PPNA manuscript No.
(will be inserted by the editor)

Multi-objective Optimization Based Privacy Preserving Distributed Data Mining in Peer-to-peer Networks

Kamalika Das · Kanishka Bhaduri · Hillol Kargupta

Received: date / Accepted: date

Abstract This paper proposes a scalable, local privacy-preserving algorithm for distributed peer-to-peer (P2P) data aggregation useful for many advanced data mining/analysis tasks such as average/sum computation, decision tree induction, feature selection, and more. Unlike most multi-party privacy-preserving data mining algorithms, this approach works in an asynchronous manner through local interactions and it is highly scalable. It particularly deals with the distributed computation of the sum of a set of numbers stored at different peers in a P2P network in the context of a P2P web mining application. The proposed optimization-based privacy-preserving technique for computing the sum allows different peers to specify different privacy requirements without having to adhere to a global set of parameters for the chosen privacy model. Since distributed sum computation is a frequently used primitive, the proposed approach is likely to have significant impact on many data mining tasks such as multi-party privacy-preserving clustering, frequent itemset mining, and statistical aggregate computation.

Keywords privacy preserving · data mining · peer-to-peer

A shorter version of this paper was published in IEEE P2P'09 conference. This work was supported by AFOSR MURI grant 2009-11

K. Das
Stinger Ghaffarian Technologies Inc., NASA Ames Research Center,
MS 269-3, Moffett Field, CA-94035
E-mail: Kamalika.Das@nasa.gov

K. Bhaduri
Mission Critical Technologies Inc., NASA Ames Research Center,
MS 269-2, Moffett Field, CA-94035
E-mail: Kanishka.Bhaduri-1@nasa.gov

H. Kargupta
CSEE Dept, University of Maryland, Baltimore County, MD-21250 and AGNIK LLC
E-mail: hillol@cs.umbc.edu

1 Introduction

Privacy-preserving data mining (PPDM) is a requirement in increasing number of multi-party applications where the data is distributed among many nodes in a network. Web mining applications in Peer-to-Peer (P2P) networks [17][6] and cross-domain network threat management systems for analyzing cyber-terrorism trends[1] are some examples where data privacy is an important issue. In such large distributed environments, PPDM algorithms are unlikely to work unless they can offer scalability and heterogeneous privacy-models. Scalability can be addressed by local algorithms in which the communication overhead is bounded by a constant or slowly growing polynomial [6][25]. In a multi-party environment such as the Internet, different users may have different requirements of privacy. Hence a heterogenous privacy model in such scenarios gives parties the autonomy to optimize their privacy cost requirements. This paper takes a step toward developing such a model for privacy preserving data aggregation in a P2P network. The main contributions of this work are two-fold: (1) multi-objective optimization-based heterogeneous privacy model, and (2) a local asynchronous algorithm for distributed data aggregation in a large network for client-side web mining [6, 17].

Data analysis in such heterogenous environments calls for a genre of algorithms which perform the analysis in a distributed fashion. One possibility is distributed data mining (DDM) which deals with the problem of data analysis in environments with distributed data, computing nodes, and users. This paper explores the problem of computing the sum of a collection of numbers distributed in a P2P network in a distributed privacy-preserving manner following the paradigm of DDM. We develop a distributed averaging technique that uses secure sum computation as a building block for scalable data aggregation useful for many advanced data mining tasks. The algorithm is provably correct. Unlike most secure multi-party computation protocols, our algorithm does not assume semi-honest adversary [5]. However, we prove that this algorithm, though not secure, is privacy preserving in a well-defined way. This paper also proposes a new multi-objective optimization-based privacy model for a heterogenous distributed environment where each node defines its own privacy requirement. Each user can specify its own set of parameters for the chosen privacy model. Under this proposed model, each peer gets to choose its own privacy and the algorithm guarantees that the privacy requirement of each peer is satisfied at the end of the protocol. We discuss ranking a set of web advertisements as a client-side web mining application of the proposed algorithm.

The rest of the paper is organized as follows. In the next section (Section 2) we present an illustrative application followed by necessary background material in Section 3. In Section 4 we first describe the optimization-based privacy model and then present the privacy preserving distributed sum computation algorithm. We analyze the algorithm in Section 5 and demonstrate its empirical performance in Section 6. In Section 7 we present some existing work related to this area of research. Finally, we conclude the paper in Section 8.

[1] `http://www.agnik.com/PursuitFlyer.pdf`

2 Illustrative Application

Consider a designer shoe manufacturing company that wants to study the market in South Asia before finalizing their advertising campaign for that geographical region. They plan to use the web for aiding their market research. They buy some web-advertisements designed for collecting user preference-statistics at a popular web-portal and inks in a deal with a web-analytics company to provide business intelligence by combining the click-stream data from the web-advertisements along with user background information collected through the IP address-based mapping of the geographical location and other related techniques. Among other things, the business intelligence provider counts the geographic distribution of clicks on different parts of the advertisements from different IP addresses. This is how it works today. However, growing concerns for online privacy protection is creating technology for protecting the identity of the user. For example, use of anonymizing networks such as TOR [2] may prevent web-servers from collecting any meaningful information regarding queries involving the geographical location of the users. In this case, the IP address associated with a click may come from randomly selected nodes in the TOR network. As a result the web-analytics may give completely misleading information. How do we solve this problem—protect privacy of the user but still be able to rip the benefits of the web mining technology?

This paper offers a solution to this type of problems. It offers a P2P framework where the user identity is protected and the web-mining task is accomplished using distributed privacy-preserving data mining algorithms. It provides a decentralized client-side solution for distributed privacy preserving data aggregation.

The web mining problem of advertisement ranking discussed here is only a representative application scenario and the algorithm can be extended to solve a variety of data aggregation tasks. Since the Internet can be viewed as a connected network of users, we pose this as a data analysis problem in a large P2P network. Every user (peer) in the network has a predefined vector of fixed size where the j-th entry of the vector corresponds to the number of clicks for the j-th advertisement. In this environment, ranking the advertisements can be framed as a global sum computation problem. As the network of users converge to the global sum for every entry in the data vector, they can locally sort the vectors to get the correct global popularity-based ranks of the advertisements. Since web browsing information can be privacy sensitive, it is important to do this sum computation in a privacy-preserving manner. This becomes particularly challenging in heterogeneous environments such as the Internet, since different users might have different requirements of privacy. Therefore the problem that this paper addresses is to compute the global sum of a data vector in a distributed, asynchronous, and privacy-preserving manner.

3 Background

In this paper we propose a privacy-preserving distributed sum computation technique. Since scalability is an important issue for large distributed computing environments, asynchronous solutions are preferred. To the best of the authors' knowledge, there does not exist any privacy-preserving asynchronous algorithm for sum computation.

[2] `http://www.torproject.org/`

The secure sum protocol [5] solves a similar problem but is highly synchronous. There exist several solutions for asynchronous distributed averaging, but are not privacy-preserving [21][20]. The algorithm proposed here uses distributed averaging for privacy preserving sum computation in a locally synchronous fashion. Note that the average computation problem can be converted to a sum computation problem by scaling up the data of each peer by the total number of peers. There exists several techniques in the literature to solve this problem. Examples include the capture-recapture method proposed by Mane *et al.* [19] and the aggregate computation as proposed by Bawa *et al.* [1].

3.1 Notations

Let $P_1, P_2, \ldots, P_d$ be the set of peers connected to each other by an underlying communication infrastructure. The network can be viewed as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{P_1, P_2, \ldots, P_d\}$ denotes the set of vertices (nodes) and $\mathcal{E}$ denotes the set of edges (communication pathways). Let $\Gamma_{i,\alpha}$ denote the set of neighbors of P_i at a distance of α from P_i and $\left|\Gamma_{i,\alpha}\right|$ denote the size of this set *i.e.* the number of neighbors in the α-neighborhood. Further, let $\Phi_{d\times d}$ denote the connectivity matrix or topology matrix of $\mathcal{G}$ representing the network where

$$\phi_{ij} = \begin{cases} 1 \text{ if } i,j \in \mathcal{E} \ \& \ i \neq j \\ -\left|\Gamma_{i,1}\right| \text{ if } i,j \in \mathcal{E} \ \& \ i = j \\ 0 \text{ otherwise} \end{cases}$$

Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_p$ denote an ordered set of advertisements common to all peers and let $\mathbf{X}$ be the $d \times p$ global data matrix where the i-th row vector corresponds to the data of peer P_i for all the advertisements and the j-th column corresponds to the advertisement $\mathcal{A}_j$ across all the peers. Furthermore, for peer P_i, x_{ij} denotes the number of clicks of advertisement $\mathcal{A}_j$. Let X be the random variable for the distribution of x_{ij}. Let S_j denote the global sum of the j-th data element x_{ij} *i.e.*, $S_j = \sum_{i=1}^{d} x_{ij}$. Finally, let n_i^* denote the size of the ring that peer P_i forms for the secure sum computation (to be discussed later).

3.2 Distributed Averaging

In distributed averaging, the objective is to compute the global average $\Delta_j = \frac{1}{d}\sum_{i=1}^{d} x_{ij}$ where every peer P_i has a real number x_{ij} and d is the size of the network. In the naive solution, all the peers can exchange information to compute the correct sum. However, this solution is highly synchronous and does not scale well for large P2P networks. Distributed approaches include the approaches proposed by Scherber and Papadopoulos [21] and Mehyar *et al.* [20]. The basic idea of all these approaches is to maintain the current estimate of Δ_j ($z_i^{(t)}$) and exchange messages with its immediate neighbors to update $z_i^{(t)}$. As iteration $t \to \infty$, $z_i^{(t)} \to \Delta_j$, *i.e.* the system asymptotically converges to the correct average.

The distributed averaging problem, as proposed in [21], is not privacy-preserving. Moreover it works only for symmetric topologies. In Section 3.4 we explain that our multi-objective optimization framework requires asymmetric network topology. To handle this, we present a modified protocol in Section 4.5.

3.3 Secure Sum Protocol

Secure sum computation [5] computes $S_j = \sum_{i=1}^{n} x_{ij}$ without disclosing the local value x_{ij} of any user. It has been widely used in privacy-preserving distributed data mining as an important primitive. The secure sum protocol requires the existence of a ring topology (or an overlay ring network) connecting the users *i.e.* for peers 2 through $d-1$, $\Gamma_{i,1} = \{P_{i-1}, P_{i+1}\}$, $\Gamma_{1,1} = \{P_d, P_2\}$ and $\Gamma_{d,1} = \{P_{d-1}, P_1\}$. Let each $x_{ij} \in \{0, 1, 2, \ldots m\}$. It is known that the sum $S_j = \sum_{i=1}^{d} x_{ij}$ to be computed takes an integer value in the range $[0, N-1]$. Assuming peers do not collude, P_1 generates a random number R uniformly distributed in the range $[0, N-1]$, which is independent of its local value x_{1j} and transmits $(R + x_{1j}) \bmod N$ to P_2. In general, for $i = 2, \ldots, d$, peer P_i executes:

$$y_{ij} = (y_{i-1j} + x_{ij}) \bmod N = (R + \textstyle\sum_{q=1}^{i} x_{qj}) \bmod N,$$

where y_{ij} is the perturbed version of local value x_{ij} to be sent to the next peer $i+1$. P_d performs the same step and sends the result y_{dj} to P_1. Then peer P_1, which knows R, can subtract R from y_{dj} to obtain the actual sum. This sum is finally broadcast to all other users.

The secure sum protocol is highly synchronous and is therefore unlikely to scale for large networks. Combining a newer variation of the distributed averaging (Section 4.5) with the secure sum protocol in a small neighborhood of a peer, we propose a privacy-preserving sum computation algorithm which (i) asymptotically converges to the correct result, and (2) being only locally synchronous, scales well with the network size.

3.4 Multi-objective Optimization and Privacy

Multi-objective optimization, also known as multi-criteria or multi-attribute optimization, is the process of simultaneously optimizing two or more possibly conflicting objectives subject to certain constraints. It can be mathematically stated as:

$$\begin{array}{rll} \text{minimize} & f(\mathbf{x}) = [f_1(\mathbf{x}) \quad \ldots \quad f_M(\mathbf{x})]^{\mathrm{T}} & \\ \text{subject to} & g_j(\mathbf{x}) \le 0, \quad \forall j = 1, \ldots, p & \\ & h_k(\mathbf{x}) = 0, \quad \forall k = 1, \ldots, q & \\ & x_i^{(\ell)} \le x_i \le x_i^{(u)}, \quad \forall i = 1 \ldots m & \end{array} \tag{1}$$

where there are M scalar objectives $f_1 \ldots f_M$ with $f_i : \mathbb{R}^m \to \mathbb{R}$, g_j and h_k are known as the constraint functions and each variable also has its own explicit bounds between $x_i^{(\ell)}$ and $x_i^{(u)}$. The solution to such a multi-objective optimization problem is a vector $\mathbf{x}^* = \{x_1^*, x_2^*, \ldots, x_m^*\} \in \mathbb{R}^m$. Next we discuss the concept of *Pareto* optimal set which is useful for comparing two optimal solutions.

The concept of dominance is intricately related to multi-objective optimization. Let $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$ be two solutions where we define the '$\prec$' operator as $\mathbf{x}_1^* \prec \mathbf{x}_2^*$ implies that solution $\mathbf{x}_1^*$ is better than solution $\mathbf{x}_2^*$ on a particular objective $f_j(\mathbf{x})$ *i.e.* $\exists j$ such that, $f_j(\mathbf{x}_1^*) < f_j(\mathbf{x}_2^*)$. Similarly, we define the '$\not\succ$' operator as $\mathbf{x}_1^* \not\succ \mathbf{x}_2^*$ implying that $\mathbf{x}_1^*$ is no worse than $\mathbf{x}_2^*$ for the objective function $f_j(\mathbf{x})$ *i.e.* $\exists j$ such that, $f_j(\mathbf{x}_1^*) \not> f_j(\mathbf{x}_2^*)$. Under such conditions, solution $\mathbf{x}_1^*$ is said to dominate solution $\mathbf{x}_2^*$. Given a finite set of solutions $\mathcal{S}$, it is always possible to find a subset of solutions $\mathcal{S}^{'} \subset \mathcal{S}$, such that any two solutions in $\mathcal{S}^{'}$ do not dominate each other. Moreover, for any solution in $\mathcal{S} \backslash \mathcal{S}^{'}$, we can always find a solution in $\mathcal{S}^{'}$ which dominates the one in $\mathcal{S} \backslash \mathcal{S}^{'}$. When the set $\mathcal{S}$ refers to the entire search space, then the set $\mathcal{S}^{'}$ is known as the *Pareto* optimal set of solutions. All solutions in the *Pareto* optimal are equivalent. Thus, solution to multi-objective optimization problem reduces to finding the entire *Pareto* optimal solution set.

In general, a multi-objective optimization problem (as defined by Equation 1) can be solved in several different ways to find the *Pareto* optimal set. In this paper, we explore a simple technique *viz. scalarization* — which combines multiple objective functions into a single objective function using a set of weights. As a result, Equation 1 can be reformulated as:

$$\begin{aligned} \text{minimize} \quad & F = \mathbf{w}^{\mathrm{T}} f(\mathbf{x}) = [w_1 f_1(\mathbf{x}) + \quad \ldots \quad + w_M f_M(\mathbf{x})] \\ \text{subject to} \quad & g_j(\mathbf{x}) \leq 0, \quad \forall j = 1, \ldots, p \\ & h_k(\mathbf{x}) = 0, \quad \forall k = 1, \ldots, q \\ & x_i^{(\ell)} \leq x_i \leq x_i^{(u)}, \quad \forall i = 1 \ldots m \end{aligned} \tag{2}$$

where $\mathbf{w}$ is a M-dimensional weight vector whose components are positive. Since multiplication by a constant does not change the optimal value, it is customary to assume that $\sum_{i=1}^{M} w_i = 1$. Note that this technique reduces the multi-objective optimization problem to an ordinary scalar optimization problem. The exact value of the weights depends on several factors: (1) the importance one associates to each objective function, and (2) if the objective functions are not all in the same scale, the weights can be used to scale them to uniformity. By varying the weight vector one can obtain possibly different *Pareto* optimal solutions of the multi-objective optimization given in Equation 1. Finally we point out that scalarization does not destroy the *Pareto* optimal set of the original problem. Therefore, (1) any solution of Equation 2 lies in the *Pareto* optimal solution set of Equation 1, and (2) the entire *Pareto* optimal set can be generated by solving the scalarized version (Equation 2) for *convex* multi-objective optimization. Interested readers are referred to [9] and [4] for proofs.

4 Privacy-Preserving Distributed Sum Computation

It is clear from the previous section that we need to measure threat and cost in order to solve the optimization problem. In this section we first state the model of privacy that we have used and then derive an expression for threat in secure sum protocol in the presence of collusion. Finally we show how the overall multi objective optimization problem can be used to develop our privacy preserving asynchronous distributed sum computation algorithm.

4.1 Bayes Optimal Privacy Model

The Bayes optimal model of privacy [18] uses prior and posterior distribution to quantify privacy breach.

Let X be a random variable which denotes the j-th data value at each node. The value at node P_i is denoted by x_{ij}. The prior probability distribution is $prior =$

$P(X = x_{ij})$. Once the data mining process is executed, the participants can have some extra information. Given this, we define the posterior probability distribution as $posterior = P(X = x_{ij}|\mathcal{B})$, where $\mathcal{B}$ models the extra information available to the adversary at the end of computation. The exist several techniques in the literature to measure privacy breach [11][10][18]. Below we state the Bayes optimal privacy breach.

Definition 1 [$\rho_1 - to - \rho_2$-privacy breach[10]] Let f_{prior} and $f_{posterior}$ denote the prior and posterior probability distribution of X. The $\rho_1 - to - \rho_2$ privacy breach happens when $f_{prior} \leq \rho_1$ and $f_{posterior} \geq \rho_2$, where $0 < \rho_1 < \rho_2 < 1$.

As noted in [18], any privacy definition which quantifies the privacy breach in terms of principle 1 or 2, is known as the Bayes optimal privacy model. However, this $\rho_1 - to - \rho_2$ privacy model is applicable only when there is a single node in the network. Below we extend this privacy framework for a distributed multi-party scenario.

Definition 2 [Multi-party $\rho_1 - to - \rho_2$ privacy breach] For the i-th peer P_i, privacy breach occurs if $f^i_{prior} \leq \rho_{1i}$ and $f^i_{posterior} \geq \rho_{2i}$. Multi-party $\rho_1 - to - \rho_2$ privacy breach occurs when the constraints are violated for any peer in the network *i.e.* $\forall i,\ f^i_{prior} \leq \rho_{1i}$ and $f^i_{posterior} \geq \rho_{2i}$, where $0 < \rho_{1i} < \rho_{2i} < 1$.

In the definition, the posterior probabilities of each peer can either be dependent or independent of each other. If the peers share the extra information ($\mathcal{B}$), their posterior distributions are also related. Since in our framework each peer solves the optimization problem locally, the dependence or the independence of the posterior probabilities does not change the privacy requirements.

Since in a distributed environment, different peers have different privacy requirements, it is difficult to achieve the distributed $\rho_1 - to - \rho_2$ privacy using a single secure sum since the $\rho_1 - to - \rho_2$ privacy is achieved in terms of the number of participants of the ring (as shown in Section 4.2). So our proposed algorithm uses multiple local sum computation protocols with different ring sizes, one for each node in the network. This approach addresses two issues: (1) it proposes a solution to privacy preservation in heterogenous environments and (2) it avoids creating a single large synchronous ring for sum computation which makes the algorithm scalable for large-scale distributed systems. The sum computation does not claim to be a secure protocol by getting rid of the semi-honest assumption, but still is privacy preserving. Before we describe the algorithm for doing the distributed averaging based local secure sum, we introduce a measure of the *threat* component in the objective function applicable to the secure sum protocol.

4.2 Threat Measure under Collusion

The secure sum computation algorithm assumes semi-honest parties who do not collude. However, it has been shown in the literature [13] that such an assumption is suboptimal and that rational parties would always try to collude in the absence of a penalizing mechanism. In this paper we adapt the expression of threat developed in [13] to estimate the threat component in our objective function. Each peer forms a ring of size n_i^* (referred to as n in this section for sake of simplicity) in our algorithm. Let us assume that there are k ($k \geq 2$) nodes acting together secretly to achieve a fraudulent purpose. Let P_i be an honest node who is worried about its privacy. Let P_{i-1} be the immediate predecessor of P_i and P_{i+1} be the immediate successor of P_i. We will only consider the case when $n-1 > k \geq 2$ and the colluding nodes contain neither P_{i-1} nor P_{i+1}, or only one of them, then P_i is disguised by $n-k-1$ other nodes' values. This can be represented as

$$\underbrace{\sum_{q=1}^{n-k-1} x_{qj}}_{\text{denoted by Y}} + \underbrace{x_{ij}}_{\text{denoted by X}} = \underbrace{S_j - \sum_{q=i+1}^{i+k} x_{qj}}_{\text{denoted by W}},$$

where W is a constant and known to all the colluding nodes. The posterior probability of x_{ij} is:

$$f_{posterior}(x_{ij}) = \frac{1}{(m+1)^{(n-k-1)}} \sum_{q=0}^{r} (-1)^q \binom{n-k-1}{q} \times \binom{n-k-1+(r-q)(m+1)+t-1}{(r-q)(m+1)+t} \tag{3}$$

where $z_j = W - x_{ij}$ and $z \in \{0, 1, \ldots, m(n-k-1)\}$. $r = \lfloor \frac{z_j}{m+1} \rfloor$, and $t = z_j - \lfloor \frac{z_j}{m+1} \rfloor (m+1)$. Note that here we assume $x_{ij} \leq W$, otherwise $f_{posterior}(x_{ij}) = 0$. This posterior probability can be used to measure the threat faced by a peer while participating in the secure sum computation protocol, if there is collusion:

$$threat = Posterior - Prior = f_{posterior}(x_{ij}) - \frac{1}{m+1} \tag{4}$$

Note that using uniform distribution as the prior belief is a reasonable assumption because it models the basic knowledge of the adversaries. This assumption was also adopted by [24] where a Bayes intruder model was proposed to assess the security of additive noise and multiplicative bias.

It can be observed from this threat measure that (1) as k increases, the posterior probability increases, and (2) as n increases, the posterior probability decreases. This implies that as the size of the network involved in the secure sum computation increases, the threat decreases for a fixed size of the colluding group. Therefore, the privacy of the data of the users in the secure sum depends on the initiator's choice of the size of the group (n). The choice of n can vary between 1 and the total number of nodes d. As the value of n increases, the threat to a user's data due to collusion decreases, assuming a constant percentage of colluding nodes in the network. However, increasing n increases the overall communication cost and synchronization requirements of the algorithm. The conflicting nature of the objective functions allows us to set up a multi-objective optimization problem.

4.3 Threat Measure for Multiple Rings

The above expression only gives us a measure of threat when there is only one ring. In the presence of multiple rings, a colluder can infer more knowledge about an honest node's data. In this section, we derive an expression for threat in the presence of multiple rings. For simplicity, we consider the situation of only two intersecting rings. The case for multiple rings can be analogously derived.

Let there be n_1 nodes in ring 1 and n_2 nodes in ring 2. The values at the nodes for the two rings be arranged as follows:

$$\text{Ring 1: } \overbrace{x_{1,j} \rightarrow \cdots \rightarrow x_{c-1,j} \rightarrow x_{c,j}}^{\text{common}} \rightarrow \overbrace{x_{a,j} \rightarrow x_{a+1,j} \rightarrow \cdots \rightarrow x_{g,j}}^{\text{not common}}$$

$$\text{Ring 2: } \overbrace{x_{1,j} \rightarrow \cdots \rightarrow x_{c-1,j} \rightarrow x_{c,j}}^{\text{common}} \rightarrow \overbrace{x_{b,j} \rightarrow x_{b+1,j} \rightarrow \cdots \rightarrow x_{h,j}}^{\text{not common}}$$

For ring 1, let the colluding nodes be $x_{c-1,j}, x_{c,j}, x_{a,j}, x_{a+1,j}$. Similarly, for the other ring, $x_{c-1,j}, x_{c,j}, x_{b,j}, x_{b+1,j}$ are the colluding nodes.

Denoting the sum of the data values in the rings by C_1 and C_2, we can write,

$$x_{1,j} + \cdots + x_{c,j} + x_{a,j} + \cdots + x_{g,j} = C_1$$
$$x_{1,j} + \cdots + x_{c,j} + x_{b,j} + \cdots + x_{h,j} = C_2$$

Subtracting, we get

$$x_{a,j} + \cdots + x_{g,j} - \big(x_{b,j} + \cdots + x_{h,j}\big) = C_1 - C_2$$

Since the values of the colluders $(x_{a,j}, x_{a+1,j}, x_{b,j}, x_{b+1,j})$ are known to the colluding group, we can even subtract these from the sum to be estimated. We are left with the following expression:

$$x_{a+2,j} + \cdots + x_{g,j} - \big(x_{b+2,j} + \cdots + x_{h,j}\big) = C_1 - C_2 - (x_{a,j} + x_{a+1,j} + x_{b,j} + x_{b+1,j})$$

Let $C_1 - C_2 - (x_{a,j} + x_{a+1,j} + x_{b,j} + x_{b+1,j}) = C$. We can now write,

$$x_{a+2,j} + \cdots + x_{g,j} - \big(x_{b+2,j} + \cdots + x_{h,j}\big) = C$$

Without loss of generality, let the node whose value is at threat be $x_{g,j}$. Thus, we can write,

$$\underbrace{x_{g,j}}_{\text{denoted by} Z} = C + \left(\underbrace{x_{b+2,j} + \cdots + x_{h,j}}_{\text{denoted by} X}\right) - \left(\underbrace{x_{a+2,j} + \cdots + x_{g-1,j}}_{\text{denoted by} Y}\right)$$

Note that X and Y are the sums of $n_2 - c - 2$ and $n_1 - c - 3$ (leaving out the one to be estimated) iid random variables respectively. Now since C is a constant, it can be shown that,

$$\begin{aligned} P(Z = z) &= P(X - Y = z) \\ &= \sum_{y=0}^{(n_1-c-3)m} P(X - Y = z | Y = y) P(Y = y) \\ &= \sum_{y=0}^{(n_1-c-3)m} P(X - y = z) P(Y = y) \\ &= \sum_{y=0}^{(n_1-c-3)m} P(X = y + z) P(Y = y) \end{aligned}$$

Using the posterior distribution value, we can write the expression for $P(Z = z)$ as,

$$\begin{aligned}P(Z = z) = \sum_{y=0}^{(n_1-c-3)m} & \frac{1}{(m+1)^{(n_2-c-2)}} \sum_{j=0}^{r_y} (-1)^j \binom{n_2 - c - 2}{j} \times \\ & \binom{(n_2 - c - 2) + (r_y - j)(m+1) + t_y - 1}{(r_y - j)(m+1) + t_y} \times \\ & \frac{1}{(m+1)^{(n_1-c-3)}} \sum_{j=0}^{q_y} (-1)^j \binom{n_1 - c - 3}{j} \times \\ & \binom{(n_1 - c - 3) + (q_y - j)(m+1) + s_y - 1}{(q_y - j)(m+1) + s_y}\end{aligned}$$

where $y \in \{0, 1, \ldots, m(n_1 - c - 3)\}$, $r_y = \lfloor \frac{y+z}{m+1} \rfloor$, $q_y = \lfloor \frac{y}{m+1} \rfloor$, $t_y = y + z - \lfloor \frac{y+z}{m+1} \rfloor (m+1)$, and $s_y = y - \lfloor \frac{y}{m+1} \rfloor (m+1)$.

4.4 Distributed Privacy Solution using Multi-objective Optimization

Privacy is a social concept. In a distributed data mining environment, different peers have different notions and requirements of privacy. Many privacy preserving distributed data mining algorithms can be modeled as an optimization problem with two conflicting objectives: maximizing the privacy (or minimizing the threat to the data) while minimizing the cost. Due to sharing of private information in the process of computation, privacy of the users' data is threatened. Every user in the network has a prior belief (assumption) about the *threat* to its data privacy. The threat that a peer's data is exposed to can be considered as a measure of the lack of privacy of its data. The amount of resources available to a peer varies across the network and hence, the cost (of computation and communication) a peer can bear to ensure its data privacy also varies. In this paper we assume that each peer has the same privacy and cost model parameterized by the size of the local ring n. Let $f_t(n)$ and $f_c(n)$ be two functions defining the threat to data privacy and the cost respectively. The constraints are different for each peer since each node has its own threshold of privacy and cost defined by its own requirements and resources. Also, the weights of the multi-objective optimization problem may be different for different participants depending on the importance they attach to threat and cost. For peer P_i, this multi-objective optimization problem can be written as,

$$\begin{aligned} \text{minimize} \quad & F = \mathbf{w_i}^{\mathrm{T}} f(n) = \left[w_{1,i} f_t(n) + w_{2,i} f_c(n) \right] \\ \text{subject to} \quad & n_i^{(\ell)} \le n \le n_i^{(u)}, \\ & w_{1,i} + w_{2,i} = 1 \\ & w_{1,i}, w_{2,i} \ge 0 \end{aligned} \tag{5}$$

Note that, in a multi-party scenario, each party can define its own optimization functions and solve them independently. But this might generate a different *Pareto* optimal set for each party. The other extreme solution is for all nodes in the network to use the same objective functions and constraints. Both of these solutions are undesirable — in the first, parties do not guarantee a global solution while in the second, each party has to abide by the same threat and cost requirements. In this paper, we guarantee a global solution based on the personalized requirements of a user. To achieve this, we require that the threat $f_t(n)$ and the cost $f_c(n)$ functions be the same for each party. The privacy model across each party may be different. For example, in the case of privacy by anonymization, parties can choose either the k-anonymity [22], ℓ-diversity [18], or the t-closeness [16] model since all of them gives rise to same $f_t(n)$. However, the choice of ϵ-differential privacy will be meaningless in this context, since the measurement of threat $f_t(n)$ will be different for this privacy model.

There can be several approaches to generating an optimal privacy/cost solution following this multi-objective optimization framework. We discuss two such approaches here.

4.4.1 Average Solution

In a collaborative environment such as the Internet, a globally correct solution would require one to centralize the constraints. Since this requires overall network synchronization, thereby leading to low scalability, we take the alternate approach of using an asynchronous sum computation technique as discussed in Section 3.2. By executing a separate average computation for both the upper and lower bounds $\sum_{i=1}^{d} n_i^{(\ell)}$ and $\sum_{i=1}^{d} n_i^{(u)}$ across all the peers, we can generate an average constraint. After this computation, the multi objective optimization problem at each node can be written as,

$$\begin{aligned} \text{minimize} \quad & F = \mathbf{w_i}^{\mathrm{T}} f(n) = \left[w_{1,i} f_t(n) + w_{2,i} f_c(n)\right] \\ \text{subject to} \quad & \overline{n^{(\ell)}} \le n \le \overline{n^{(u)}}, \\ & w_{1,i} + w_{2,i} = 1 \\ & w_{1,i}, w_{2,i} \ge 0 \end{aligned} \tag{6}$$

where $\overline{n^{(\ell)}}$ and $\overline{n^{(u)}}$ are the average lower and upper bound computed via distributed averaging. Using Eq. 3, and a linear function for the cost $f_c(n) = w_2 g n$, where g is a constant, we can write the objective function as,

$$F = w_{1,i} \frac{1}{(m+1)^{(n-k-1)}} \sum_{q=0}^{r} (-1)^q \binom{n-k-1}{q} \binom{n-k+t-2+(r-q)(m+1)}{(r-q)(m+1)+t} + w_{2,i} g n.$$

There does not exist a closed form derivative for this function, so we approximate $f_t(n)$ by $F^{'} = \frac{1}{(m+1)^{(n-k-1)}}$. Since $F^{'} < f_t(n)$, and the objective is to minimize $f_t(n)$, this approximation does not result any loss of accuracy. Note that both of these functions are convex if $n > k+1$. Therefore, using the results of Section 3.4, we can guarantee that the entire *Pareto* optimal set of the multi-objective optimization problem can be enumerated. The new objective function can be written as,

$$F = w_{1,i} \frac{1}{(m+1)^{(n-k-1)}} + w_{2,i} g n.$$

We first compute the first order partial derivative with respect to n and set it to 0:

$$\begin{aligned} \frac{\partial F}{\partial n} &= w_{1,i} \frac{-(n-k-1)}{(m+1)^{(n-k-2)}} + w_{2,i} g = 0 \\ &\Rightarrow \frac{w_{1,i}}{w_{2,i}} = \frac{g(m+1)^{(n-k-2)}}{n-k-1} \end{aligned}$$

Solving for n from the above equation will gives the optimal value n^*. Now since each peer has the average of the constraints, we can write:
(1) when $n^* = \overline{n^{(\ell)}}$,

$$w_1 = \frac{g(m+1)^{(\overline{n^{(\ell)}}-k-2)}}{\overline{n^{(\ell)}} - k - 1 + g(m+1)^{(\overline{n^{(\ell)}}-k-2)}}$$

$$w_2 = \frac{\overline{n^{(\ell)}} - k - 1}{\overline{n^{(\ell)}} - k - 1 + g(m+1)^{(\overline{n^{(\ell)}}-k-2)}}$$

and (2) when $n^* = \overline{n^{(u)}}$,

$$w_1 = \frac{g(m+1)^{(\overline{n^{(u)}}-k-2)}}{\overline{n^{(u)}} - k - 1 + g(m+1)^{(\overline{n^{(u)}}-k-2)}}$$

$$w_2 = \frac{\overline{n^{(u)}} - k - 1}{\overline{n^{(u)}} - k - 1 + g(m+1)^{(\overline{n^{(u)}}-k-2)}}.$$

Therefore the following ranges of w_1 and w_2 generate the entire *Pareto* optimal set:

$$\frac{g(m+1)^{(\overline{n^{(\ell)}}-k-2)}}{\overline{n^{(\ell)}} - k - 1 + g(m+1)^{(\overline{n^{(\ell)}}-k-2)}} < w_1 < \frac{g(m+1)^{(\overline{n^{(u)}}-k-2)}}{\overline{n^{(u)}} - k - 1 + g(m+1)^{(\overline{n^{(u)}}-k-2)}}$$

$$\frac{\overline{n^{(\ell)}} - k - 1}{\overline{n^{(\ell)}} - k - 1 + g(m+1)^{(\overline{n^{(\ell)}}-k-2)}} < w_2 < \frac{\overline{n^{(u)}} - k - 1}{\overline{n^{(u)}} - k - 1 + g(m+1)^{(\overline{n^{(u)}}-k-2)}}.$$

This solution to the multi-objective function provides an average privacy/cost to all the peers in the network. In the next section we develop another solution to the same problem in which the maximum privacy of all the nodes in the network is satisfied.

4.4.2 Worst-case Solution

In this case we assume that each peer has maximum threat and cost constraints. The goal for each peer is to find out the maximum value of the ring size n based on the local constraints. The multi-objective optimization function can be written as:

$$\max_n \left[w_{1,i} f_t(n) - w_{2,i} f_c(n)\right]$$

subject to the following constraints: $f_c < c_i$ and $f_t < t_i$ where $f_t(n)$ is given by Equation 4 and $f_c(n) = w_{2,i} gn$. g is the proportionality constant and c_i and t_i are constants for every peer and denote the cost threshold and privacy threshold that each peer is willing to withstand. Below is a solution to this optimization problem.

Lemma 1 *Given the thresholds for threat t_i and cost c_i, the solution to the optimization problem*

$$\max_n \left[w_{1,i} f_t(n) - w_{2,i} f_c(n)\right]$$

is given by

$$1 + k + \frac{log(w_{1,i}) - log(t_i)}{log(m+1)} \le n^* \le \frac{c_i}{w_{2,i} g}$$

Proof

$$h(n) = \sum_{q=0}^{r} (-1)^q \binom{n-k-1}{q} \binom{n+t-k-2+(r-q)(m+1)}{(r-q)(m+1)+t}$$

Now, we know that for $a \geq b$, $\binom{a}{b} \geq 1$. Since $h(n)$ is a probability distribution, $h(n) > 0$. Also, $h(n)$ being product of combinations must be integer. Therefore, $h(n) \geq 1$. Using the constraint, $f_t \leq t_i$ we know that

$$\frac{w_{1,i}}{(m+1)^{(n-k-1)}} \times h(n) \leq t_i$$

Using these results, we can write,

$$\begin{aligned}
1 &\leq h(n) \\
\Rightarrow \frac{w_{1,i}}{(m+1)^{(n-k-1)}} &\leq \frac{w_{1,i}}{(m+1)^{(n-k-1)}} \times h(n) \leq t_i \\
\Rightarrow \frac{w_{1,i}}{(m+1)^{(n-k-1)}} &\leq t_i \\
\Rightarrow (m+1)^{(n-k-1)} &\geq \frac{w_{1,i}}{t_i} \\
\Rightarrow (n-k-1)\log(m+1) &\geq \log(w_{1,i}) - \log(t_i) \\
\Rightarrow n &\geq 1 + k + \frac{\log(w_{1,i}) - \log(t_i)}{\log(m+1)} \qquad (7)
\end{aligned}$$

Similarly, using the constraint on cost, we get

$$\begin{aligned}
w_{2,i} g n &\leq c_i \\
\Rightarrow n &\leq \frac{c_i}{w_{2,i} g} \qquad (8)
\end{aligned}$$

Using Equations 7 and 8, we get the optimal value of n^* as,

$$1 + k + \frac{\log(w_{1,i}) - \log(t_i)}{\log(m+1)} \leq n_i^* \leq \frac{c_i}{w_{2,i} g} \qquad (9)$$

□

Now, depending on its personal preference, each peer can choose the number of nodes (n_i^*) for computing the sum in a privacy preserving fashion, even in the presence of colluding parties.

4.5 Distributed Averaging for Asymmetric Topologies

In this section we present the iterative distributed algorithm for computing the global sum of a set of data vectors. Our solution is inspired by the distributed averaging algorithms proposed in [21] and [20].

The distributed averaging technique that we are exploring asymptotically converges to the global average. It can easily be used to compute the sum if each peer multiplies its data by the total number of peers in the network. Therefore, for the given scenario, each peer P_i contains a real number $d \times x_{ij}$ where d is the size of the entire network and the objective is to compute $\Delta_j = \frac{1}{d}\sum_{i=1}^{d} d \times x_{ij}$ *i.e.* the sum of the numbers. There exist several techniques in the literature to estimate the network size. Examples include the capture-recapture method proposed by Mane *et al.* [19] and Bawa *et al.* [1]. Moreover at any time, the number of nodes in the network can be estimated efficiently using heartbeat mechanisms or retransmissions. From now on we assume that each entry x_{ij} of the data has been multiplied by the total number of peers so that distributed averaging gives the global sum and not the global average.

Let x_{ij} denote the j-th data of peer P_i. $\mathbf{z}_j^{(t)} = \left[z_{1j}^{(t)} z_{2j}^{(t)} \dots z_{dj}^{(t)}\right]^T$ denotes the estimate of the global sum $\Delta_j = \frac{1}{d}\sum_{i=1}^{d} x_{ij}$ by d peers at the t-th iteration. The initialization is $\mathbf{z}_j^{(0)} = \left[x_{1j}x_{2j}\dots x_{dj}\right]^T$. The proposed algorithm works as follows: at any iteration, each peer P_i gets the estimate from all of its neighbors (the $z_{ij}^{(t-1)}$'s for $i \in \Gamma_{i,1}$) and then generates the estimate for round t (*i.e.* $z_{ij}^{(t)}$) based on those received estimates and its local data. This algorithm is asynchronous and local since each node gets update from its neighbors only. The update rule used is first order: $\mathbf{z}_j^{(t)} = \mathbf{W}\mathbf{z}_j^{(t-1)}$. Any choice of $\mathbf{W}$ guarantees asymptotic convergence if $\mathbf{W}$ satisfies the following properties: (i) $\mathbf{W}.\mathbf{1} = \mathbf{W}^T.\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ denotes a $d \times 1$ vector of all ones and (ii) the eigenvalues of $\mathbf{W}$, λ_i when arranged in descending order are such that $\lambda_1 = 1$ and $|\lambda_i| < 1$. Setting $\mathbf{W} = \mathbf{I} + \rho\Phi$ satisfies these conditions; where ρ is a small number which determines the stability of the solution and the convergence rate, and $\mathbf{I}$ denotes the identity matrix.

From Section 4.2, it is clear that depending on the solution to the optimization problem, each peer can have a different value of n_i^*, *i.e.* number of nodes it wants to communicate with. This means that if peer P_i chooses peer P_j to be part of its sum computation, it is not necessary that P_j would choose P_i to be part of its sum computation ring. This implies that even if P_j is a neighbor of P_i, P_i need not be a neighbor of P_i (in terms of adjacency matrix). This implies that the resulting topology matrix is asymmetric. Note that if we use $\mathbf{W} = \mathbf{I} + \rho\Phi$, the resulting $\mathbf{W}$ does not satisfy the requirements stated above. Therefore, asymmetric topology matrices cannot be directly used for generating the update matrix $\mathbf{W}$. Now, an asymmetric topology matrix can be converted to a symmetric one as follows: $\Phi^{''} = \Phi + \Phi^T$, where Φ^T is the transpose of Φ. Since Φ is a square matrix, $\Phi^{''}$, by definition, is a symmetric matrix. In order for $\mathbf{W}$ to satisfy the properties stated above, it can be generated using the transformation $\mathbf{W} = \mathbf{U} + \rho\Phi^{''}$ where each entry of $\mathbf{U}_{d\times d}$ is such that

$$u_{ii} = \begin{cases} 1 - \rho\sum_{j=1}^{d}\phi_{ij}^{''} \\ 0 \text{ otherwise} \end{cases}$$

In Section 5, we analyze the convergence and correctness of this proposed distributed averaging algorithm. Based on the above transformation, every peer updates its estimate of Δ_j using an update rule that depends on the ring it forms. The following lemma (Lemma 41) states the update rule for our proposed distributed averaging problem.

Lemma 41 *For the modified distributed averaging technique, the update rule for any peer can be written as*

$$z_{ij}^{(t)} = \left\{1 - 2\rho\left|\Gamma_{i,1}\right| - \rho(n_i^* - \left|\Gamma_{i,1}\right|)\right\} z_{ij}^{(t-1)} + 2\rho \sum_{q\in\Gamma_{i,1}} z_{qj}^{(t-1)} + \rho \sum_{q=1}^{n_i^*-|\Gamma_{i,1}|} z_{qj}^{(t-1)}$$

.

Proof At the t-th time step, the update for the next time instance $t+1$ can be written as:

$$\mathbf{z}_j^{(t)} = \mathbf{W}\mathbf{z}_j^{(t-1)} = \left[\mathbf{U} + \rho\Phi^{''}\right]\mathbf{z}_j^{(t-1)}$$

Since $\Omega^{''}$ is symmetric, it will have the following structure:

$$\Phi^{''} = \begin{pmatrix} -2\left|\Gamma_{1,1}\right| & 2 & \dots & 2 \\ 2 & -2\left|\Gamma_{2,1}\right| & 1 & 2 \\ \vdots & & & \end{pmatrix}$$

We can write:

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1-\rho & 0 & 0 \\ \vdots & & & \end{pmatrix}$$

Thus, W matrix can be written as:

$$\mathbf{W} = \begin{pmatrix} 1-2\rho\left|\Gamma_{1,1}\right| & 2\rho & \dots & 2\rho \\ 0 & 1-\rho-2\rho\left|\Gamma_{2,1}\right| & \rho & 2\rho \\ \vdots & & & \end{pmatrix}$$

Generalizing the above expression we can write the update rule for each peer as:

$$z_{ij}^{(t)} = \left\{1 - 2\rho\left|\Gamma_{i,1}\right| - \rho(n_i^* - \left|\Gamma_{i,1}\right|)\right\} z_{ij}^{(t-1)} + 2\rho\sum_{q\in\Gamma_{i,1}} z_{qj}^{(t-1)} + \rho\sum_{q=1}^{n_i^*-|\Gamma_{i,1}|} z_{qj}^{(t-1)}$$

□

4.6 Overall Algorithm

In this section we present the overall algorithm. We have two different algorithms: namely, the local ring formation algorithm (**L-Ring**) which is executed only once, offline. The second algorithm is the iterative local privacy preserving sum computation algorithm (**L-PPSC**).

4.6.1 Local Ring Formation Algorithm (L-Ring)

For distributed averaging, peer P_i updates its current state based on the information it gets from its n_i^* neighbors. In order to preserve privacy, P_i does not get the raw data from its neighbors; rather a ring is formed among n_i^* neighbors and sum computation is performed in that ring. We call this ring the *local* ring since each ring is only formed in a peer's neighborhood. This has the advantage that (1) the algorithm is only synchronous in a peer's local neighborhood and (2) the communication is bounded due to local peer interactions.

L-Ring takes as input the predefined values of cost and threat threshold, *i.e.* c_i and t_i. When the algorithm starts, each peer solves a local optimization problem based on local constraints c_i and t_i to choose a value of n_i^*, the size of the ring for sum computation. It then launches n_i^* random walks in order to select n_i^* nodes uniformly from the network to participate in P_i's ring. The random walk we have used is the Metropolis-Hastings random walk which gives uniform samples even for skewed networks. We do not present the details here, interested readers are referred to [6]. Whenever one random walk ends at P_j, it first checks if $n_i^* < n_j^*$. If this is true, it poses a potential privacy breach for P_j. Hence P_j may choose not to participate in P_i's call by sending a **NAC** message along with its n_j^*. Otherwise P_j sends an **ACK** message to P_i. If P_i has received any **NAC** message, it computes $max(n_j^*)$ and checks if it violates its cost constraint. If the constraint is violated, P_i chooses a different peer P_q by launching a different random walk. Otherwise, it then sends out all of the $max(n_j^*)$ invitations again which satisfies the privacy constraints of all the participants. The pseudocode is presented in Alg. 1.

Algorithm 1 L-Ring

```
Input of peer $P_i$:
  Threat $t_i$ and cost $c_i$ that peer $P_i$ is willing to tolerate
Initialization:
  Find the optimal value of $n_i^*$ using $t_i$ and $c_i$.
If $P_i$ initializes a ring:
  Contact the neighbors as dictated by $n_i^*$ by launching $n_i^*$ parallel random walks
When a random walk ends in node $P_j$:
  Fetch the value of $n_i^*$ as sent by $P_i$
  IF ($n_i^* < n_j^*$) Send (NAC,$n_j^*$) to $P_i$ ELSE Send ACK to $P_i$
  ENDIF
On receiving NAC, $n_j^*$ from $P_j$:
  IF replies received from everyone
    IF $n_j^*$ violates cost constraint
      Contact different neighbor $P_q$
    ELSE $max = argmax_j\{n_j^*\}$; Set $n_i^* = max$
      Send invitation $I(n_i^*)$ to $P_j$ with $n_i^*$ value
    ENDIF
  ENDIF
```

Once the rings are formed offline, the local sum computations start.

4.6.2 Local Privacy Preserving Sum Computation Algorithm (L-PPSC)

In the local privacy preserving distributed sum computation algorithm (**L-PPSC**), initially all peers in the network have a data vector of size p which represents the number of clicks for each of the p advertisements under consideration. The j-th entry of this vector corresponds to the number of clicks of advertisement $\mathcal{A}_j$. Below we discuss the algorithm with respect to only one sum computation (a scalar quantity). In practice, the secure sum will be computed over a vector of size p, the number of advertisements. Assuming that each peer has agreed on a ring in its local neighborhood, each initiator peer starts a round of sum computation based on the secure sum computation. The message sent by the initiator node for any sum computation contains: (1) the ID of the initiator, (2) the data which needs to be added for the local sum, (3) the size of the local ring that it has constructed for the sum, and (4) which peer needs to multiply the data by 2 (according to Lemma 41).

This algorithm differs from traditional secure sum computation protocol in the update rule and the enforcement of the ring topology. In the traditional version, the initiator sends its data masked by a random number while all others in the ring add their numbers as is and pass the sum on. Here, however, the initiator specifies in its message the parameters of the update rule: the amount of scaling that some of the peers might need to do to their data before adding them to the received sum and passing them on. This is essential to guarantee convergence of the algorithm to the correct result, following Lemma 41.

These steps are executed by every peer in the system. The algorithm is locally synchronous in that, during every round of sum computation, the initiator has to wait for all peers in its rings to complete their previous round. This is essential since this algorithm is based on the working of first order LTI systems where the update in the t-th round uses data from the $(t-1)$-st round. Algorithm 2 lists the steps in a pseudo-code format.

Algorithm 2 Local Privacy Pres. Sum Comp. (L-PPSC)

```
Input of peer $P_i$:
  Convergence rate $\rho$, local data $x_i$, $round$, set of $n_i^*$-local neighbors arranged in a ring or
$\{ring_{i,n^*}\}$, random number $R$, and the max range of the sum $N$
Initialization:
  Initialize $\{ring_{i,n^*}\}$, $\rho$, $x_i$; Set $round \leftarrow 1$
  Set $j \leftarrow$ first entry of $\{ring_{i,n^*}\}$
  $\{ring_{i,n^*}\} \leftarrow \{ring_{i,n^*}\} \setminus j$
  Send $(R + x_i, \{ring_{i,n^*}\}, round)$ to $j$
On receiving a message $(data, \{ring\}, rnd)$ from $P_j$:
  IF $\{ring\} = \emptyset$
    Update $z_i^{(round)}$ using $(data - R)$ and Lemma 41;
    $round \leftarrow round + 1$;
    Set $j \leftarrow$ first entry of $\{ring_{i,n^*}\}$
    $\{ring_{i,n^*}\} \leftarrow \{ring_{i,n^*}\} \setminus j$
    Send $\left(z_i^{(round)}, \{ring_{i,n^*}\}\ round\right)$ to $j$
    Check if any node is waiting on this peer
    Send data to all such nodes
  ELSE IF $round < rnd$ Wait
    ELSE
      Set $y = (data + z_i^{rnd}) \mathrm{mod} N$; Set $j \leftarrow$ first entry of $\{ring\}$
      $\{ring\} \leftarrow \{ring\} \setminus j$; Send $(y, ring, rnd)$ to $P_j$
  END
```

Using **L-PPSC** algorithm the peers can compute the sum of the number of clicks for each advertisement in a privacy preserving (not secure) fashion. Once that is done, ranking them by popularity becomes a sorting problem which each peer can solve independently.

4.7 Illustration

In this section we illustrate the working of the **L-Ring** and the **L-PPSC** algorithm. Figure 1 shows a small arbitrary peer-to-peer network. The next sequence shows how

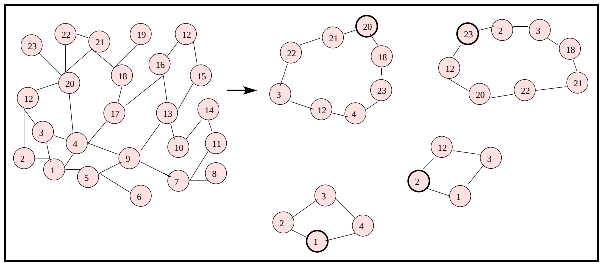


Fig. 1 Figure showing how local rings are formed based on **L-Ring** protocol. It shows four rings with the initiators highlighted. Note that a given node (*e.g.* node 12) is part of multiple rings.

the rings are formed. The peers shown in bold are the initiator nodes for the respective rings. For example, for the two smaller rings, the initiators are peers P_1 and P_2 respectively. For the two larger rings, the initiators are peers P_{20} and P_{23}. To illustrate, assume that P_{20}'s privacy value is high ($n^*_{20} = 7$). Hence there are seven other peers in P_{20}'s ring. Now, if P_{23} wants to include P_{20} in its ring, it must satisfy the privacy requirements of P_{20} as well. As a result, there are seven other peers in the ring initiated by P_{23} (although it is possible that initially $n^*_{23} < 7$). For peer P_{20}, $\Gamma_{20,1} = \{P_4, P_{12}, P_{21}, P_{22}, P_{23}\}$ and so $\left|\Gamma_{20,1}\right| = 5$. Since $n^*_{20} = 7$, $n^*_{20} - \left|\Gamma_{20,1}\right| = 2$.

Using Lemma 41, the update rule for peer P_{20} can be written as:

$$\begin{aligned} z^{(t)}_{20j} &= \{1 - 2\rho\,|\Gamma_{20,1}| - \rho(n^*_{20} - |\Gamma_{20,1}|)\}\, z^{(t-1)}_{20j} + 2\rho \sum_{\ell\in\Gamma_{20,1}} z^{(t-1)}_{\ell j} + \rho \sum_{\ell=1}^{n^*_{20}-|\Gamma_{20,1}|} z^{(t-1)}_{\ell j} \\ &= \{1 - 2\rho\times 5 - \rho(7-5)\} z^{(t-1)}_{20j} + 2\rho\left(z^{(t-1)}_{12j} + z^{(t-1)}_{23j} + z^{(t-1)}_{22j} + z^{(t-1)}_{21j} + z^{(t-1)}_{4j}\right) \\ &\quad +\rho\left(z^{(t-1)}_{3j} + z^{(t-1)}_{18j}\right) \\ &= (1 - 12\rho)\, z^{(t-1)}_{20j} + 2\rho\left(z^{(t-1)}_{12j} + z^{(t-1)}_{23j} + z^{(t-1)}_{22j} + z^{(t-1)}_{21j} + z^{(t-1)}_{4j}\right) + \rho\left(z^{(t-1)}_{3j} + z^{(t-1)}_{18j}\right) \end{aligned}$$

The coefficients of the update rule are passed on by P_{20} at the beginning of any sum computation.

5 Algorithm Analysis

In this section we analyze the properties the **L-Ring** and **L-PPSC** algorithms.

5.1 L-Ring Running Time

The running time of **L-Ring** algorithm is $O(max(n^*_i, n^*_j))$, where n^*_i is the optimal value for node P_i and n^*_j is the value required by node P_j where P_i and P_j belong to the same ring for the sum computation. This can be proved by considering these two cases:

1. For all $P_j \in \Gamma_{i,1}$, if $n^*_i > n^*_j$, then the running time is upper bounded by the maximum time required by P_i to contact all its neighbors *i.e.* $O(n^*_i)$.

2. Without loss of generality, assume that

$$\Xi = \{P_1, \ldots, P_{S_i}\} \subseteq \Gamma_{i,1}$$

be the set of nodes whose n_j^*, for all $P_j \in \Xi$ is greater than n_i^* *i.e.* $\forall P_j \in \Xi, n_j^* \geq n_i^*$. These are the number of **NAC** messages received by P_i from all $P_j \in \Gamma_{i,1}$. Computing the maximum of all entries in Ξ takes $O(|\Xi|)$. In order to accommodate all the nodes in its neighborhood, P_i increases its ring size to $\max_{P_j \in \Xi}\{n_j^*\}$. In this case, computation on this ring takes time $O(n_j^*)$.

Therefore the overall running time is $O(\max(n_i^*, n_j^*))$.

5.2 L-PPSC Privacy

Lemma 51 *For any P_i, the ρ_{1i}-to-ρ_{2i} privacy is satisfied in the **L-PPSC** protocol.*

Proof For any node P_i, there are two rings in which it participates in the computation. P_i's initiated ring must satisfy the privacy model since it is a solution to the optimization problem with the parameters of the model as the constraints. Similarly, for any invitation that P_i receives, it only joins if $n_j^* > n_i^*$ which also guarantees conformity to the ρ_{1i}-to-ρ_{2i} model for P_i. Therefore, using the privacy model defined in 2, the **L-PPSC** protocol is distributed ρ_1-to-ρ_2 privacy preserving. □

In the **L-PPSC** algorithm, it is assumed that each ring has fewer than $(n_i^* - 2)$ bad nodes. If this condition is violated, then we know that privacy breach will surely occur. Next we derive an expression for the probability of this happening and show that it is very low.

Lemma 52 *Let θ be the probability of a node being good. Then the probability that in a ring of size n_i^*, there are at most $(n_i^* - 2)$ bad nodes is given by $1 - (1-\theta)^{n_i^*-1}$.*

Proof Let ε count the number of bad nodes. Then,

$$\begin{aligned} Prob(\varepsilon \leq n_i^* - 2) &= 1 - Prob(\varepsilon = n_i^* - 1) - Prob(\varepsilon = n_i^*) \\ &= 1 - \theta(1-\theta)^{n_i^*-1} - (1-\theta)^{n_i^*} \\ &= 1 - (1-\theta)^{n_i^*-1} \end{aligned}$$

□

The above expression shows that the probability of selecting less than $n_i^* - 2$ bad nodes increases with increase in the (1) probability of a good node θ, and (2) ring size n_i^*. Figure 2 (top) shows how the probability varies as a function of θ and n_i^*. As shown, the probability increases with increasing θ. This is intuitive, since with increasing θ, there is a higher chance that each contacted node is good. Also for a fixed θ, as n_i^*, the ring size increases and the probability of contacting less than $n_i^* - 2$ bad nodes goes to 1 faster.

Now consider another scenario in which there is the possibility of a privacy breach. Consider two intersecting rings which contains only one honest node. Now the probability of this occurring is given by $\theta(1-\theta)^{n_i^*+n_j^*-1}$, where n_i^* and n_j^* are the sizes of the two rings. Figure 2 (bottom) demonstrates the variation of this expression with θ, n_i^* and n_j^*. As seen in the figure, the probability is very low and decreases with increasing size of the ring. Also, for a fixed ring size, as θ increases, the probability decreases.

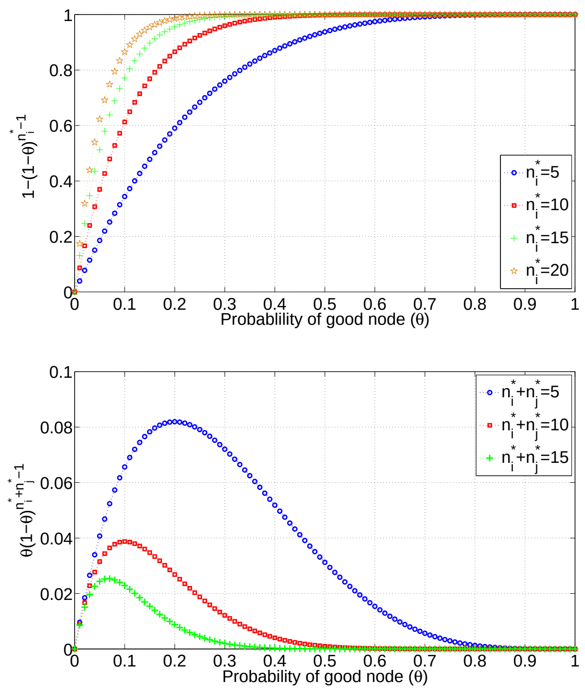


Fig. 2 The top figure shows the probability that less than $n_i^* - 2$ nodes are bad in a ring of size n_i^*. The bottom figure demonstrates the variation of $\theta(1-\theta)^{n_i^*+n_j^*-1}$ vs. θ, n_i^* and n_j^*.

5.3 Correctness and Convergence

L-PPSC protocol is based on the distributed averaging protocol proposed by Scherber and Papadopoulos [21]. The correctness proof of **L-PPSC** can be derived based on two observations analogous to [21]: (1) $\mathbf{W}.\mathbf{1} = \mathbf{W}^T.\mathbf{1} = \mathbf{1}$ and (2) the eigenvalues of $\mathbf{W}$, λ_i when arranged in descending order are such that $\lambda_1 = 1$ and $|\lambda_i| < 1$. It is easy to prove these two statements based on the proof given in [21].

Following similar arguments in [21], we can show that the error (between the true average and the estimate at each peer) tends to zero exponentially fast as the number of iterations tend to infinity.

5.4 Locality

There are several definitions of locality proposed in the literature. The locality concept proposed by Das *et al.* [6] is characterized by two quantities — (1) α – which is the number of neighbors a peer contacts in order to find answer to a query and (2) γ – which is the total size of the response which a peer receives as the answer to all the queries executed throughout the lifetime of the algorithm.

In case of **L-PPSC**, the choice of α is guided by the optimal solution of the objective function defined earlier. In the worst case, a peer may choose α to be equal to the size of the entire network. Therefore, $\alpha = O(d)$ in the worst case. Next we derive a value of γ based on the error induced and the convergence rate.

Lemma 53 *Let ϵ be the error between the true sum (Δ) and the node estimates ($\boldsymbol{z}_j^{(t)}$) as induced by the **L-PPSC** algorithm after t rounds of computation. Then $t \geq \frac{log(\epsilon)-log(d)}{log(\lambda_{max}^2)}$, where λ_i's are the eigenvalues of the topology matrix Φ''.*

Proof From [21], we know that the error at the t-th step is bounded by $d\lambda_{max}^{2t}$ *i.e.*

$$\left|\left|\mathbf{z}_j^{(t)} - 1\Delta_j\right|\right|^2 = \textstyle\sum_{j=2}^{d} \lambda_i^{2t} \left|\mathbf{z}_j^{(0)}\right|^2 \text{ (assuming } \left|\mathbf{z}_j^{(0)}\right|^2 = 1)$$

Now let the error be bounded by ϵ. Thus,

$$d\lambda_{max}^{2t} < \epsilon \Rightarrow t \geq \frac{log(\epsilon) - \log(d)}{\log(\lambda_{max}^2)}$$

□

Theorem 1 *The total size of the messages exchanged (γ) by any peer is upper bounded by*

$$\frac{log(\epsilon) - \log(d)}{\log(\lambda_{max}^2)} \left[\log(z_{max}^{(t)}) + n_i^*\right],$$

where $z_{max}^{(t)}$ is the maximum of data values at any peer in a ring at round t.

Proof At round t, the number of bits necessary to store the maximum of all the $z_i^{(t)}$-s is $\log(z_{max}^{(t)})$. While performing the secure sum at any round ℓ, peer P_i with $\Gamma_{i,1} = \{P_{i-1}, P_{i+1}\}$ does the following computation: $(z_{i-1}^{(\ell)} + z_i^{(\ell)}) \mod N$, where N is the parameter of the sum computation protocol. Hence for every peer, the number of bits required to represent the new sum will increase by 1 at most. Therefore, the total number of bits required for each message is upper bounded by $\left[\log(z_{max}^{(\ell)}) + n_i^*\right]$. In each round of the sum computation, a peer exchanges only one message (due to ring topology). Hence, for t rounds, we get the total number of bits exchanged as $t\left[\log(z_{max}^{(t)}) + n_i^*\right]$. Using Lemma 53,

$$\gamma \leq \frac{log(\epsilon) - \log(d)}{log(\lambda_{\max}^2)} \left[\log(z_{max}^{(t)}) + n_i^*\right].$$

□

Lemma 54 ***L-PPSC** algorithm is $\left(O(d), \frac{\log(\epsilon)-\log(d)}{\log(\lambda_{\max}^2)}\left[\log(z_{max}^{(t)}) + n_i^*\right]\right)$-local.*

Proof As stated, for any node P_i the maximum size of ring is equal to the size of the network. So according to the definition of locality, $\alpha = O(d)$. Also as shown in Theorem 1,

$$\gamma \leq \frac{\log(\epsilon) - \log(d)}{\log(\lambda_{\max}^2)} \left[\log(z_{max}^{(t)}) + n_i^*\right]$$

. Therefore, **L-PPSC** algorithm is $\left(O(d), \frac{\log(\epsilon)-\log(d)}{\log(\lambda_{\max}^2)}\left[\log(z_{max}^{(t)}) + n_i^*\right]\right)$-local. □

6 Experimental Results

To validate the performance of the proposed **L-PPSC** algorithm, we have conducted experiments on a simulated network of peers. The topology is generated using BRITE[3]. We have used the Barabasi Albert (BA) model in BRITE since it is often considered a reasonable model for the Internet. In all our experiments, we have used the following default values of the system and algorithm parameters: size of the network $(d) = 1000$, the maximum range of the sum for the secure computation $(N) = x_i \times d$ and $\rho = \max_i \frac{1}{|\Phi_{ii}|}$.

6.1 Experiments on Synthetic Dataset

In this section we first discuss about the dataset and then describe the convergence and scalability results.

As already noted, each peer agrees on a predefined set of advertisements. We assume that there are 5 advertisements A, B, C, D and E with arbitrary counts. The goal is to find the sum of all the clicks on advertisements over all the peers. A data set was generated consisting of tuples from different random distributions. Each advertisement is generated from a fixed uniform distribution (with a fixed range). Thus, there are as many different distributions as the number of advertisements. This centralized data set was then split among a fixed number of neighbors such that each peer has a fraction of the count of all the advertisements (0 if none exists). Note that this requires a separate privacy-preserving sum algorithm to be invoked for each advertisement/category. For the rest of this section we will present our results with respect to one sum computation only.

As shown in Figure 3 (top), the algorithm converges to the correct sum with respect to a centralized algorithm, where a centralized algorithm is one which has access to all the data of all the peers. In this figure we have plotted the estimate of all the peers at each time instance *i.e.* the $\mathbf{z}_j^{(t)}$ values for each t. To start with, each peer is assigned a data value which corresponds to the number of clicks of a particular advertisement. Hence, initially the estimate of each peer is close to its local data. As time progresses, the peers slowly converge to the correct sum. Figure 3 (bottom) demonstrates the number of messages per peer.

In Figure 4 (top), we show the correctness result of the **L-PPSC** algorithm (in triangle) when the number of peers vary from 100 to 2000. Also shown in the figure are the results computed by a centralized algorithm on the same data (using the circles). The graph shows that our algorithm converges to the correct result for varying sizes of the network. The cost of the algorithm with increasing network size is demonstrated in Figure 4 (bottom). It can be noted that the number of messages per peer is almost a constant. Hence, our algorithm is highly scalable.

6.2 Results on Real Dataset

Finally in this section we describe the results of the experiments with a real data set. Volunteers at UMBC were asked to search for the following five categories in the

[3] `http://www.cs.bu.edu/brite/`

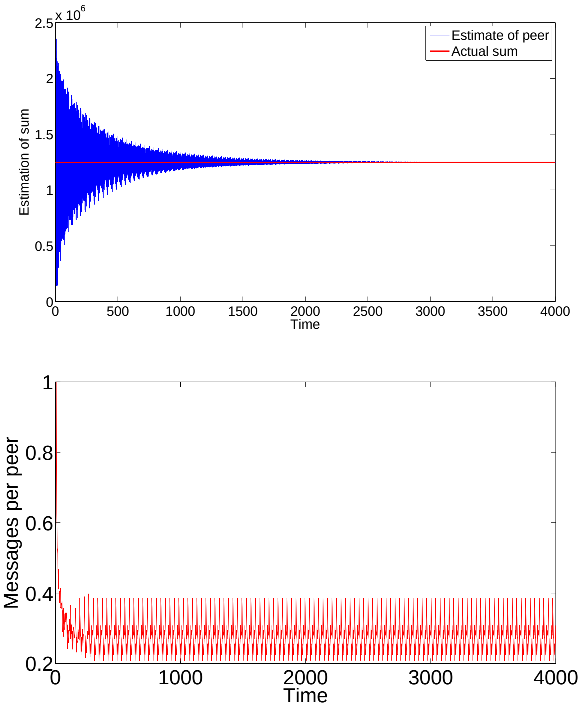


Fig. 3 Convergence to global sum and communication cost per peer.

popular search engines: (1) digital camera, (2) auto insurance, (3) cars, (3) laptop, and (4) gps systems. They were also asked to store the web urls which they found as the closest match for each of these categories. In the experimental setup, we list all these links in a single file (for all categories) and for each link, count the number of times it has been reported by the volunteer. In order to simulate the P2P setup, we then divide this data file randomly among 100 peers, such that each peer contains only fraction of the data — either links or count for each link. If a peer does not have a link, it may add a value of zero in order to participate in the **L-PPSC** protocol. In total there were 1000 links. Once the rings were formed using the **L-Ring** protocol, we ran 1000 sum computations in parallel. Figure 5 shows the results of the **L-PPSC** protocol on this data set. The x-axis in the quality figure (top) refers to the 1000 links grouped per category. The y-axis shows the total count per link for the **L-PPSC** protocol (circles). Also shown in the figure are the true counts per link (diamonds) which we call the centralized execution scenario. As easily verified, the counts of the links in the distributed experiments is very close to those found in the centralized situation. Similarly, the cost figure (bottom) shows the number of messages exchanged per peer per unit of time which varies between 0.5 and 1. A value of x at a particular time instance means that only x% of all the peers send message at that time instance.

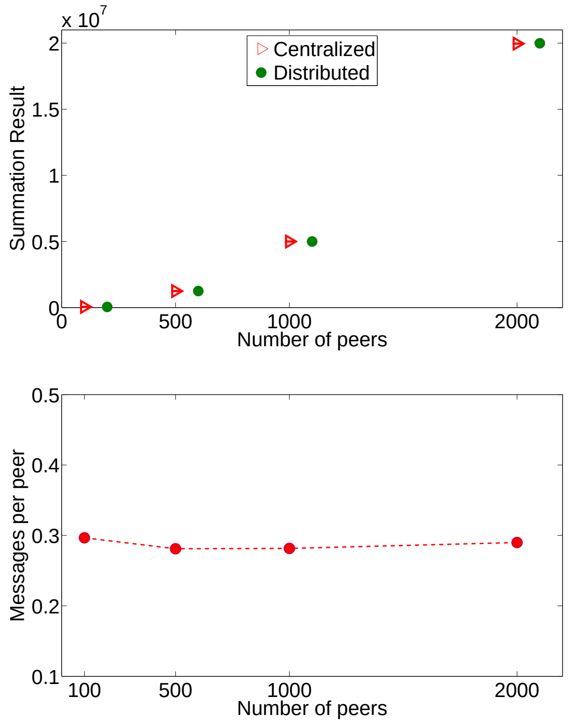


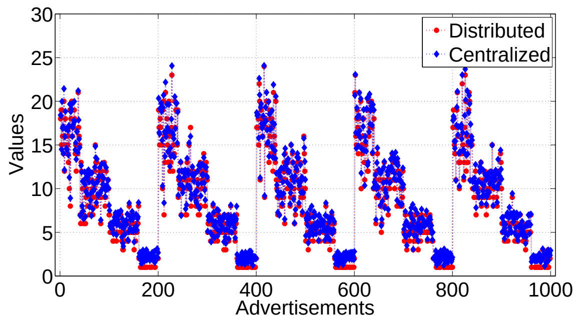
Fig. 4 Scalability of the algorithm as the number of peers is increased: quality (top) and cost (bottom).

7 Related Work

Data mining in P2P networks is at its nascency. In this section we present some previous work related to this area of research.

7.1 P2P Data Mining

Distributed data mining (DDM) deals with the problem of data analysis in environments with distributed data, computing nodes, and users. Kargupta [14][15] presents a detailed overview of this topic. A more recent research topic is data mining in large P2P networks. This paradigm of computing poses several challenges for which the algorithms need to be asynchronous, communication-efficient and scalable. Datta *et al.* [7] presents an overview of this topic. Examples of scalable distributed P2P data mining algorithms include the association rule mining algorithm [26], k-Means clustering [8], top-l inner product identification [6], decision tree induction [3], expectation maximization [2] and more.



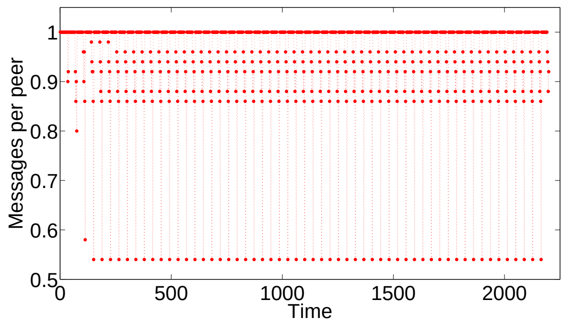


Fig. 5 Results on the real advertisement data set: relative orderings (top) and messages exchanged (bottom).

7.2 Privacy in P2P Network

P2P networks have recently emerged as huge information systems for collaborative computation such as file sharing, distributed computing, real-time telephone and tv, and academic applications. However, free flow of information is frequently prohibited by legal obligations or by commercial and personal concerns. Privacy preserving data mining in P2P networks aims to solve this problem by allowing users to share their data without revealing the sensitive information. In a large scale cooperative computation environment, each node has a private input x_i. They wish to jointly compute the output $f(x_1, x_2, \ldots, x_n)$ of some common function f. At the end of the data mining process, nothing but the output should be revealed. PPDM solves this problem by allowing useful data patterns to be extracted without revealing sensitive data *e.g.* the SMC protocols [5][27]. Gilburd *et al.* presents a privacy model called k-TTP for large-scale distributed environments [12]. Kargupta *et al.* [13] presents a game theoretic solution for dealing with the collusion problem in secure sum computation in a large distributed environment. Teng and Du [23] present a hybrid technique for PPDM. They combine randomization and SMC protocols to achieve better accuracy and lower running times than these algorithms acting independently. Most of these techniques suffer from one major drawback — scalability — which hinders their deployment in large P2P networks. The technique proposed in this paper is highly scalable and asynchronous, thus enabling privacy preserving data mining in P2P networks.

8 Conclusion

In this paper we have presented a local privacy-preserving peer-to-peer data aggregation algorithm for doing data mining in a large P2P setting. Due to the constant communication complexity and locally synchronous nature of the algorithm, it is highly scalable. We have framed privacy and cost as a multi-objective optimization problem local to each peer and shown that our proposed algorithm is privacy-preserving. To the best of the authors' knowledge, this is one of the first solutions which blends in the concept of local asynchronous distributed averaging with secure sum protocol to develop a scalable privacy preserving sum computation algorithm tailored to accommodate every participant's privacy and cost constraints. This algorithm is, therefore, applicable for large scale heterogeneous distributed systems such as the Internet and has various applications that require privacy preserving data mining.

As future work, we would like to incorporate the situation in which each peer can choose its own privacy model. Currently we only allow peers to select from the same type of privacy model such as k-anonymity, ℓ-diversity or t-closeness model. It does not allow a peer to choose a different privacy model such as ϵ-differential privacy since the nature of the optimization problem changes with the choice of the model. We plan to develop a distributed multi-objective optimization solution for a class of optimization problems so that we can extend this application to a wide range of existing privacy models and other data mining applications.

References

1. M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating Aggregates on a Peer-to-Peer Network. Technical report, Stanford University, April 2003.
2. K. Bhaduri and A. Srivastava. A Local Scalable Distributed Expectation Maximization Algorithm for Large Peer-to-Peer Networks. In *Proceedings of ICDM'09*, pages 31–40, Miami, FL, 2009.
3. K. Bhaduri, R. Wolff, C. Giannella, and H. Kargupta. Distributed Decision Tree Induction in Peer-to-Peer Systems. *Statistical Analysis and Data Mining (SAM)*, 1(2):85–103, 2008.
4. S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.
5. C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu. Tools for Privacy Preserving Distributed Data Mining. *ACM SIGKDD Explorations*, 4(2), 2003.
6. K. Das, K. Bhaduri, K. Liu, and H. Kargupta. Distributed Identification of Top-l Inner Product Elements and its Application in a Peer-to-Peer Network. *TKDE*, 20(4):475–488, 2008.
7. S. Datta, K. Bhaduri, C. Giannella, R. Wolff, and H. Kargupta. Distributed Data Mining in Peer-to-Peer Networks. *IEEE Internet Computing*, 10(4):18–26, 2006.
8. S. Datta, C. Giannella, and H. Kargupta. K-Means Clustering over a Large, Dynamic Network. In *Proceedings of SDM'06*, pages 153–164, MD, 2006.
9. K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* Wiley, 2001.
10. A. Evfimevski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proc. of SIGMOD'03*, San Diego, CA, June 2003.
11. A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*, July 2002.
12. B. Gilburd, A. Schuster, and R. Wolff. k-TTP: A New Privacy Model for Large-Scale Distributed Environments. In *Proc. of KDD'04*, pages 563–568, Seattle, 2004.
13. H. Kargupta, K. Das, and K. Liu. Multi-Party, Privacy-Preserving Distributed Data Mining using a Game Theoretic Framework. In *Proc. of PKDD'07*, pages 523–531, 2007.
14. H. Kargupta and K. Sivakumar. *Existential Pleasures of Distributed Data Mining. Data Mining: Next Generation Challenges and Future Directions.* AAAI/MIT press, 2004.

15. Hillol Kargupta and Philip Chan, editors. *Advances in Distributed and Parallel Knowledge Discovery.* MIT Press, 2000.
16. N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and ℓ-diversity. In *Proceedings of ICDE'07*, pages 106–115, 2007.
17. K. Liu, K. Bhaduri, K. Das, P. Nguyen, and H. Kargupta. Client-side Web Mining for Community Formation in Peer-to-Peer Environments. *SIGKDD Explorations*, 8(2):11–20, 2006.
18. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramanian. l-diversity: Privacy beyond k-anonymity. In *Proc. of ICDE'06*, page 24, GA, April 2006.
19. S. Mane, S. Mopuru, K. Mehra, and J. Srivastava. Network Size Estimation In A Peer-to-Peer Network. Technical Report 05-030, University of Minnesota, September 2005.
20. M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. Murray. Distributed Averaging on Peer-to-Peer Networks. In *Proc. of CDC'05*, Spain, 2005.
21. D. Scherber and H. Papadopoulos. Distributed Computation of Averages Over ad hoc Networks. *IEEE Journal on Selected Areas in Communications*, 23(4):776–787, 2005.
22. L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
23. Zhouxuan Teng and Wenliang Du. Hybrid Multi-group Approach for Privacy-preserving Data Mining. *Knowl. Inf. Syst.*, 19(2):133–157, 2009.
24. M. Trottini, S. Fienberg, U. Makov, and M. Meyer. Additive Noise and Multiplicative Bias as Disclosure Limitation Techniques for Continuous Microdata: A Simulation Study. *J. Comp. Methods in Sci. and Eng.*, 4(1,2):5–16, 2004.
25. R. Wolff, K. Bhaduri, and H. Kargupta. A Generic Local Algorithm for Mining Data Streams in Large Distributed Systems. *IEEE Transactions on Knowledge and Data Engineering*, 21(4):465–478, 2009.
26. R. Wolff and A. Schuster. Association Rule Mining in Peer-to-Peer Systems. *IEEE SMC - Part B*, 34(6):2426 – 2438, 2004.
27. A. C. Yao. How to Generate and Exchange Secrets (Extended Abstract). In *FOCS*, pages 162–167, 1986.

Author Biographies

Kamalika Das received her B.Tech degree in CSE from Kalyani University, India, in 2003 and her MS and PhD degrees in CS from University of Maryland Baltimore County, in 2005 and 2009 respectively. Currently she is working as a postdoc with the Intelligent Data Understanding group at the NASA Ames Research Center. Kamalika has published several conference and journal papers in PKDD, SDM, P2P, IEEE Transactions and SIGKDD Explorations. Her work has been nominated for the 2007 PET award, and invited for fast-track submission to Peer-to-Peer Networking and Applications journal as one of the best papers of P2P 2009. She has received the 2008 Grace Hopper Celebration of Women in Computing Scholarship Award and the NSF-sponsored SIAM Student travel award for SDM 2009. Kamalika serves regularly as a reviewer for SDM, SIGKDD, ICDM conferences and TKDE, TIFS and SMC journals. More information about her can be found at http://www.csee.umbc.edu/˜kdas1.

Kanishka Bhaduri is currently working as a research scientist in the Intelligent Data Understanding group at NASA Ames Research Center. He received his B.E. in Computer Science and Engineering from Jadavpur University India (2003) and PhD from the University of Maryland Baltimore County (2008). His research interests include distributed and P2P data mining, data stream mining, and statistical data mining. Several of his papers in top conferences have been selected as 'best' papers. Kanishka regularly serves as a reviewer and/or PC member for SDM, KDD, ICDM conferences and TKDE, SMC and DMKD journals. More information about him can be found at http://ti.arc.nasa.gov/profile/kbhaduri/.

Hillol Kargupta is a Professor at the Department of CSEE, University of Maryland Baltimore County. He received his Ph.D. in Computer Science from University of Illinois at Urbana-Champaign in 1996. He is also a co-founder of AGNIK LLC, a ubiquitous data intelligence company. His research interests include distributed data mining, data mining in ubiquitous environment, and privacy-preserving data mining. Dr. Kargupta won a US National Science Foundation CAREER award in 2001 for his research on ubiquitous and distributed data mining. He received the best paper award at the 2003 IEEE International Conference on Data Mining. He has published more than 90 peer-reviewed articles in journals, conferences, and books. He is an associate editor of the IEEE TKDE, the IEEE SMC Part B, and the SAM Journal. He regularly serves on the organizing and program committees of many data mining conferences. More information about him can be found at http://www.csee.umbc.edu/˜hillol.