Computing Node Clustering Coefficients Securely

Katchaguy Areekijseree
Syracuse University
kareekij@syr.edu

Yuzhe Tang
Syracuse University
ytang100@syr.edu

Sucheta Soundarajan
Syracuse University
susounda@syr.edu

***Abstract*—When performing any analysis task, some information may be leaked or scattered among individuals who may not willing to share their information (e.g., number of individual's friends and who they are). Secure multi-party computation (MPC) allows individuals to jointly perform any computation without revealing each individual's input. Here, we present two novel secure frameworks which allow node to securely compute its clustering coefficient, which we evaluate the trade off between efficiency and security of several proposed instantiations. Our results show that the cost for secure computing highly depends on network structure.**

1. Introduction

While network analysis is important, in many real applications, data may be scattered among parties who wish to keep their information private. In such cases, it is still often desirable to conduct standard graph analysis tasks, such as community detection, identifying influential nodes, etc. Existing algorithms for distributed network analysis (e.g., MapReduce) leak a great deal of private information. On the other hand, secure multi-party computation (MPC) is to allow mutually-distrusting individuals to jointly compute the output of some function of their individual inputs.

While any algorithm can be converted into a fully-secure representation (e.g., using Yao's Garbled Circuit [1]), these methods are mostly conceptual, as they are dramatically inefficient. Our long-term goals are to develop a 'library' of secure MPC techniques for performing graph operations, which can then be combined into more sophisticated network analysis techniques. In this paper, we present two different frameworks of converting the clustering coefficient computation into different MPC primitives, and for each framework, give multiple instantiations to demonstrate the efficiency-security tradeoff.

ASONAM '19, August 27-30, 2019, Vancouver, Canada

ACM ISBN 978-1-4503-6868-1/19/08.
http://dx.doi.org/10.1145/3341161.3342946

2. Computing Clustering Coefficient

Here, we demonstrate multiple ways in which a node u can compute its own triangle participation count (TPC). From this, the node can trivially compute its clustering coefficient. In a non-secure setting, a node's TPC is computed by iterating over all pairs of its neighbors and counting how many of those pairs are themselves connected. In a secure setting, a node does not know its neighbors' connections. Here is a list of MPC operations, which can be computed securely in various ways [2]:

1) **Secret Sharing** splits some object (a number, a set, etc.) among several parties so that no single party can recover the value of the object.
2) **Private Set Intersection Cardinality (PSIC)**, **Private Set Union (PSU)**, and **Secure Sum (SECSUM)** compute the sizes of the sets intersection, the union of multiple sets, and the sum of numbers, and secret share the results.

This suggests a natural construction ($C1$) for counting triangles. Let u's neighbors are $nb(u) = \{v_1, ..., v_k\}$. For every $v_i \in nb(u)$, u and v_i run the PSIC subprotocol between the $nb(u)$ and $nb(v_i)$ with the cardinality result secret-shared between u and v_i. Then u and $v_i \in nb(u)$ run $SECSUM$ among themselves to obtain the count of triangles. We can consider four instantiations of $C1$, as follows:

$C1_1$ naturally follows $C1$ by scheduling $PSIC$ on u and $v_i \in nb(u)$, secret-sharing this result between u and v_i, and performing $SECSUM$ on u, v_{1-k}. $SECSUM$ requires $(k+1)$-party, but $deg(u)$ is leaked. In $C1_2$, u randomly selects a 'representative' node $v^{'} \in nb(u)$. For each $PSIC$, neighbor v_i secret-shares its neighborhood between u and v', and u performs $PSIC$ between itself and v'. $SECSUM$ is then run only on u and v'. Thus, it makes $C1_2$ much more efficient than $C1_1$, but $deg(u)$ is still leaked to v'. $C1_3$ is similar to $C1_1$, except that u only runs the $PSIC$ secure operation. This instantiation leaks the $PSIC$ results (u learns common neighbors of each v_i). $C1_3$ is fast because fewer computations are done securely. $C1_4$ has no secure computations (fastest but insecure). Information of u's and its neighbors' connections are leaked.

We can also define construction $C2$: Given node u, $C2$ runs PSU on $nb(v_1), ..., nb(v_k)$ to generate a multiset of u's 2-hop neighbors, and then runs a $PSIC$ between the set of u's neighbors and the multiset of u's 2-hop neighbors. The same security-efficiency tradeoff that we saw for $C1$ can be obtained for $C2$ by defining instantiations $C2_1, ..., C2_4$ in a similar manner.

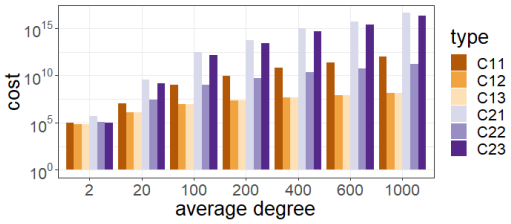
The Costs of MPC: This security comes at a cost of gross inefficiency. Instead, secure algorithm designers typically build an secure algorithm, and accept some information leakage. The costs of $PSIC$ or PSU operations depends on sizes of the sets being considered and the number of parties in the operation, but the cost of $SECSUM$ depends on the number of parties involved.

To estimate the cost, we express secure algorithms in a Boolean/arithmetic circuit and count the number of AND gates, using the classic GMW MPC protocol [2]. We use the number of AND gates to model the computation cost, which is defined as $C = m \cdot n \cdot (n-1)/2$, where n and m is the number of parties and AND gates. So, we get: $C[PSIC] = (M_1 + M_2) \cdot log(M_1 + M_2) \cdot (n^2 - n)/2$, $C[PSU] = (\sum_{i=1}^{n} M_i) \cdot log(\sum_{i=1}^{n} M_i) \cdot (n^2 - n)/2$, and $C[SECSUM] = (n^3 - n^2)/2$.

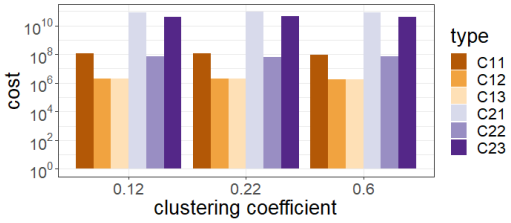
3. Experiments

As mentioned, the costs of the primitive operations depend on the number of parties and sizes of the input sets. Each node runs its own MPC protocol, so we are not interested in the *overall* computation cost of all nodes in the network, but rather the $total$ cost per node. We do not show results for $C1_4$ or $C2_4$, do not use MPC, and so are fast.

Experimental Setup: We consider three network properties that could computational cost: average degree, degree distribution, and clustering coefficient. We generate 2000-node networks according to the Erdos-Renyi (binomial degree distribution) and LFR [4] (power law) models.



(a) Results when varying average degree



(b) Results when varying clustering coefficient

Figure 1: Average total computation cost of each node on networks with different average degrees and clustering coefficients. Costs increase with degree.

Results: Figure 1 shows the average total computation cost of every node as average degree and clustering coefficient vary, on ER networks (results on LFR networks were similar).

The computational cost increases as average degree increases, since it highly depends on number of parties (neighboring nodes) involved during the secure computation. Surprisingly, clustering coefficient (which plays a role in the size of the overlap between a node's neighbors' neighborhoods) has little effect on the computation cost. Instantiations $C1_2$ and $C1_3$ have similar computation cost, but $C1_3$ is less secure ($PSIC$ is done outside MPC protocol, the information of node's neighbors may be leaked). $C2_1$ and $C2_3$ consistently have the highest costs. Overall, the $C2$ construction is more expensive than $C1$.

3.1. Related Work

There has been limited work on secure graph mining. Nayak, et al. introduce GraphSC, which outsources computation to two cloud providers [5]. Wu, et al. consider finding shortest paths while preserving privacy [6]. Hu, et al. show how to use random walks to find communities, so that a node's community membership and neighborhood information are not leaked outside its community [7].

Recent research has been dedicated to building practical MPC software systems [8]. Practical systems are built on top of MPC protocols including Yao's garbled circuits [1] or the GMW protocol [2]. There is an important set of research on privacy-preserving graph mining [9].

4. Conclusions

The purpose of this work was to demonstrate, using clustering coefficient as an example, how one can design secure MPC algorithms for conducting secure, distributed network analysis. We proposed two high-level constructions, $C1$ and $C2$, and for each, described four instantiations corresponding to different levels of security. Our long-term goal is to develop a library of secure graph primitive operations, which can then be combined into more sophisticated techniques, such as community detection or link prediction.

References

[1] A. C. Yao, "How to generate and exchange secrets (extended abstract)," in *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, 1986, pp. 162–167.

[2] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, 1987, pp. 218–229.

[3] A. Ben-David, N. Nisan, and B. Pinkas, "Fairplaymp: a system for secure multi-party computation," in *ACM Conference on Computer and Communications Security*, 2008, pp. 257–266.

[4] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical review E*, vol. 78, no. 4, p. 046110, 2008.

[5] K. Nayak, X. S. Wang, S. Ioannidis, U. Weinsberg, N. Taft, and E. Shi, "Graphsc: Parallel secure computation made easy," in *2015 IEEE Symposium on Security and Privacy, SP*, 2015, pp. 377–394.

[6] D. J. Wu, J. Zimmerman, J. Planul, and J. C. Mitchell, "Privacy-preserving shortest path computation," in *Network and Distributed System Security Symposium*, 2016.

[7] P. Hu, S. Chow, and W. C. Lau, "Secure friend discovery via privacy-preserving and decentralized community detection," in *ICML Workshop on Learning, Security, and Privacy*, 2014.

[8] B. Kreuter, A. Shelat, B. Mood, and K. R. B. Butler, "Pcf: A portable circuit format for scalable two-party secure communication," in *USENIX Security Symposium*, 2013, pp. 321–336.

[9] X. Wu, X. Ying, K. Liu, and L. Chen, "A survey of privacy-preservation of graphs and social networks," *Managing and Mining Graph Data*, vol. 40, pp. 421–453, 2010.