DB}'_i$ which satisfies the predefined properties without revealing any more information about $\mathcal{DB}_i$'s and $\mathcal{DB}'_i$'s.

The Turing machine game, defined by Goldreich, Micali, and Wigderson [9], is a subclass of the distributed transformation problem defined above. Informally, the Turing machine game can be described as: N parties, respectively owning secret inputs $s_1,\ldots,s_N$, are to *correctly* run a given Turing machine $\mathcal{M}$ on $s_1,\ldots,s_N$ while keeping the maximum possible privacy of all parties. Clearly, the Turing machine game with Turing machine $\mathcal{M}$

[2] It is usually assumed that the formula $\mathcal{CF}$ of a distributed transformation problem is random polynomial time computable in the sense that we can construct a *random* algorithm $\mathcal{A}_{\mathcal{CF}}$ which on each tuple of inputs $(s_1,\ldots,s_N)$ outputs a tuple of output, in random polynomial time, a tuple $(z_1,\ldots,z_N)$ such that $\mathcal{CF}(s_1,\ldots,s_N,z_1,\ldots,z_N) = 1$.

is a distributed transformation problem with formula $\mathcal{CF}_{\mathcal{M}}$:

$$\mathcal{CF}_{\mathcal{M}}(x_1,\ldots,x_N,y_1,\ldots,y_N) = 1 \; if \; y_1 = y_2 = \ldots = y_N = \mathcal{M}(x_1,\ldots,x_N)$$

A distributed protocol $\mathcal{DP}$ is a s-**secure** protocol for a distributed transformation problem with formula $\mathcal{CF}$, if the following condition is satisfied.

For all conspiracy C among a set of s dishonest users, for all pairs of $2N$-ary tuples $(x_1,\ldots,x_N,u_1,\ldots,u_N)$ and $(y_1,\ldots,y_N,v_1,\ldots,v_N)$ with

$$\mathcal{CF}(x_1,\ldots,x_N,u_1,\ldots,u_N) = \mathcal{CF}(y_1,\ldots,y_N,v_1,\ldots,v_N) = 1$$

and $x_i = y_i$ if $\mathcal{U}_i$ is dishonest, for all $k \in \mathcal{N}$,

$$prob\{C(x_1,\ldots,x_N,u_1,\ldots,u_N) = 1\} - prob\{C(y_1,\ldots,y_N,v_1,\ldots,v_N) = 1\} \leq \frac{1}{(N+z+K_N)^k}$$

where z is the input size which equals to the maximum binary-length of x_i and y_i.

Informally, the above condition says that $(x_1,\ldots,x_N)$ and $(y_1,\ldots,y_N)$ are *polynomial time indistinguishable* to the dishonest users.

A distributed protocol $\mathcal{DP}$ is s-**verifiable** for a distributed transformation problem with formula $\mathcal{CF}$ if for all inputs $S = (s_1,\ldots,s_N)$, the probability that $\mathcal{CF}(S,Z) = 1$, where $Z = (z_1,\ldots,z_N)$ is the output of $\mathcal{DP}$ on $s_1,\ldots,s_N$, is at least $1 - \frac{1}{(N+z+V_N)^k}$ for all $k \in \mathcal{N}$, provided no more than s users are dishonest.

A distributed protocol $\mathcal{DP}$ is an **optimally secure and verifiable** protocol for a distributed transformation problem iff it is s-secure and s-verifiable for all $1 \leq s \leq N$.

3 Complete Sets and Universal Problems

Throughout the development of computational complexity theory, an important notion has been the *reduction* among a class $\mathcal{CP}$ of problems. Informally, reduction from one problem to another shows that the first problem is essentially no harder than the second. The notion of reduction introduces a partial order among problems in $\mathcal{CP}$. A problem $\mathcal{P}$ is **complete** or **universal** for the whole class of problems if all problems in $\mathcal{CP}$ are reducible to $\mathcal{P}$. More generally, a subset $\mathcal{S} \subseteq \mathcal{CP}$ is a *complete set* if all problems in $\mathcal{CP}$ are reducible to $\mathcal{S}$.

The completeness of a problem $\mathcal{P}$ is often used as a strong evidence that $\mathcal{P}$ is intractable up to certain computation power. For example, if a problem $\mathcal{P}$ is complete for the class of recursive functions (NP, P) under recursive reduction (polynomial-time reduction, NC-reduction, respectively), then $\mathcal{P}$ is undecidable (unlikely in P, unlike in NC, respectively). However, in the case where $\mathcal{P}$ admits an efficient solution, a constructive proof of completeness provides a systematic method for solving all problems in $\mathcal{CP}$. In this case, we also call $\mathcal{P}$ a universal problem for the class $\mathcal{CP}$.

Informally, the reduction from a multi-party problem $\mathcal{P}$ to another multi-party problem $\mathcal{P}'$ means that $\mathcal{P}$ can be solved by alternating applications of local computation by individual parties, and a secure and verifiable protocol for $\mathcal{P}'$. More specifically, each distributed protocol can be decomposed into a sequence of local transformation where each user computes locally and securely; and distributed transformation where all users work together to transform a tuple of secret inputs to a tuple of secret outputs satisfying some predefined conditions. Let $Program(S)$ be the set of all distributed programs consisting of alternating local transformation and distributed transformation protocols from S. Informally, a set of protocols is *complete* iff for all multi-party problems P, there is a distributed program from $Program(S)$ that is secure and verifiable for P. A multi-party problem Q is *universal* iff each secure and verifiable protocol for Q by itself forms a complete set.

3.1 Local Transformation vs Distributed Transformation

The class of distributed transformation problem can be partitioned into two subclasses according to the input–output dependency. Let us first see some examples:

Problem 3.1 *There are N users. User i has a secret value s_i. User i wants to compute the largest perfect square which is smaller than s_i. In other words, problem 3.1 is a distributed transformation problem with formula CF:*

$$\mathcal{CF}(x_1,\ldots,x_N,y_1,\ldots,y_N)=1 \mathit{iff}\ y_i=\max\{z^2 \mid z^2\le x_i\}$$

Problem 3.2 *There are N users. User i has a secret value s_i. User i wants to compute $\sum_{j=1}^{N} s_j^i$.*

In problem 3.1, each user can locally compute its secret output from its secret input; while in problem 3.2, the secret output of each user depends on the secret inputs of all other users. Hence, each user, by itself, can not obtain the correct secret output. In order to perform the computation, each user has to communicate with other users.

In general, a distributed transformation problem $\mathcal{P}$ with formula $\mathcal{CF}$ is *locally computable* if there are N functions $f_1,\ldots,f_N\in\mathcal{RPU}$, such that for all

$$\mathcal{CF}(s_1,\ldots,s_N,z_1,\ldots,z_N)=1 \Longleftrightarrow z_i=f_i(s_i),$$

where $\mathcal{RPU}$ stands for the class of probabilistic polynomial time computable unary functions.

The computation of a locally computable distributed transformation problem is called *local transformation*, and the computation of a distributed transformation problem which involves inter-user communication is called a *distributed transformation.*

Using the probabilistic public-key cryptosystem of Goldwasser and Micali [11] and two party zero knowledge proof protocols from [12,10,3], or using the verifiable secret sharing (VSS) [2,5,17], each party can prove to all other parties the correctness of its local computation without leaking any information about its secrets. Such a scheme can be found in [1,2,5,14]. Thus, it is assumed that the local transformation of each party in all distributed protocols is performed securely with verifiable correctness.

3.2 Reducibility

Two operators are defined on the set of distributed protocols to formalize the concept of reduction from one multi-party problem to another.

Definition 3.1 (Composition) *Let $\mathcal{DP}_1$ and $\mathcal{DP}_2$ be two N-party distributed protocols, and $F = (f_1, \ldots, f_N) \in \mathcal{RPU}^N$. The F-composition of $\mathcal{DP}_1$ and $\mathcal{DP}_2$ forms a new N-party distributed protocol, denoted by $\mathcal{DP}_2 \odot_F \mathcal{DP}_1$, which is composed of the following three steps: (1) apply $\mathcal{DP}_1$ on a tuple of secret input $(s_1, \ldots, s_N)$ to compute a tuple of secret outputs $(u_1, \ldots, u_N)$; (2) each party $\mathcal{U}_i$ performs a local transformation to compute $f_i(u_i)$; (3) apply $\mathcal{DP}_2$ on $(f_1(u_1), \ldots, f_N(u_N))$ to compute the final tuple of secret outputs $(z_1, \ldots, z_N)$.*

Definition 3.2 (Combination) *Let $\mathcal{DP}_1, \ldots, \mathcal{DP}_k$ be k N-party distributed protocols, the combination of these k protocols defines a new N-party distributed protocol, denoted by $\uplus_{i=1}^{k} \mathcal{DP}_i$, which is specified as: on a tuple of secret inputs $((s_{1,1}, , , s_{k,1}), \ldots, (s_{1,N}, \ldots, s_{k,N}))$, for $i = 1$ to k, apply $\mathcal{DP}_i$ on S_i to compute a tuple of secret outputs $(z_{i,1} \ldots, z_{i,N})$. Then the final tuple of secret output is $((z_{1,1}, , , z_{k,1}), \ldots, (z_{1,N}, \ldots, z_{k,N}))$.*

Let $\mathcal{IDP}_N$ denote the identity distributed protocol whose application on any tuple of secret inputs $(s_1, \ldots, s_N)$ outputs the tuple of secret outputs $(s_1, \ldots, s_N)$.

Definition 3.3 (Protocol Circuit) *An N-party protocol circuit C is a labeled directed acyclic simple graph with a unique sink r_C in which each vertex v is labeled by an ordered pair $(F_v, \mathcal{DP}_v)$, where $F_v \in \mathcal{RPU}^N$ and $\mathcal{DP}_v$ is a N-party distributed protocol. The* value *of each vertex v in a protocol circuit is a N-party distributed protocol which is defined inductively:*

- *If v is a leaf vertex with label $(F_v, \mathcal{DP}_v)$, then $value(v) = \mathcal{DP}_v \odot_{F_v} \mathcal{IDP}_N$.*

- *If v is an internal vertex, labeled by $(F_v, \mathcal{DP}_v)$ and with children $w_1, \ldots, w_k$, then:*

$$value(v) = \mathcal{DP}_v \odot_{F_v} (\biguplus_{i=1}^{k} value(w_i))$$

The distributed protocol defined a protocol circuit C, denoted by protocol(C), is value(r_C).

We can evaluate a protocol circuit C on a tuple of secret inputs $(s_1, \ldots, s_N)$ according to the definition of composition and combination. Note that the evaluation is composed of an alternating applications of local transformation within each party and some distributed protocols associated with the vertices in C. This yields a general paradigm for solving multi-party problems.

Definition 3.4 (Reduction) *A set S of multi-party problems is* **reducible** *to another set $\mathcal{T}$ of multi-party problems iff for all $\mathcal{P} \in S$, there exist protocol circuit C, with protocol labels only from the set of protocols which are secure and verifiable for problems in S, that defines a secure and verifiable protocol for $\mathcal{P}$.*

Let $F = (f_1, \ldots, f_N)$ be an N-tuple of random polynomial computable unary functions. Let $\mathcal{DP}_1, \ldots, \mathcal{DP}_k$ be k $2N$-ary formulas. The *F-composition* of $\mathcal{CF}_1$ and $\mathcal{CF}_2$, denoted by $\mathcal{CF}_2 \odot_F \mathcal{CF}_1$, is defined as: for all N-tuples, $X = (x_1, \ldots, x_N), Y = (y_1, \ldots, y_N)$,

$$\mathcal{CF}_2 \odot_F \mathcal{CF}_1(X, Y) = 1, \ iff \ \exists U = (u_1, \ldots, u_N), \mathcal{CF}_1(X, F(U)) = \mathcal{CF}_2(F(U), Y) = 1 \quad (1)$$

Where $F(U) = (f_1(u_1), ..., f_N(u_N))$.

The *combination* of $\mathcal{CF}_1, \ldots, \mathcal{CF}_k$, denoted by $\uplus_{i=1}^{k} \mathcal{CF}_i$, is defined as $1 \le i \le k$, $1 \le j \le N$, for all $S_i = (s_{i,1}, ..., s_{i,N})$, $Z_i = (z_{i,1}, ..., z_{i,N})$, $U_j = (s_{1,j},,, s_{k,j})$, $V_j = (z_{1,j},,, z_{k,j})$,

$$\uplus_{i=1}^{k} \mathcal{CF}_i\{(U_1, V_1), ..., (U_N, V_N)\} = 1 \ iff \ \prod_{i=1}^{k} \mathcal{CF}_i(S_i, Z_i) = 1 \quad (2)$$

If for all vertex v is a protocol circuit, $\mathcal{DP}_v$ is a distributed protocol for a distributed transformation problem with formula $\mathcal{CF}_v$, then C defines a formula, denoted by *formula*(C), by Relation (1), and Relation (2) in a natural way. We can prove the following lemma.

Lemma 3.1 *If for all v, $\mathcal{DP}_v$ is a s-verifiable distributed protocol for the distributed transformation problem with formula $\mathcal{CF}_v$, then protocol(C) is a s-verifiable distributed protocol for the distributed transformation problem with formula formula(C).*

4 The Distributed Sum Problem

Formally, the distributed sum problem is a distributed transformation problem with formula

$$\mathcal{CF}(x_1,\ldots,x_N,y_1,\ldots,y_N)=1\ iff\ y_1=y_2=\ldots=y_N=\sum_{i=1}^{N}x_i.$$

where the problem domain is a subset of $\mathcal{Z}$, the set of integers. It will be shown in next section that the distributed sum problem is universal over all multi-party problems.

An optimally secure and verifiable protocol is presented in [14] based on the efficient construction of *perfectly secure patterns*.

Lemma 4.1 ([14]) *There is an optimally secure and verifiable protocol for the distributed sum problem.*

Other secure and verifiable protocols for the distributed sum problem are also implied in [2,5,9,8].

An important variance of the distributed sum problem, denoted by $\mathcal{DSP}_i^N$, is the one that after the computation, only the i^{th} party correctly computes the sum, and all other parties can extract no information about the sum. In other words, $\mathcal{DSP}_i^N$ is a distributed transformation problem with formula:

$$\mathcal{CF}(x_1,\ldots,x_N,y_1,\ldots,y_N)=1\ iff\ y_i=\sum_{j=1}^{N}x_i$$

Lemma 4.2 *the distributed sum problem and $\{\mathcal{DSP}_i^N \mid 1\le i\le N\}$ are reducible between each other.*

[PROOF] It can be easily shown that the distributed sum problem is reducible to $\{\mathcal{DSP}_i^N \mid 1\le i\le N\}$. We now show that $\{\mathcal{DSP}_i^N \mid 1\le i\le N\}$ is reducible to the distributed sum problem $\mathcal{DSP}^N$. Let $(s_1,\ldots,s_N)$ be a tuple of secret inputs, let $s=\sum_{i=1}^{N}s_i$, the application of $\mathcal{DSP}_i^N$ on $(s_1,\ldots,s_N)$ can be done by:

1. The i^{th} party $\mathcal{U}_i$ randomly chooses $w_1,w_2\in\mathcal{Z}$, such that $w_1+w_2=s_i$.
2. Apply a protocol for the distributed sum problem on $(s_1,..,s_{i-1},w_1,..,s_N)$ to produce $(y,...,y)$, where $y=w_1-s_i+\sum_{j=1}^{N}s_j=s-w_2$.
3. $\mathcal{U}_i$ locally compute w_2+y to get s.

Note that y contains no information about s, therefore the above protocol is s-secure, if the protocol for the distributed sum problem is s-secure. □

5 Universality of the Distributed Sum Problem

A natural universal problem for multi-party distributed computation is sought. And the simpler the universal problem, the better. In this section, the very simple the distributed sum problem is proven to be a universal multi-party problem. Moreover, the proof is constructive.

Theorem 5.1 (Main Theorem) *The distributed sum problem is a universal multi-party problem.*

Corollary 5.1 *For all multi-party problem, there is a secure and verifiable protocol assuming honest majority.*

5.1 Distributed Boolean Circuit Problem

The proof of Theorem 5.1 consists of a sequence of reductions. The first step is to reduce the general distributed transformation problem to a special distributed transformation problem, the *distributed boolean circuit problem.*

The distributed boolean circuit problem is proposed by the following observation.

For each formula $\mathcal{CF}(x_1,\ldots,x_N,Y_1,\ldots,Y_N)$, we can construct a *probabilistic* algorithm $\mathcal{A}_{\mathcal{CF}}$ which on each tuple of inputs $(s_1,\ldots,s_N)$ outputs a tuple $(z_1,\ldots,z_N)$ such that $\mathcal{CF}(s_1,\ldots,s_N,z_1,\ldots,z_N)=1$. In turn, we can construct a Boolean circuit[3], $C_{\mathcal{CF}}$ to implement $\mathcal{A}_{\mathcal{CF}}$ such that the size of C_{CF} is polynomially bounded by the time complexity of $\mathcal{A}_{\mathcal{CF}}$. In the context of secure distributed computation, $\mathcal{N}$-parties, each holding some secret inputs to $C_{\mathcal{CF}}$, want to evaluate $C_{\mathcal{CF}}$ to correctly compute their corresponding secret output, i.e. $\mathcal{U}_i$ holding secret input s_i is to securely and correctly compute the value of z_i. In circuit C_{CF}, s_i corresponds to a subset of input Boolean variables, and z_i to a subset of output Boolean variables. The distributed boolean circuit problem is defined formally as:

Definition 5.1 (Distributed Boolean circuit problem) *Given a Boolean circuit C of m input variables $x_1,\ldots,x_m$, and n output variables $b_1,\ldots,b_n$. Each party $\mathcal{U}_i$ owns a nonempty subset of input variables X_i such that $\cup_{i=1}^N X_i=\{x_1,\ldots,x_m\}$ and $X_i\cap X_j=\phi$, for all $1\le i\ne j\le N$. The distributed Boolean circuit problem with circuit C is a distributed transformation problem with formula $\mathcal{CF}_C$:*

$$\mathcal{CF}_C(X_1,\ldots,X_N,Y_1,\ldots,Y_N)=\bigwedge_{j=1}^{n}\{\mathcal{F}_j(X_1,\ldots,X_N)=(\sum_{i=1}^{N}y_{i,j}) \bmod 2\}$$

[3]A Boolean circuit is a labeled directed acyclic graph in which the leaves are labeled by distinct Boolean variables, and the internal nodes are labeled from the set of Boolean operators. Each node v in the Boolean circuit is associated with a Boolean formula which is defined in a natural way.

Where $\mathcal{F}_i$ is the boolean formula defined by the Boolean circuit C on output variable z_i, $Y_i = \{y_{i,1}, ..., y_{i,n}\}$.

Lemma 5.1 *The distributed transformation problem is reducible to the distributed Boolean circuit problem and the distributed sum problem.*

[PROOF] Given a distributed transformation problem with formula $\mathcal{CF}$, first we construct a Boolean circuit $C_{\mathcal{CF}}$ of output Boolean variables $b_1, \ldots, b_n$, then we apply the secure and verifiable protocol for the distributed Boolean circuit problem on $C_{\mathcal{CF}}$. Suppose b_j is a bit in the secret output z_i of $\mathcal{U}_i$, we apply the secure and verifiable protocol for $\mathcal{DSP}_i$ on $(y_{1,j}, ..., y_{N,j})$ to transform b_j securely and correctly to $\mathcal{U}_i$. The verifiability and the security of the above reduction can be easily verified. □

5.2 Two Primitives for the Distributed Boolean Circuit Problem

We will construct a distributed protocol which evaluates a Boolean circuit sequentially gate after gate in such a way that after evaluating one gate b, each party $\mathcal{U}_i$ obtains a fraction of information y_i about the value of b defined on the secret inputs $S_1, \ldots, S_N$, and $\sum_{i=1}^{N} y_i \bmod 2 = value(b)$. Moreover, no proper subset less than $N/2$ parties can extract any information about $value(b)$ and S_i's more than those contained in their secret inputs. Note that $\wedge, \oplus_2$ are **complete** in zero-one Boolean Algebra in the sense that all boolean operators can be respented by those two operators. Therefore, for all Boolean circuits C of size n, there is an equivalent Boolean circuit C' built up by $\oplus_2$ and $\wedge$ only, whose size is polynomial in n, computes the same function as C does. This reduces the distributed Boolean circuit problem to the following set of problems.

- **Distributed $\oplus_2$-problem:** a distributed transformation problem with formula:

$$\mathcal{CF}((x_1,y_1),...,(x_N,y_N),z_1,\ldots,z_N) = 1 \ iff \sum_{i=1}^{N} z_i \bmod 2 = (\sum_{i=1}^{N} x_i \bmod 2) \oplus_2 (\sum_{i=1}^{N} y_i \bmod 2)$$

- **Distributed $\wedge$-problem:** a distributed transformation problem with formula:

$$\mathcal{CF}((x_1,y_1),...,(x_N,y_N),z_1,\ldots,z_N) = 1 \ iff \sum_{i=1}^{N} z_i \bmod 2 = (\sum_{i=1}^{N} x_i \bmod 2) \wedge (\sum_{i=1}^{N} y_i \bmod 2)$$

We can prove the following lemma:

Lemma 5.2 *The distributed Boolean circuit problem is reducible to the distributed $\oplus_2$-problem, distributed $\wedge$-problem, and the distributed sum problem.*

Motivated by finding reduction from the distributed $\wedge$-problem and the distributed $\oplus_2$-problem to the distributed sum problem, we introduce the following set of equivalent distributed transformation problems.

Let $\mathcal{Z}[x]$ denote the set of polynomials in x whose coefficients are in $\mathcal{Z}$. Let $\mathcal{Z}[x]^N$ stand for the set of integral polynomials of degree N. We define a function $\mathcal{PARITY}$: $\mathcal{Z}[x] \rightarrow \{0,1\}$ as: for all $a[x] = \Sigma_{i=1}^{N} a_i x^i \in \mathcal{Z}[x]$,

$$\mathcal{PARITY}(a[x]) = \sum_{i=1}^{N} a_i \bmod 2$$

• $\oplus_2$**-simulation-Problem** *is a distributed transformation problem with formula:*

$$\mathcal{CF}(F,H) = 1 \ iff \ \mathcal{PARITY}\left(\sum_{i=1}^{N} h_i[x]\right) = \mathcal{PARITY}\left(\sum_{i=1}^{N} f_i[x]\right) \oplus_2 \mathcal{PARITY}\left(\sum_{i=1}^{N} g_i[x]\right) \quad (3)$$

• $\wedge$**-simulation-problem** *is a distributed transformation problem with formula:*

$$\mathcal{CF}(F,H) = 1 \ iff \ \mathcal{PARITY}\left(\sum_{i=1}^{N} h_i[x]\right) = \mathcal{PARITY}\left(\sum_{i=1}^{N} f_i[x]\right) \wedge \mathcal{PARITY}\left(\sum_{i=1}^{N} g_i[x]\right) \quad (4)$$

Where $F = ((f_1[x], g_1[x]), ..., (f_N[x], f_N[x]))$, and $f_i[x], g_i[x], h_i[x] \in \mathcal{Z}[x]^N$.

We observe that $\mathcal{PARITY}$ defines a homomorphism from $\mathcal{Z}[x]$ to $\mathcal{F}_2 = \{0, 1, \oplus_2, \wedge\}$. Therefore, the solution to the above two problems can be applied for N parties to perform secure $\wedge$ and $\oplus_2$ operations. This reduces the distributed $\oplus_2$-problem and the distributed $\wedge$-problem to the $\oplus_2$-simulation problem and the $\wedge$-simulation problem.

Observer that, for all $i : 1 \leq i \leq N$, letting $h_i[x] = f_i[x] + g_i[x]$,

$$\mathcal{PARITY}\left(\sum_{i=1}^{N} h_i[x]\right) = \mathcal{PARITY}\left(\sum_{i=1}^{N} f_i[x]\right) \oplus_2 \mathcal{PARITY}\left(\sum_{i=1}^{N} g_i[x]\right)$$

Hence, the $\oplus_2$-simulation problem can be solved solely by local computation. Consequently, we have:

Lemma 5.3 *The distributed Boolean circuit problem, hence the distributed transformation problem, is reducible to the $\wedge$-simulation problem, and the distributed sum problem.*

5.3 Reducing the $\wedge$-Simulation Problem to the Distributed Sum Problem

In this section, we complete the proof of the main theorem by showing that the $\wedge$-simulation problem is reducible to the distributed sum problem.

Let $f[x] = \sum_{j=1}^{N} f_j[x]$, $g[x] = \sum_{j=1}^{N} g_j[x]$, and $h[x] = f[x]g[x]$. Then $h[x]$ is a polynomial of degree no more than $2N$, and

$$\mathcal{PARITY}(h[x]) = \mathcal{PARITY}(f[x]) \wedge \mathcal{PARITY}(g[x])$$

Lemma 5.4 *Given a secure and verifiable protocol for the distributed sum problem, there is a secure and verifiable protocol to transform the tuple of secret inputs*

$$((f_1, g_1), \cdots, (f_i, g_i), \cdots, (f_N, g_N))$$

to the tuple of secret outputs

$$((h[2], h[2N+2], h[4N+2]), \cdots, (h[2i], h[2N+2i], h[4N+2]), \cdots, (h[2N], h[4N], h[4N+2])).$$

[PROOF] Protocol 1:

1. for all i, locally, $\mathcal{U}_i$ computes $f_i[2j]$ and $g_i[2j]$, $1 \le j \le 2N+1$
2. for $i = 1$ to N, apply a protocol $\mathcal{DSP}_i^N$ on tuples of secret inputs $(f_1[2i], \cdots, f_N[2i])$ and $(f_1[2N+2i], \cdots, f_N[2N+2i])$, $(g_1[2i], \cdots, g_N[2i])$ and $(g_1[2N+2i], \cdots, g_N[2N+2i])$ to transfer $f[2i] = \sum_{j=1}^N f_j[2i]$, $f[2N+2i] = \sum_{j=1}^N f_j[2N+2i]$, $g[2i] = \sum_{j=1}^N g_j[2i]$, $g[2N+2i] = \sum_{j=1}^N g_j[2N+2i]$ securely and correctly to $\mathcal{U}_i$.
3. apply a protocol for the distributed sum problem on the tuple of secret inputs $(f_1[4N+2], \cdots, f_N[4N+2])$ and $(g_1[4N+2], \cdots, g_N[4N+2])$ to transfer $f[4N+2] = \sum_{j=1}^N f_j[4N+2]$, and $g[4N+2] = \sum_{j=1}^N g_j[4N+2]$ securely and correctly to all parties.
4. Each party $\mathcal{U}_i$ computes $h[2i]$, $h[2N+2i]$ and $h[4N+2]$ locally. □

By **interpolation law**, we have:

$$h[x] = \sum_{k=1}^{2N+1} \frac{\prod_{j\neq k}(x-2j)}{\prod_{j\neq k}(2k-2j)} h[2k] = \frac{1}{A} \sum_{k=1}^{2N+1} H_k[x]$$

Where $A = \frac{1}{\prod_{1\le i<j\le 2N+1}(2i-2j)}$, and for all $k : 1 \le k \le 2N+1$,

$$H_k[x] = \prod_{j\neq k}(x-2j) \prod_{1\le a<b\le 2N+1, a\neq k, b\neq k} (2a-2b)h[2k]$$

Let $H_k[x] = \sum_{j=0}^{2N} H_{k,j}x^j$, let $c_j = \sum_{k=1}^{2N+1} H_{k,j}$. Then $d_j = c_j/A \in \mathcal{Z}$.
The following procedure forms a reduction for the $\wedge$-simulation-problem to $\{\mathcal{DSP}_i \mid 1 \le i \le N\}$ and the distributed sum problem.

Reduction Protocol:

1. apply Protocol 1.

2. By interpolation law, each party $\mathcal{U}_i$ computes polynomial $H_i[x]$, $H_{N+i}[x]$ and $H_{2N+1}[x]$ locally. Then for all $j : 0 \leq j \leq 2N$, $\mathcal{U}_1$ computes $c_{i,j} = H_{1,j} + H_{N+1,j} + H_{2N+1,j}$, and all other parties $\mathcal{U}_i$ computes $c_{i,j} = H_{i,j} + H_{N+i,j}$ locally.

3. for $i = 1$ to N, apply a protocol for $\mathcal{DSP}_i^N$ on tuples of secret inputs $(c_{1,i}, ..., c_{N,i})$ and $(c_{1,N+i}, ..., c_{N,N+i})$ to transfer $c_i = \sum_{k=1}^{2N+1} H_{k,i}$, and $c_{N+i} = \sum_{k=1}^{2N+1} H_{k,N+i}$, securely and correctly to $\mathcal{U}_i$.

4. apply a protocol for the distributed sum problem on the tuple of secret inputs $(c_{1,0}, ..., c_{1,0})$ to transfer $c_0 = \sum_{k=1}^{2N+1} H_{k,0}$, securely and correctly all parties.

5. *Local Transformation:* each party $\mathcal{U}_i$ computes $d_i = \frac{c_i}{A}$, $d_{N+i} = \frac{c_{N+i}}{A}$ and $d_0 = \frac{c_0}{A}$ locally. Then for all $j : 1 \leq j \leq N$, $\mathcal{U}_1$ randomly generate a polynomial $h_1[x] \in \mathcal{Z}[x]^N$ locally such that $\mathcal{PARITY}(h_1) = (d_1 + d_{N+1} + d_0) \bmod 2$, and all other parties $\mathcal{U}_i$ randomly generate a polynomial $h_i[x] \in \mathcal{Z}[x]^N$ locally such that $\mathcal{PARITY}(h_i) = (d_i + d_{N+i}) \bmod 2$.

By interpolation law, we have:

$$\mathcal{PARITY}\left(\sum_{j=1}^{N} h_j[x]\right) = \mathcal{PARITY}(h[x]) = \mathcal{PARITY}(f[x]) \wedge \mathcal{PARITY}(g[x])$$

The verifiability of the above reduction can be easily verified. We now show that the above reduction is secure under the assumption of honest majority.

We first make an observation based on the classical information theory.

Lemma 5.5 (Composition Lemma) *If $a_1, \ldots, a_k$ be k $(K > 1)$ random numbers in $\mathcal{Z}$, then for all $X \subset \{a_1, \ldots, a_k\}$, X implies no information about $\Sigma_{i=1}^k a_i$.*

It follows from the Composition Lemma (Lemma 5.5), any proper subset of $\{f_1, \ldots, f_N\}$ or $\{g_1, \ldots, g_N\}$ contains no information about $\mathcal{PARITY}(f[x])$ and $\mathcal{PARITY}(g[x])$. So, after running the protocol, the only additional information obtained by the i^{th} party is the values of $f[2i], f[2N+2i], f[4N+2]$, $g[2i], g[2N+2i], g[4N+2]$, hence $h[2i], h[2N+2i], h[4N+2]$. It follows that for any subset $S \subset \{1, ..., N\}$, the information on the parity of f, g and h that can be obtained by S is all that is implied by the interpolated values held by the party in S. Treating the coefficients of f (g, h) as variables, each value $f[2j]$ ($g[2j]$, $h[2j]$ respectively) determines a linear equation in the coefficients of f (g, h respectively). So the values held by the users in S determines a linear system L. Consequently, the security of our scheme relies on the parity of the solutions in the solution space of L.

The following Lemma can be proved via linear algebraic analysis (the proof will appear in the full paper).

Lemma 5.6 *For all $p[x] = \sum_{i=0}^{N} p_i x^i \in \mathcal{Z}[x]$, for all $X \subset \{2,4,6...,4N+2\}$, if $| X | \leq N$, then:*

$$\frac{| SOLU_0 |}{| SOLU_1 |} = 1$$

Where for $j \in \{0,1\}$

$$SOLU_j = \{(c_0, c_1, \ldots, c_N) \mid \sum_{i=0}^{N} c_i = j mod2 \ \& \ \forall x \in X, p[x] = \sum_{i=0}^{N} c_i x^i\}$$

Therefore, it follows from the above lemmas that any subset of s dishonest users can extract no more information about $\mathcal{PARITY}(f_i)[x]$, $\mathcal{PARITY}(g_i)[x]$, $\mathcal{PARITY}(h_i[x])$, $\mathcal{PARITY}(f[x])$, $\mathcal{PARITY}(g[x])$, and $\mathcal{PARITY}(h[x])$ provided all applications of the protocols for the distributed sum problem and its variance are s-secure, and $s \leq \lceil \frac{N}{2} \rceil$.

Lemma 5.7 *The $\wedge$-simulation problem is reducible to the distributed sum problem.*

Remark 5.1 *It is assumed in Lemma 5.6 that the domain of the distributed sum problem is $\mathcal{Z}$, the set of integers. This assumption is not realistic in the sense that there is no bound on the size of integers. The following are some results when the size of the integers is bounded.*

Let S_N be a security parameter agreed upon all users, let $d_N = 2^{S_N}$ and $D_N = \{-d_N, ..., d_N\}$, and for all $i : N-1 \leq i \leq 0$, $d_i = \Theta(N^2 d_{i+1})$ and $D_i = \{-d_i, ..., d_i\}$. The following Lemma can be proved via linear algebraic analysis.

Lemma 5.8 *For a random polynomial $p[x] = \sum_{i=0}^{N} p_i x^i \in \mathcal{Z}[x]$ such that $p_i \in \mathcal{D}_i$, for all $X \subset \{2,4,6...,4N+2\}$, if $| X | \leq N$, then with probability at least $1 - poly(N)(\frac{1}{2})^{S_N}$:*

$$\frac{| SOLU_0 |}{| SOLU_1 |} = 1 \pm poly(N) \cdot (\frac{1}{2})^{S_N}$$

Where $poly(N)$ mean a polynomial in N.

Assuming the coefficients of the secret polynomials f and g are bounded as in the above lemma, it follows from the above lemma that, with very high probability, no subset of s dishonest users can extract any more information about $\mathcal{PARITY}(f[x])$, $\mathcal{PARITY}(g[x])$ and $\mathcal{PARITY}(h[x])$, where $s \leq \lceil \frac{N}{2} \rceil$.

Remark 5.2 *Note that, if s, the number of dishonest users, is greater than $\lfloor \frac{N}{2} \rfloor$, they can compute $f[x]$, $g[x]$ by interpolation law, thus $\mathcal{PARITY}(h[x])$. Hence, our scheme is not secure against the dishonest majority.*

Acknowledgement: We would like to thank Len Adleman and Gary Miller for helpful discussion.

References

[1] J. Benaloh. Secret sharing homomorphisms: keeping shares of a secret secret. In *CRYPTO*, 1986.

[2] Michael Ben-or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 1–10, ACM, May 1988.

[3] Gilles Brassard and Claude Crepeau. Non-transitive transfer of confidence: a perfect zero-knowledge interactive protocol for sat and beyond. In *27th Annual Symposium on Foundations of Computer Science*, pages 188–195, IEEE, October 1986.

[4] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *CACM*, 24():84–88, Feb. 1981.

[5] David Chaum, Claude Crepeau, and Ivan Damgdra. Multi-party unconditionally secure protocols. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 11–19, ACM, May 1988.

[6] J. Cohen and M. Fisher. A robust and verifiable cryptographically secure election scheme. In *FOCS25*, pages 372–382, IEEE, October 1985.

[7] R. A. Demillo, N. A. Lynch, and M. J. Merritt. Cryptographic protocols. In *Proceedings of the 14h Annual ACM Symposium on Theory of Computing*, pages 383–400, ACM, May 1982.

[8] Z. Galil, S. Haber, and M. Yung. Cryptographic computation: secure fault-tolerant protocols and the public-key model. In *CRYPTO*, 1987.

[9] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, ACM, 1987.

[10] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *27th Annual Symposium on Foundations of Computer Science*, pages 174–187, IEEE, 1986.

[11] S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, 28(2):270–299, 1984.

[12] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge of complexity of interactive proof-systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, ACM, May 1985.

[13] Ming-Deh A. Huang and Shang-Hua Teng. Election schemes of optimal security and verifiability. manuscript, usc. 1988.

[14] Ming-Deh A. Huang and Shang-Hua Teng. Secure and verifiable schemes for election and general distributed computing problems. In *7th Annual Symposium on Principles of Distributed Computing*, ACM, 1988.

[15] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *JACM*, 27:228–234, 1980.

[16] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digitial signatures and public-key cryptosystems. *CACM*, 21(2):120–126, 1978.

[17] A. Shamir. How to share a secret. *CACM*, 22(11):612–613, 1979.

[18] A. Yao. Protocols for secure computations. In *23th Annual Symposium on Foundations of Computer Science*, pages 160–164, IEEE, 1982.

[19] A. Yao. Theory and application of trapdoor functions. In *23th Annual Symposium on Foundations of Computer Science*, pages 80–91, IEEE, 1982.