Location Privacy for Group Meetups

A.K.M. Mustafizur Rahman Khan
University of Melbourne
Victoria, Australia
khank@student.unimelb.edu.au

Lars Kulik
University of Melbourne
Victoria, Australia
lkulik@ unimelb.edu.au

Egemen Tanin
University of Melbourne
Victoria, Australia
etanin@ unimelb.edu.au

ABSTRACT

A Group Nearest Neighbor (GNN) query finds a point of interest (POI) that minimizes the aggregate distance for a group of users. In current systems, users have to reveal their exact, often sensitive locations to issue a GNN query. This calls for private GNN queries. However, existing methods for private GNN queries either are computationally too expensive for mobile phones or cannot resist sophisticated attacks. Our approach can efficiently and effectively process an important variant of private GNN queries: queries that minimize the maximum distance for any user in the group. To achieve high efficiency we develop a distributed multi-party private protocol to compute the maximum function. Our method exploits geometric constraints to prune POIs and avoids unnecessary data disclosure. In contrast to current state of the art multi-party private protocols, our proposed protocol does not rely on cryptography and has a fast runtime. Importantly, a user does not have to provide a location directly, even in imprecise form.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS

Keywords

Location Privacy, location based services, spatial data

1. INTRODUCTION

Group Nearest Neighbor (GNN) queries enable a group of users such as friends or family members to determine an optimal meeting location such as a restaurant, picnic place and so forth. We consider the case of a private GNN query where group members issue a query to find out the point of interest (POI) which minimizes their aggregate distance (AD), i.e., the maximum distance. In a privacy preserving GNN query, a user's location is kept private from other members as well as from the LSP. If a user's distances to three POIs are known, the user's location can be calculated by two-dimensional trilateration. Therefore, the group is required to compute the ADs of POIs without revealing the users' locations or individual distances to POIs.

SIGSPATIAL'16, October 31-November 03, 2016, Burlingame, CA, USA

DOI: http://dx.doi.org/10.1145/2996913.2996966

In our scenario, neither the LSP nor the users trust each other. For example, the LSP may collude with group members. In line with existing research, we consider only passive attacks. In these attacks, an attacker or a group of colluding attackers compute distances of an individual user to multiple POIs using gained information during the distance aggregation process. Then these distances are used in 2D trilateration to recover the exact location of that user. We do not consider the attacks where all users collude to attack a single user.

In order to calculate the aggregate maximum distance of a POI from a group of user, the users have to compute the maximum of their distances without revealing their individual distances. Current state of the art private multi-party computation protocols [10, 6, 8] that compute an aggregate such as the maximum are based on cryptography and thus requires expensive computation. Moreover, our problem requires aggregate maximum distance computation for a large number of POIs, which may incur prohibitive computation cost on mobile devices once cryptographic approaches are used. Xiong et al. [9] proposed a non-cryptographic technique to find the maximum value from multiple private values. The participants form a ring and the update a current maximum value sequentially. If the input of a participant is larger than the current maximum value, the participant updates the current maximum value with value chosen randomly between the current maximum value and her own input. In this way the participants keep on updating the current maximum value and if there is no update in one complete round, the process terminates. The current maximum value is declared as the maximum. Although a participant cannot relate the value of current maximum to her previous participant in the ring, if two participants collude they can monitor the participants who are positioned in between them.

The novelty of our work is twofold. First, we propose a non-cryptographic privacy preserving multi-party computation protocol to compute the maximum of private values. Using a non-cryptographic approach, our protocol significantly outperforms existing approaches. A key feature of our proposed protocol is that the maximum can be computed for different levels of imprecision. Our proposed protocol can be readily applied in other non-spatial privacy preserving applications such as privacy preserving auction systems or anonymous sorting. Second, we develop a two stage pruning strategy and algorithms to search the GNN efficiently while preserving each group member's privacy.

2. RELATED WORK

Hashem et al. [3] first proposed a privacy preserving GNN query processing technique. The users provide their imprecise locations as regions instead of exact points to an LSP and thus preserve their location privacy. The LSP returns a set of candidate POIs with respect to the provided regions. The LSP calculates lower bounds on the maximum distance of each POI from the users using their im-

precise locations. Then these values are passed to the users. The users sequentially update the value (lower bound) for each POI. Talouki et al. [2] have shown that Hashem et al's [3] technique is vulnerable to a Partial Collusion (PC) attack and a user's location can be recovered. Huang et al. used Garbled circuit, a cryptographic technique, to preserve privacy in GNN queries [5]. Creating and evaluating a Garbled circuit when there were 10 users and 2000 POIs took 20 seconds in a centralized model and 4 seconds in a distributed model in a desktop computer in their experiment. In [4], Huang et al. reported that Garbled circuit implementations on smartphones are about three orders of magnitude slower than desktop computers and thus require unrealistic response time. Talouki et al. [2, 1] and Khan et al. [7] proposed solutions to find the POI that minimizes the aggregate total distance of the users and are not applicable to our problem.

3. OUR SOLUTION

In our model, all users can communicate with the LSP and the users can communicate with each other without the help of the LSP. A user sends all group members' identities and the desired POI type to the LSP and inquires about the GNN. The LSP communicates with the users using our Private Aggregate Maximum protocol to compute and compare ADs and direct the GNN search.

3.1 Private Aggregate Maximum (PAM) Protocol

We propose the Private Aggregate Maximum protocol (PAM) to calculate the maximum of inputs from multiple users when users do not reveal their inputs. We use PAM to compute aggregate maximum distances of a group of users from points privately. The maximum value is calculated in a bit by bit fashion. Let $\{u_1, u_2, \ldots, u_n\}$ be a group of users. The distance from a user u_i to a point is d_i, which is kept private. The AD is computed as

$$D = \max_{i=1}^{n} d_i.$$

Let the distances be N_b bit binary values. The protocol to solve this problem is divided into two phases. The first phase is an initialization step. Random values are exchanged between users which are used in the second phase to hide distance values. In the second phase, the users send transformed values to the LSP to compute their aggregate distance while concealing their individual distances.

In the first phase, the users communicate with each other and exchange values. Each user u_i does:

1. Create an $n \times N_b$ matrix B_i whose elements are zero.
2. Randomly select ϕ_i rows of B_i and replace the values in these rows by random values. Here $0 \leq \phi_i \leq n$ and u_i keeps ϕ_i private.
3. Send the j-th row of B_i to user u_j, where $j = \{1, 2, \ldots, n\}$ and $j \neq i$.
4. Receive the i-th row of B_j from u_j , where $j = \{1, 2, \ldots, n\}$ and $j \neq i$.
5. Set candidancy value $C_i = 1$. C_i is the availability of chance of d_i being the maximum.

The values of un-selected rows are zero. Therefore, a user only need to send the selected ϕ_i rows to the corresponding users. If a user does not receive values from user u_j, s/he sets the received values from u_j to zero. In this way the number of messages sent between the users can be decreased. C_i means whether d_i is still a candidate of being the maximum or not. Users update C_i in the second phase if necessary. In the first phase, the users exchange random values among them. Therefore, a user's distance cannot be recovered by other users even if all of them collude.

The second phase is performed once for computation of each bit of the maximum value. This computation is done in an order from the MSB to the LSB. In this phase, the users send values to the coordinator. An LSP or any user or any third party can act as a coordinator. Suppose that the most significant $(m-1)$ bits of the maximum value has been computed and the m-th bit is to be computed. Here $1 \leq m \leq N_b$. Each user u_i does the following:

1. Set $\Delta_i = r_i d_i[m] C_i - \sum_{k=1, k\neq i}^{n} B_i[k][m]$. Here $d_i[m]$ is the m-th bit of d_i and r_i is a random positive value.
2. Calculate $\Delta_i^{'}$ as the sum of the m-th elements of the row vectors received from other users. $\Delta_i^{'} = \sum_{j=1, j\neq i}^{n} B_j[i][m]$.
3. Calculate $\delta_i = \Delta_i + \Delta_i^{'}$ and send δ_i to the coordinator.

The coordinator calculates the aggregate total value for the m-th bit as

$$\delta = \sum_{i=1}^{n} \delta_i.$$

If $\delta > 0$, $D[m] = 1$, otherwise $D[m] = 0$. $D[m]$ is the m-th bit of D. The coordinator publishes $D[m]$ to all users. Each user u_i sets C_i to 0 if $D[m] > d_i[m]$.

A random positive value r_i is used to hide the information of the number of candidates for the maximum. If r_i is set to 1, δ is equal to the number of candidates whose m-th bit is 1.

A user u_i discloses only the j-th row of B_i to u_j and δ_i to the coordinator. The user keeps all other information private. When calculating ADs to a set of points, users exchange an matrix of values instead of just one value in each communication. PAM cannot protect a user if all other users collude. Therefore, the group size must be at least three.

3.1.1 A run through example

Suppose there are four users u_1, u_2, u_3 and u_4 and their distance from a point is 1101, 0111, 1011 and 1100 respectively. We use PAM to compute the aggregate maximum distance of the point. In the first phase:

1. The number of users is $n = 4$ and the number of bits in distance value is $N_b = 4$. Each user u_i creates a $n \times N_b = 4 \times 4$ matrix B_i whose elements are zero.
2. u_1 selects a random number between 0 and 4, say 2 as the value of ϕ_1. u_1 randomly selects the second and fourth row of B_1 and replaces the values with random values. B_1 becomes a matrix of random values where the elements in the 1st and 3rd rows are zero. Let the resultant value of B_1 be

$$B_1 = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 6 & 3 & 4 & 1 \\ 0 & 0 & 0 & 0 \\ 5 & 6 & 0 & 2 \end{vmatrix}$$

   Similarly suppose the random matrix created by u_2 (selects the 1st and 2nd row), u_3 (does not select any row) and u_4 (selects the 2nd row) are

$$B_2 = \begin{vmatrix} 3 & 7 & 6 & 2 \\ 2 & 3 & -5 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix} B_3 = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix} B_4 = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 7 & 2 & 1 & -3 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

3. u_1 sends the 2nd and 4th row of B_1 to u_2 and u_4. Similarly u_2 sends the 1st row of B_2 to u_1 and u_4 sends the 2nd row of B_4 to u_2.
4. The row vectors (ignoring the all zero rows) received by u_1, u_2

and u_4 are

$$B_1^{'} = \begin{vmatrix} 3 & 7 & 6 & 2 \end{vmatrix} B_2^{'} = \begin{vmatrix} 6 & 3 & 4 & 1 \\ 7 & 2 & 1 & -3 \end{vmatrix} B_4^{'} = \begin{vmatrix} 5 & 6 & 0 & 2 \end{vmatrix}$$

u_3 does not receive any value. Thus $B_3^{'} = \begin{vmatrix} 0 & 0 & 0 & 0 \end{vmatrix}$

5. Candidacy value is set to 1 and thus $C_1 = C_2 = C_3 = C_4 = 1$.

The computation of maximum starts from the MSB and thus m=1. The coordinator and users run the 2nd phase of the PAM protocol.

1. u_1 computes Δ_1. u_1 selects a random positive value for r_i, say 6. Here $d_1[1] = 1$. Thus $r_1 d_1[m] C_1 = 6 \times 1 \times 1 = 6$. $\sum_{k=1,k\neq 1}^{n} B_1[k][1]$ is the total of all values except the first value in the first column of B_1. Thus $\sum_{k=1,k\neq 1}^{n} B_1[k][1] = 6+0+5 = 11$ and $\Delta_1 = 6-11 = -5$. Suppose the random values selected by u_2, u_3 and u_4 are $r_2 = 2$, $r_3 = 8$ and $r_4 = 9$. Similarly, u_2, u_3 and u_4 compute $\Delta_2 = 2 \times 0 \times 1 - 3 = -3$, $\Delta_3 = 8 \times 1 \times 1 - 0 = 8$ and $\Delta_4 = 9 \times 1 \times 1 - 7 = 2$.
2. u_1 computes $\Delta_1^{'}$ as a sum of the first values of the received vectors. $\Delta_1^{'}$ is the sum of the values of the first column of $B_1^{'}$. Thus $\Delta_1^{'} = 3$. Similarly, u_2, u_3 and u_4 compute $\Delta_2^{'} = 6+7 = 13$, $\Delta_3^{'} = 0$ and $\Delta_4^{'} = 5$.
3. u_1 calculates $\delta_1 = \Delta_1 + \Delta_1^{'} = -5+3 = -2$. Similarly, u_2, u_3 and u_4 compute $\delta_2 = -3+13 = 10$, $\delta_3 = 8+0 = 0$ and $\delta_4 = 2+5 = 7$. Each user u_i sends the computed value δ_i to the coordinator.

The coordinator computes $\delta = -2+10+8+7 = 23 > 0$. Thus the coordinator publishes the MSB of the aggregate value $D[1] = 1$. u_2 updates $C_2 = 0$ since $d_2[1] < D[1]$.

The coordinator and users run the 2nd phase of the PAM protocol again to compute the 2nd bit ($m = 2$). Suppose, the selected random values are $r_1 = 6, r_2 = 5, r_3 = 1, r_4 = 3$. Each user u_i computes Δ_i. Thus $\Delta_1 = 6 \times 1 \times 1 - (3+0+6) = -3$, $\Delta_2 = 5 \times 1 \times 0 - 7 = -7$, $\Delta_3 = 0-0 = 0$ and $\Delta_4 = 3-2 = 1$. Next, each user u_i computes $\Delta_i^{'}$. Thus $\Delta_1^{'} = 7$, $\Delta_2^{'} = 5$, $\Delta_3^{'} = 0$ and $\Delta_4^{'} = 6$. Then each user compute δ_i and sends to the coordinator. $\delta_1 = 4, \delta_2 = -2, \delta_3 = 0$ and $\delta_4 = 7$. The coordinator computes $\delta = 4-2+0+7 = 9 > 0$ and thus $D[2] = 1$. u_3 updates $C_3 = 0$ since $d_3[2] < D[2]$.

In this way, the coordinator and the users collectively compute $D[3] = 0$ and $D[4] = 1$ and thus obtain the aggregate maximum as $D = 1101$.

3.2 Privacy Preserving Algorithms

An LSP divides the whole search space, which includes all POIs, into grids and selects the center points of cells as *probe points* (PP). Then the LSP prunes POIs and computes a candidate POI list. Finally, the LSP sends the location information of the un-pruned candidate POIs to the users and uses our proposed private GNN searching algorithm to search the GNN.

The LSP employs a two stage pruning strategy to compute candidate POIs. In the first stage, the AD of the PPs are calculated and compared using PAM. The lower and upper bound on ADs of POIs in each cell is computed and the cells are pruned using these bounds. At the second stage, the LSP generates some representative points using the location distribution of the un-pruned POIs of 1st stage. Then the LSP estimates the lower bounds on ADs of POIs using the ADs of representative points and further prune POIs.

3.2.1 Computing Candidate POIs

In the first stage, the lower and upper bound of ADs of PPs are calculated. The ADs are calculated in a bit by bit fashion starting from the MSBs. Suppose the most significant m bits of the value of AD of a PP q is calculated at a certain point of time. The lower bound of AD of the PP is computed by filling the remaining bits with 0. The upper bound on AD is computed by filling the remaining bits by 1. For example, the first two bits of a four bit value is 10. The lower and upper bound of the value is 1000 and 1011. Let $D_{min}(q)$ and $D_{max}(q)$ refer to the lower and upper bound on AD of a point q. According to triangle inequality theorem, if the distance of a POI p from q is $d(p,q)$, the lower bound on AD of p is $D_{min}(p) = D_{min}(q) - d(p,q)$ and the upper bound is $D_{max}(p) = D_{max}(q) + d(p,q)$. Let the length of a cell is l. The maximum possible distance of the farthest POI from the center is $l/\sqrt{2}$. Hence the lower bound on the AD of POIs in a cell whose center is q is

$$DCell_{min}(q) = D_{min}(q) - l/\sqrt{2} \quad (1)$$

The LSP computes the nearest POI to each PP. Let the distance between a PP q and its nearest POI be $d_{min}(q)$. The upper bound of ADs of the POIs in a cell whose center is q is

$$DCell_{max}(q) = D_{max}(q) + d_{min}(q) \quad (2)$$

Each user calculates distance to each PP and keep the distance private. For each PP each user performs the first phase of PAM protocol. The LSP coordinates with the users and performs the 2nd phase of PAM protocol to calculate the MSBs of ADs of all PPs. Then the LSP uses the equation (1) and (2) to calculate upper bound and lower bound of ADs of POIs in each cell. Let D_{max} be the minimum of $DCell_{max}$ of all cells. D_{max} is a upper bound on the AD of the GNN. If a cell is found whose $DCell_{min}$ is larger than D_{max}, the cell is pruned as it cannot contain the POI with minimum AD. Next the LSP and users calculates the 2nd most significant bits of ADs of each un-pruned PPs by following the 2nd phase of the PAM protocol. The LSP updates the $DCell_{min}$ and $DCell_{max}$ of each un-pruned cell using equation (1) and (2). Then the LSP updates D_{max}. The LSP prunes cells by applying the lower and upper bound conditions on AD. In this way, the LSP and users jointly calculate the ADs of PPs and update $DCell_{min}$ and $DCell_{max}$ of cells and D_{max} until only one cell is left.

In the second stage, the LSP randomly selects some POIs from the un-pruned cells. Then it generates a representative point (RP) from each selected POI by adding a random noise to the location of the POI. If the location of a selected POI is (x,y), the LSP generates a RP with location $(x+\varepsilon_x, y+\varepsilon_y)$. Here ε_x and ε_y are random values such that $|\varepsilon_x| < \varepsilon$and $|\varepsilon_y| < \varepsilon$. Next the LSP and users collectively compute the imprecise ADs of the RPs. The users perform the first phase of PAM for the RPs. Then the LSP and the users collectively perform the second phase of PAM N_b times to compute the ADs of all RPs.

Let the nearest RP of a POI p be r. According to triangular inequality theorem, the lower bound on the AD of p is $D_{min}(p) = D_{min}(r) - d(r,p)$. The upper bound on the AD of p is $D_{max}(p) = D_{max}(r) + d(r,p)$. The LSP computes the lower and upper bounds on ADs of all POIs in the un-pruned cells. If $D_{max}(p)$ for any POI p is smaller then the current upper bound on AD of the GNN D_{max}, D_{max} is updated to $D_{max}(p)$. If the lower bound on AD of a POI is larger than D_{max}, the POI is pruned. In this way, after pruning the POIs the LSP gets a final candidate list P.

3.2.2 Searching for the GNN in Candidate POIs

The LSP sends the candidate POI list P to the users. Each user calculates distance to each POI and keep the result private. For each POI each user performs the first phase of PAM protocol. Then the LSP and users collectively calculate the 1st most significant bits of the ADs of all POIs by executing the 2nd phase of PAM. If the

most significant bit of a value is larger than that of another value, the first value must be larger than the second value. If the most significant bit of AD of a POI is 0, the POIs whose AD have 1 as the most significant bit cannot have the minimum AD and thus can be pruned. Then the LSP and users calculate the 2nd most significant bit of ADs of the remaining unpruned POIs using 2nd phase of PAM. Similar to the case of the most significant bit, the unpromising POIs are pruned. In this way, ADs of POIs are calculated and compared bit by bit until there is only one POI left or the least significant bit is calculated. If there is only one POI left in the candidate list, the POI has the minimum AD. If the LSB is calculated, all POIs in the candidate list have the minimum AD and thus all of them are the GNN.

4. EXPERIMENTAL EVALUATION

We generated synthetic data sets using uniform distributions. We varied the size of datasets between 1K to 64K point locations. In our default scenario, we imagine 10 friends who want to select a restaurant from nearby 16000 restaurants to have a dinner. After the first stage pruning, the LSP selected 10% of the un-pruned POIs. Then it added random displacement error noise of maximum 500m to the location of the selected POIs to generate representative point locations. We considered 100 private GNN queries for each set of experiments, evaluated our method for each of these GNN queries and determined the average experimental results. For each query we generated new set of POIs' locations and users' locations. We ran our experiments on a desktop computer with Intel Core i7-2600 3.40GHz processor and 8GB RAM.

Very few privacy preserving techniques are currently available that minimize the maximum distance of users while executing GNN queries. We do not consider Hashem et. al.'s [3] method and Huang et. al.'s method [5] for comparison because of their limited privacy protection and extremely high cost as discussed in Section 1. The computation time reported by Huang et. al [5] is three to four orders of magnitude larger than the computation time of our method. We measured computation time for both users and the LSP. *T-user* refers to the computation time for a user and *T-LSP* denotes the computation time for the LSP. We find that in almost all cases, the computation time for the users and the LSP are within 1 ms.

4.1 Effect of the Number of POIs

We studied the scalability of our method by varying the number of POIs in the range of 1K to 64K. The number of users was 10 and the location data had uniform distribution. Figure 1(a) shows that the computation time for LSP increases linearly. The computation time for users increases and becomes double when the number of POIs increases from 1K to 64K. Even if there are 64K POIs the computation time for the LSP is about 2.1 ms and for the users about 0.7 ms. Even if a smartphone is three orders of magnitude slower than a desktop computer [4], the computation of our method will still take less than one second, making it practical. In a compatible setup for only 2000 POIs, Huang et al. [5] reported 4 seconds per user in a decentralized approach and 20 seconds in a centralized approach using a desktop PC. These values translate to unrealistic processing times on today's smartphones.

4.2 Effect of the Number of Users

We studied the effect of the number of users by varying the number of users between 5 and 80. The number of POIs was 16K and the location data had uniform distribution. In our method, a user sends values to all other users and receives values from them. As a result the processing time for the users increases when the number of users increases (Seec Figure 1(b)). Even for 80 users, T-user is about 1.25 ms. The number of users has no effect on the processing time of the LSP. T-LSP remains around 0.6 ms.

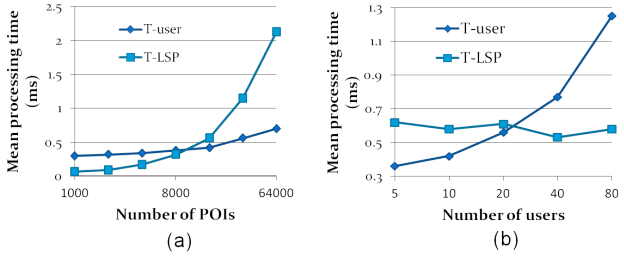

Figure 1: a) Effect of the number of POIs on mean processing time; b) Effect of the number of users on mean processing time.

5. CONCLUSION

We proposed a framework for privacy preserving GNN queries. To calculate the maximum function privately we developed a distributed non-cryptographic multi-party private protocol which is light weight in computation. A key feature of the protocol is that the maximum can be computed with different levels of imprecision. The protocol may have many applications in other domains, such as privacy preserving auctions. Our method exploits geometric constraints to prune POIs and avoids unnecessary data disclosure from an LSP. Our method is a practically applicable solution. In the future, we intend to extend our work to road networks.

6. REFERENCES

[1] M. Ashouri-Talouki and A. Baraani-Dastjerdi. Homomorphic encryption to preserve location privacy. *International Journal of Security and Its Applications*, 6(4):183–189, 2012.

[2] M. Ashouri-Talouki, A. Baraani-Dastjerdi, and A. Aydın Selçuk. Glp: A cryptographic approach for group location privacy. *Computer Communications*, 35(12):1527–1533, 2012.

[3] T. Hashem, L. Kulik, and R. Zhang. Privacy preserving group nearest neighbor queries. In *international conference on Extending Databse Technologies*, pages 489–500, 2010.

[4] Y. Huang, P. Chapman, and D. Evans. Privacy preserving applications on smartphones. In *USENIX Workshop on Hot Topics in Security*, pages 4–10, 2011.

[5] Y. Huang and R. Vishwanathan. Privacy preserving group nearest neighbour queries in location-based services using cryptographic techniques. In *IEEE GLOBECOM*, pages 1–5, 2010.

[6] I. Ioannidis and A. Grama. An efficient protocol for yao's millionaires' problem. In *International Conference on System Sciences*, pages 6–12. IEEE, 2003.

[7] A. M. R. Khan, T. Hashem, E. Tanin, and L. Kulik. Location oblivious privacy protection for group nearest neighbor queries. In *Geographic Information Science*, pages 301–317. Springer, 2014.

[8] H. Lipmaa, N. Asokan, and V. Niemi. Secure vickrey auctions without threshold trust. In *Financial Cryptography*, pages 87–101. Springer, 2003.

[9] L. Xiong, S. Chitti, and L. Liu. Topk queries across multiple private databases. In *Distributed Computing SystemsICDCS 2005*, pages 145–154. IEEE, 2005.

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