Incentive Compatible Privacy-Preserving Data Analysis

Murat Kantarcioglu and Wei Jiang

Abstract—In many cases, competing parties who have private data may collaboratively conduct privacy-preserving distributed data analysis (PPDA) tasks to learn beneficial data models or analysis results. Most often, the competing parties have different incentives. Although certain PPDA techniques guarantee that nothing other than the final analysis result is revealed, it is impossible to verify whether participating parties are truthful about their private input data. Unless proper incentives are set, current PPDA techniques cannot prevent participating parties from modifying their private inputs. This raises the question of how to design incentive compatible privacy-preserving data analysis techniques that motivate participating parties to provide truthful inputs. In this paper, we first develop key theorems, then base on these theorems, we analyze certain important privacy-preserving data analysis tasks that could be conducted in a way that telling the truth is the best choice for any participating party.

Index Terms—Privacy, secure multiparty computation, noncooperative computation

1 Introduction

Privacy and security, particularly maintaining confidentiality of data, have become a challenging issue with advances in information and communication technology. The ability to communicate and share data has many benefits, and the idea of an omniscient data source carries great value to research and building accurate data analysis models. For example, for credit card companies to build more comprehensive and accurate fraud detection system, credit card transaction data from various companies may be needed to generate better data analysis models. Department of Energy supports research on building much more efficient diesel engines [7]. Such an ambitious task requires the collaboration of geographically distributed industries, national laboratories, and universities. Those institutions (including potentially competing industry partners) need to share their private data for building data analysis models to understand the underlying physical phenomena.

An omniscient data source eases misuse, such as the growing problem of identity theft. To prevent misuse of data, there is a recent surge in laws mandating protection of confidential data, such as the European Community privacy standards [9], U.S. health-care laws [17], and California SB1386. However, this protection comes with a real cost through both added security expenditure and penalties and costs associated with disclosure. What we need is the ability to compute the desired "beneficial outcome" of data sharing for analyzing without having to actually share or disclose data. This would maintain the security provided by separation of control while still obtaining the benefits of a global data source.

Secure multiparty computation (SMC) [11], [44], [45] has recently emerged as an answer to this problem. Informally, if a protocol meets the SMC definitions, the participating parties learn only the final result and whatever can be inferred from the final result and their own inputs. A simple example is Yao's millionaire problem [44]: two millionaires, Alice and Bob, want to learn who is richer without disclosing their actual wealth to each other. Recognizing this, the research community has developed many SMC protocols, for applications as diverse as forecasting [5], decision tree analysis [33] and auctions [37] among others.

Nevertheless, the SMC model does not guarantee that data provided by participating parties are truthful. In many real-life situations, data needed for building data analysis models are distributed among multiple parties with potentially conflicting interests. For instance, a credit card company that has a superior data analysis model for fighting credit card fraud may increase its profits as compared to its peers. An engine design company may want to exclusively learn the data analysis models that may enable it to build much more efficient diesel engines. Clearly, as described above, building data analysis models is generally performed among parties that have conflicting interests.

In SMC, we generally assume that participating parties provide truthful inputs. This assumption is usually justified by the fact that learning the correct data analysis models or results is in the best interest of all participating parties. Since SMC-based protocols require participating parties to perform expensive computations, if any party does not want to learn data models and analysis results, the party should not participate in the protocol. Still, this assumption does not guarantee the truthfulness of the private input data when participating parties want to learn the final result exclusively. For example, a drug company may lie about its private data so that it can exclusively learn the data analysis

- *M. Kantarcioglu is with the Jonsson School of Engineering and Computer Science, The University of Texas at Dallas, 800 West Campbell Road, Richardson, TX. E-mail: muratk@utdallas.edu.*
- *W. Jiang is with the Department of Computer Science, Missouri University of Science and Technology, 310 Computer Science Building, 500 W. 15th St., Rolla, MO 65409-0350. E-mail: wjiang@mst.edu.*

Manuscript received 11 Apr. 2011; revised 27 Oct. 2011; accepted 17 Feb. 2012; published online 9 Mar. 2012.
Recommended for acceptance by E. Ferrari.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2011-04-0198.
Digital Object Identifier no. 10.1109/TKDE.2012.61.

model. Although SMC protocols guarantee that nothing other than the final data analysis result is revealed, it is impossible to verify whether or not participating parties are truthful about their private input data. In other words, unless proper incentives are set, current SMC techniques cannot prevent input modification by participating parties.

To better illustrate this problem, we consider a case from management where competing companies (e.g., Texas Instruments, IBM and Intel) establish a consortium (e.g., Semiconductor Manufacturing Technology[1]). The companies send the consortium their sales data, and key manufacturing costs and times. Then, the consortium analyzes the data and statistically summarizes them in a report of industry trends, which is made available back to consortium members. In this case, it is in the interest of companies to learn *true* industry trends while revealing their private data as little as possible. Even though SMC protocols can prevent the revelation of the private data, they do not guarantee that companies send their true sales data and other required information. Assume that n companies would like to learn the sample mean and variance of the sales data for a particular type of product.

Example 1.1. Let x_i be the ith company's sales amount. In order to estimate the sample mean, companies need to calculate $\mu = \frac{1}{n} \cdot \sum_{i=1}^{nxi}$ and similarly $s^2 = \frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_i - \mu)^2$ for sample variance. Any company may *exclusively* learn the *correct* result by lying about its input. Company i may report x_i' instead of the correct x_i. Given the wrong mean μ' and variance s'^2 (computed based on x_i' and truthful values from the other parties), the company i can calculate the *correct* sample mean μ by setting

$$\mu = \mu' + \frac{x_i - x_i'}{n}.$$

The *correct* sample variance s^2 can be calculated as

$$s^2 = s'^2 + \frac{x_i^2 - x_i'^2}{n-1} + \frac{n(\mu'^2 - \mu^2)}{n-1}.$$

As illustrated above, any company may have the incentive to lie about its input in order to learn the result exclusively, and at the same time, the correct result (e.g., μ) can be computed from its original input, modified input and the incorrect final result (e.g., x_i, x_i', and μ'). If this situation always occurred, no company would have the incentive to be truthful. Fortunately, the intrinsic nature of a function determines whether the situation (demonstrated by the above example) could occur.

1.1 Our Contributions

In this paper, we analyze what types of distributed functionalities could be implemented in an incentive compatible fashion. In other words, we explore which functionalities can be implemented in a way that participating parties have the incentive to provide their true private inputs upon engaging in the corresponding SMC protocols. We show how tools from theoretical computer science in general and noncooperative computation (NCC) [40] in particular could be used to analyze incentive issues in distributed data analysis framework. This is significant because input modification *cannot* be prevented *before* the execution of any SMC-based protocol. (Input modification could be prevented *during* the execution of some SMC-based protocols, but these protocols are generally expensive.) The theorems developed in the paper can be adopted to analyze whether or not input modification could occur for computing a distributed functionality. If the answer is positive, then there is no need to design complicated and generally inefficient SMC-based protocols.

In this paper, we assume that the number of malicious or dishonest participating parties can be at most $n-1$, where n is the number of parties. This assumption is very general since most existing works in the area of privacy-preserving data analysis assume either all participating parties are honest (or semi-honest) or the majority of participating parties are honest. Thus, we extend the noncooperative computation definitions to incorporate cases where there are multiple dishonest parties. In addition, we show that from incentive compatibility point of view, most data analysis tasks need to be analyzed only for two party cases. Furthermore, to show the applicability of our developed theorems, we use these theorems to analyze under what conditions, common data analysis tasks, such as mean and covariance matrix estimation, can be executed in an incentive compatible manner.

The paper is organized as follows: Section 2 provides an overview of the works closely related to this paper and background regarding the concept of noncooperative computation. In Section 3, we propose several important theorems along with formal proofs. Based on these theorems, Section 4 analyzes some common distributed data analysis tasks that are either incentive compatible or not incentive compatible in the context of this paper. Section 6 concludes the paper with a discussion of possible future research directions.

TABLE 1
Notations and Terminologies

NCC	Non-Cooperative Computation
DNCC	Deterministic NCC
PPDA	Privacy Preserving Data Analysis
SMC	Secure Multi-party Computation
TTP	Trusted Third Party

2 Related Work and Background

We begin with an overview of privacy-preserving distributed data analysis. Then, we briefly discuss the concept of noncooperative computation. Table 1 provides common notations and terminologies used extensively for the rest of this paper. In addition, the terms *secure* and *privacy preserving* are interchangeable thereafter.

2.1 Privacy-Preserving Data Analysis

Many privacy-preserving data analysis protocols have been designed using cryptographic techniques. Data are generally assumed to be either vertically or horizontally partitioned. In the case of horizontally partitioned data, different sites collect the same set of information about different

1. www.sematech.org.

entities. For example, different credit card companies may collect credit card transactions of different individuals. Privacy-preserving distributed protocols have been developed for horizontally partitioned data for many different data mining tasks such as building decision trees, [32], mining association rules, [26], and generating k-means clusters [31] and k-nn classifiers.

In the case of vertically partitioned data, we assume that different sites collect information about the same set of entities, but they collect different feature sets. For example, both a university pay roll and the university's student health center may collect information about a student. Again, privacy-preserving protocols for the vertically partitioned case have been developed for many different data mining tasks such as association rules, [41], building decision trees [8] and k-means clusters [19]. (See [43] for a survey of the results.) To our knowledge, all the existing techniques assume that each participating party use its true data during the distributed data mining protocol execution.

In addition to existing techniques that consider honest-but-curious model, there are techniques developed against malicious adversaries, such as [16], [24]. Especially, in [16], authors discuss how to prevent lying about inputs using "input-consistency checks." Basically, authors suggest checking whether the inputs satisfy some conditions that are known to be true about the inputs (e.g., a binary input vector cannot consist of all zeros). Although such approach could be useful in practice, it cannot prevent lying about inputs that satisfy the domain constraints (e.g., an adversary can lie about its binary vector by making sure that he does not use binary vectors that consists of all zeros as input). In our case, we suggest a different solution. Basically, we try to "incentivize" truth telling instead of preventing lying.

Shoham and Tennenholtz [40] define the class of NCC, or *noncooperatively computable* functions, and define specifically the Boolean functions which are NCC. In addition, the paper defined two additional classes, p-NCC and s-NCC, which stand for probabilistic-NCC and subsidized-NCC, respectively. p-NCC are the functions which are computable with some probability noncooperatively, and s-NCC are the functions which are computable when external monetary motivation is allowed. This was expanded to consider different motivations [35] and coalitions [4]. Our work is basically inspired by [40], but we consider general data mining functions and develop additional techniques to show whether such functions are in NCC.

Much work seeks to include a game-theoretic model in standard secure multiparty computation. Instead of considering players who are honest, semihonest or malicious, the work simply considers players to be rational in the game theoretic sense. Much of this work concentrates on the problem of *secret sharing*; that is, dividing a secret number among players such that any quorum (sufficiently large subset) of them can reconstruct the secret number. This was first studied by Halpern and Teague [14], and later reexamined by Gordon and Katz [13]. Other protocols for this problem were outlined in [1] and [34]. The paper by Ong et al. [39] hybridizes the two areas within the realm of secret sharing by assuming that some players are honest and a majority of players are rational. Other work seeks a broader realm of computation, such as [18] and [29] that build their computation model on a secret sharing model. There is other work that attempts to combine game theoretic and cryptographic methodologies, many of which are surveyed in [28]. Although these rational secure computation systems could be used to ensure privacy of the data mining techniques discussed in this paper, like other secure computation systems, they make no guarantees about the truthfulness of the inputs.

More closely related to the work in this paper, some work has attempted to enforce honest behavior among the participants in a data sharing protocol. Agrawal and Terzi [2] present a model which enforces honesty in data sharing through the use of auditing mechanisms. Layfield et al. in [30] present strategies which enforce honesty in a distributed computation without relying on a mediator. Jiang et al. in [20] integrate the auditing mechanism with secure computation to convert existing protocols into rationally secure protocols. Finally, the work of Kargupta et al. [27] analyzes each step of a multiparty computation process in terms of game theory, with the focus of preventing cheating within the process and removing coalitions from gameplay. Each of these deals with the problem of ensuring truthfulness in data mining. However, each one requires the ability to verify the data after the calculation. Although verification-based techniques are very useful, there are cases where verification is not feasible due to legal, social, and privacy concerns. For example, if two intelligence agencies from different countries are collaborating, one agency may not allow others to verify its database due to legal and security concerns. Our work enables new applications by showing what is possible when verification is not feasible and complements the existing verification based work. In addition, Nix and Kantarcioglu [25] present a model that enforces honesty in distributed data mining using monetary payments. In our case, we do not use any monetary payments to incentivize truth telling.

2.2 Noncooperative Computation

Recently, research issues at the intersection of computer science and game theory have been studied extensively. Among those research issues, algorithmic mechanism design and noncooperative computation are closely related to our work.

The field of algorithmic mechanism design tries to explore how private preferences of many parties could be combined to find a global and socially optimal solution [38]. Usually, in algorithmic mechanism design, there exists a function that needs to be maximized based on the private inputs of the parties, and the goal is to devise mechanisms and payment schemes that force individuals to tell their true private values. In our case, since it is hard to measure the monetary value of the data analysis results, devising a payment scheme that is required by many mechanism design models is not viable (e.g., Vickrey-Groves-Clarke mechanisms [38]). Instead, we adopt the noncooperative computation model [40] that is designed for parties who want to jointly compute the correct function results on their private inputs. Since data analysis algorithms can be seen as a special case, modifying noncooperative computation model for our purposes is a natural choice.

The noncooperative computation model can be seen as an example of applying game theoretical ideas in a distributed computation setting [40]. In NCC, each party participates in a protocol to learn the output of some function f over the joint inputs of the parties. First, all participating parties send their private inputs securely to a trusted third party (TTP), then TTP computes f and sends back the result to every participating party. The NCC model makes the following assumptions:

1. ***Correctness***. The first priority for every participating party is to learn the *correct* result.
2. ***Exclusiveness***. If possible, every participating party prefers to learn the *correct* result *exclusively*.

In other words, learning the correct result is the most important objective of every party. Other factors such as privacy and voyeurism could be also considered in the NCC setting. We omit such discussion here. Additional details can be found in [35]. In this paper, we use the NCC setting where each party wants to learn the data mining result *correctly*, if possible prefers to learn it *exclusively*. Also, we assume that revealing only the result does not violate privacy.

Under the *correctness* and *exclusiveness* assumptions, the NCC model is formally defined as follows: Given a set of n parties, for a party i, we denote its private input as $v_i \in D_i$, where D_i is the domain of the possible inputs of party i. (D_i could be any domain that can be used to represent the possible data sets of the participants. For many practical applications, it could be the set of binary strings with size less than some value k.) For simplicity, we assume that all $D_i = D$ for all i. Parties joint input is represented as $v = (v_1, \ldots, v_n)$, where $v \in D^n$. We use v_{-i} to represent $(v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)$, and (v_i, v_{-i}) to denote the reconstruction of v. It is also assumed that the v values are distributed according to some probability function, and the probability of seeing any $v \in D^n$ is always nonzero. In the NCC model, for calculating any n party function $f : D^n \longmapsto R$ with range R, we use the following simple protocol:

1. Each party i sends v_i' (not necessarily the correct private input) to a TTP.
2. The TTP computes $f(v') = f(v_1', \ldots, v_n')$ and sends the results back to the participating parties.
3. Each party i computes $f(v)$ based on $f(v')$ received from TTP and v_i.

Considering the above simple protocol does not limit its generality. Under the literature of SMC, the TTP can be replaced such that the required functionality (represented by f) is still computable without violating privacy regarding each participating party's private input [14]. The next definition states the conditions a function needs to satisfy under the NCC model.

Definition 2.1 [40]. *Let n, f be as above. Then, f is deterministically noncooperatively computable (DNCC), if the following holds: For any party i, every strategy (t_i, g_i), and every $v_i \in D$, it is the case that*

- *Either $\exists v_{-i} \in D_{-i}$, $g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i})$.*
- *Or $\forall v_{-i} \in D_{-i}$, $f(t_i(v_i), v_{-i}) = f(v_i, v_{-i})$.*

The above definition simply states what function could be computed in NCC setting deterministically (i.e., computation result is correct with probability one), and no party could correctly compute the correct result once the party lies about his or her inputs in a way that changes the original function result. In other words, if a party i replaces its true input v_i with v_i' and if $f(v_i', v_{-i}) \neq f(v_i, v_{-i})$, then party i should not be able to calculate the correct $f(v_i, v_{-i})$ from $f(v_i', v_{-i})$ and v_i. Note that strategy (t_i, g_i) means that the way the input is modified, denoted by t_i, and the way the output is calculated, denoted by g_i. In Example 1.1, t_i can be considered as choosing a value different from the actual input, and g_i can be considered as the ways the correct μ and s^2 are computed. Another implication of the above definition is that for any t_i, the corresponding g_i should be deterministic, because each party want to exactly compute the "correct" result.

In this paper, we mainly focus on the DNCC model instead of considering a probabilistic extension due to the following observations. First, as shown in [40], if the v_i values are independent, then a Boolean function[2] is in DNCC if and only if it is probabilistically noncooperatively computable. Second, even if v_i values are not independent, it is shown that if a Boolean function is in DNCC, then it is also probabilistically noncooperatively computable [40]. Because of these observations, DNCC provides a good basis for understanding incentive issues in privacy preserving data analysis. We leave other possible extensions as a future work.

3 Characteristics of DNCC Functions

As discussed previously, the term *incentive compatible* means that participating parties have the incentive or motivation to provide their actual inputs when they compute a functionality. Although SMC-based privacy-preserving data analysis protocols (under the malicious adversary model) can prevent participating parties from modifying their inputs once the protocols are initiated, they cannot prevent the parties from modifying their inputs before the execution. On the other hand, parties are expected to provide their true inputs to correctly evaluate a function that satisfies the NCC model. Therefore, any functionality that satisfies the NCC model is inherently *incentive compatible* under the assumption that participating parties prefer to learn the function result *correctly*, and if possible *exclusively*. Now, the question is which functionalities or data analysis tasks satisfy the NCC model. For the rest of the paper, we first develop certain key theorems regarding NCC functions. Based on these theorems, we subsequently analyze functionalities that can be implemented under (or satisfy) the NCC model.

Based on Definition 2.1, it is difficult to prove whether a function f is in DNCC because the proof needs to consider all possible t_i and g_i pairs. The strategy t_i defines a way to change the input, and the strategy g_i defines a method to reconstruct the actual result based on the true input, modified input and the result computed based on the modified input and other parties' input data.

2. The distinction between probabilistic and deterministic non-Boolean functions is still an open problem in the literature of NCC.

Example 3.1. For instance, according to Example 1.1 in Section 1, the strategy t_i can be considered as

$$t_i(x_i) = x_i + c = x_i',$$

where c could be any real number. In addition, to compute the sample mean, the strategy g_i is defined as

$$g_i(x_i, x_i', \mu') = \mu' + \frac{x_i - x_i'}{n} = \mu.$$

To compute the sample variance, g_i is defined as

$$g_i(x_i, x_i', s'^2) = s'^2 + \frac{x_i^2 - x_i'^2}{n-1} + \frac{n(\mu'^2 - \mu^2)}{n-1} = s^2.$$

For complex functionalities, it is very difficult to enumerate all possible t_i and g_i pairs. To avoid this issue, we instead develop the following theorem that describes a simpler way to prove that a function is in DNCC.

Theorem 3.1. *A function $f : D^n \mapsto R$ is in DNCC if for any given $v_i \in D$, for every t_i, it is true that*

- *Either $\exists v_{-i}, y_{-i} \in D_{-i}$, $f(t_i(v_i), v_{-i}) \neq f(v_i, v_{-i})$ **and** $f(t_i(v_i), v_{-i}) = f(t_i(v_i), y_{-i})$ **and** $f(v_i, v_{-i}) \neq f(v_i, y_{-i})$.*
- ***Or** $\forall v_{-i} \in D_{-i} f(t_i(v_i), v_{-i}) = f(v_i, v_{-i})$.*

Before we prove the theorem, we need to emphasize the fact that in DNCC setting, a party only lies if it can always compute the *correct* result from the wrong result (based on its modified input and the other parties' inputs) and its original input. Therefore, for any t_i, the corresponding g_i is deterministic. Using this fact, it can be proved that if f satisfies the conditions of the above theorem, then f is in DNCC.

Proof. Please note that Theorem 3.1 is very similar to Definition 2.1. If for every strategy t_i and for any $v_i \in D$, $\forall v_{-i} \in D_{-i}$, $f(t_i(v_i), v_{-i}) = f(v_i, v_{-i})$, then this automatically satisfies the Definition 2.1.

For the case where for all $v_i \in D$ and for all t_i, $\exists v_{-i}, y_{-i} \in D_{-i}$ such that $f(t_i(v_i), v_{-i}) \neq f(v_i, v_{-i})$ **and** $f(t_i(v_i), v_{-i}) = f(t_i(v_i), y_{-i})$ **and** $f(v_i, v_{-i}) \neq f(v_i, y_{-i})$, if we can prove that for all g_i, there exists v_{-i}, such that $g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i})$, then f satisfies Definition 2.1, and we can conclude that f is in DNCC. We will achieve this by proving that for all g_i functions either $g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i})$ or $g_i(f(t_i(v_i), y_{-i}), v_i) \neq f(v_i, y_{-i})$.

First, please note that g_i is deterministic and $f(t_i(v_i), v_{-i}) = f(t_i(v_i), y_{-i})$, then we know that

$$g_i(f(t_i(v_i), y_{-i}), v_i) = g_i(f(t_i(v_i), v_{-i}), v_i). \quad (1)$$

Now, consider the following two cases:

Case 1. Assume that $g_i(f(t_i(v_i), v_{-i}), v_i) = f(v_i, v_{-i})$. Using (1), we infer that

$$g_i(f(t_i(v_i), y_{-i}), v_i) = f(v_i, v_{-i}).$$

Since we are given that $f(v_i, v_{-i}) \neq f(v_i, y_{-i})$, we can conclude that

$$g_i(f(t_i(v_i), y_{-i}), v_i) \neq f(v_i, y_{-i}).$$

Case 2. Assume that $g_i(f(t_i(v_i), y_{-i}), v_i) = f(v_i, y_{-i})$. Using (1), we infer that

$$g_i(f(t_i(v_i), v_{-i}), v_i) = f(v_i, y_{-i}).$$

Since we are given that $f(v_i, v_{-i}) \neq f(v_i, y_{-i})$, we can conclude that

$$g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i}).$$

Using above equations, we can conclude that either $g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i})$ or $g_i(f(t_i(v_i), y_{-i}), v_i) \neq f(v_i, y_{-i})$. Thus, for all g_i, there exists v_{-i}, such that $g_i(f(t_i(v_i), v_{-i}), v_i) \neq f(v_i, v_{-i})$. □

We would like to stress that Theorem 3.1 only states a sufficient condition for a function to be in DNCC. In Section 4, we show how Theorem 3.1 provides guidance for proving some common functionalities that satisfy the DNCC model.

We illustrate how the above theorem could be used for proving certain functions are in DNCC using the following function $f((Y_1, C_1), (Y_2, C_2)) = (Y_1 + Y_2)/(C_1 + C_2)$, where each (Y_i, C_i) belongs to party i.

Example 3.2. Let Y_i be a real number ($Y_i \in \mathcal{R}$) and let C_i a positive integer ($C_i \in \mathcal{Z}^+$). Then, we can prove that $f((Y_1, C_1), (Y_2, C_2)) = (Y_1 + Y_2)/(C_1 + C_2)$ is in DNCC by showing the conditions stated in Theorem 3.1 holds for any t_i. Assume that party 1 (without loss of generality) uses a t_1 to modify its Y_1 and C_1 input. Let Y_1' and C_1' be those modified inputs. Now, consider the case where $f(t_1(Y_1', C_1'), (Y_2, C_2)) = u'$, $f((Y_1, C_1), (Y_2, C_2)) = u$, and $u' \neq u$ for some u' and u. Note that $f(t_1(Y_1', C_1'), (Y_2, C_2)) = f(t_1(Y_1, C_1), (Y_2 + ku', C_2 + k)) = u'$ for any positive integer k. This implies that given $t_1(v_1) = (Y_1', C_1')$, we can find $v_2 = (Y_2, C_2)$ and $y_2 = (Y_2 + ku', C_2 + k)$ that satisfies the $f(t_i(v_i), v_{-i}) = f(t_i(v_i), y_{-i})$ requirement of Theorem 3.1. Next, we consider whether or not $f(v_i, v_{-i}) \neq f(v_i, y_{-i})$ requirement is satisfied. Without loss of generality and assuming $k = 1$, $f(v_1, v_2) = f(v_1, y_2)$ iff $(Y_1 + Y_2)/(C_1 + C_2) = (Y_1 + Y_2 + u')/(C_1 + (C_2 + 1))$. This implies that $f(v_1, v_2) = f(v_1, y_2)$ iff $u' = (Y_1 + Y_2)/(C_1 + C_2) = u$. This contradicts the initial assumption that $u' \neq u$. Therefore, we can conclude that $f(v_1, v_2) \neq f(v_1, y_2)$.

Although, the above proof assumes Y_i is a real number, the proof could be directly applied if $Y_i \in \mathcal{R}^{p \times p}$.

3.1 Collusion Regarding the NCC Model

When evaluating certain privacy-preserving protocols in practice, we need to consider the case where an adversary may control a subset of the parties involved in the protocol. Such an adversary may force the parties it controls to submit wrong inputs. In order to analyze functionalities that are incentive compatible when collusion is possible, the current DNCC model needs to be extended to include the possibility of collusion. In other words, we need to understand that given f is in DNCC, whether or not f is still in DNCC when collusion occurs. To analyze this, we continue to follow the *correctness* and *exclusiveness* assumptions under the NCC model. Based on the assumptions, we

next define the DNCC functions for the case where an adversary controls at most t fixed parties.

Definition 3.1. *Suppose $t < n$, f is (n,t)-deterministically noncooperatively computable (or (n,t)-DNCC) if the following holds: For any set $S \subset \{1,\ldots,n\}$ (where $|S| \leq t$), every strategy (t_S, g_S), and every $v_S = (v_{i_1}, v_{i_2}, \ldots, v_{i_{|S|}})$ (where $i_j \in S$), it is the case that*

- *Either*

$$\exists v_{-S} \in D_{-S}, g_S(f(t_S(v_S), v_{-S}), v_S) \neq f(v_S, v_{-S}).$$

- *Or $\forall v_{-S} \in D_{-S}$, $f(t_S(v_S), v_{-S}) = f(v_S, v_{-S})$.*

Intuitively, the above definition indicates that any adversary that controls at most t parties is not able to *exclusively* learn the correct function result. Clearly, if we can prove that a function is $(n, n-1)$-DNCC, then this implies that an adversary that controls at most $n-1$ parties is not able to exclusively learn the correct result by modifying the inputs.

For many distributed data analysis tasks, we need to compute functions that have a special structure. For example, assuming that data sets are horizontally partitioned, any distributed data mining function $f(d_1,\ldots,d_n)$ defined over n databases $d_1,\ldots,d_n$ could be rewritten as $f(d_i, w(d_{-i}))$, where $w(d_{-i})$ is an inputs combining function, e.g., union, intersection, max or min. In other words, w determines how these inputs are used in f. In general, for any function $f(v_1,\ldots,v_n)$ which can be rewritten as $f(v_i, w(v_{-i}))$ for any i and some function w, we can show that f is $(n, n-1)$-DNCC if and only if f is $(2,1)$-DNCC.

Theorem 3.2. *If $f(v_1, v_2, \ldots, v_n) = f(v_i, w(v_{-i}))$ for any i, any v_i and some function $w : D^{n-1} \mapsto D$, then f is $(n, n-1)$-DNCC if and only if f is $(2,1)$-DNCC.*

Proof. If f is $(n, n-1)$-DNCC for any n, f is $(2,1)$-DNCC (by setting $n = 2$). Next, we need to show that if f is not $(n, n-1)$-DNCC for some n (say $n = 3$), then f is not $(2,1)$-DNCC. Suppose f is not $(n, n-1)$-DNCC. Then, according to Definition 3.1, $\exists S \subset \{1,\ldots,n\}$, $\exists (t_S, g_S)$ and $\exists v_S$ such that the following holds simultaneously:

- $\forall v_{-S}$: $g_S(f(t_S(v_S), v_{-S}), v_S) = f(v_S, v_{-S})$.
- $\exists v_{-S}$: $f(t_S(v_S), v_{-S}) \neq f(v_S, v_{-S})$.

Using such t_S, g_S, we can define a cheating strategy for $(2,1)$-DNCC case. Set $v_i = w(v_S)$, and define $t_i(v_i)$ for the two party case as follows: Since v_i is equal to $w(v_S)$, set $t_i(v_i) = w(t_S(v_S))$ and $g_i(f(t_i(v_i), v_{-i}), v_i) = g_S(f(t_i(v_i), v_{-i}), v_S)$. This strategy works if t_S, g_S exist because (without loss of generality, assume $|S| = n-1$):

$$\begin{aligned} g_S(f(t_i(v_i), v_{-i}), v_S) &= g_S(f(w(t_S(v_S)), v_{-i}), v_S) \\ &= g_S(f(t_S(v_S), v_{-i}), v_S) \\ &= f(v_S, v_{-i}) \\ &= f(w(v_S), v_{-i}) \\ &= f(v_i, v_{-i}). \end{aligned}$$

Note that in the above case $v_{-S} = v_{-i}$. Also we know that $f(t_S(v_S), v_{-S}) \neq f(v_S, v_{-S})$ for some v_{-S}. This implies that for some v_{-i} and $v_i = w(v_S)$, $f(w(t_S(v_S)), v_{-i}) \neq f(v_i, v_{-i})$. □

The effects of an adversary that controls multiple parties have been studied in SMC domain extensively. The general results indicate that any function could be evaluated privately (i.e., nothing other than the function result is revealed) if an adversary is computationally bounded and does not control the majority of the parties (i.e., an adversary controls at most $\lfloor \frac{n-1}{2} \rfloor$ [12]). This result is still valid if the adversary is rational [14]. Using the ideas from [14], we can easily show that every function in DNCC has a $\lfloor \frac{n-1}{2} \rfloor$ private evaluation without requiring a trusted third party.

4 Analyzing Data Analysis Tasks under the DNCC Model

So far, we have developed techniques to prove whether or not a function is in DNCC. Combining the two concepts DNCC and SMC, we can analyze privacy-preserving data analysis tasks (without utilizing a TTP) that are incentive compatible. We next prove several such important tasks that either satisfy or do not satisfy the DNCC model. Also, note that the data analysis tasks analyzed next have practical SMC-implementations.

4.1 Function with Boolean Output

From SMC literature, we know that there are few functions that can be evaluated if the adversary controls $n-1$ parties. Here, we prove that functions with Boolean outputs that are $n-1$ private are not in DNCC.

Theorem 4.1 [6]. *A function from $f : D_1 \times D_2 \times \cdots \times D_n \mapsto \{0,1\}$ is $n-1$-private if there exits a protocol f so that no coalition of size $\leq n-1$ can infer any additional information from the execution, other than the function result. Further more, f is $n-1$ private if and only if it can be represented as*

$$f(v_1, v_2, \ldots, v_n) = f_1(v_1) \oplus f_2(v_2) \oplus \cdots \oplus f_n(v_n),$$

where f_is are arbitrary functions with Boolean outputs and $\oplus$ is the binary XOR operation.

Theorem 4.2. *There does not exit any nonconstant $n-1$ private DNCC function with Boolean output.*

Proof. According to Theorem 4.1, we know that any $n-1$ private function is of the form: $f(v_1, v_2, \ldots, v_n) = f_1(v_1) \oplus f_2(v_2) \oplus \cdots \oplus f_n(v_n)$. Clearly, for any t_i, we can define the g_i as

$$g_i(f(t_i(v_i), v_{-i}), v_i) = f((t_i(v_i), v_{-i})) \oplus f_i(t_i(v_i)) \oplus f_i(v_i).$$

Note that g_i function will always give the correct result for all possible v_{-i} because

$$\begin{aligned} g_i(f(t_i(v_i), v_{-i}), v_i) &= f((t_i(v_i), v_{-i})) \oplus f_i(t_i(v_i)) \oplus f_i(v_i) \\ &= (\oplus_{j \neq i} f_j(v_j)) \oplus f_i(t_i(v_i)) \\ &\quad \oplus f_i(t_i(v_i)) \oplus f_i(v_i) \\ &= (\oplus_{j \neq i} f_j(v_j)) \oplus f_i(v_i) \\ &= f(v_i, v_{-i}). \end{aligned}$$

To complete the proof that f is not in DNCC, we also need to show that there exists v_{-i} such that $f(t_i(v_i), v_{-i}) \neq f(v_i, v_{-i})$. Clearly, if f is a nonconstant function, there exits v_i, v_i' for some i and for some v_{-i} such that $f(v_i, v_{-i}) \neq f(v_i', v_{-i})$. Then, we can define $t_i(v_i) = v_i'$. □

4.2 Set Operations

Set operations are commonly used in privacy-preserving data analysis protocols. (See [26], [41] for examples). Here, we show that common set operations like intersection and union are not in DNCC. Let us assume that each party i has a set S_i, a subset of publicly known universal set U. For example, assuming that each party is a financial institution, S_i could be a set of customers' social security numbers of party i and U could be the set of all nine digit numbers. As before, let S_{-i} denote $(S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_n)$.

Theorem 4.3. *Let $f(S_1, \ldots, S_n) = S_1 \cup \cdots \cup S_n$. Then, f is not in $(n, 1)$-DNCC.*

Proof. In order to prove that f is not in $(n, 1)$-DNCC, we need to provide a correct (t_i, g_i) pair that works on S_i and S_{-i}. We also need to prove that t_i prevents the correct function evaluation for some S_{-i} (i.e., $\exists S_{-i}$ such that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$). We can define $t_i(S_i) = S_i \setminus S'$ where S' is any nonempty subset of S_i. The corresponding g_i could be defined as

$$\begin{aligned} g_i(f(t_i(S_i), S_{-i}), S_i) &= f(t_i(S_i), S_{-i}) \cup S' \\ &= (\cup_{j \neq i} S_j) \cup (S_i \setminus S') \cup S' \\ &= S_1 \cup \cdots \cup S_n = f(S_i, S_{-i}). \end{aligned}$$

The above g_i works because $S' \subseteq S_i$ implies $S_i = (S_i \setminus S') \cup S'$. To conclude, we need to show there exists S_{-i} such that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$. Note that for any S' and S_{-i} where $S' \cap (\cup_{j \neq i} S_j) = \emptyset$, we know that

$$\begin{aligned} f(t_i(S_i), S_{-i}) &= (\cup_{j \neq i} S_j) \cup (S_i \setminus S') \\ &= f(S_i, S_{-i}) \setminus S'. \end{aligned}$$

This implies that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$. □

Theorem 4.4. *Let $f(S_1, \ldots, S_n) = S_1 \cap \cdots \cap S_n$. Then, f is not in $(n, 1)$-DNCC.*

Proof. We need to define (t_i, g_i) that works for a chosen S_i and show there exists S_{-i} for any S_i such that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$. Let $S' \subseteq U$ and $S_i \cap S' = \emptyset$. Define t_i as $t_i(S_i) = S_i \cup S'$. Then, define g_i as

$$\begin{aligned} g_i(f(t_i(S_i), S_{-i}), S_i) &= f(t_i(S_i), S_{-i}) \setminus S' \\ &= (\cap_{j \neq i} S_j) \cap (S_i \cup S') \setminus S' \\ &= S_1 \cap \cdots \cap S_n \\ &= f(S_i, S_{-i}). \end{aligned}$$

The g_i works correctly because $S_i \cap S' = \emptyset$ and $(S_i \cup S') \setminus S' = S_i$. Also, we need to show that for any S_i, there exists S_{-i} such that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$. Let us assume $S' \cap (\cap_{j \neq i} S_j) \neq \emptyset$. In that case,

$$\begin{aligned} f(t_i(S_i), S_{-i}) &= (\cap_{j \neq i} S_j) \cap (S_i \cup S') \\ &= f(S_i, S_{-i}) \cup ((\cap_{j \neq i} S_j) \cap S'). \end{aligned}$$

Since $S' \cap (\cap_{j \neq i} S_j) \neq \emptyset$, we can conclude that $f(t_i(S_i), S_{-i}) \neq f(S_i, S_{-i})$. □

4.3 Multivariate Statistics

In many distributed data mining tasks, we may need to learn the mean and the covariance of the underlying data set. For example, to build a mixture of Gaussian model for classification, we may want to learn the mean and the covariance matrix of each class [10]. Let us assume that $X_1, X_2, \ldots, X_N$ are identically and independently distributed $1 \times p$ row vectors (or a tuple in a data set) where $E(X_k) = \mu$ and $\Sigma = E((X_k - \mu)^T (X_k - \mu))$. Let X_k^v denotes the vth column of the row vector X_k. We can estimate μ and Σ as $\bar{\mu}$ and $\bar{\Sigma}$:

$$\bar{\mu} = \frac{1}{N} \sum_{k=1}^{N} X_k$$

$$\bar{\Sigma} = \frac{1}{N} \sum_{k=1}^{N} (X_k - \bar{\mu})^T (X_k - \bar{\mu}).$$

In our analysis, we consider two different data partition cases: the horizontally partitioned data and vertically partitioned data. In the case of horizontally partitioned data, each party i owns a set S_i where $S_i \cap S_j = \emptyset$ for any i and j, and $\cup_{i=1}^{n} S_i = \{X_1, \ldots, X_N\}$. In the case of vertically partitioned data, for all k, each party i owns all $X_k^{v_1}, \ldots, X_k^{v_i}$, where $1 \leq k \leq N$ and $\{v_1, \ldots, v_i\} \subset \{1, \ldots, p\}$ (p indicates the number of attributes or columns). Next, we discuss whether $\bar{\mu}$ and $\bar{\Sigma}$ could be evaluated in the DNCC model on horizontally partitioned and vertically partitioned data.

4.3.1 Horizontally Partitioned Data

For the horizontally partitioned case, we will show that if the total number of vectors are private (i.e., N is private), then the functions computing $\bar{\mu}$ and $\bar{\Sigma}$ are in $(2, 1)$-DNCC. If N is a public information, then the functions computing $\bar{\mu}$ and $\bar{\Sigma}$ are not in $(n, 1)$-DNCC. Consider Example 1.1, in this example, the total number of vectors is equal to the total number of parties. Since the total number of parties is a public information, the functions computing the mean and the variance are not in DNCC.

To prove that the functions computing $\bar{\mu}$ and $\bar{\Sigma}$ are not in DNCC when N is public, we first prove that any linear function is not in DNCC. Then, we will show functions computing $\bar{\mu}$ and $\bar{\Sigma}$ are linear functions if N is public.

Theorem 4.5. *Any linear function $f(v_1, v_2, \ldots, v_n) = f_1(v_1) + f_2(v_2) + \cdots + f_n(v_n)$ is not in $(n, 1)$-DNCC.*

Proof. For any t_i, we can define g_i as follows:

$$g_i(f(t_i(v_i), v_{-i}), v_i) = f(t_i(v_i), v_{-i}) - f_i(t_i(v_i)) + f_i(v_i).$$

Note that g_i function will always give the correct result for all possible v_{-i} because

$$\begin{aligned} g_i(f(t_i(v_i), v_{-i}), v_i) &= f(t_i(v_i), v_{-i}) - f_i(t_i(v_i)) + f_i(v_i) \\ &= \sum_{j \neq i} f_j(v_j) + f_i(t_i(v_i)) \\ &\quad - f_i(t_i(v_i)) + f_i(v_i) \\ &= f(v_i, v_{-i}). \end{aligned}$$

To conclude the proof, we need to show that there exists v_{-i} such that $f(t_i(v_i), v_{-i}) \neq f(v_i, v_{-i})$. Clearly, if f is a nonconstant function, there exits v_i, v_i' for some i and for some v_{-i} such that $f(v_i, v_{-i}) \neq f(v_i', v_{-i})$. Then, we can define $t_i(v_i) = v_i'$. □

Using Theorem 4.5, we can easily prove that functions computing $\bar{\mu}$ and $\bar{\Sigma}$ are not in DNCC.

Theorem 4.6. *If N is public, then the functions computing $\bar{\mu}$ and $\bar{\Sigma}$ on the horizontally partitioned data are not in $(n,1)$-DNCC.*

Proof. Each party i can locally compute $Y_i = \frac{1}{N}\sum_{X_j \in S_i} X_j$. It is easy to see that $\bar{\mu}$ is a linear function of Y_i values (i.e., $\bar{\mu} = \sum_{i=1}^{n} Y_i$). Therefore, function computing $\bar{\mu}$ is not in $(n,1)$-DNCC due to Theorem 4.5.

Similar argument is valid for the function computing $\bar{\Sigma}$. Here, each party i locally compute $Y_i = \frac{1}{N}\sum_{X_j \in S_i} (X_j - \bar{\mu})^T (X_j - \bar{\mu})$, and $\bar{\Sigma}$ is a linear function of Y_is (i.e., $\bar{\Sigma} = \sum_{i=1}^{n} Y_i$). Thus, function computing $\bar{\Sigma}$ is not in $(n,1)$-DNCC due to Theorem 4.5. □

If we assume that $|S_i|$ values are private (i.e., the total number of vectors denoted by N is a private information), then we can prove functions computing $\bar{\mu}$ and $\bar{\Sigma}$ on horizontally partitioned data are in $(n, n-1)$-DNCC. To prove the above statement, we first prove that $f : (\mathcal{R}, \mathcal{Z}^+)^n \mapsto R$ defined below is in $(n, n-1)$-DNCC

$$f((Y_1, C_1), \ldots, (Y_n, C_n)) = \frac{\sum_i^n Y_i}{\sum_i^n C_i}.$$

Note that if we set $Y_i = \sum_{X_j \in S_i} X_j$ and $C_i = |S_i|$, then $\bar{\mu} = f((Y_1, C_1), \ldots, (Y_n, C_n))$. Likely, if we set

$$Y_i = \sum_{X_j \in S_i} (X_j - \bar{\mu})^T (X_j - \bar{\mu}),$$

then we can compute $\bar{\Sigma} = f((Y_1, C_1), \ldots, (Y_n, C_n))$.

To prove that $f((Y_1, C_1), \ldots, (Y_n, C_n))$ is in $(n, n-1)$-DNCC, we use Theorem 3.2. Let $v_i = (Y_i, C_i)$. First note that f satisfies the requirements of Theorem 3.2, due to the fact that $f(v_i, v_{-i}) = f(v_i, w(v_{-i}))$ for $w(v_{-i}) = (\sum_{j \neq i} Y_j, \sum_{j \neq i} C_j)$. Therefore, showing that f is in $(2,1)$-DNCC will automatically imply that f is in $(n, n-1)$-DNCC. Please note that for two party case, f becomes $f((Y_1, C_1), (Y_2, C_2)) = (Y_1 + Y_2)/(C_1 + C_2)$ and using the proof given in Example 3.2, we can conclude that f is in $(2,1)$-DNCC.

4.3.2 Vertically Partitioned Data

In the case of vertically partitioned data, each party i owns all $X_k^{v_1}, \ldots, X_k^{v_i}$, where $1 \leq k \leq N$ and $\{v_1, \ldots, v_i\} \subset \{1, \ldots, p\}$. Note that the vth column of the $\bar{\mu}$ can be calculated by the party who owns all X_k^v values, for $1 \leq k \leq N$ (i.e., $\bar{\mu}^v = \frac{1}{N}\sum_{k=1}^{N} X_k^v$. Thus, in order to estimate $\bar{\mu}$, each party could calculate the $\bar{\mu}^v$ for v values it owns and announce the results. Clearly, each party can lie about the $\bar{\mu}^v$ value as in the example given in Section 1. Therefore, for the vertically partitioned data case, the function computing $\bar{\mu}$ is not in $(n,1)$-DNCC.

Also, we can show that computing $\bar{\Sigma}$ is not in $(n,1)$-DNCC. First note that $\bar{\Sigma}_{uv}$ could be computed as

$$\bar{\Sigma}_{uv} = \frac{1}{N}\sum_{k=1}^{N} (X_k^u - \bar{\mu}^u)(X_k^v - \bar{\mu}^v)$$

If party i knows all the X_k^u and X_k^v values, it can calculate the $\bar{\Sigma}_{uv}$ locally. If party i knows all the X_k^u values and party j knows all the X_k^v values, they may jointly compute $\bar{\Sigma}_{uv}$. Unfortunately, even in that case, the function that computes $\bar{\Sigma}_{uv}$ is not in $(2,1)$-DNCC.

Since computing $\bar{\Sigma}_{uv}$ can be seen as a dot product of two vectors, we can show that computing $\bar{\Sigma}_{uv}$ is not in $(2,1)$-DNCC by showing that computing the dot product of vectors of real numbers are not in $(2,1)$-DNCC. As before, we show that computing the dot product of vectors of real numbers are not in $(2,1)$-DNCC by specifying t_i and g_i functions.

Theorem 4.7. *Let $f(Y_1, Y_2) = \sum_{k=1}^{N} (Y_1^k \cdot Y_2^k)$ where Y_i is a $1 \times p$ row vector with real number entries and Y_i^k is the k^{th} column of Y_i. Then, f is not in $(2,1)$-DNCC.*

Proof. Without loss of generality, assume $t_1(Y_i) = \alpha \cdot Y_1$ where α is a nonzero scalar number. Since $f(t_1(Y_1), Y_2) = \alpha \cdot f(Y_1, Y_2)$, we can define g_1 as

$$g_i(f(t_1(Y_1), Y_2), Y_1) = f(t_1(Y_1), Y_2)/\alpha.$$

Clearly, (t_i, g_i) works for any nonzero scalar α. Also, $f(Y_1, Y_2) \neq f(t_1(Y_1), Y_2)$ for any $f(Y_1, Y_2) \neq 0$. □

4.4 Privacy-Preserving Association Rule Mining

In this section, we first summarize the association rule mining and analyze whether the association rule mining can be done in an incentive compatible manner over horizontally and vertically partitioned databases.

4.4.1 Overview of Association Rule Mining

The association rules mining problem can be defined as follows [3]. Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of items. Let DB be a set of transactions, where each transaction T is an item set such that $T \subseteq I$. Given an item set $X \subseteq I$, a transaction T *contains* X if and only if $X \subseteq T$. An association rule is an implication of the form $X \Rightarrow Y$ where $X \subseteq I, Y \subseteq I$ and $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ has *support* s in the transaction database DB if s percent of transactions in DB contain $X \cup Y$. The association rule holds in the transaction database DB with *confidence* c if c percent of transactions in DB that contain X also contain Y. An item set X with k items called k-item set. The problem of mining association rules is to find all rules whose support and confidence are higher than certain user specified minimum support and confidence.

In this simplified definition of the association rules that we use in this paper, missing items and negative quantities are not considered. In this respect, transaction database DB can be seen as $0/1$ matrix where each column is an item and each row is a transaction.

4.4.2 Horizontally Partitioned Data

The above problem of mining association rules can be extended to distributed environments. Let us assume that a transaction database DB is horizontally partitioned among n parties where $DB = DB_1 \cup DB_2 \cup \cdots \cup DB_n$, and DB_i resides at party i's site ($1 \leq i \leq n$). The item set X has *local* support count of $X.sup_i$ at party i if $X.sup_i$ of the

transactions contain X. The *global* support count of X is given as $X.sup = \sum_{i=1}^{n} X.sup_i$. An item set X is *globally supported* if $X.sup \geq s \cdot \sum_{i=1}^{n} |DB_i|$. Global confidence of a rule $X \Rightarrow Y$ can be given as $\{X \cup Y\}.sup/X.sup$.

The set of large item sets $L_{(k)}$ consists of all k-item sets that are globally supported. The aim of distributed association rule mining is to find the sets $L_{(k)}$ for all $k > 1$ and the support counts for these item sets, and from this, association rules with the specified minimum support and confidence can be computed.

In [26], authors discuss how to convert a fast distributed association rule mining algorithm to a privacy-preserving association rule mining algorithm. Although, for efficiency purposes, authors use secure set union to calculate the union of locally large item sets, this calculation reveals some information and may not be desirable for some applications [26]. As discussed in [26], to enable a more strictly privacy-preserving version, it is enough to securely check whether a candidate large item set is globally supported. Thus, if the protocol is strictly secure (e.g., without leaking additional information like the secure set union protocol) under the SMC definitions, it is sufficient to show checking if a candidate item set is globally supported is in DNCC.

First note that, to check if a candidate item set X is supported globally, all we need to know is whether $X.sup \geq s \cdot |DB|$. The following allows us to reduce this to a comparison against the sum of local values (the *excess support* at each party)

$$X.sup \geq s * |DB| = s \cdot \sum_{i=1}^{n} |DB_i|$$

$$\sum_{i=1}^{n} (X.sup_i - s \cdot |DB_i|) \geq 0.$$

Let Y_i be the local excess support at party i (i.e., $Y_i = (X.sup_i - s \cdot |DB_i|)$). We next show the predicate $f(Y_1, \ldots, Y_n)$: $(\sum_i^n (Y_i) \geq 0)$ is in $(n, n-1)$-DNCC.

Theorem 4.8. *Given a candidate item set X, checking $X.sup \geq s \cdot |DB|$ is in $(n, n-1)$-DNCC.*

Proof. As noted above checking $X.sup \geq s \cdot |DB|$ is equivalent to evaluate the predicate $f(Y_1, \ldots, Y_n)$: $(\sum_i^n (Y_i) \geq 0)$ for $Y_i = (X.sup_i - s \cdot |DB_i|)$. Note that for $w(Y_{-i}) = \sum_{j \neq i} Y_j$, we get $f(Y_i, Y_{-i}) = f(Y_i, w(Y_{-i}))$. Based on Theorem 3.2, to show f is $(n, n-1)$-DNCC, it is sufficient to show that f is in $(2, 1)$-DNCC. For the two party case, we can define the predicate as $f(Y_1, Y_2)$: $(Y_1 \geq -Y_2)$. As a result, we merely need to show that the comparison function is in $(2, 1)$-DNCC for arbitrary real number inputs.

If we can show that the conditions stated in Theorem 3.1 hold, we can conclude f is in $(2, 1)$-DNCC for any t_i. Suppose $v_1 = Y_1$ ($v_2 = -Y_2$) is the true input of the first (second) user. We define $t_1(v_1)$ as the modified input of party 1 and let $t_1(v_1) - v_1 = r$, for some value r. When r is 0, we say t_i is the identity function, and for the identity function, the condition "Or $\forall v_{-i} \in D_{-i}$, $f(t_i(v_i), v_{-i}) = f(v_i, v_{-i})$" holds automatically. For the rest of the proof, we consider the situation where $r \neq 0$. Based on the r value, we consider two cases:

1. Assume $r > 0$, consider the case where $v_1 + r > v_2 > v_1 > y_2$. Note that $f(t_1(v_1), v_2) = f(v_1 + r, v_2) = f(v_1 + r, y_2)$, but $f(v_1, v_2) \neq f(v_1, y_2)$. Also, $f(t_1(v_1), v_2) \neq f(v_1, v_2)$. This implies that the first condition of Theorem 3.1 is satisfied.
2. Assume $r < 0$, consider the case where $y_2 > v_1 > v_2 > v_1 + r$. Note that $f(t_1(v_1), v_2) = f(v_1 + r, v_2) = f(v_1 + r, y_2)$, but $f(v_1, v_2) \neq f(v_1, y_2)$. Also, $f(t_1(v_1), v_2) \neq f(v_1, v_2)$. This implies that the first condition of Theorem 3.1 is satisfied.

As a result, we conclude that f is in $(2, 1)$-DNCC. □

Again as shown in [26], we can use the f described above to check if the confidence threshold is satisfied for any candidate rule of the form $X \Rightarrow Y$. One interesting consequence of the above result is that composition of the DNCC functions should be carefully checked to see whether or not the composition is in DNCC. Note that even though $g(Y_1, \ldots, Y_n)$: $\sum_i^n (Y_i)$ is **not** in DNCC, $f(Y_1, \ldots, Y_n)$: $(\sum_i^n (Y_i) \geq 0)$ is in DNCC. (Refer to Claim 5.1 in Section 5 for more details.)

4.4.3 Vertically Partitioned Data

Given the transaction database DB as $0/1$ matrix, where each column is an item and each row is a transaction, the DB is considered vertically partitioned if different parties know different columns of the DB. To mine association rules over vertically partitioned data, it has been shown that you need to calculate a dot product with $0/1$ vectors, where each vector represents whether a certain set of items are present in a transaction or not [41]. Therefore, if we can show that functions calculating dot product with binary vectors is in DNCC, then using the results from [41], we can conclude that calculating an support count of an item set is also in DNCC.

Below we show that calculating the dot product of nonzero binary vectors is in $(2, 1)$-DNCC. (In this context, we call a vector, nonzero vector if at least one of the entries is nonzero.) Before we proceed with the proof, we stress that the following theorem is not a contradiction with our earlier result that shows that the function computing dot product of real-valued vectors is not in DNCC. Note that t_i described for the dot product of real valued vectors will no longer work in the context of binary vectors since you can only multiply each entry with zero or one. Another important detail to note is we assume that the binary vectors are nonzero vectors. This assumption is important because if we allow zero vectors, the dot product result could be exactly determined even without any computation by the owner of the zero vector. Thus, the owner of the zero vector could lie about its input easily.

We believe that the assumption of nonzero binary vectors is realistic because with fairly large databases, it is highly likely that there exists at least one transaction that supports the required item.

Theorem 4.9. *Let f be denoted as $f(v_1, v_2) = \sum_{j=1}^{k} v_1^j \cdot v_2^j$, where v_i is a nonzero binary vector with k rows and v_i^j is the value of the jth row of vector v_i, then f is in $(2, 1)$-DNCC.*

Proof. Let U be the set of indexes from $\{1, \ldots, k\}$. Let S_1 be any subset of U such that $\forall j \in S_1, v_1^j = 1$. Also, we know

that $|S_1| > 0$ since v_1 is a nonzero vector. Without loss of generality, we assume that t_1 modifies some subset of v_1's rows. (Note that any t_1 that transforms v_1 could be represented as the set of indexes that are negated.) Let C be the set of indexes negated by t_1. Suppose $C \neq \emptyset$, and because $C = \emptyset$ implies $t_1(v_1) = v_1$, we can rewrite $f(v_1, v_2)$ as

$$f(v_1, v_2) = \sum_{j=1}^{k} \left(v_1^j \cdot v_2^j\right) = \sum_{j \in S_1} v_2^j.$$

All v_1^js, where $j \notin S_1$, are equal to zero, and all v_1^js, where $j \in S_1$, are equal to one. Similarly, given t_1 and the associated set C, we can represent $f(t_1(v_1), v_2)$ as

$$f(t_1(v_1), v_2) = \sum_{j \in S_1 \setminus C} v_2^j + \sum_{j \in C \setminus S_1} v_2^j.$$

Note that all v_1^j values, where $j \in S_1 \cap C$, are converted to zero by t_1. Also, the v_1^j values, where $j \in C \setminus S_1$, that are zero are converted to one by t_1. We next show that for any t_1, we can find y_2 and v_2 such that $f(t_1(v_1), v_2) = f(t_1(v_1), y_2)$ and $f(v_1, v_2) \neq f(v_1, y_2)$. In addition, if $f(t_1(v_1), v_2) \neq f(v_1, v_2)$, then the first condition of Theorem 3.1 would be satisfied. Otherwise, it means that $f(t_1(v_1), v_2) = f(v_1, v_2)$ for all v_2. To prove the existence of such y_2 and v_2 for any t_1, we consider two different cases:

Case 1: $S_1 \cap C = \emptyset$. Implies that some zero values in v_1 are converted to one by t_1

$$f(t_1(v_1), v_2) = \sum_{j \in S_1} v_2^j + \sum_{j \in C} v_2^j.$$

Now, consider y_2 and v_2 such that $\sum_{j \in S_1}(v_2^j - y_2^j) = 1$ (this is possible because $S_1 \neq \emptyset$) and $\sum_{j \in C}(v_2^j - y_2^j) = -1$. It is easy to see that

$$\begin{aligned} f(t_1(v_1), v_2)) &= \sum_{j \in S_1} v_2^j + \sum_{j \in C} v_2^j \\ &= \sum_{j \in S_1} \left(y_2^j\right) + 1 + \sum_{j \in C} \left(y_2^j\right) - 1 \\ &= f(t_1(v_1), y_2). \end{aligned}$$

Clearly, $f(v_1, v_2) \neq f(v_1, y_2)$ since $\sum_{j \in S_1}(v_2^j - y_2^j) = 1$.

Case 2: $S_1 \cap C \neq \emptyset$. Now, consider any y_2 and v_2 such that $y_2^j = v_2^j$ for $j \notin S \cap C$, and $v_2^j = 1$ and $y_2^j = 0$ for $j \in S \cap C$. Note that $f(t_1(v_1), v_2) = f(t_1(v_1), y_2)$ because v_2^j and y_2^j are exactly the same except $j \in S \cap C$. Also, note that $f(v_1, v_2) \neq f(v_1, y_2)$

$$\begin{aligned} &\sum_{j \in S_1 \cap C} v_2^j \neq \sum_{j \in S_1 \cap C} y_2^j \\ &\sum_{j \in S_1 \setminus C} v_2^j + \sum_{j \in S_1 \cap C} v_2^j \neq \sum_{j \in S_1 \setminus C} v_2^j + \sum_{j \in S_1 \cap C} y_2^j \\ &f(v_1, v_2) \neq f(v_1, y_2). \end{aligned}$$

Therefore, we can conclude that f is in $(2, 1)$-DNCC. □

4.5 Privacy-Preserving Naive Bayes Classification

In naive Bayes classification [15], building the data mining model involves determining the probability that an instance is of a certain class given that it has certain values for its other attributes. To classify an unlabeled instance, we simply compute our estimated likelihood for each class as follows: $Pr(C = c) \prod_{i=1}^{k} Pr(A_i = v_i | C = c)$ for class value c and attribute values $v_1, \ldots, v_k$. This implies that to build privacy-preserving Naive Bayes classification models, we need to estimate the probability $Pr(A_i = v_i | C = c)$ that attribute A_i has value v_i given that class value is c. Below we show that for both vertically and horizontally partitioned data, computing such probabilities is in DNCC.

4.5.1 Horizontally Partitioned Data

To compute $Pr(A_i = v_i | C = c)$ in the horizontally partitioned case, in [42], it is shown that each party needs to compute a function of the form

$$f((Y_1, C_1), \ldots, (Y_n, C_n)) = \frac{\sum_i^n Y_i}{\sum_i^n C_i}$$

where (Y_i, C_i) belongs to party i. As shown in Theorem 4.6, this function is $(n, n-1)$-DNCC. Therefore, learning Naive Bayes models in horizontally partitioned data case is in $(n, n-1)$-DNCC.

4.5.2 Vertically Partitioned Data

Suppose one party has the class attribute of the data, and another party has some other attribute A_i. To build the model, we would need to compute $Pr(A_i = v_i | C = c)$ for each value v_i and class value c. One way of doing this is, for each value v_i, to create a vector of length n (where n is the number of instances), whose jth location is one if jth instance's A_i attribute has value v_i, and zero otherwise. Similarly, we can create vectors for each class value c where jth location is 1 if jth instance's class value is c, and zero otherwise. Then, for each pair (v_i, c) for attribute A_i, we compute the dot product of the corresponding two vectors, which gives us the number of instances where $A_i = v_i$ and $C = c$. This can be divided by the total number of instances where $C = c$ to estimate the probability $Pr(A_i = v_i | C = c)$. Note that the dot products involved in this case are computed over binary vectors. Therefore, we can use Theorem 4.9 to prove that learning Naive Bayes models in vertically partitioned case is in (2,1)-DNCC.

4.6 Privacy-Preserving Decision Tree Classification

The aim of a decision tree is to provide classification criteria based on the attributes of a data set. At each node of a decision tree, the data are "split" into several subsets of data based on the criterion of *information gain*. The information gain of a split is defined as the average difference in entropy between the original data set, and each data set formed by the split.

In the Id3 and C4.5 decision trees [15], the sets for the split are chosen based on a single attribute. The construction of the tree proceeds as follows: 1) Determine the attribute that has the greatest information gain (or equivalently minimum conditional entropy). 2) Use that attribute to split the data set. 3) Repeat this process within each subset until no splits give significant information gain. At this point the tree is complete.

4.6.1 Horizontally Partitioned Data

In [33], it is shown that in the horizontally partitioned data case, after secure computation step, the random shares of

the conditional entropy for an attribute are distributed to each party. Given the random share of party j for conditional entropy of attribute A_i ($S^j_{A_i}$), parties (assuming there are n parties) need to find the attribute A_k for which the $\sum_{j=1}^{n}(S^j_{A_k}) \bmod F$ (for some large prime F) is minimum. Note that such minimum could be computed using series of comparisons. Since we proved that comparison is in (2,1)-DNCC and from Claim 5.1 (given in Section 5), we can conclude that learning decision trees in horizontally partitioned case is in (2,1)-DNCC.

4.6.2 Vertically Partitioned Data

One way to calculate the probabilities used in the conditional entropy for the vertically partitioned data case is to have both parties create a zero-one vector with the length of the data set, where the value at position i is one if the ith instance could be in the partition, and have class C according to the party's data, and zero otherwise. The dot product of these two vectors gives the number of rows which could belong to the given node according to *both* parties' data, and therefore gives the number of rows which belong to that node. We would again compute the dot product, and divide it by the total number of instances in the partition. From Theorem 4.9, learning decision tree models in vertically partitioned case is in (2,1)-DNCC.

5 Using Non-DNCC Functions in an Incentive-Compatible Way

According to our previous analyses, some functions like sum, set union and set intersection are not in DNCC, but they can still be used in an incentive-compatible way in PPDA applications. The reason is that in many applications, primitives like sum, set union and intersection are not used alone, and they often act as subroutines. To have a completely secure protocol, the subroutines can only return random shares of the expected results. For example, suppose a PPDA application uses the set intersection as a subroutine to compute the intersection between two sets D_1 and D_2. To achieve the best security, the subroutine produces two random numbers α_1 and α_2 (from a certain field), such that $\alpha_1 + \alpha_2 = |D_1 \cap D_2|$. Subsequently, α_1 and α_2 will be used as input values to the next subroutine in the PPDA application. Because of this observation, we have the following claim.

Claim 5.1. Suppose a function f is a sequential composition of n subroutines $f_1, \ldots, f_n$, and for $1 \leq i \leq n-1$, f_i returns random shares of its expected results. If f_n is in DNCC, then f is in DNCC.

Note that the random shares produced from f_i are uniformly distributed from the viewpoint of an individual participating party (denoted by P). Therefore, if P modifies his or her input to f_i, it is impossible to derive the actual result from the random shares returned by f_i. In addition, any change to the actual input or the intermediate random shares can change the input to f_n. Thus, if f_n is in DNCC, so is f. Using the above claim, we next show several additional PPDA applications or sequential composite functions are in DNCC.

In information retrieval, the Jaccard coefficient (JC), the Dice coefficient (DC) and the Cosine similarity (CS) are extensively used as similarity metrics to identify relevant information. For illustration purposes, assume D_1 and D_2 are two sets owned by two parties P_1 and P_2, respectively. The metrics are defined as

- $\mathrm{JC}(D_1, D_2) = \frac{|D_1 \cap D_2|}{|D_1 \cup D_2|} = \frac{|D_1 \cap D_2|}{|D_1| + |D_2| - |D_1 \cap D_2|}$.
- $\mathrm{DC}(D_1, D_2) = \frac{2|D_1 \cap D_2|}{|D_1| + |D_2|}$.
- $\mathrm{CS}(D_1, D_2) = \frac{\sum_{i=1}^{m} D_1[i] \times D_2[i]}{\sqrt{\sum_{i=1}^{m} D_1[i]^2} \times \sqrt{\sum_{i=1}^{m} D_2[i]^2}}$.

In [23], a two-party protocol is proposed to securely compute JC. The protocol consists of two stages:

- **Stage 1**—Computing Random Shares of $|D_1 \cap D_2|$. At the end, P_1 has a random number α_1 and P_2 has a random number α_2, such that $\alpha_1 + \alpha_2 = |D_1 \cap D_2|$.
- **Stage 2**—Computing JC Score. P_1 sets $\beta_1 = |D_1| - \alpha_1$ and P_2 sets $\beta_2 = |D_2| - \alpha_2$. Both parties securely compute $\frac{\alpha_1 + \alpha_2}{\beta_1 + \beta_2}$.

Since stage 1 returns random shares and the function $f((\alpha_1, \beta_1), (\alpha_2, \beta_2)) = \frac{\alpha_1 + \alpha_2}{\beta_1 + \beta_2}$ is in DNCC (Example 3.2), the protocol for computing JC is in DNCC from Claim 5.1. Similar protocols can be developed to compute DC and CS using the above stages with minor modifications. Thus, the protocols or the composite functions that compute DC and CS are also in DNCC.

In addition, these metrics are commonly used to measure intracluster and intercluster distances among text documents. Therefore, text clustering techniques using these metrics are in DNCC. Moreover, secure similar document detection [21], [22], [36] is another PPDA application. Because it directly uses CS to measure similarity, this PPDA application is also in DNCC.

6 Conclusion and Future Work

Even though privacy-preserving data analysis techniques guarantee that nothing other than the final result is disclosed, whether or not participating parties provide truthful input data cannot be verified. In this paper, we have investigated what kinds of PPDA tasks are incentive compatible under the NCC model. Based on our findings, there are several important PPDA tasks that are incentive driven. Table 2 classifies the common data analysis tasks studied in this paper into DNCC or Non-DNCC categories. Most often, data partition schemes can make a difference in determining DNCC or Non-DNCC classifications.

For any functions, Theorems 3.1 and 3.2 provide a general way to determine if a function is in DNCC. In addition, Claim 5.1 can be used to analyze if a composite function is in DNCC, and it also provides a method to design a PPDA protocol that guarantees to be incentive compatible under the DNCC definition. For instance, a PPDA task can have many variations, and one common variation is to place a filter at the last step of the task to make the PPDA protocols more secure (e.g., secure similar document detection versus its threshold-based variation). According to Claim 5.1, as long as the last step in a PPDA task is in DNCC, it is always possible to make the entire

TABLE 2
Classification of Common Data Mining Tasks

Data Analysis Tasks	DNCC		Non-DNCC	
	Vertical	Horizontal	Vertical	Horizontal
Comparison		√		
Mean & Variance			√	
Mean & Variance (assuming N is public)				√
Mean & Variance (assuming N is private)		√		
$f(v_1, v_2, \ldots, v_n) = f_1(v_1) \oplus f_2(v_2) \oplus \cdots \oplus f_n(v_n)$				√
$f(S_1, \ldots, S_n) = S_1 \cup \cdots \cup S_n$				√
$f(S_1, \ldots, S_n) = S_1 \cap \cdots \cap S_n$				√
$f(v_1, v_2, \ldots, v_n) = f_1(v_1) + f_2(v_2) + \cdots + f_n(v_n)$				√
$f((Y_1, C_1), \ldots, (Y_n, C_n)) = \frac{\sum_i^n Y_i}{\sum_i^n C_i}$		√		
Dot product of vectors containing real values				√
Dot product of non-zero binary vectors		√		
The predicate $f(Y_1, \ldots, Y_n)$: $(\sum_i^n (Y_i) \geq 0)$		√		
Association rule mining (finding frequent itemsets)	√	√		
Naive Bayes classification	√	√		
Decision tree classification	√	√		
Jaccard coefficient		√		
Dice coefficient		√		
Cosine similarity		√		
Secure similar document detection		√		

PPDA task satisfying the DNCC model. Therefore, when designing a PPDA protocol, it is in our best interests to make the last step of the PPDA task incentive-compatible whenever possible.

As a part of future research direction, we will investigate incentive issues in other data analysis tasks, and extend the proposed theorems under the probabilistic NCC model. Another important direction that we would like to pursue is to create more efficient Secure Multiparty Computation techniques tailored toward implementing the data analysis tasks that are in DNCC.

ACKNOWLEDGMENTS

The first author's contribution to this work was partially supported by Air Force Office of Scientific Research MURI Grant FA9550-08-1-0265, National Institutes of Health Grant 1R01LM009989, US National Science Foundation (NSF) Grant Career-CNS-0845803, and NSF Grants CNS-0964350, CNS-1016343. The second author's contribution to this work was supported by the Office of Naval Research under Award No. N000141110256 and NSF under Award No. CNS-1011984.

REFERENCES

[1] I. Abraham, D. Dolev, R. Gonen, and J. Halpern, "Distributed Computing Meets Game Theory: Robust Mechanisms for Rational Secret Sharing and Multiparty Computation," *Proc. 25th Ann. ACM Symp. Principles of Distributed Computing,* pp. 53-62, 2006.

[2] R. Agrawal and E. Terzi, "On Honesty in Sovereign Information Sharing," *Proc. Int'l Conf. Advances in Database Technology,* pp. 240-256, 2006.

[3] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. Int'l Conf. Very Large Data Bases (VLDB '94),* pp. 487-499, Sept. 1994.

[4] I. Ashlagi, A. Klinger, and M. Tenneholtz, "K-NCC: Stability Against Group Deviations in Non-Cooperative Computation," *Proc. Third Int'l Conf. Internet and Network Economics,* pp. 564-569, 2007.

[5] M.J. Atallah, M. Bykova, J. Li, and M. Karahan, "Private Collaborative Forecasting and Benchmarking," *Proc. Second ACM Workshop Privacy in the Electronic Soc. (WPES),* Oct. 2004.

[6] B. Chor and E. Kushilevitz, "A Zero-One Law for Boolean Privacy," *Proc. 21st Ann. ACM Symp. Theory of Computing (STOC '89),* pp. 62-72, 1989.

[7] Doe News, www.doe.gov, Feb. 2005.

[8] W. Du and Z. Zhan, "Building Decision Tree Classifier on Private Data," *Proc. IEEE Int'l Conf. Data Mining Workshop Privacy, Security, and Data Mining,* C. Clifton and V. Estivill-Castro, eds., vol. 14, pp. 1-8, Dec. 2002.

[9] "Directive 95/46/EC of the European Parliament and of the Council of 24 Oct. 1995 on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of Such Data," *Official J. European Communities,* vol. 281, pp. 31-50, Oct. 1995.

[10] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.

[11] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game - A Completeness Theorem for Protocols with Honest Majority," *Proc. 19th ACM Symp. the Theory of Computing,* pp. 218-229, 1987.

[12] O. Goldreich, "General Cryptographic Protocols," *The Foundations of Cryptography,* vol. 2, Cambridge Univ. Press, 2004.

[13] S.D. Gordon and J. Katz, "Rational Secret Sharing, Revisited," *Proc. Int'l Conf. Security and Cryptography for Networks,* p. 229, 2006.

[14] J. Halpern and V. Teague, "Rational Secret Sharing and Multiparty Computation: Extended Abstract," *Proc. Ann. ACM Symp. Theory of Computing (STOC '04),* pp. 623-632, 2004.

[15] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2006.

[16] S. Han and W.K. Ng, "Preemptive Measures against Malicious Party in Privacy-Preserving Data Mining," *Proc. SIAM Int'l Conf. Data Mining (SDM),* pp. 375-386, 2008.

[17] "Standard for Privacy of Individually Identifiable Health Information," *Fed. Register,* vol. 67, no. 157, pp. 53181-53273, Aug. 2002.

[18] S. Izmalkov, S. Micali, and M. Lepinski, "Rational Secure Computation and Ideal Mechanism Design," *Proc. 46th Ann. IEEE Symp. Foundations of Computer Science (FOCS '05),* pp. 585-594, 2005.

[19] G. Jagannathan and R.N. Wright, "Privacy-Preserving Distributed *k*-Means Clustering over Arbitrarily Partitioned Data," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 593-599, Aug. 2005.

[20] W. Jiang, C. Clifton, and M. Kantarcoğlu, "Transforming Semi-Honest Protocols to Ensure Accountability," *Data and Knowledge Eng.,* vol. 65, no. 1, pp. 57-74, 2008.

[21] W. Jiang, M. Murugesan, C. Clifton, and L. Si, "Similar Document Detection with Limited Information Disclosure," *Proc. 24th Int'l Conf. Data Eng. (ICDE '08),* Apr. 2008.

[22] W. Jiang and B.K. Samanthula, "A Secure and Distributed Framework to Identify and Share Needed Information," *Proc. IEEE Int'l Conf. Privacy, Security, Risk and Trust (PASSAT '11),* Oct. 2011.

[23] W. Jiang and B.K. Samanthula, "N-Gram Based Secure Similar Document Detection," *Proc. 25th Ann. WG 11.3 Conf. Data and Applications Security and Privacy (DBSec '11),* July 2011.

[24] M. Kantarcioglu and O. Kardes, "Privacy-Preserving Data Mining in the Malicious Model," *Int'l J. Information and Computer Security,* vol. 2, pp. 353-375, Jan. 2009.

[25] M. Kantarcioglu and R. Nix, "Incentive Compatible Distributed Data Mining," *Proc. IEEE Int'l Conf. Soc. Computing/IEEE Int'l Conf. Privacy, Security, Risk and Trust,* pp. 735-742, 2010.

[26] M. Kantarcoğlu and C. Clifton, "Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 9, pp. 1026-1037, Sept. 2004.

[27] H. Kargupta, K. Das, and K. Liu, "A Game Theoretic Approach toward Multi-Party Privacy-Preserving Distributed Data Mining," *Proc. 11th European Conf. Principles and Practice of Knowledge Discovery in Databases,* pp. 523-531, Sept. 2007.

[28] J. Katz, "Bridging Game Theory and Cryptography: Recent Results and Future Directions," *Proc. Fifth Conf. Theory of Cryptography,* p. 251, 2008.

[29] G. Kol and M. Naor, "Cryptography and Game Theory: Designing Protocols for Exchanging Information," *Proc. Conf. Theory of Cryptography,* p. 320, 2008.

[30] R. Layfield, M. Kantarcioglu, and B. Thuraisingham, "Incentive and Trust Issues in Assured Information Sharing," *Proc. Fourth Int'l Conf. Collaborative Computing: Networking, Applications and Worksharing,* p. 113, 2009.

[31] X. Lin, C. Clifton, and M. Zhu, "Privacy Preserving Clustering with Distributed EM Mixture Modeling," *Knowledge and Information Systems,* vol. 8, no. 1, pp. 68-81, July 2005.

[32] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *Proc. Int'l Conf. Advances in Cryptology (CRYPTO '00),* pp. 36-54, Aug. 2000.

[33] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *J. Cryptology,* vol. 15, no. 3, pp. 177-206, 2002.

[34] A. Lysyanskaya and N. Triandopoulos, "Rationality and Adversarial Behavior in Multi-Party Computation," *Proc. Ann. Int'l Conf. Advances in Cryptology,* pp. 180-197, 2006.

[35] R. McGrew, R. Porter, and Y. Shoham, "Towards a General Theory of Non-Cooperative Computation (Extended Abstract)," *Proc. Conf. Theoretical Aspects of Rationality and Knowledge (TARK IX),* 2003.

[36] M. Murugesan, W. Jiang, C. Clifton, L. Si, and J. Vaidya, "Efficient Privacy-Preserving Similar Document Detection," *VLDB J.,* vol. 19, pp. 457-475, Jan. 2010.

[37] M. Naor, B. Pinkas, and R. Sumner, "Privacy Preserving Auctions and Mechanism Design," *Proc. First ACM Conf. Electronic Commerce,* 1999.

[38] N. Nisan and A. Ronen, "Algorithmic Mechanism Design (Extended Abstract)," *Proc. Ann. ACM Symp. Theory of Computing (STOC '99),* pp. 129-140, 1999.

[39] S.J. Ong, D. Parkes, A. Rosen, and S. Vadhan, "Fairness with an Honest Minority and a Rational Majority," *Proc. Sixth Theory of Cryptography Conf. (TCC),* 2009.

[40] Y. Shoham and M. Tennenholtz, "Non-Cooperative Computation: Boolean Functions with Correctness and Exclusivity," *Theoretical Computer Science,* vol. 343, nos. 1/2, pp. 97-113, 2005.

[41] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '02),* pp. 639-644, July 2002.

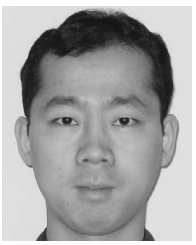
[42] J. Vaidya, M. Kantarcioglu, and C. Clifton, "Privacy-Preserving Naive Bayes Classification," *VLDB J.,* vol. 17, pp. 879-898, July 2008.

[43] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-Art in Privacy Preserving Data Mining," *SIGMOD Record,* vol. 33, no. 1, pp. 50-57, 2004.

[44] A.C. Yao, "Protocols for Secure Computation," *Proc. 23rd IEEE Symp. Foundations of Computer Science,* pp. 160-164, 1982.

[45] A.C. Yao, "How to Generate and Exchange Secrets," *Proc. 27th IEEE Symp. Foundations of Computer Science,* pp. 162-167, 1986.

Murat Kantarcioglu received the BS degree in computer engineering from Middle East Technical University, and the MS and PhD degrees in computer science from Purdue University. He is an associate professor in the Computer Science Department and director of the UTD Data Security and Privacy Lab at the University of Texas at Dallas. He received the US National Science Foundation (NSF) Career award and Purdue CERIAS Diamond award for academic excellence. His research focuses on creating technologies that can efficiently extract useful information from any data without sacrificing privacy or security. He has published more than 75 papers in peer-reviewed journals and conferences. His research has been supported by grants from NSF, AFOSR, ONR, NSA, and NIH. Some of his research work has been covered by the media outlets such as Boston Globe, ABC News, etc., and has received two best paper awards.



Wei Jiang received the bachelor's degrees in both computer science and mathematics from the University of Iowa, Iowa City, Iowa, in 2002 and the master's degree in computer science and the PhD degree from Purdue University, West Lafayette, IN, in 2004 and 2008, respectively. He is an assistant professor in the Department of Computer Science, Missouri University of Science and Technology. His research interests include privacy-preserving data mining, data integration, privacy issues in federated search environments, and text sanitization. His current research is funded by grants from the US National Science Foundation (NSF) and ONR.