Statistical Multiparty Computation Based on Random Walks on Graphs

Liangliang Xiao[1], Mulan Liu[2], and Zhifang Zhang[2]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, 100080, China
[2] Academy of Mathematics and System Sciences, Key Laboratory of Mathematics Mechanization, Chinese Academy of Sciences, Beijing, 100080, China
{mlliu, zfz}@amss.ac.cn

Abstract. With respect to a special class of access structures based on connectivity of graphs, we start from a linear secret sharing scheme and turn it into a secret sharing scheme with perfect security and exponentially small error probability by randomizing the reconstruction algorithm through random walks on graphs. It reduces the polynomial work space to logarithmic. Then we build the corresponding statistical multiparty computation protocol by using the secret sharing scheme. The results of this paper also imply the inherent connections and influences among secret sharing, randomized algorithms, and secure multiparty computation.

Keywords: statistical multiparty computation, random walks on graphs, linear secret sharing scheme, monotone span program

1 Introduction

The problem of secure multiparty computation (MPC for short) is fundamental in cryptography and distributed computation. A solution of the multiparty computation problem implies in principle a solution to any cryptographic protocol problem. After it was proposed by Yao [3] for two party case and Goldreich, Micali, Wigderson [8] for multiparty case, it has become an active and developing field of information security. Although the study and construction of statistical multiparty computation is relatively less than that of the perfect and computational cases, it is a natural idea to convert a perfect multiparty computation protocol to a statistical one by the technique of randomizing the corresponding algorithm. As the advantage of randomized algorithms, it will raise efficiency and make the multiparty computation protocol more applicable, although increasing error probability as well as losing perfect security sometimes from the perfect situation to the statistical situation. By this idea we do design a statistical multiparty computation protocol based on random walks on graphs. In detail, we start from a linear secret sharing scheme and turn it into a secret sharing scheme with perfect security and exponentially small error probability by randomizing the reconstruction algorithm through random walks on graphs.

It reduces the polynomial work space to logarithmic. Then we discuss how to build the corresponding statistical multiparty computation protocol by using the secret sharing scheme. The results of this paper also imply the inherent connections and influences among secret sharing, randomized algorithms, and secure multiparty computation.

The paper is organized as follows. In Section 2 we review some related concepts, such as secret sharing schemes, random walks on graphs and multiparty computation. Section 3 defines a special class of access structures based on connectivity of graphs and devises a secret sharing scheme to realize it through random walks on graphs. Section 4 discusses how to build a statistical multiparty computation protocol by using the secret sharing scheme constructed in Section 3. The last section is a conclusion.

2 Preliminaries

In this section, we recall some basic concepts and results about secret sharing, random walks on graphs, and secure multiparty computation. Throughout this paper let $\mathcal{K}$ denote a finite field and $P = \{P_1, \cdots, P_n\}$ be the set of n participants.

2.1 Secret Sharing Schemes

Secret sharing schemes were first independently proposed by Blakley [4] and Shamir [2] for the purpose of key manegement in 1979. Henceforth, it develops quickly and gets wide applications in the field of information security.

Informally, a secret sharing scheme is a protocol to share a secret among a set of participants P such that only participants in an authorized set can recover the secret together from their shares. We call the collection of all authorized sets the *access structure* over P, denoted by AS, and it satisfies the monotone ascending property: for any $A' \in AS$ and $A \subset P$, $A' \subset A$ implies $A \in AS$. Because of the monotone ascending property, for any access structure AS it is enough to consider the corresponding *minimum access structure* AS_m, defined as $AS_m = \{A \in AS | \forall B \subsetneqq A \Rightarrow B \notin AS\}$.

Suppose that S is the secret-domain, R is the set of random inputs, and S_i is the share-domain of P_i where $1 \leq i \leq n$. A secret sharing scheme with respect to an access structure AS is composed of the distribution function $\Pi : S \times R \rightarrow S_1 \times \cdots \times S_n$, $\Pi(s, r) = (\Pi_1(s, r), \cdots, \Pi(s, r))$ and the reconstruction function Re: for any $A \in AS$, $Re|_A : (S_1 \times \cdots \times S_n)|_A \rightarrow S$, such that the following two requirements are satisfied.

1. Correctness requirement: for any $A \in AS, s \in S$ and $r \in R$, it holds that $Re|_A(\Pi(s, r)|_A) = s$.

2. Security requirement: for any $B \notin AS$, $H(S|\Pi(S, R)|_B) \leq H(S)$, where $H(\cdot)$ is the entropy function.

In the security requirement, if $H(S|\Pi(S, R)|_B) = H(S)$, then we call it a *perfect secret sharing scheme* which we are interested in. Furthermore, a perfect

secret sharing scheme is *linear* (LSSS for short), if it also satisfies the following two conditions:

3. Suppose $S = \mathcal{K}$, then the share-domain S_i for $1 \leq i \leq n$ and the set of random inputs R are finite dimensional linear subspaces over $\mathcal{K}$, that is, there exist positive integers d_i for $1 \leq i \leq n$ and l such that $S_i = \mathcal{K}^{d_i}$ and $R = \mathcal{K}^l$.

4. The reconstruction function is linear. Precisely, for any set $A \in AS$, there exists a set of constants $\{\alpha_{kj} \in \mathcal{K} | P_k \in A, 1 \leq j \leq d_k\}$ such that for any $s \in \mathcal{K}$ and $r \in R = \mathcal{K}^l$, $s = \sum_{p_k \in A} \sum_{j=1}^{d_k} \alpha_{kj} \Pi_{kj}(s, r)$, where $\Pi_k(s, r) = (\Pi_{k1}(s, r), \cdots, \Pi_{kd_k}(s, r)) \in \mathcal{K}^{d_k}$.

Karchmer and Wigderson [7] introduced monotone span programs (MSP for short) as linear models computing monotone Boolean functions. Usually we denote a MSP by $\mathcal{M}(\mathcal{K}, M, \psi)$, where M is a $d \times l$ matrix over $\mathcal{K}$ and $\psi : \{1, ..., d\} \to \{P_1, ..., P_n\}$ is a surjective labelling map which actually distributes to each participant some rows of M. We call d the *size* of the MSP. For any subset $A \subseteq P$, there is a corresponding characteristic vector $\overrightarrow{\delta_A} = (\delta_1, ..., \delta_n) \in \{0, 1\}^n$ such that for $1 \leq i \leq n$, $\delta_i = 1$ if and only if $P_i \in A$. On the other hand, for any $\overrightarrow{\delta} \in \{0, 1\}^n$, there is a subset $A \subset P$ such that $\overrightarrow{\delta_A} = \overrightarrow{\delta}$. Because of the corresponding relation between vectors in $\{0, 1\}^n$ and subsets of P, in the following we denote the vector in $\{0, 1\}^n$ in terms of a characteristic vector of some subset in P. Consider a monotone Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ which satisfies that $f(\overrightarrow{\delta_B}) = 1$ implies $f(\overrightarrow{\delta_A}) = 1$ for any $A \subseteq P$ and $B \subseteq A$. We say that a MSP $\mathcal{M}(\mathcal{K}, M, \psi)$ computes the monotone Boolean function f with respect to a target vector $\overrightarrow{v} \in \mathcal{K}^l \setminus \{(0, ..., 0)\}$, if it holds that $\overrightarrow{v} \in span\{M_A\}$ if and only if $f(\overrightarrow{\delta_A}) = 1$, where M_A denotes M restricted to those rows i with $\psi(i) \in A$ and $\overrightarrow{v} \in span\{M_A\}$ means that there exists a vector $\overrightarrow{w}$ such that $\overrightarrow{v} = \overrightarrow{w} M_A$.

Beimel [1] proved the equivalence of devising a LSSS with respect to an access structure AS and constructing a MSP computing the monotone Boolean function f_{AS} which satisfies $f_{AS}(\overrightarrow{\delta_A}) = 1$ if and only if $A \in AS$. Particularly, we will show how to build a LSSS from a MSP in detail in Section 3.2.

2.2 Random Walks on Graphs for Undirected $s - t$ Connectivity Problem

Let $G(V, E)$ be an undirected graph where V is the set of vertices and E is the set of edges. First we introduce the undirected $s - t$ connectivity (USTCON) problem in $G(V, E)$: given two vertices s and t in V, decide whether s and t are in a connected component. It is easy to see that a standard graph search algorithm such as depth-first search solves the problem in $O(|E|)$ steps using workspace $O(|V|)$. But by using random walks on graphs we can devise a randomized algorithm A to solve the problem [9]: starting from the vertex s, then we randomly choose a neighbor of s, say v_1, and walk into v_1. At the next step, we randomly choose a neighbor of v_1, say v_2, walk into v_2, and so on. Taking at most $2|V|^3$ steps, if the random walk meets the vertex t, the algorithm returns

"YES"; otherwise it returns "NO". It is well known that the probability

$$Pr[A \text{ returns "YES"}] = \begin{cases} \geq \frac{1}{2}, & if\ s \text{ and } t \text{ are connected,} \\ 0, & if\ s \text{ and } t \text{ are not connected,} \end{cases}$$

that is, the algorithm has a error probability at most $\frac{1}{2}$. Furthermore, the algorithm takes $O(|V|^3)$ steps and uses $O(\log |V|)$ space which is much less than the standard graph search algorithms. Repeating the algorithm k times independently, we can reduce the error probability to $\frac{1}{2^k}$.

2.3 Secure Multiparty Computation

The problem of secure MPC for one function has been studied by many people and it can be stated as follows: n players $P_1, ..., P_n$ are to securely compute an agreed function $f(x_1, ..., x_n) = (y_1, ..., y_n)$ against an adversary, where P_i holds private input x_i and is to get the output y_i. The security means that the correctness of the outputs and the privacy of players' inputs are guaranteed even when the adversary corrupts some subset of the players. The collection of all subsets that the adversary can corrupt is called the *adversary structure*, denoted by $\mathcal{A}$, and it satisfies the monotone descending property: for any $A' \in \mathcal{A}$ and $A \subset P$, $A \subset A'$ implies $A \in \mathcal{A}$. The adversary is called an $\mathcal{A}$-adversary. Similar to the access structure, for any adversary structure $\mathcal{A}$, it is enough to consider the *maximum adversary structure* $\mathcal{A}_m$, defined as $\{A \in \mathcal{A} | \forall A \subsetneq B \Rightarrow B \notin \mathcal{A}\}$. Obviously, if $\mathcal{A}$ is an adversary structure over P, then $AS = 2^P - \mathcal{A}$ is an access structure over P, vice versa. Furthermore, an adversary structure is called Q2, *resp.* Q3, if no two, *resp.* no three of the sets in the structure cover the full player set P. According to the corrupting way, an adversary may be *passive* or *active*, i.e., he may just monitor corrupted players or take full control. Meanwhile he may be *static* or *adaptive*, i.e., all corruptions take place before the protocol starts, or happen dynamically during the protocol.

We assume that throughout this paper the communication is synchronous and a broadcast channel is given. Then in the information theoretic model, *i.e.*, the players can communicate over pairwise secure channels and the adversary has unbounded computing power, every function can be securely computed with exponentially small error probability against an adaptive $\mathcal{A}$-adversary if and only if $\mathcal{A}$ is Q2 [6]. We call MPC with information theoretic security and exponentially small error probability as statistical MPC. Some papers [10],[12] studied this problem mainly in the threshold case, and their protocols can be generalized easily to provide security against general Q2 adversaries.

3 Secret Sharing Schemes Based on Random Walks on Graphs

In this section, we first define an access structure based on connectivity of graphs, then by the algorithm of random walks on graphs we give a secret sharing scheme to realize the access structure where the reconstruction algorithm runs in polynomial time and uses logarithmic space.

3.1 Access Structures Based on Connectivity of Graphs

Let m be a positive integer, $n = \binom{m}{2}$, and $P = \{P_1, \cdots, P_n\}$ be the set of participants. Let $G(V, E)$ be a undirected complete graph with the vertex set $V = \{v_0, v_1 \cdots, v_{m-1}\}$ and edge set $E = \{v_i v_j \mid 0 \leq i < j \leq m-1\}$. Suppose $f : P \to E$ is a bijection corresponding each participant with an edge. For any subset $A \subset P$, $G(V, E_A)$ is a spanning subgraph of $G(V, E)$ where $E_A = \{v_i v_j \in E | v_i v_j \in f(A)\}$. Define the access structure

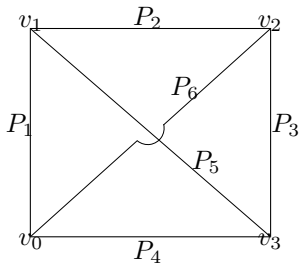
$$AS = \{A \subset P | G(V, E_A) \text{ is a connected graph}\}. \tag{1}$$

Obviously AS satisfies the monotone ascending property.

Proposition 1. *Suppose AS is given by (1) and $\mathcal{A} = 2^P - AS$ is the adversary structure. Then $\mathcal{A}$ is $Q2$.*

Proof. Let $G(V, E')$ be an disconnected graph with $E' \subset E$. In order to prove $\mathcal{A}$ is $Q2$, it suffices to prove that $G(V, E - E')$ is a connected graph, that is, for every pair of vertices v and v', they are connected in the graph $G(V, E - E')$. Suppose the graph $G(V, E')$ has k connected components, $k \geq 2$. If the vertices v and v' are in different connected components of $G(V, E')$, then the edge $vv' \notin G(V, E')$. So the edge $vv' \in G(V, E - E')$ and it implies v and v' are connected in the graph $G(V, E - E')$. If the vertices v and v' are in the same connected component of $G(V, E')$, then we consider the vertex v'' in another connected component. We have v and v'' are connected, v' and v'' are connected in the graph $G(V, E - E')$. Hence v and v' are connected in the graph $G(V, E - E')$.

Example 1. Let $m = 4$, $n = 6$, and $V = \{v_0, v_1, v_2, v_3\}$. Let $P = \{P_1, \cdots, P_6\}$, $f(P_1) = v_0 v_1$, $f(P_2) = v_1 v_2$, $f(P_3) = v_2 v_3$, $f(P_4) = v_0 v_3$, $f(P_5) = v_1 v_3$, $f(P_6) = v_0 v_2$. See the figure.



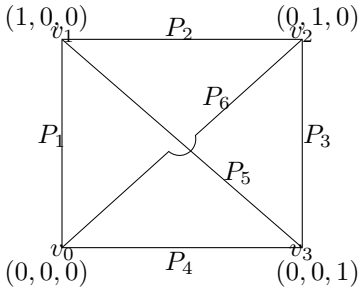
It's easy to have $AS_m = \{\{P_1, P_2, P_3\}, \{P_2, P_3, P_4\}, \{P_3, P_4, P_1\}, \{P_4, P_1, P_2\}, \{P_1, P_2, P_5\}, \{P_2, P_3, P_6\}, \{P_3, P_4, P_5\}, \{P_4, P_1, P_6\}, \{P_1, P_5, P_3\}, \{P_1, P_6, P_3\}, \{P_2, P_6, P_4\}, \{P_2, P_5, P_4\}, \{P_1, P_5, P_6\}, \{P_3, P_5, P_6\}, \{P_2, P_5, P_6\}, \{P_4, P_5, P_6\}\}$.

Let $\mathcal{A} = 2^P - AS$, then $\mathcal{A}_m = \{\{P_1, P_3\}, \{P_2, P_4\}, \{P_5, P_6\}, \{P_1, P_2, P_6\}, \{P_2, P_3, P_5\}, \{P_3, P_4, P_6\}, \{P_4, P_1, P_5\}\}$. It's easy to verify that $\mathcal{A}$ is $Q2$.

3.2 The Secret Sharing Scheme Realizing The Access Structure

First we set $S = \mathcal{K} = \mathbb{F}_2$. As to a larger secret-domain, we can put it into $\mathbb{F}_{2^k}$ for a proper positive integer k and share the secret bit by bit independently. It is obvious that the correctness and security requirements are still satisfied by doing so. We associate the vertex v_i with the vector $\overrightarrow{e_i} = (0, ..., 0, \overset{i}{1}, 0, ..., 0) \in \mathbb{F}_2^{m-1}$, where $0 \leq i \leq m-1$ and $\overrightarrow{e_0} = \overrightarrow{0}$. For $1 \leq i \leq n$, if $f(P_i) = v_{i_1}v_{i_2}$, then let $\overrightarrow{r_{P_i}} = \overrightarrow{e_{i_1}} + \overrightarrow{e_{i_2}}$ where $0 \leq i_1 < i_2 \leq m-1$.

Example 2. (following Example 1) Associate vertex v_0 with $(0,0,0)$, vertex v_1 with $(1,0,0)$, vertex v_2 with $(0,1,0)$, vertex v_3 with $(0,0,1)$.



Then $\overrightarrow{r_{P_1}} = (1,0,0)$, $\overrightarrow{r_{P_2}} = (1,1,0)$, $\overrightarrow{r_{P_3}} = (0,1,1)$, $\overrightarrow{r_{P_4}} = (0,0,1)$, $\overrightarrow{r_{P_5}} = (1,0,1)$, $\overrightarrow{r_{P_6}} = (0,1,0)$.

Construct a MSP $\mathcal{M}(\mathbb{F}_2, M, \psi)$ as follows: M consists of all the row vectors $\overrightarrow{r_{P_i}}$ for $1 \leq i \leq n$ and ψ maps the row $\overrightarrow{r_{P_i}}$ to the player P_i. Obviously, M is a $n \times (m-1)$ matrix over $\mathbb{F}_2$. In order to devise a scheme to realize AS, we introduce the access structure

$$AS_{v_0v_i} = \{A \subset P \mid \text{the vertex } v_0 \text{ and the vertex } v_i \text{ are connected in } G(V, E_A)\},$$

and claim that $\mathcal{M}(\mathbb{F}_2, M, \psi)$ can compute the Boolean function $f_{AS_{v_0v_i}}$ with respect to the target vector $\overrightarrow{e_i}$ where $1 \leq i \leq m-1$. Before giving the proof, we see the example below.

Example 3. (following Example 2) So $M = \begin{pmatrix} 1\,0\,0 \\ 1\,1\,0 \\ 0\,1\,1 \\ 0\,0\,1 \\ 1\,0\,1 \\ 0\,1\,0 \end{pmatrix}$, and $\psi(i) = P_i$ for $1 \leq i \leq 6$. The three corresponding access structures are the following: $(AS_{v_0v_1})_m = \{\{P_1\}, \{P_2, P_6\}, \{P_4, P_5\}, \{P_2, P_3, P_4\}, \{P_3, P_5, P_6\}\}$, $(AS_{v_0v_2})_m = \{\{P_6\}, \{P_3, P_4\}, \{P_1, P_2\}, \{P_1, P_3, P_5\}, \{P_2, P_4, P_5\}\}$, and $(AS_{v_0v_3})_m = \{\{P_4\}, \{P_3, P_6\}, \{P_1, P_5\}, \{P_1, P_2, P_3\}, \{P_2, P_5, P_6\}\}$. It can be easily verified that $\mathcal{M}(\mathbb{F}_2, M, \psi)$ computes $f_{AS_{v_0v_i}}$ with respect to the target vector $\overrightarrow{e_i}$ where $1 \leq i \leq 3$.

Proposition 2. *The MSP $\mathcal{M}(\mathbb{F}_2, M, \psi)$ constructed above can compute the monotone Boolean function $f_{AS_{v_0v_i}}$ with respect to the target vector $\overrightarrow{e_i}$ where $1 \leq i \leq m-1$.*

Proof. It is equivalent to prove that for $1 \leq i \leq m-1$, $\overrightarrow{e_i} \in span\{M_A\}$ if and only if $A \in AS_{v_0v_i}$, *i.e.*, the vertex v_0 and v_i are connected in $G(V, E_A)$.

Suppose that $A \in AS_{v_0v_i}$, then there is a path from the vertex v_0 to v_i in the subgraph $G(V, E_A)$, denoted by $v_{i_0} - v_{i_1} - \cdots - v_{i_k}$, where $v_{i_0} = v_0$, $v_{i_k} = v_i$ and $0 < i_1, ..., i_k \leq m-1$. Assume that $f^{-1}(v_{i_j}v_{i_{j+1}}) = P_{i_j}$ for $0 \leq j \leq k-1$, then $P_{i_j} \in A$ and $\overrightarrow{r_{P_{i_j}}} = \overrightarrow{e_{i_j}} + \overrightarrow{e_{i_{j+1}}}$. So

$$\overrightarrow{e_i} = \overrightarrow{e_{i_0}} + \overrightarrow{e_{i_k}} = (\overrightarrow{e_{i_0}} + \overrightarrow{e_{i_1}}) + (\overrightarrow{e_{i_1}} + \overrightarrow{e_{i_2}}) + \cdots + (\overrightarrow{e_{i_{k-1}}} + \overrightarrow{e_{i_k}}) = \overrightarrow{r_{P_{i_0}}} + \overrightarrow{r_{P_{i_1}}} + \cdots + \overrightarrow{r_{P_{i_{k-1}}}}, \tag{2}$$

that is, $\overrightarrow{e_i} \in span\{M_A\}$.

On the other hand, suppose that $\overrightarrow{e_i} \in span\{M_A\}$. Without loss of generality, assume that $\overrightarrow{e_i} = \overrightarrow{r_{P_{i_0}}} + \cdots + \overrightarrow{r_{P_{i_{k-1}}}}$, where $0 \leq i_0, ..., i_{k-1} \leq m-1$ and $P_{i_j} \in A$ for $0 \leq j \leq k-1$. Denote $f(P_{i_j}) = v_{h_j}v_{t_j}$, then

$$\overrightarrow{e_i} = (\overrightarrow{e_{h_0}} + \overrightarrow{e_{t_0}}) + \cdots + (\overrightarrow{e_{h_{k-1}}} + \overrightarrow{e_{t_{k-1}}}), \tag{3}$$

where $\overrightarrow{e_{h_j}}, \overrightarrow{e_{t_j}} \in \{\overrightarrow{e_0}, \overrightarrow{e_1}, ..., \overrightarrow{e_{m-1}}\}$ for $0 \leq j \leq k-1$. Because $\overrightarrow{e_1}, ..., \overrightarrow{e_{m-1}}$ are linearly independent, $\overrightarrow{e_i}$ and $\overrightarrow{e_0}$ must appear in the right of (3) for odd times, respectively, and $\overrightarrow{e_j}$ appears even times where $0 < j \leq m-1$ and $j \neq i$. Thus the expression of (3) actually determines a walk from v_0 to v_i in $G(V, E_A)$. Hence $A \in AS_{v_0v_i}$.

For $1 \leq i \leq m-1$, we build a LSSS, denoted by $LSSS_i$, realizing the access structure $AS_{v_0v_i}$ from the MSP $\mathcal{M}(\mathbb{F}_2, M, \psi)$: for a given secret s^i, the dealer randomly selects $\rho_1, \cdots, \rho_{m-1}$ in $\mathbb{F}_2$ and secretly transmits to P_j the share $M_{P_j}(\rho_1, \cdots, \rho_{i-1}, s^i, \rho_{i+1}, \cdots, \rho_{m-1})^\tau$ for $1 \leq j \leq n$, where "τ" denotes the transpose, M_{P_j} denotes M restricted to those rows k with $\psi(k) = P_j$ and in our construction $M_{P_i} = \overrightarrow{r_{P_i}}$. For any $A \in AS_{v_0v_i}$, from the proof above and the equality (2) we have that

$$s^i = \overrightarrow{e_i} \cdot \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{i-1} \\ s^i \\ \rho_{i+1} \\ \vdots \\ \rho_{m-1} \end{pmatrix} = \overrightarrow{r_{P_{i_0}}} \cdot \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{i-1} \\ s^i \\ \rho_{i+1} \\ \vdots \\ \rho_{m-1} \end{pmatrix} + \cdots + \overrightarrow{r_{P_{i_{k-1}}}} \cdot \begin{pmatrix} \rho_1 \\ \vdots \\ \rho_{i-1} \\ s^i \\ \rho_{i+1} \\ \vdots \\ \rho_{m-1} \end{pmatrix}. \tag{4}$$

Since $\overrightarrow{r_{P_{i_j}}}(\rho_1, \cdots, \rho_{i-1}, s^i, \rho_{i+1}, \cdots, \rho_{m-1})^\tau$ is actually P_{i_j}'s share, the equality (4) implies that the secret can be recovered by finding a path from v_0 to v_i and adding up the shares of the participants associated to the edges of the path. In implementation, we can use the algorithm of random walks on graphs

described in Section 2.2 to find the path, thus we uses only logarithmic space. But the usual deterministic reconstruction algorithm is to obtain a vector $\overrightarrow{w}$ by solving the linear equations $\overrightarrow{e_i} = \overrightarrow{w}M_A$ and then the secret is recovered by $s^i = \overrightarrow{w}(M_A(\rho_1, \cdots, \rho_{i-1}, s^i, \rho_{i+1}, \cdots, \rho_{m-1})^\tau)$. This algorithm uses polynomial space.

Example 4. (following Example 3) We build the secret sharing scheme $LSSS_1$ to realize $AS_{v_0v_1}$ from the MSP $\mathcal{M}(\mathbb{F}_2, M, \psi)$: for a given secret s^1, the dealer randomly selects $\rho_2, \rho_3 \in \mathbb{F}_2$ computes $M(s^1, \rho_2, \rho_3)^\tau$ and transmits the share $s_1 = s^1$ to P_1, $s_2 = s^1 + \rho_2$ to P_2, $s_3 = \rho_2 + \rho_3$ to P_3, $s_4 = \rho_3$ to P_4, $s_5 = s^1 + \rho_3$ to P_5 and $s_6 = \rho_2$ to P_6.

when reconstructing the secret, we first find a path from v_0 to v_1, say $v_0 - v_2 - v_1$ which passes P_6, P_2 in order. Then adding up P_2 and P_6's shares, we get $s_2 + s_6 = (s^1 + \rho_2) + \rho_2 = s^1$. Note that all operations are done over $\mathbb{F}_2$.

So far we have built the secret sharing scheme $LSSS_i$ with respect to $AS_{v_0v_i}$ where $1 \leq i \leq m-1$. Our propose is to build a secret sharing scheme to realize the access structure $AS = \{A \subset P | G(V, E_A) \text{ is a connected graph}\}$. It is obvious that $AS = \bigcap_{i=1}^{m-1} AS_{v_0v_i}$, so we can build the secret sharing scheme as follows [5]:

Distribution Phase: For a given secret $s \in \mathbb{F}_2$, the dealer randomly selects $s^1, \cdots, s^{m-2} \in \mathbb{F}_2$ and sets $s^{m-1} = s - \sum_{i=1}^{m-2} s^i$. Then he shares s^i through the scheme $LSSS_i$ constructed above where $1 \leq i \leq m-1$.

Reconstruction Phase: For any $A \in AS$, since $A \in AS_{v_0v_i}$, participants in A can recover s^i for $1 \leq i \leq m-1$. Then the secret can be recovered by $\sum_{i=1}^{m-1} s^i = s$. Precisely, suppose that for $1 \leq i \leq n$, P_i gets share $(s_{i1}, ..., s_{i(m-1)}) \in \mathbb{F}_2^{m-1}$ after the distribution phase where s_{ij} is the share of s^j from $LSSS_j$ and we call it as the j-th share of P_i, $1 \leq j \leq m-1$. We use the following **Process Rec** i to reconstruct s^i where $1 \leq i \leq m-1$.

Process Rec i
(1)Set the counter $t = 1$; k is the security parameter;
(2)Set $s^i = 0$;
(3)Starting from the vertex v_0, execute a random walk of $2|V|^3$ steps. Each step consists of randomly and uniformly choosing an edge leaving the current vertex and renewing s^i by adding to s^i the i-th share of the player associated to that edge through the map f.
- If the vertex v_i is reached, store the current value s^i and stop;
- If v_i is not met after the $2|V|^3$ steps, set the counter $t \leftarrow t + 1$ and return to (2) while $t \leq k$;
- If $t > k$, set s^i be a value randomly and uniformly chosen in $\mathbb{F}_2$ and stop.

After the $m-1$ processes, we have $m-1$ values $s^1, ..., s^{m-1}$, then the secret is recovered by $s = \sum_{i=1}^{m-1} s^i$ over $\mathbb{F}_2$. From the proof of Proposition 2, we know that the randomized reconstruction algorithm above can output the correct secret except with error probability at most $\frac{m-1}{2^k}$ which is negligible. Usually, set the security parameter be the number of players, *i.e.*, $k = n$. Then our reconstruction

algorithm runs in time $poly(n)$ and uses $O(\log n)$ space with error probability $O(\frac{1}{2^n})$. The deterministic reconstruction algorithm by solving linear equations runs in time $poly(n)$ and uses $O(n)$ space. So our algorithm has more advantage in space usage with negligible error probability.

4 Devising Corresponding MPC Protocols

Since secret sharing schemes are primary tool for MPC, in this section we discuss how to build a statistical MPC protocol by using the secret sharing scheme constructed in Section 3.2. From Proposition 1 we know that the adversary structure $\mathcal{A} = 2^P - AS = \{A \subset P | G(V, E_A)$ is a disconnected graph$\}$ is Q2, then every function can be securely computed against an adaptive $\mathcal{A}$-adversary in the information theoretic model where a broadcast channel is given [6]. Cramer et al. [11] gave a general construction to build a MPC protocol from any LSSS. Next, through a specific example we show how it works. For simplicity, we only deal with passive adversaries and the adversary structure is assumed to be $\mathcal{A}' = 2^P - AS_{v_0v_1} = \{A \subset P \mid v_0$ and v_1 are disconnected in $G(V, E_A)\}$ which is obviously Q2.

Example 5. (following Example 3)

Suppose $P_1, ..., P_6$ are to jointly and securely compute an agreed function $f = x_1 + x_2x_3$ where P_i holds private input x_i for $1 \leq i \leq 3$. Based on the MSP $\mathcal{M}(\mathbb{F}_2, M, \psi)$ which computes $f_{AS_{v_0v_1}}$ with respect to $\overrightarrow{e_1}$, they execute the following steps.

Step 1. For $1 \leq i \leq 3$, P_i shares his private input x_i through $\begin{pmatrix} 1\,0\,0 \\ 1\,1\,0 \\ 0\,1\,1 \\ 0\,0\,1 \\ 1\,0\,1 \\ 0\,1\,0 \end{pmatrix} \begin{pmatrix} x_i \\ \alpha_i \\ \beta_i \end{pmatrix}$, where α_i and β_i are randomly chosen in $\mathbb{F}_2$ by P_i. After that, P_1 gets x_i, P_2 gets $x_i + \alpha_i$, P_3 gets $\alpha_i + \beta_i$, P_4 gets β_i, P_5 gets $x_i + \beta_i$ and P_6 gets α_i.

Step 2. P_1 locally computes x_2x_3 and reshares the result by $M(x_2x_3, \alpha', \beta')^\tau$ where α' and β' are secretly and randomly chosen in $\mathbb{F}_2$ by P_1. So P_1 gets x_2x_3, P_2 gets $x_2x_3 + \alpha'$, P_3 gets $\alpha' + \beta'$, P_4 gets β', P_5 gets $x_2x_3 + \beta'$ and P_6 gets α'. We do this way because of the speciality of the access structure, and a general way is in [11].

Step 3. Every player locally adds up the share for x_1 he gets from Step 1 and the share for x_2x_3 obtained from Step 2, that is P_1 gets $x_1 + x_2x_3$, P_2 gets $x_1 + x_2x_3 + \alpha' + \alpha_1$, P_3 gets $\alpha' + \beta' + \alpha_1 + \beta_1$, P_4 gets $\beta' + \beta_1$, P_5 gets $x_1 + x_2x_3 + \beta' + \beta_1$ and P_6 gets $\alpha' + \alpha_1$. Actually, by doing this every player gets shares for $x_1 + x_2x_3$ through $M(x_1 + x_2x_3, \alpha' + \alpha_1, \beta' + \beta_1)^\tau$.

Step 4. Every player can finally get $x_1 + x_2x_3$ by the reconstruction algorithm of the secret sharing scheme.

When the adversary is active, we need to replace the secret sharing scheme used above by a corresponding verifiable secret sharing scheme where each player

can verify the validity of the shares showed by others [10]. The whole process is complicated and so omitted here. However, we pointed out that by building verifiability into our secret sharing scheme, every player can reconstruct the final result from true shares through random walks on graphs, which needs only logarithmic space much less than general schemes.

5 Conclusion

The advantage of randomized algorithms in efficiency and complexity encourages us to convert a scheme with perfect correctness and security to a scheme with exponentially small error probability and statistical security by randomizing the corresponding algorithms. By this idea we do obtain a statistical multiparty computation protocol based on random walks on graphs. Furthermore, the results of this paper also imply the inherent connections and influences among secret sharing, randomized algorithms, and secure multiparty computation.

References

1. A. Beimel, Secure Schemes for Secret Sharing and Key Distribution, PhD thesis, Technion - Israel Institute of Techonlogy, 1996.
2. A. Shamir, How to share a secret, *Communications of the ACM*, 1979, 22:612-613.
3. A. Yao. Protocols for Secure Computation. *Proc. of IEEE FOGS '82*, pp. 160-164, 1982.
4. Blackley G.R., Safeguarding cryptographic keys, *Proc. of the 1979 AFIPS National Computer Conference*, 1979, 48:313-317.
5. L. Xiao, M. Liu, Linear secret sharing schemes and rearrangements of access structures, *Acta Mathematicae Applicatae Sinica, English Series*, Vol. 20, No. 4, 2004, pp.685-694.
6. M. Hirt, U. Maurer, Player simulation and general adversary structures in perfect multi-party computation, *Journal of Cryptology*, vol.13, NO. 1, pp.31-60, 2000.
7. M. Karchmer and A. Wigderson, On span programs, *Proc. 8th Ann. Symp. Structure in complexity Theory, IEEE 1993*, pp. 102-111.
8. O. Goldreich, S. Micali ,A. Wigderson. How to play ANY mental game. *Proceedings of the nineteenth annual ACM conference on Theory of computing*, pp.218-229, January 1987, New York, New York, United States.
9. Rajeev Motwani, Prabhakar Raghavan. Randomized Algorithms. Cambridge University Press,1995.
10. R. Cramer, I. Damgard, S. Dziembowski, M. Hirt and T. Rabin: Efficient Multiparty Computations with Dishonest Minority, *Proceedings of EuroCrypt 99*.
11. R. Cramer, I. Damgard, U. Maurer., General Secure Multi-Party Computation from any Linear Secret-Sharing Scheme, *Proc. EUROCRYPT '00*, Springer Verlag LNCS, vol 1807, pp. 316–334. Full version available from IACR eprint archive, 2000.
12. T. Rabin, M. Ben-Or, Verifiable Secret Sharing and Multiparty Protocols with Honest majority, *Proc. ACM STOC'89*, pp. 73-85