# Secure Multiparty Computation of a Social Network

Varsha Bhat Kukkala*, S.R.S Iyengar† and Jaspal Singh Saini‡
*Department of Computer Science and Engineering, Indian Institute of technology Ropar*
*Punjab, India - 140001*
*Email: *varsha.bhat@iitrpr.ac.in, †sudarshan@iitrpr.ac.in, ‡jaspal.singh@iitrpr.ac.in*

*Abstract*—**The recent explosion of online networked data and the discovery of universal topological characteristics in real world networks has led to the emergence of a new domain of research, namely, social networks. However, much research in this domain remains unexplored due to the unavailability of sensitive networks, which include hate networks, trust networks and sexual relationship networks. This paper proposes a secure multiparty protocol which allows a set of parties to compute the underlying network on them. The proposed protocol is information theoretic secure, and its security is further enhanced by a list of security tests, which include, k-anonymity test, check for self loops and weighted edges. Although some solutions have been proposed for this problem earlier, the practicality of each one of those is questionable.**

*Index Terms*—**Multiparty computation, Social networks, Anonymization**

## 1. Introduction

Social networking platforms such as Facebook, Twitter, LinkedIn, LiveJournal allow individuals to stay connected. These networks can be visualized as graphs of social actors (individuals and/or organizations) seen as nodes, and the dyadic ties bridging them as edges. These edges in the network can represent a variety of positive as well as negative relationships - friendship/enmity, common interests/dissimilarity, trust/distrust. Although these platforms serve as a rich source of data related to the social interactions, accessibility of this data is not at par with its availability. This is primarily attributed to the privacy concerns of the involved individuals. With the emerging interest in the analysis of these social networks, the difficulty in accessing the underlying network has been a major issue.

Social Network Analysis (SNA) is a three step process involving the collection of data (i.e. the network), its analysis and inferring the theories that explain the patterns observed in these structures [34]. It is known that the network structure affects the underlying dynamics such as information flow, social behavior, small world phenomenon, as well as structural properties such as existence of communities and hubs. A few case studies where the analysis of the underlying topology has proven to be beneficial are: the transmission of diseases [29], [31], classification of the influence or popularity of individuals [28], [10], evolution of a social network [2], the flow of information in a social network [11], measuring influence of a publication [23]. However, the usage of SNA is subject to the availability of the underlying graph. That is, only with the social network data at our disposal can we proceed with its analysis.

There are several studies conducted on networks with sensitive information, and the data has mostly been acquired through surveys [19], [21], [29] and then anonymized[1]. The fear of sensitive information being leaked prevents most of the users from sharing their local network data [6], [27], [16] or could even lead to reporting false information. Hence, the survey results are generally less reliable. Thus, there is a necessity for a protocol that generates the underlying network securely, by amalgamating the data that is available distributedly. It must be done in a way that privacy and integrity of the inputs is ensured while guaranteeing the correctness of the output. This is precisely what constitutes a *multiparty computation*. It involves a set of parties who follow a specific protocol for computing a function of their private data. The process must ensure that nothing but the final result is revealed. The first MPC protocol was proposed by Yao [35], which allowed two parties to compare and determine who among the two are richer, without revealing each others wealth.

The current paper proposes a protocol that allows a bunch of parties to get together and securely compute their underlying network, where the edges are held distributedly. This is achieved without the help of any third party. A few instances where the protocol to compute the graph can be put to use include - generation of *sexual networks* by the concerned individuals themselves; analyzing the stability of professional relationships among employees, through the underlying *hatred network*; studying the correlation of productivity with the underlying *trust network* in a team. The proposed protocol will help determine the structural properties of sensitive networks, which would otherwise

1. Anonymization is a process in which, only the interconnections between the social actors are retained. Every other information about the social actor is removed (i.e. name, email-address etc.)

not be possible. It can further aid in understanding the effects of the network structure on social behavior [30]. The structural properties of real world networks that have been previously studied - scale free property [3], [4], [24], [32], presence of communities [17], presence of structural holes [9], navigability of the network [7], structural balance in signed networks [22], etc., can also be validated.

## 2. Related Work

General protocols have been proposed for securely evaluating any computable function [18], [12], [5]. These theoretical models cannot be put to practical use due to large communication and computation overhead involved. Thus, efficient protocols have been further proposed for specific problems, such as computing approximations on distributed data [13], auctions [26], private matching and set intersection [14], secure rank computation [1], privacy preserving data classification [37] and data mining [25].

Some researchers have also looked at computing certain aspects of a network securely. Brickell and Shmatikov [8] look at two party protocols for computing graph algorithms securely. Here, they assume that the underlying graph is already known, and is used as the party's private data in the protocol. Hu, Chow and Lau, in their work [20], discuss on how one can detect people belonging to the same community with minimum information being leaked. Such a detection allows to suggest friends in a social network. Zeng et al. also propose a technique for secure link prediction in online social networks [36].

Securely generating the underlying graph has been previously studied by Frikken and Golle [15]. It is assumed that the network information is held in a distributed manner where each individual possesses some partial information of the network. The drawbacks of the protocol is that it uses additional dummy parties called authorities, who help compile the collected data into the required graph. Also, the use of *threshold Elgamal encryption* scheme and *re-encryption mix nets* in the protocol, amounts to increased communication and computation cost. It is to be noted that the protocol in [15] is cryptographically secure. The protocol proposed in the current paper avoids the use of dummy parties and is information theoretically secure, thereby overcoming the above mentioned drawbacks.

## 3. Preliminaries and Definitions

The individuals who wish to compute the underlying network are termed as *parties*, labelled as $P_1, P_2, \ldots, P_n$. The information regarding the edges of underlying graph on these parties, is assumed to be held distributedly. Each party contributes her out going links, as private input to the protocol. The protocol consists of parties communicating messages amongst each other, across a secure channel. A message from $P_i$ intended only to $P_j$ is known to no other party, unless they choose to share it.

The adjacency vector of a party $P_i$ is represented as $v_i = [a_{i1}, a_{i2}, \ldots, a_{in}]$, where $a_{ij}$ equals 1 if there exists a directed edge from $P_i$ to $P_j$, otherwise it equals 0. Therefore, the adjacency matrix $A = [a_{ij}]_{n\times n}$.

Each party $P_i$ possess a share $A_i$ of the adjacency matrix $A$, where the $n \times n$ matrix $A_i$ contains the vector $v_i$ in the $i^{th}$ row and zeroes in all the other entries of the matrix.

Two graphs $G$ and $G'$ (with the adjacency matrices $A$ and $A'$ respectively) are isomorphic if there exists a bijection $\sigma$ from the vertex set of $G$ to the vertex set of the $G'$, such that the edges are preserved. For simplicity, $A'$ is represented as $\sigma(A)$. An illustration of applying a permutation on a graph is as shown in Figure 1.

$$\overset{\text{A}}{\begin{array}{c|cccc|} & c_1 & c_2 & c_3 & c_4 \\ r_1 & 0 & 1 & 1 & 0 \\ r_2 & 1 & 0 & 1 & 0 \\ r_3 & 0 & 0 & 0 & 1 \\ r_4 & 1 & 0 & 0 & 0 \end{array}} \xrightarrow{\sigma} \overset{\text{A'}}{\begin{array}{c|cccc|} & c_1 & c_4 & c_2 & c_3 \\ r_1 & 0 & 0 & 1 & 1 \\ r_4 & 1 & 0 & 0 & 0 \\ r_2 & 1 & 0 & 0 & 1 \\ r_3 & 0 & 1 & 0 & 0 \end{array}}$$

Figure 1. Action of $\sigma$ = (2 4 3) in cycle notation of a permutation

Throughout the paper, all calculations are assumed to be in arithmetic modulo $q$. A matrix $R \in_R \mathbb{R}_{n\times n}$ represents a random $n \times n$ matrix, where all its elements are picked uniformly at random from $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$.

*Secret Sharing Protocol*: allows a party $P$ to share her secret with $n$ parties (including $P$) such that each party has a share (some information) of the secret. The information contained in a share owned by a party is not enough to compute the secret. But, once all the parties (including $P$) combine their shares, the secret can be computed.

Let $x \in \mathbb{Z}_q$ be the secret of party $P$ which she wants to share. Party $P$ chooses $n$ random numbers $\{x_1, x_2, \ldots, x_n\}$ from $\mathbb{Z}_q$ such that $x = \sum_{i=1}^{n} x_i$. Each $x_i$ would be a share of the secret $x$. Party $P$ gives each party exactly one share. Therefore, only when all the parties collaborate together can they generate the secret $x$, by summing up their shares.

*Protocol for Secure Addition*: allows the $n$ parties to compute the sum of their secrets, without revealing their individual secret to anyone. A stepwise description is as given below:

- Each party shares her secret $x_i$ with all the parties using the secret sharing protocol mentioned above. The $n$ shares of secret $x_i$ are denoted as

$\{x_{i1}, x_{i2}, \ldots, x_{in}\}$, where the share $x_{ij}$ is sent by party $P_i$ to party $P_j$.

- Each party $P_i$ computes the sum $S_i$ over all the shares she receives, i.e. $S_i = \sum_{j=1}^{n} x_{ji}$. This partial sum of few shares is released in public.
- All the parties now compute $\sum_{i=1}^{n} S_i$ to get the final desired sum.

## 4. A Naive Solution

The crux of the solution is to find an isomorphic graph of the original graph. Our aim, therefore, is to construct a permuted adjacency matrix $\sigma(A)$. The permutation $\sigma$ should be known to no individual, since one can easily backtrack to $A$, using the permutation $\sigma$ and permuted adjacency matrix $\sigma(A)$.

The protocol begins with each party $P_i$ picking two random matrices $R_{i1}$ and $R_{i2}$, such that $A_i = R_{i1} + R_{i2}$. Here, $A_i$ is party $P_i$'s share of the adjacency matrix $A$, supplied as her secret input to the protocol. Further, each party $P_i$ passes the first share $R_{i1}$ of her secret $A_i$ to party $P_1$. The second share $R_{i2}$ is sent to party $P_2$. Parties $P_1$ and $P_2$ then sum over all the corresponding shares they received. Hence, the adjacency matrix $A$ is currently held distributedly by parties $P_1$ and $P_2$. Next, both the parties $P_1$ and $P_2$ privately agree on a permutation $\sigma_1$. They further apply the permutation $\sigma_1$ to their shares and pass it to party $P_3$. Party $P_3$ can compute $\sigma_1(A)$ by summing over the received two shares. Since, $\sigma_1$ is not known to party $P_3$, she cannot compute $A$ from the pieces of information she holds. Party $P_3$ further applies a permutation $\sigma_2$ and release $\sigma_2(\sigma_1(A))$ in public. A schematic representation of the same is shown in Figure 2.

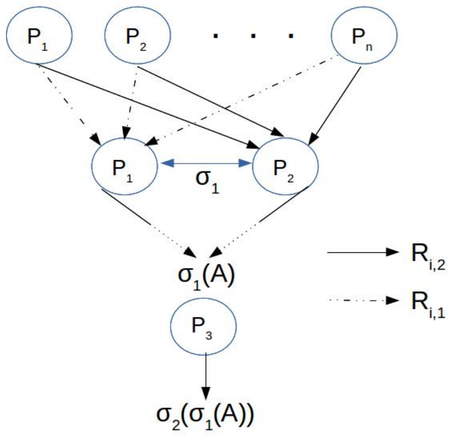

Figure 2. Naive Protocol

Although the protocol seems to be secure, it has certain pitfalls. Clearly, parties $P_1, P_2, P_3$ are more powerful than the others. Collaboration of these powerful parties would lead to breach in privacy. Consider the case where parties $P_1$ and $P_2$ collaborate. They can compute $A$ directly by summing over all their shares. Another dangerous collaboration is that of $P_3$ with either $P_1$, $P_2$ or both. Even when all the parties are honest and do not collaborate, there are still chances of information leak due to the structural properties of the output graph. For example, consider the case where a party $P_j$ has a unique out-degree in the network. This would imply that party $P_j$ can easily re-identify herself and her neighbours in the final isomorphic version.

The paper proposes a secure multiparty graph computation protocol (SMPGC) in the next section. The SMPGC protocol improves on this naive protocol by distributing the work equally among all parties, thus making no party important or special. It also minimizes the chances of information leak due to structural properties of the graph using various techniques discussed in Section 6.1.

## 5. SMPGC Protocol

The SMPGC protocol can be briefly decomposed into two stages. The first stage securely computes the isomorphic version of the underlying graph, held distributedly. Therefore, no individual party has the entire isomorphic graph, at this stage. The second stage performs a few security tests (discussed in Section 6.1) to determine if the computed result can be made public or if the protocol needs to be aborted. The isomorphic graph is released in public only after the desired security checks are met.

The protocol requires active participation of $m^2$ of the $n$ parties, selected at random, such that $m = \lfloor \sqrt{n} \rfloor$. These $m^2$ parties are arranged in a layered structure as shown in Figure 3. $P_{i,j}$ would represent the $j^{th}$ party in the $i^{th}$ layer of the given structure. The stepwise description of the protocol is given below:

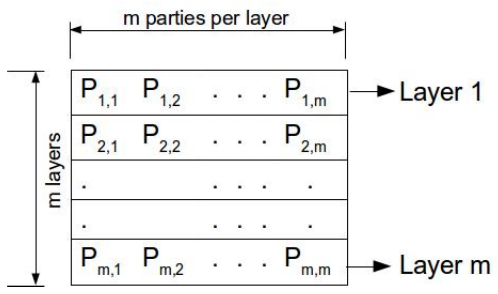

Figure 3. Layered structure

Step 1: The focus here lies on data collection. Each party $P_i$ shares her secret $A_i$ with the parties in layer 1 using the secret sharing protocol. Hence party $P_{1,j}$ receives the $R_{i,j}$ ($j^{th}$ share of $A_i$). Therefore, each party in layer 1 receives $n$ random matrices. In particular, party $P_{1,j}$ receives $R_{1,j}, R_{2,j}, \ldots, R_{n,j}$. Each party $P_{1,j}$ in layer 1 now computes its secret $S_{1,j}$ as given below:

$$S_{1,j} = \sum_{i=1}^{n} R_{i,j} \qquad (\forall 1 \leq j \leq m)$$

Step 2: The data collected is processed at each level and communicated to parties in the next layer as shown below:

```
for l := 1 to m-1 do                                  ▷ l iterates over all the layers
    Party $P_{l,1}$ picks a random permutation $\sigma_l$ and shares it with $P_{l,2}, P_{l,3} \dots, P_{l,m}$.
    for i := 1 to m do
        Using secret sharing protocol, $P_{l,i}$ sends $m$ random shares of $\sigma_l(S_{l,i})$ to all the parties in layer $l+1$ i.e. parties $P_{l+1,1}, P_{l+1,2}, \dots, P_{l+1,m}$.
    end for
    for j := 1 to m do
        Each party $P_{l+1,j}$, adds all the shares received to obtain $S_{l+1,j}$
    end for
end for
```

Similarly, parties in the last layer fix on a permutation $\sigma_m$ and apply it on their secrets. At the end of this step, the parties in the last layer collectively hold $\sigma(A)$, where

$$\sigma = \prod_{i=1}^{m} \sigma_i = \sigma_m \circ \sigma_{m-1} \circ \dots \circ \sigma_1 \qquad (1)$$

This marks the end of stage 1 of the protocol.

Step 3: $P_{m,1}$ picks a bijection $f : M_{n \times n} \to M_{n \times n}$ uniformly at random, satisfying the following two properties:

- The rows of the matrix are shuffled
- For each row in the matrix, its entries are shuffled; independent of the shuffling on the other rows

Party $P_{m,1}$ then shares $f$ with all the $m$ parties in the last layer. Each of the $(n!)^n$ possibilities for the function $f$ is termed as *row-wise permutation functions* henceforth. An example depicting the application of $f$ on a matrix $M$ is shown in the Figure 4.

$$\overset{\text{M}}{\begin{array}{c} r_1 \\ r_2 \\ r_3 \\ r_4 \end{array} \begin{pmatrix} 10 & 4 & 6 & 3 \\ 2 & 19 & 7 & 4 \\ 24 & 3 & 4 & 5 \\ 3 & 16 & 13 & 9 \end{pmatrix}} \xrightarrow{f} \overset{\text{f(M)}}{\begin{array}{c} r_2 \\ r_3 \\ r_1 \\ r_4 \end{array} \begin{pmatrix} 19 & 4 & 7 & 2 \\ 24 & 4 & 3 & 5 \\ 3 & 6 & 4 & 10 \\ 13 & 9 & 16 & 3 \end{pmatrix}}$$

Figure 4. Action of a bijection $f$ on matrix $M$

Each party $P_{l,i}$ in the last layer shares $f(S_{l,i})$ with layer 1 using secret sharing protocol.

Step 4: Each party $P_{1,i}$ in layer 1 sums over all the matrices she receives from the last layer and these new matrices are represented as $S'_{1,i}$. We further run the below shown algorithm.

```
for l := 1 to m-1 do
    $P_{l,1}$ picks a row-wise permutation function $f_l$ uniformly at random and share it with $P_{l,2}, P_{l,3} \dots, P_{l,m}$
    for i := 1 to m do
        Party $P_{l,i}$ sends random shares of $f_l(S'_{l,i})$ to all the parties in layer $l+1$
    end for
    for j := 1 to m do
        Each party $P_{l+1,j}$, adds all the shares received to obtain $S'_{l+1,j}$
    end for
end for
```

Step 5: Parties in the last layer $P_{m,1}, P_{m,2}, \dots, P_{m,m}$ perform secure addition of their secrets $S'_{m,1}, S'_{m,2}, \dots, S'_{m,m}$ respectively and release the sum (lets say $S'$) in public.
Few security tests described in Section 6.1 are performed on this output matrix $S'$. If no malicious behaviour by any party is detected, $\sigma(A)$ calculated at the end of Step 2 is released in public, else the protocol is aborted.

## 6. Security of The Protocol

This section discusses the security tests to be performed in the second stage of the SMPGC protocol. These tests are meant to detect possibility of information leak from the final output, which could be either due to malicious behaviour of parties in the protocol execution or due to the output graph structure itself. Further we discuss on the various security models in MPC. In particular, we discuss on the Honest model, the Semi-honest model and the Malicious model.

### 6.1. Security Tests

The SMPGC protocol ensures that the parties do not learn any additional information in the intermediate stages (Section 6.2). However, the final graph computed might inherently posses structure such that re-identification is possible. For example, a party with unique out degree can positively identify herself from the final graph and hence learn some information regarding her neighbours as well. There is also a possibility for a party to deviate from the protocol to facilitate re-identification. For all of these reasons, we propose a set of tests that identify scenarios that may lead to information leak.

**Check for Self Loops**: In most of the scenarios, the underlying network may not have any self loops, as in the case of frenemy networks and sexual networks. The appearance of self loops in such cases suggest misbehaviour and need to be avoided. This can be taken care of with a minor addition to Step 2 of the protocol. At the end of step 2, all the parties in the last layer contain shares of $\sigma(A)$. We use the fact that the diagonal entries of $A$ continue to be the diagonal entries of $\sigma(A)$ (although permuted). Each party on the last layer can create a vector containing the diagonal entries of the shares of $\sigma(A)$. They can further perform a secure sum of these vectors to obtain the permuted entries of the diagonal of $A$. Hence they can check whether all the entries are zero or not.

**Check for Weighted Edges**: To generate unweighted networks securely, the adjacency matrix of the output isomorphic graph must contain just zeroes and ones. A malicious party may falsely report weighted edges to facilitate re-identification of nodes in the output adjacency matrix. Since all the entries of $A$ are present in $\sigma(A)$ and hence in $S'$ (which is available in public), we can look for non-zero-one entries in this matrix $S'$ and confirm that the underlying network is unweighted.

**k-Anonymity Test**: If a party in the underlying network has a unique degree, she can find her position in the released isomorphic graph easily and this can lead to information leakage. As a precaution, *k-anonymity* test was proposed in [33], and it has since been an active area of research. A graph is said to be k-anonymized if for every individual in the network, there exist atleast $k-1$ other individuals in the network with the same degree. The publicly available matrix $S'$ helps us perform the k-anonymity test on the underlying graph securely. However, the real challenge lies in making the underlying graph k-anonymized (in case it is not). This is done by adding and/or removing a few edges from the network and can be achieved using the following technique:

1) All the parties agree on a matrix $K$ which should have been added to the matrix $S'$ so that the underlying graph is k-anonymized.
2) The last layer distributedly hold the matrix $K$ using the secret sharing protocol. Each party on the last layer performs $f_l^{-1}$ on her share and passes it to the upper layer using the secret sharing protocol.
3) The above process is repeated until parties in layer one perform $f_1^{-1}$ on their shares and pass it to the last layer.
4) Each party $P_{l,i}$ on the last layer can now add the received share to her previously held share $S_{l,i}$, and release the sum in public. Therefore, the final output matrix is a k-anonymized perturbed matrix of the original isomorphic graph.

**A Lower Bound and an Upper Bound on Degree**: In some scenarios, we may want to bound the out-degree of all the parties. For example, in a friendship graph of size 5000, we may want to exclude individuals with an out degree as high as 500. This requires a constraint on all the participants to report degree lower than 500. The maximum and minimum degree of the graph can be checked using the matrix $S'$ and hence this test can be performed with ease.

## 6.2. Security Models

There are several behavioural aspects of a party that could affect the security of the protocol. Each case is considered separately and its security discussed in detail.

**Honest Model**: A party is said to be *honest* when she neither deviates from the protocol nor does she share her *view* with other parties. View of a party here refers to all the data she sees throughout the execution of the protocol. Since each party sees only random matrices throughout the protocol and a single permutation of the net $m$ permutations masking the adjacency matrix $A$ (Eqn 1), the protocol is secure in the honest model.

**Semi-honest Model**: In this model, no party deviates from the protocol. However a set of corrupt parties may collaborate with the aim of revealing the private data of the honest parties. There are only two types of collaborations which can leak information in such a case:

- Since $\sigma(A)$ is known publicly, one method to reveal $A$ is to determine all the $\sigma_i$'s. Therefore, the adjacency matrix $A$ can be leaked if there exists atleast one corrupt party from each layer. This implies that atleast $m$ corrupt parties are required for breaking the protocol.
- Consider another case, where all the parties in a layer $l$ collaborate. A secure sum over their secrets $S_{l,i}$'s would reveal $\sigma_{l-1}(\sigma_{l-2}(\ldots(\sigma_1(A))))$ to the corrupt parties. To further reveal $A$, a corrupt party is required from each of the layers $1, 2, \ldots, l-1$. Therefore, atleast $(m + l - 1)$ corrupt parties are required in this case.

From the above discussion, we conclude that the threshold[2] for the number of corrupt parties in the semi-honest model is $\lfloor\sqrt{n}\rfloor$.

**Malicious Model**: In this model, a corrupt party may even deviate from the protocol with the hope to reveal some information about the honest parties. If a corrupt party tries to add self loops, add weighted edges or make her degree unique, such behaviour can be detected using the security tests proposed in Section 6.1. However, if a malicious party $P_{l,i}$ makes a change of $\pm 1$ to an entry of her share $S_{l,i}$, then it may go undetected if that particular change results in the entry converting from 0 to 1 or vice versa. For $k$ such changes (let's say from 1 to 0) of a malicious party to go undetected, the malicious party must always pick an entry which corresponds to 1. Therefore, the probability of this event occurring is $\approx \left(|E|/\binom{n}{2}\right)^k$ i.e. it exponentially decreases as $k$ increases, where $|E|$ is the number of the edges in the network.

## 7. Conclusion

Despite the recent explosion of networked data due to the WWW, sensitive data remains scarcely accessible due

2. We say that the threshold of a protocol is $k$ if no subset of $k-1$ corrupt parties can reveal any information about the honest parties

to privacy concerns. This paper proposes an information theoretic secure protocol for computing a social network present on a set of individuals. Various tests including the k-anonymity test, self loops check and weighted edges check help further strengthen the security of the protocol. The proposed protocol is efficient in terms of the computation and communication cost. The SMPGC protocol also provides a sense of flexibility, since it can be easily fine tuned to handle the case of unweighted graphs and labelled nodes/edges. This protocol can be used to test the various hypothesis of real world networks (like scale free degree distribution, clustering, community structure, homophily, etc) on currently hidden sensitive social networks. Hence this work can be of great value to the network science community.

## References

[1] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the k th-ranked element. In *Advances in Cryptology-EUROCRYPT 2004*, pages 40–55. Springer, 2004.

[2] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54. ACM, 2006.

[3] A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69–77, 2000.

[4] A.-L. Barabási et al. Scale-free networks: a decade and beyond. *science*, 325(5939):412, 2009.

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 1–10. ACM, 1988.

[6] J. Black. The perils and promise of online schmoozing. *BusinessWeek Online, February*, 20:2004, 2004.

[7] M. Boguna, D. Krioukov, and K. C. Claffy. Navigability of complex networks. *Nature Physics*, 5(1):74–80, 2009.

[8] J. Brickell and V. Shmatikov. Privacy-preserving graph algorithms in the semi-honest model. In *Advances in Cryptology-ASIACRYPT 2005*, pages 236–252. Springer, 2005.

[9] R. S. Burt. Structural holes and good ideas1. *American journal of sociology*, 110(2):349–399, 2004.

[10] M. Cha, H. Haddadi, F. Benevenuto, and P. K. Gummadi. Measuring user influence in twitter: The million follower fallacy. *ICWSM*, 10(10-17):30, 2010.

[11] M. Cha, A. Mislove, and K. P. Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 721–730, New York, NY, USA, 2009. ACM.

[12] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19. ACM, 1988.

[13] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. J. Strauss, and R. N. Wright. Secure multiparty computation of approximations. In *Automata, Languages and Programming*, pages 927–938. Springer, 2001.

[14] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology-EUROCRYPT 2004*, pages 1–19. Springer, 2004.

[15] K. B. Frikken and P. Golle. Private social network analysis: How to assemble pieces of a graph privately. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 89–98. ACM, 2006.

[16] L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer-Mediated Communication*, 3(1):0–0, 1997.

[17] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[18] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229. ACM, 1987.

[19] S. Helleringer and H.-P. Kohler. Sexual network structure and the spread of hiv in africa: evidence from likoma island, malawi. *Aids*, 21(17):2323–2332, 2007.

[20] P. Hu, S. S. Chow, and W. C. Lau. Secure friend discovery via privacy-preserving and decentralized community detection. *arXiv preprint arXiv:1405.4951*, 2014.

[21] H. Ibarra. Homophily and differential returns: Sex differences in network structure and access in an advertising firm. *Administrative science quarterly*, pages 422–447, 1992.

[22] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1361–1370. ACM, 2010.

[23] N. Li and D. Gillet. Identifying influential scholars in academic social media platforms. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 608–614. ACM, 2013.

[24] F. Liljeros, C. R. Edling, L. A. N. Amaral, H. E. Stanley, and Y. Åberg. The web of human sexual contacts. *Nature*, 411(6840):907–908, 2001.

[25] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in CryptologyCRYPTO 2000*, pages 36–54. Springer, 2000.

[26] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 129–139. ACM, 1999.

[27] A. Newitz. Defenses lacking at social network sites. *SecurityFocus, December*, 31, 2003.

[28] M. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. *Complex networks*, pages 337–370, 2004.

[29] J. Potterat, L. Phillips-Plummer, S. Muth, R. Rothenberg, D. Woodhouse, T. Maldonado-Long, H. Zimmerman, and J. Muth. Risk network structure in the early epidemic phase of hiv transmission in colorado springs. *Sexually transmitted infections*, 78(suppl 1):i159–i163, 2002.

[30] D. Robson. Anti-social network: Health risks of love-hate friends, 2015.

[31] M. Salathe, M. Kazandjieva, J. W. Lee, P. Levis, M. W. Feldman, and J. H. Jones. A high-resolution human contact network for infectious disease transmission. *Proceedings of the National Academy of Sciences*, 107(51):22020–22025, 2010.

[32] A. Schneeberger, C. H. Mercer, S. A. Gregson, N. M. Ferguson, C. A. Nyamukapa, R. M. Anderson, A. M. Johnson, and G. P. Garnett. Scale-free networks and sexually transmitted diseases: a description of observed patterns of sexual contacts in britain and zimbabwe. *Sexually transmitted diseases*, 31(6):380–387, 2004.

[33] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[34] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[35] A. C.-C. Yao. Protocols for secure computations. In *FOCS*, volume 82, pages 160–164, 1982.

[36] Y. Zheng, B. Wang, W. Lou, and Y. T. Hou. Privacy-preserving link prediction in decentralized online social networks. In *Computer Security–ESORICS 2015*, pages 61–80. Springer, 2015.

[37] Z. Y. S. Zhong and R. N. Wright. Privacy-preserving classification of customer data without loss of accuracy. In *SIAM International Conference on Data Mining (SDM), Newport Beach*. SIAM, 2005.