Differentially Private Multi-Party High-Dimensional Data Publishing

Sen Su †1, Peng Tang †2, Xiang Cheng †3, Rui Chen ‡, Zequn Wu †4
†State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications, Beijing, China
{susen1, tangpeng2, chengxiang3, leizibupt4}@bupt.edu.cn
‡Samsung Research America
rui.chen1@samsung.com

***Abstract*—In this paper, we study the novel problem of publishing high-dimensional data in a distributed multi-party environment under differential privacy. In particular, with the assistance of a semi-trusted curator, the involved parties (i.e., local data owners) collectively generate a synthetic integrated dataset while satisfying ε-differential privacy for any local dataset. To solve this problem, we present a differentially private sequential update of Bayesian network (*DP-SUBN*) solution. In DP-SUBN, the parties and the curator collaboratively identify the Bayesian network $\mathbb{N}$ that best fits the integrated dataset D in a sequential manner, from which a synthetic dataset can then be generated. The fundamental advantage of adopting the sequential update manner is that the parties can treat the statistical results provided by previous parties as their prior knowledge to direct how to learn $\mathbb{N}$. The core of DP-SUBN is the construction of the search frontier, which can be seen as a priori knowledge to guide the parties to update $\mathbb{N}$. To improve the fitness of $\mathbb{N}$ and reduce the communication cost, we introduce a correlation-aware search frontier construction (*CSFC*) approach, where attribute pairs with strong correlations are used to construct the search frontier. In particular, to privately quantify the correlations of attribute pairs without introducing too much noise, we first propose a non-overlapping covering design (*NOCD*) method, and then introduce a dynamic programming method to find the optimal parameters used in NOCD to ensure that the injected noise is minimum. Through formal privacy analysis, we show that DP-SUBN satisfies ε-differential privacy for any local dataset. Extensive experiments on a real dataset demonstrate that DP-SUBN offers desirable data utility with low communication cost.**

I. Introduction

A. Background and Motivation

In many real-life applications, a mass of high-dimensional data are stored among multiple distributed parties. When integrated, such distributed high-dimensional data are valuable sources for making better decisions and providing high-quality services. For example, it is essential to integrate multiple sources of data for financial institutions to detect and prevent fraud; it is also beneficial for online businesses to collectively analyze their user data for better personalized recommendations. However, since such high-dimensional data may contain highly sensitive personal information, data sharing in the distributed multi-party setting needs to be conducted in a way that no private information is revealed to other participating entities or any other potential adversaries.

In recent years, *privacy-preserving data publishing* [14] has received considerable attention as a promising approach to sharing data while preserving individual privacy. After many attempts to formally define the privacy requirement in data sharing, *differential privacy* [12] has been the most widely accepted privacy model. Unlike the traditional privacy models (e.g., k-anonymity [27] and l-diversity [20]), differential privacy provides strong theoretical guarantees on the privacy of released data. In this paper, we study the problem of publishing high-dimensional data in the distributed multi-party setting while satisfying ε-differential privacy for each local database.

B. Limitation of Prior Art

Some pioneering works [2], [18] have started to explore the distributed multi-party setting. Alhadidi et al. [2] address the problem of private data publishing in the two-party setting. However, their algorithm is specific to classification tasks. Hong et al. [18] propose a solution to multi-party search log publication, which enables different parties to collaboratively generate search logs with boosted utility. However, their solution is not generic to handle other types of data. Moreover, this solution only satisfies the weaker (ε, δ)-differential privacy notion. Thus, it is imperative to develop an approach to multi-party high-dimensional data publication for different data analysis purposes, while satisfying ε-differential privacy.

There have been some existing works [8], [28], [31] in addressing differentially private high-dimensional data publication in the *centralized* setting. A simple attempt is for each distributed party to use the above methods to independently sanitize its data and then share the data for further integration. Unfortunately, this will result in multiple shares of noise in the integrated data, rendering it useless for future data analysis. Therefore, tailored solutions are needed to achieve desirable data utility in our problem.

C. Our Solution

The crux of our problem is the convergence of two independent technical challenges: the *high dimensionality* of the underlying data and the *distributed multi-party setting*. In the centralized setting, the Bayesian network model has been proven effective to release high-dimensional data under differential privacy. For instance, Zhang et al. [31] propose *PrivBayes*, a differentially private method for releasing high-dimensional data. To address our problem, a straightforward attempt is to ask a semi-trusted curator to build the Bayesian network from the private information provided by different parties. However, since the number of candidate attribute-parent (AP) pairs required to construct the Bayesian network is enormous, such an approach suffers from poor data utility and high communication cost.

In this paper, we carefully design solutions to overcome the above difficulties. We note that the correlation between attributes is relatively stable within different local datasets [29]. Motivated by this fact, we propose a differentially private sequential update of Bayesian network (*DP-SUBN*) solution, in which the curator and the parties collaboratively learn the Bayesian network in a sequential manner. The fundamental advantage of adopting such a manner is that the parties can treat the statistical results provided by previous parties as prior knowledge to direct how to learn the Bayesian network.

The key challenge of DP-SUBN is the construction of the search frontier, which can be seen as priori knowledge to guide the parties to update the Bayesian network. It consists of a set of candidate AP pairs and their corresponding marginal distributions, which form the space of all candidate Bayesian networks for the next update. To improve the fitness of the Bayesian network and reduce the communication cost, we should reasonably restrict the size of the search frontier while making it contain as much useful information as possible. To this end, we propose a *correlation-aware search frontier construction* (*CSFC*) approach, where attribute pairs having strong correlations are greedily selected to construct the search frontier. In particular, the key of quantifying the correlations of attribute pairs is to privately obtain their 2-way marginals. To reduce the amount of noise injected into the 2-way marginals, we first propose a *non-overlapping covering design* (*NOCD*) method, which finds larger non-overlapping marginals to reconstruct all 2-way marginals. Then, we propose a dynamic programming method to find the optimal parameters used in NOCD to ensure that the injected noise is minimum.

D. Key Contributions

In addressing the problem of publishing differentially private high-dimensional data in the distributed multi-party setting, we make the following key contributions:

(1) We present a differentially private sequential update of Bayesian network (*DP-SUBN*) solution for multi-party high-dimensional data publication under differential privacy, which can substantially reduce the magnitude of noise and the communication cost. We formally prove that DP-SUBN satisfies ε-differential privacy. The idea of sequential update can also be adopted by other differentially private data analysis tasks in the distributed multi-party setting whenever the statistical results obtained by some parties can be used as prior knowledge.

(2) In DP-SUBN, to improve the fitness of the Bayesian network and reduce the communication cost, we propose a correlation-aware search frontier construction (*CSFC*) approach, where attribute pairs with strong correlations are greedily used to construct the search frontier.

(3) We develop a non-overlapping covering design (*NOCD*) method for generating all 2-way marginals of a given set of attributes. We further propose a dynamic programming method to find the optimal parameters used in NOCD to ensure that the injected noise is minimum. The general idea of NOCD is to find a set of larger *non-overlapping* marginal tables to reconstruct 2-way marginals. We show that NOCD achieves better utility than the state-of-the-art technique [26].

(4) We conduct an extensive experimental study over a real dataset. The experimental results not only demonstrate the effectiveness of NOCD, but also show that DP-SUBN offers desirable data utility with low communication cost.

The rest of our paper is organized as follows. We provide a literature review in Section II. Section III presents necessary background on differential privacy and Bayesian networks. In Section IV, we formulate our problem. In Section V, we present a first-cut solution and point out its limitations. In Section VI, we present our DP-SUBN solution, along with its privacy and communication cost analysis. Comprehensive experimental results are reported in Section VII. Finally, we conclude the paper in Section VIII.

II. Related work

Private Multi-Party Data Publishing. Zhong et al. [32] present two formulations and corresponding solutions for securely building k-anonymous tabular data held by distributed sites. Jiang et al. [19] address the distributed k-anonymity problem, where data are vertically partitioned and owned by two parties. Mohammed et al. [23] extend the above work to securely anonymize distributed data held by multiple parties under k-anonymity. Goryczka et al. [16] raise a privacy notion called m-privacy to bound the number of colluding parties in distributed anonymization. Mohammed et al. [24] propose the LKC-privacy model to address the anonymization problem for both centralized and distributed healthcare data. Different from these works, we focus on designing a multi-party data publishing approach under differential privacy, which provides a stronger privacy guarantee. Based on differential privacy and secure multi-party computation, Alhadidi et al. [2] present a two-party protocol for publishing horizontally partitioned data. In contrast, our solution is designed for multiple parties. Recently, Hong et al. [18] propose the CELS protocol, which enables different parties to collaboratively generate search logs with boosted utility while satisfying (ε, δ)-differential privacy. We solve a more general problem, i.e., the published data are not necessarily search logs. In addition, our solution provides the stronger ε-differential privacy guarantee.

Other Private Multi-Party Computation. Differential privacy has also been studied in the distributed multi-party setting for goals other than data publishing. Chan et al. [6] consider the problem where an untrusted aggregator continually monitors the heavy hitters over a set of distributed streams. Acs and Castelluccia [1] present a differentially private smart metering system based on a novel cryptographic protocol that allows an untrusted server to learn the aggregated power consumption without revealing any individual consumption. Hardt and Nath [17] apply differential privacy to personalized mobile advertising. They design a more efficient protocol than the one in [1] to gather statistics over distributed mobile users by assuming the existence of a server and a proxy that do not collude with anyone. Chen et al. [7] introduce a distributed differentially private system that enables an analyst to make statistical queries over multiple clients' data via an honest-but-curious proxy. Friedman et al. [13] design a framework to monitor general threshold functions over statistics derived from distributed streams in a differentially private manner. Goryczka and Xiong [15] study secure multi-party addition protocols using different encryption techniques (e.g., Shamirs secret sharing) in the distributed multi-party setting. All these works differ from ours which aims for distributed high-dimensional data publishing.

Private High-Dimensional Data Publishing. In the centralized setting, there have been some works on differentially private high-dimensional data publishing. To improve data utility, the crux is to overcome the curse of dimensionality. Xiao et al. [28] make use of wavelet transformation to improve the accuracy of range queries. Zhang et al. [31] propose *PrivBayes* to approximate the joint distribution of an input high-dimensional dataset from a set of well-chosen low-dimensional conditional distributions, which can significantly reduce the amount of injected noise. Very recently, Chen et al. [8] present *JTree*, which identifies a set of marginal tables from the dependency graph to approximate the joint distribution while minimizing the resultant error. Our problem can be indirectly solved by using the above techniques, i.e., each party generates a synthetic dataset based on its own data, and all such synthetic datasets compose the entire integrated dataset. However, this straightforward approach normally incurs multiple shares of noise in the integrated data. There are also some other approaches to releasing statistics of high-dimensional data under differential privacy. Ding et al. [11] privately release datacubes by searching for meaningful "subcubes" of the datacube representation. Based on Fourier decompositions, Barak et al. [4] propose a holistic solution to the problem of contingency table release. Qardaji et al. [26] propose *PriView* to differentially privately release marginal tables, in which a number of attribute sets, known as *views*, are strategically selected to reconstruct any desired α-way marginals. Day et al. [10] present *DPSense* to publish statistical information (i.e., column counts) of input datasets under differential privacy via sensitivity control. In general, these approaches cannot be directly used for the purpose of data publication.

III. Preliminaries

A. Differential Privacy

Differential privacy [12] has recently emerged as the *de facto* standard for individual privacy protection in data publishing. It provides a strong privacy guarantee to ensure that the output of a computation is insensitive to any particular record in the database. Differential privacy is defined based on the concept of *neighboring databases*, i.e., databases that have the same cardinality and differ in only one record. The formal definition of differential privacy is given as follows.

Definition 1: (Differential Privacy [12]). A random algorithm $\mathcal{A}$ achieves ε-differential privacy, if for any pair of *neighboring databases* D_1 and D_2, and for any possible anonymized dataset D',

$$\Pr[\mathcal{A}(D_1) = D'] \leq e^{\varepsilon} \times \Pr[\mathcal{A}(D_2) = D'], \tag{1}$$

where the probability $\Pr[\cdot]$ is taken over the coin tosses of the algorithm $\mathcal{A}$.

Differential privacy can be achieved by two standard mechanisms, the *Laplace mechanism* [12] and the *exponential mechanism* [22]. Both of them are based on the concept of the *sensitivity* of a function F.

Definition 2: (Sensitivity [12]). Let F be a function that maps a database into a fixed-size vector of real numbers. For all neighboring databases D_1 and D_2, the sensitivity of F is defined as:

$$S(F) = \max_{D_1, D_2} \left\|F(D_1) - F(D_2)\right\|_1, \tag{2}$$

where $\left\|\cdot\right\|_1$ denotes the L_1 norm.

For a function F whose outputs are real numbers, the Laplace mechanism [12] works by adding random noise drawn from a Laplace distribution to the true outputs. The Laplace distribution follows the probability density function $p(x) = \frac{1}{2\lambda} e^{-|x|/\lambda}$, where the magnitude $\lambda = S(F)/\varepsilon$ is determined by the desired privacy budget ε and the function's sensitivity $S(F)$.

The exponential mechanism [22] is mainly used for functions whose outputs are categorical instead of numeric. Its general idea is to sample an output O from the output space $\mathcal{O}$ according to a *score function* f_s. It assigns exponentially greater probabilities of being selected to outputs of higher scores so that the final output would be close to the optimum with respect to f_s. More specifically, given a database D, the exponential mechanism samples $O \in \mathcal{O}$ with probability proportional to $\exp(f_s(D, O)/2\Delta)$, where Δ is a factor that controls the degree of privacy protection. The exponential mechanism satisfies ε-differential privacy if $\Delta \geq S(f_s)/\varepsilon$, where $S(f_s)$ is defined as:

$$S(f_s) = \max_{D_1, D_2, O} \left|f_s(D_1, O) - f_s(D_2, O)\right|, \tag{3}$$

for any two neighboring databases D_1 and D_2, and any element $O \in \mathcal{O}$.

For a sequence of differentially private algorithms, the composability properties [21] guarantee the overall privacy.

Theorem 1: (Sequential Composition). Let $\mathcal{A}_1, \ldots, \mathcal{A}_k$ be k algorithms, each providing ε_i-differential privacy. A sequence of algorithms $\mathcal{A}_i(D)$ over a database D provides $(\sum \varepsilon_i)$-differential privacy.

Theorem 2: (Parallel Composition). Let $\mathcal{A}_1, \ldots, \mathcal{A}_k$ be k algorithms, each providing ε_i-differential privacy. A sequence of algorithms $\mathcal{A}_i(D_i)$ over disjoint databases D_i provides $\max(\varepsilon_i)$-differential privacy.

In the distributed multi-party setting, given a function F and K distributed local databases, one might aim to compute $F(\bigcup_{k=1}^{K} D_k)$ while satisfying ε-differential privacy on each D_k. To prevent the result from containing multiple shares of noise in the distributed multi-party setting, Pathak et al. [25] propose a multi-party Laplace mechanism. Its general idea is to let the parties interact with a semi-trusted curator to construct a private aggregated result, while retaining only one share of noise in it. In particular, each party P_k first computes $F(D_k)$ over the local dataset D_k and generates Laplace noise $\eta_k = Lap(S(F)/\varepsilon)$. Then, the curator generates a key-pair (pk, sk) for an additive homomorphic function $\varsigma(\cdot)$ and a size-K indicator vector u, where $u_i = 1$ and all other elements are 0. The curator element-wisely encrypts u to get $\varsigma_{pk}(u)$ and sends it to the parties. Next, the parties mutually agree on a permutation π, and calculate $\pi(\varsigma_{pk}(u)) = \varsigma_{pk}(\pi(u))$. We denote $\pi(u)$ by u' and u'_k is assigned to P_k. After that, the parties calculate $\prod_{k=1}^{K} (\varsigma_{pk}(u'_k))^{\eta_k} \cdot \varsigma_{pk}(F(D_k)) = \prod_{k=1}^{K} (\varsigma_{pk}(u'_k \cdot \eta_k + F(D_k)))$ and send the result to the curator. Finally, the curator decrypts the received result to get $\sum_{k=1}^{K} (u'_k \cdot \eta_k + F(D_k)) = F(\bigcup_{k=1}^{K} D_k) + \sum_{k=1}^{K} (u'_k \cdot \eta_k)$. Theoretical analysis indicates that:

Theorem 3: In the multi-party setting, given a function F and K distributed local databases, the multi-party Laplace mechanism preserves ε-differential privacy on each local database in computing $F(\bigcup_{k=1}^{K} D_k)$.

B. Bayesian Network

A *Bayesian network* $\mathbb{N}$ is an annotated directed acyclic graph (*DAG*) that represents a joint probability distribution of a set of random variables.

Formally, a Bayesian network can be represented by a pair $\mathbb{N} = \langle G, \Theta \rangle$. The first component G is a directed acyclic graph whose vertices correspond to the random variables and whose edges represent direct dependencies between the variables. If there exists an edge from X_j to X_i, we define X_j as a *parent* of X_i. The set of all parents of X_i is defined as its *parent set*, denoted by Π_i. The graph G encodes the following independence assumption: each variable X_i is independent of its non-descendants given its parents Π_i in G. The second component Θ represents the set of parameters that quantifies the network. It contains a table of conditional probabilities for each variable, denoted by $\Pr[X_i|\Pi_i]$. A Bayesian network $\mathbb{N}$ defines a unique joint probability distribution over an attribute set $\mathcal{X}$ given by: $\Pr_{\mathbb{N}}[\mathcal{X}] = \prod_{i=1}^{n} \Pr[X_i|\Pi_i]$.

Given a dataset D with d attributes $\mathcal{X} = \{X_1, \ldots, X_d\}$, learning a Bayesian network on D is to find a network $\mathbb{N}$ that best matches D. The common approach to this problem is to introduce a measure function that evaluates the "fitness" of networks with respect to D, and then search for the best network according to this measure function. In this paper, we use the *KL*-divergence described in [9]. Typically, the *KL*-divergence of $\Pr_{\mathbb{N}}[\mathcal{X}]$ from $\Pr[\mathcal{X}]$ is used to measure the difference between the two probability distributions, which is defined as:

$$KL(\Pr[\mathcal{X}], \Pr_{\mathbb{N}}[\mathcal{X}]) = \sum_{i=1}^{d} H(X_i) - \sum_{i=1}^{d} I(X_i, \Pi_i) - H(\mathcal{X}),$$

where $H(\cdot)$ denotes the *entropy* of a random variable and $I(\cdot,\cdot)$ denotes the *mutual information* between two variables, which is defined as:

$$I(X, \Pi) = \sum_{x \in X} \sum_{\pi \in \Pi} \Pr[X = x, \Pi = \pi] \log \frac{\Pr[X = x, \Pi = \pi]}{\Pr[X = x] \Pr[\Pi = \pi]},$$

where $\Pr[X, \Pi]$ is the joint distribution of the attribute-parent (*AP*) pair (X, Π), and $\Pr[X]$ and $\Pr[\Pi]$ are the marginal distributions of X and Π respectively. According to the *KL*-divergence, learning a Bayesian network $\mathbb{N}$ is to construct the network with the minimum *KL*-divergence. Since $\sum_{i=1}^{d} H(X_i) - H(\mathcal{X})$ is fixed once D is given, the construction of $\mathbb{N}$ can be modeled as choosing a parent set Π_i for each attribute $X_i \in \mathcal{X}$ to maximize $\sum_{i=1}^{d} I(X_i, \Pi_i)$. This learning process is NP-hard. In practice, heuristic algorithms (e.g., greedy algorithms) are often used to learn Bayesian networks.

IV. Problem Formulation

A. Problem Definition

The problem of *differentially private multi-party high-dimensional data publishing* is given as follows. There exist K parties (i.e., local data owners) $P_1, \ldots, P_K$. Each of the parties P_k holds a local database D_k. Given the K local databases and a privacy budget ε, the K parties would like to jointly release a synthetic integrated database D' to the public while enjoying ε-differential privacy on each local database D_k, such that no party will disclose any sensitive information in its local database. It is assumed that all the local databases have the same schema $\mathcal{X} = (X_1, X_2, \cdots, X_d)$ and that a single individual's information is exclusively possessed by a single party. That is, the database $D = \bigcup_{k=1}^{K} D_k$ can be viewed as horizontally partitioned among the K parties.

B. System and Threat Model

When the data is distributedly stored at multiple parties, two settings can be used for privacy-preserving data publishing. One setting is the *independent setting*, in which each party P_k $(1 \le k \le K)$ first publishes its own local database D_k independently, and then the synthetic databases (i.e., $D'_1, \ldots, D'_K$) generated by the parties are aggregated to form the integrated database D' to publish. The other is the *collaborative setting*. In the collaborative setting, we introduce a semi-trusted curator. With the assistance of the curator, all parties publish the integrated database collaboratively. Compared with the independent setting, the collaborative setting could substantially improve the utility of integrated data. Therefore, in this paper, we focus on the collaborative setting. Under this setting, our system model consists of three roles: K parties (i.e., local data owners), a semi-trusted curator T and users (e.g., data analysts). Each party P_k holds a local database D_k. The curator, who is semi-trusted, assists the K parties to release the integrated dataset D. The published data will be used by the users for different data analysis purposes.

In the collaborative setting, we assume that the parties and the curator are "honest-but-curious", which means that the parties and the curator will correctly follow the designed protocols, but act in a "curious" fashion and they may try to infer (other) local data owners' sensitive information. Moreover, it is assumed that there exists no collusion between any two parties or between any party and the curator.

V. A First-cut Solution

We start by proposing the differentially private concurrent construction of Bayesian network (*DP-CCBN*) solution, which is an extension of *PrivBayes* [31], a solution designed for the centralized setting. Before presenting the details of DP-CCBN, we first give an overview of *PrivBayes*.

PrivBayes aims to seek a k-degree Bayesian network $\mathbb{N}$ for an input dataset D under differential privacy, such that the *KL*-divergence between the original full distribution of D and the approximate distribution defined by $\mathbb{N}$ is small. *PrivBayes* is composed of three steps: (1) Using the mutual information between attributes and their candidate parent sets as the score function, it first selects a set of parents Π_i for each attribute X_i successively by employing the exponential mechanism, and then constructs a directed acyclic graph (DAG) G with the largest $\sum_{i=1}^{d} I(X_i, \Pi_i)$; (2) it generates the noisy marginal distributions of the attribute-parent (AP) pairs $\Pr(X_i, \Pi_i)$ with Laplace noise to calculate the noisy conditional distributions of each attribute given its parents in G; (3) it generates a synthetic dataset D' from the joint probability distribution derived from these conditional probability distributions by the equation $\Pr_{\mathbb{N}}[\mathcal{X}] = \prod_{i=1}^{d} \Pr[X_i|\Pi_i]$.

Now we are ready to present DP-CCBN, which consists of the following three phases.

1. Structure learning. The parties and the curator learn the structure of the Bayesian network $\mathbb{N}$. In particular, each party first generates the noisy marginal distributions of the candidate AP pairs on its local dataset and sends them to the curator. Then, the curator aggregates them to calculate the mutual information between the attributes and their candidate parent sets. Finally, the curator selects the best parent set for each attribute according to the mutual information. In this phase, the multi-party Laplace mechanism is used to reduce the amount of injected noise while satisfying differential privacy.

2. Parameter learning. After identifying the structure of $\mathbb{N}$, the parties generate the noisy marginal distributions of all the AP pairs in $\mathbb{N}$ from their local data and send them to the curator, then the curator aggregates them to learn conditional distributions of each attribute. In this phase, the multi-party Laplace mechanism is also employed.

3. Data synthesizing. Using the learned structure and parameters of $\mathbb{N}$, the curator samples tuples from the approximate distribution defined by $\mathbb{N}$ to construct the synthetic dataset D'.

In DP-CCBN, the total privacy budget ε is divided into 2 portions, ε_1 and ε_2. Each portion is used in one of the first two phases. The third phase does not require any privacy budget.

Privacy Analysis. In the first phase, $(d-1)$ iterations are needed to identify the parent set for each attribute (except the first attribute whose parent set is $\emptyset$). We uniformly assign a privacy budget $\frac{\varepsilon_1}{d-1}$ to each iteration. In each iteration, each party satisfies $\left(\frac{\varepsilon_1}{d-1}\right)$-differential privacy by calibrating the sensitivity of all candidate AP pairs' marginal distributions. By sequential composition, the first phase satisfies ε_1-differential privacy for each party. In the second phase, each party generates d noisy marginal distributions. Similarly, each party satisfies ε_2-differential privacy by calibrating the sensitivity. Again by sequential composition, DP-CCBN as a whole gives ε-differential privacy for each party, where $\varepsilon = \varepsilon_1 + \varepsilon_2$.

Communication Cost. In DP-CCBN, there exist two phases, structure learning and parameter learning, which incur communication costs. In both of these two phases, the multi-party Laplace mechanism is used to calculate the marginal distributions of the AP pairs. In particular, the curator and the parties execute the multi-party Laplace mechanism once to calculate each item of an AP pair' marginal distributions, e.g., $\Pr(X_1 = x_1, \Pi_1 = \pi_1)$. In this mechanism, an additive homomorphic function $\varsigma(\cdot)$ is employed to perform operations on the ciphertext elements. Let t_1 denote the number of bits in a ciphertext element. In DP-CCBN, we take $t_1 = 1024$. Now, we consider two parameters to evaluate the communication cost: the number of communication rounds and the size of the message sent in each phase. To calculate the number of communication rounds, we need to count the number of AP pairs to be calculated. In the structure learning phase, there are $d-1$ iterations, where d denotes the size of the attribute set. In the i^{th} iteration, there are $(d-i)$ candidate attributes and C_i^k candidate parent sets, where k denotes the degree of the Bayesian network. Thus the number of AP pairs is $(d-i)\,C_i^k$. Summing over all iterations, the number is $\sum_{i=1}^{d-1}(d-i)\,C_i^k \approx dC_d^{k+1}$. As each AP pair has 2^{k+1} items, the number of communication rounds is $dC_d^{k+1}2^{k+1}$. Based on this result, we can calculate the size of the message sent in the structure learning phase which is $dKt_1C_d^{k+1}2^{k+2}$, where K is the number of parties. In the parameter learning phase, the number of communication rounds is $(d-k)\,2^{k+1}$ and the size of the message sent is $Kt_1\,(d-k)\,2^{k+2}$.

Limitations. DP-CCBN discussed above, however, suffers from poor data utility and high communication cost. In particular, due to the high dimensionality, there exist a large number of candidate AP pairs (i.e., dC_d^{k+1}). Since the amount of noise added to the statistics is proportional to the number of candidate AP pairs, a large amount of noisy needs to be injected, which seriously reduces the "fitness" of the learned Bayesian network and thus results in poor data utility. Moreover, as the number of candidate AP pairs is enormous, there exist a large amount of statistics that need to be sent, leading to a high communication cost (i.e., $dKt_1C_d^{k+1}2^{k+2}$).

VI. Our DP-SUBN Solution

As shown in the previous section, DP-CCBN suffers from poor data utility and high communication cost. The root cause is that there exist a huge amount of candidate AP pairs in the structure learning phase. How to reduce the number of candidate AP pairs is the key challenge of our problem. In the distributed multi-party setting, although the data are partitioned over multiple parties, the conditional dependencies of the attributes are still relatively stable [29]. Using the learned dependencies of the attributes by previous parties as prior knowledge, one party can substantially reduce the number of candidate AP pairs to consider. Therefore, we propose an optimized solution, namely differentially private sequential update of Bayesian network (*DP-SUBN*). In DP-SUBN, the parties learn the Bayesian network $\mathbb{N}$ over all local datasets in a sequential manner under differential privacy.

The core of DP-SUBN is to construct a *search frontier* $\mathbb{F}$, which can be seen as priori knowledge to guide the parties to update $\mathbb{N}$. $\mathbb{F}$ consists of a set of candidate AP pairs and their corresponding marginal distributions, which form the space of all candidate Bayesian networks for the next update. In $\mathbb{F}$, each attribute X_i has a set of parent sets. The union of these parent sets is referred to as the parent set of X_i in $\mathbb{F}$, denoted by $pa\,(X_i)$. We regard the attributes as vertices, and add an directed edge between the attribute and each of its parents. All the vertices and edges constitute a directed acyclic graph (*DAG*), which is referred to as the structure of $\mathbb{F}$. We refer to the marginal distributions $\Pr(X_i, pa\,(X_i))$ as the *sufficient statistics* of X_i. The *sufficient statistics* of all the attributes form the *sufficient statistics* of $\mathbb{F}$, denoted by $\mathbb{S}$.

Since the search frontier contains candidate AP pairs, to reduce the amount of injected noise and the communication cost, we should restrict the size of the search frontier $\mathbb{F}$ (i.e., the number of candidate AP pairs contained in $\mathbb{F}$). However, simply restricting the size of $\mathbb{F}$ will result in information loss. In general, for an attribute X_i, the attributes which have stronger correlations to X_i are more likely to be its parents. Based on this observation, we propose a *correlation-aware search frontier construction* (CSFC) approach to construct the search frontier by using only the attribute pairs with strong correlations. In particular, to privately quantify the correlations of the attribute pairs without injecting too much noise, we first propose a non-overlapping covering design (NOCD) method, and then design a dynamic programming method to determine the optimal parameters used in NOCD.

In the following subsections, we provide an overview of DP-SUBN in Section VI-A, present the details of CSFC in Section VI-B, introduce NOCD in Section VI-C and discuss how to determine the optimal parameters of NOCD in Section VI-D, and finally analyze the privacy guarantee and the communication cost of DP-SUBN in Section VI-E and VI-F.

A. *Overview*

DP-SUBN consists of five phases. The first four phases demand access to the raw data and thus need to be guarded by differential privacy. Here, we uniformly divide the total privacy budget ε into 4 portions (i.e., $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = \varepsilon_4 = \frac{\varepsilon}{4}$) and assign them to each of the first four phases, respectively.

1. Correlation quantification. The parties and the curator collaboratively quantify the correlations of all attribute pairs, which will be used to construct the search frontier. We discuss the details of this phase in Section VI-C. By using the multi-party Laplace mechanism, we ensure that this phase satisfies ε_1-differential privacy on each local dataset.

2. Structure initialization. Instead of asking a single party to initialize the structure of the initial Bayesian network $\mathbb{N}_0$, we let the parties and the curator collaboratively initialize it. First, the curator specifies an order over the K parties, e.g., $P_1, \ldots, P_K$. Then, the parties initialize the structure of $\mathbb{N}_0$ under differential privacy in the following manner.

P_1 first splits the attribute set $\mathcal{X}$ into two subsets, U and V, where U is composed of attributes whose parent sets have already been identified and V is composed of the others. Then, P_1 randomly selects an attribute X_0 from V and sets its parent set to $\emptyset$. X_0 is moved to U. Next, in each of the following iterations, P_1 considers all attributes $X \in V$ as candidate attributes and all subsets $\Pi \subseteq U$ with size $\min(k, |U|)$ as candidate parent sets, and selects an AP pair (X_i, Π_i) by using the exponential mechanism, where the mutual information $I(X_i, \Pi_i)$ is used as the score function. Π_i is treated as the parents of X_i, and X_i is moved from V to U. After $d_1 = \left\lfloor \frac{d}{K} \right\rfloor$ iterations, P_1 learns the parents for d_1 attributes. Finally, P_1 sends U, V and learned AP pairs to the next party P_2. Based on the received information, P_2 learns the parents for $d_2 = \left\lfloor \frac{d}{K} \right\rfloor$ new attributes in the same way. After P_K learns the parents for the last $d_K = d - (K-1)\left\lfloor \frac{d}{K} \right\rfloor$ attributes, the structure of $\mathbb{N}_0$ is determined, which is sent to the curator. To achieve ε_2-differential privacy, each party P_k is assigned $\frac{\varepsilon_2}{d_k}$ privacy budget to learn each AP pair over its local dataset.

Based on the correlations of the attribute pairs (quantified in the first phase) and the learned structure of $\mathbb{N}_0$, the curator constructs the search frontier $\mathbb{F}_0$ and sets its corresponding sufficient statistics $\mathbb{S}_0$ to $\emptyset$. We detail the procedure of constructing the search frontier in Section VI-B. Notice that, as quantifying the correlations of all the attribute pairs and learning the structure of the Bayesian network are both done in a differentially private way, the construction and use of the search frontier will not lead to a privacy breach.

3. Structure updating. The parties and the curator update the structure of the initial Bayesian network $\mathbb{N}_0$ in a sequential manner. The update procedure consists of K iterations. In each iteration (e.g., the k^{th} $(1 \leq k \leq K)$ iteration), the curator and the parties interact as follows.

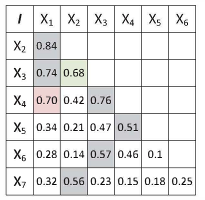
At the beginning, the curator holds the structure of the search frontier $\mathbb{F}_{k-1}$ and its corresponding sufficient statistics $\mathbb{S}_{k-1}$, which are initialized in the second phase or learned in the previous iteration, and sends $\mathbb{F}_{k-1}$ to the party P_k. According to $\mathbb{F}_{k-1}$, P_k calculates its local sufficient statistics over the database D_k, denoted by $\mathbb{S}_k^l$, and sends $\mathbb{S}_k^l$ back to the curator. To achieve differential privacy, P_k injects Laplace noise with the privacy budget ε_3 into $\mathbb{S}_k^l$. After receiving $\mathbb{S}_k^l$ from P_k, the curator updates the sufficient statistics by adding $\mathbb{S}_k^l$ to $\mathbb{S}_{k-1}$ and gets $\mathbb{S}_k$. Clearly, simply adding $\mathbb{S}_k^l$ to $\mathbb{S}_{k-1}$ will make $\mathbb{S}_k$ contain multiple shares of noise. To reduce the amount of injected noise without sacrificing the privacy level, the curator can employ a simple secure function evaluation protocol [5] to retain one share of noise in the sufficient statistics. More specifically, suppose the noisy marginal distributions of (X, Π) in $\mathbb{S}_{k-1}$ and $\mathbb{S}_k^l$ are $\sum_i^{k-1} v_i + \eta_{k-1}$ and $v_k + \eta_k$, respectively. P_{k-1}, P_k and the curator collaboratively calculate $\left(\sum_i^{k-1} v_i + \eta_{k-1}\right) + (v_k + \eta_k) - \eta_{k-1} = \sum_i^k v_i + \eta_k$ in a secure way, and get the noisy marginal distributions of (X, Π) contained in $\mathbb{S}_k$. Using the updated $\mathbb{S}_k$, the curator calculates the mutual information of the candidate AP pairs and selects a set of parents for each attribute to construct a new Bayesian network $\mathbb{N}_k$. According to $\mathbb{N}_k$ and the correlations of attribute pairs, the curator constructs a new search frontier $\mathbb{F}_k$. After K iterations, the structure of $\mathbb{N}_K$ is learned.

4. Parameter learning. This phase learns the parameters of $\mathbb{N}_K$. The procedures in this phase are the same as those in the second phase of DP-CCBN. The multi-party Laplace mechanism is used to enforce ε_4-differential privacy.

5. Data synthesizing. The curator samples tuples from the approximate distribution defined by $\mathbb{N}_K$ to construct a synthetic integrated dataset.

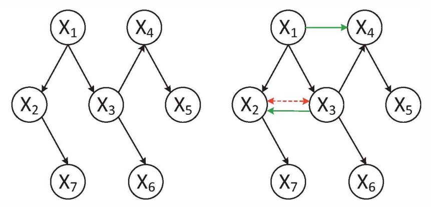
B. *Correlation-Aware Search Frontier Construction*

The search frontier $\mathbb{F}$ can be constructed in the following manner. First, the curator constructs the structure of $\mathbb{F}$ based on the current k-degree Bayesian network $\mathbb{N}$. Then, according to the structure of $\mathbb{F}$, the parties calculate the noisy local sufficient statistics of $\mathbb{F}$ on their local datasets and send them to the curator. Finally, the curator aggregates the received noisy local sufficient statistics to obtain the *sufficient statistics* $\mathbb{S}$ of $\mathbb{F}$. To reduce the amount of noise injected in $\mathbb{S}$ and achieve a low communication cost, it seems that we can restrict the size of $\mathbb{F}$ by limiting its degree to k' $(k' > k)$. However, restricting the size of $\mathbb{F}$ inevitably leads to the loss of statistics. To deal with such information loss, $\mathbb{F}$ should contain more useful candidate AP pairs in it. In other words, it should retain more potential parents for each attribute. Since the attributes with stronger correlations to an attribute X_i are more likely to be X_i's parents, they should be first considered as new candidate parents. Therefore, we formulate the problem of search frontier construction as follows. Given a k-degree Bayesian network $\mathbb{N}$ and the correlations of all attribute pairs, the curator constructs a k'-degree search frontier $\mathbb{F}$ by adding edges to $\mathbb{N}$, with the aim to maximize $\sum_{(X_i, X_j) \in E} I(X_i, X_j)$, while satisfying that $\mathbb{F}$ is a DAG, where E denotes the set of newly added edges.



I	X_1	X_2	X_3	X_4	X_5	X_6
X_2	0.84					
X_3	0.74	0.68				
X_4	0.70	0.42	0.76			
X_5	0.34	0.21	0.47	0.51		
X_6	0.28	0.14	0.57	0.46	0.1	
X_7	0.32	0.56	0.23	0.15	0.18	0.25

(a) Attribute pairs' mutual information M_I



(b) 1-degree Bayesian network (c) Adding edges

Fig. 1. Adding edges to the search frontier in CSFC

We propose a correlation-aware search frontier construction (*CSFC*) approach to solving the above problem. CSFC consists of two basic steps: select an attribute pair with strong correlation; determine which attribute is the parent of the other (i.e., determine the direction of the added edge between two attributes). With the quantified correlations of the attribute pairs, the curator can easily select the attribute pair with strong correlation. The challenge is to determine the direction of the added edge to guarantee that $\mathbb{F}$ is still a DAG. To this end, we construct a partially ordered set $\langle S, \leq_S \rangle$ according to $\mathbb{N}$. If there exists an edge or a path from X to Y in $\mathbb{N}$, we denote $X \leq Y$. If $X \leq Y$ or $Y \leq X$, X and Y are *comparable*; otherwise, they are *incomparable*.

For a selected attribute pair (X_i, X_j), if they are *comparable* (e.g., $X_i \leq X_j$), we set the direction of the added edge from X_i to X_j. For example, in Fig. 1(b), since $X_1 \leq X_4$, we set the direction of the added edge from X_1 to X_4 (as shown in Fig. 1(c)). If the selected attribute pair (X_i, X_j) are *incomparable*, setting the direction of the added edge from X_i to X_j or from X_j to X_i can both guarantee that $\mathbb{F}$ is a DAG. In this case, we introduce two concepts, *sparsity* and *impact*, to predict which direction we choose would more likely result in a larger $\sum_{(X_i,X_j)\in E} I(X_i, X_j)$. In particular, *sparsity* and *impact* are defined as follows.

Definition 3: (Sparsity). Given the search frontier $\mathbb{F}$ being constructed, let $\langle S, \leq_S \rangle$ be the partially ordered set of $\mathbb{F}$. The *sparsity* of an element X in the set S is defined as:

$$Sp\,(X) = k' \cdot |F_X| - \sum\nolimits_{X' \in F_X} |Pa(X')|, \tag{4}$$

where k' is the degree of $\mathbb{F}$, $F_X = \{X' | X' \in S \wedge X' \leq X\}$, $Pa(X') = \{Z | Z \in S \wedge Z \text{ is a parent of } X'\}$, and $|\cdot|$ denotes the size of a set.

Definition 4: (Impact). Given the search frontier $\mathbb{F}$ being constructed, let $\langle S, \leq_S \rangle$ be the partially ordered set of $\mathbb{F}$. For any two elements X and Y in the set S, the *impact* of Y on X is defined as:

$$Im\,(X,Y) = \sum\nolimits_{Y' \in D_Y} \sum\nolimits_{X' \in F_X} I\,(X', Y') - I\,(X,Y)\,, \tag{5}$$

where $D_Y = \{Y' | Y' \in S \wedge Y \leq Y'\}$.

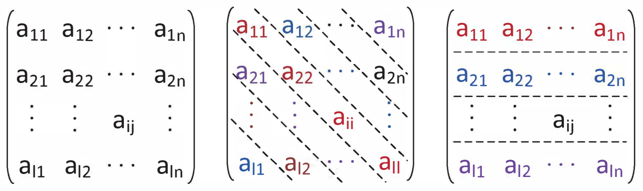
Intuitively, $Sp\,(X)$ is the sum of the maximum number of parents that can still be added to each attribute to form the search frontier, and $Im\,(X,Y)$ is the sum of the mutual information of every attribute pair (X', Y') $(X' \in F_X, Y' \in D_Y)$ except (X, Y). After adding an edge from X to Y, we have $X \leq Y$ and $X' \leq Y'$ $(X' \in F_X, Y' \in D_Y)$. To ensure that $\mathbb{F}$ is a DAG, there are $|D_Y| \cdot |F_X| - 1$ attribute pairs that cannot be considered when adding *parents* of the attributes in F_X. Thus, a larger value of $Im\,(X,Y)$ indicates that more candidate *parents* with larger mutual information of the attributes in F_X cannot be considered after adding an edge from X to Y.

For an attribute pair (X_i, X_j), if $Sp(X_j) \leq Sp(X_i)$, setting the direction of the added edge from X_j to X_i will more likely get a larger $\sum_{(X_i,X_j)\in E} I(X_i, X_j)$. In addition, if $Im\,(X_j, X_i) \leq Im\,(X_i, X_j)$, setting the direction of the added edge from X_j to X_i will more likely retain more attribute pairs with larger mutual information to be considered. Based on the above analysis, to leave more room for further optimization, we determine the direction of the added edge between two incomparable attributes X_i and X_j in the following manner. If $Sp(X_j) \cdot Im\,(X_j, X_i) \leq Sp(X_i) \cdot Im\,(X_i, X_j)$, we set the direction of the added edge from X_j to X_i; otherwise, we set it from X_i to X_j. For example, in Fig. 1, suppose $k = 1$ and $k' = 2$, we can calculate $Sp\,(X_2) = 3$, $Sp\,(X_3) = 3$, $Im\,(X_2, X_3) = 2.83$ and $Im\,(X_3, X_2) = 1.39$, and get $Sp\,(X_3) \cdot Im\,(X_3, X_2) < Sp\,(X_2) \cdot Im\,(X_2, X_3)$. Thus, we set the direction of the added edge from X_3 to X_2.

Algorithm 1 Correlation-Aware Search Frontier Construction

```
Input: $\mathbb{N}$, $M_I$, $k$, $k'$
Output: Search frontier $\mathbb{F}$
 1: Initialize $\mathbb{F}$ with $\mathbb{N}$ ;
 2: $\langle S, \leq_S \rangle \leftarrow \mathbb{F}$, $Q_I \leftarrow M_I$;
 3: while $Q_I \neq \emptyset$ do
 4:     $q = pop\,(Q_I)$, $X = q.x, Y = q.y$;
 5:     if there is no edge between $X$ and $Y$ then
 6:         if $|Pa\,(X)| < k'$ and $|Pa\,(Y)| < k'$ then
 7:             if $X < Y$ then
 8:                 $ed_{new} =< X, Y >$;
 9:             else if $Y < X$ then
10:                 $ed_{new} =< Y, X >$;
11:             else
12:                 calculate $Im\,(X,Y)\,, Im\,(Y,X)\,, Sp\,(X)\,, Sp\,(Y)$;
13:                 if $Sp\,(X) \cdot Im\,(X,Y) \geq Sp\,(Y) \cdot Im\,\langle Y, X \rangle$ then
14:                     $ed_{new} =< Y, X >$;
15:                 else
16:                     $ed_{new} =< X, Y >$;
17:                 end if
18:             end if
19:         else if $|Pa(X)| < k'$ and $Y < X$ then
20:             $ed_{new} =< Y, X >$;
21:         else if $|Pa(Y)| < k'$ and $X < Y$ then
22:             $ed_{new} =< X, Y >$;
23:         end if
24:         $\mathbb{F} = \mathbb{F} \cup ed_{new}$, $\langle S, \leq_S \rangle = \langle S, \leq_S \rangle \cup ed_{new}$;
25:     end if
26: end while
27: return $\mathbb{F}$;
```

The detailed process of CSFC is shown in Algorithm 1. It starts with a k-degree Bayesian network $\mathbb{N}$ and all the attribute pairs' mutual information M_I (shown in Fig. 1(a)). First, the curator initializes the search frontier $\mathbb{F}$ with $\mathbb{N}$ (Line 1). Then, the curator constructs the partially ordered set $\langle S, \leq_S \rangle$ according to $\mathbb{F}$, and sorts M_I by the mutual information in descending order to get the sorted queue Q_I (Line 2). Next, the curator adds edges to $\mathbb{F}$ (Lines 3-26). Specifically, the curator pops the first attribute pair q in Q_I. If the numbers of *parents* that $q.x$ and $q.y$ already have are both smaller than k', the curator will further consider whether they are comparable or not. If they are comparable, the curator determines the direction of the edge between them according to their order (Lines 6-10); otherwise, the curator will calculate their *sparsity* and *impact*, and set the direction from the one with greater product to the other (Lines 11-16). Finally, when there is no attribute to add parents, the curator obtains the k'-degree $\mathbb{F}$.



(a) $l \times n$ matrix A (b) n views generated as $m = 1$ in step 3 (c) l views generated in step 4

Fig. 2. The matrix and some generated views of NOCD

C. Quantifying the Correlations of Attribute Pairs

The correlations of attribute pairs play an important role in the construction of the search frontier. In DP-SUBN, such correlations are quantified by the mutual information of each attribute pair, which can be calculated by using its 2-way marginals. Therefore, the key of this phase is to calculate all attribute pairs' 2-way marginals. As the curator has no access to the local datasets, a straightforward approach is to let the parties calculate the noisy 2-way marginals on their local datasets, and let the curator aggregate these 2-way marginals provided by the parties. However, since the number of 2-way marginals can be very large in a high-dimensional dateset, directly generating 2-way marginals will lead to poor utility. To solve this problem, Qardaji et al. [26] propose *PriView* to differentially privately release α-way marginals, In *PriView,* a small number of "midsize" marginal tables called *views* are generated, and the *views* are used to reconstruct all α-way marginals. Intuitively, we could utilize *PriView* to improve the straightforward approach. However, the views generated by *PriView* may be overlapping in the sense that a 2-way marginal may be covered by multiple views. It implies that such an approach will need more views to cover all 2-way marginals, and gain less privacy budgets to generate each view's marginal distributions. To address this problem, we propose a novel method, *non-overlapping covering design* (*NOCD*), to generate the views for 2-way marginal reconstruction. NOCD can generate a set of non-overlapping views, while guaranteeing that all 2-way marginals can be reconstructed from these views.

In particular, NOCD is composed of four main steps:

(1) Given an attribute set $\mathcal{X}$ with size d, we first determine a pair of parameters l and n, while ensuring that $(l-1)\cdot n < d \le l \cdot n \wedge l \le n \wedge n \in \mathbb{P}$, where $\mathbb{P}$ is the set of all primes.

(2) Then, using l and n, we divide $\mathcal{X}$ into l disjoint groups. In each of the first $(l-1)$ groups, there are n attributes. In the last group, there are $(d - n\cdot(l-1))$ attributes. We store these groups in an $l \times n$ matrix A (shown in Fig. 2(a)). The j^{th} attribute in the i^{th} group is denoted by a_{ij}.

(3) Next, given m and j $(0 \le m \le n-1,\ 1 \le j \le n)$, we select exactly one attribute $a_{i,(j+(i-1)\cdot m) \bmod n}$ from each row to make up a view $V_{mj} = \{a_{1j}, \ldots, a_{l,(j+(l-1)\cdot m) \bmod n}\}$. For example, when $m = 1, j = 1$, we get $V_{11} = \{a_{11}, a_{22}, \ldots, a_{ll}\}$. Fig. 2(b) shows the n views generated when $m = 1$. By varying m and j, n^2 views are generated.

(4) Finally, the attributes in the same row make up a view, and thus l views are generated (shown in Fig. 2(c)).

As such, NOCD guarantees that the generated views are non-overlapping, and that these views can be used to reconstruct all 2-way marginals, which is formally stated in Theorem 4.

Theorem 4: Given an attribute set $\mathcal{X} = \{X_1, \ldots, X_d\}$, a set of views $\mathcal{V}$ generated by using NOCD is non-overlapping, i.e., there exists no attribute set $\{X_p, X_q\}$ to satisfy $\{X_p, X_q\} \subset V_i \cap V_j$, where $1 \le p, q \le d$ and $V_i \in \mathcal{V} \wedge V_j \in \mathcal{V} \wedge i \ne j$. In addition, any 2-way marginal can be reconstructed by the generated views, i.e., for any attribute set $\{X_{p'}, X_{q'}\}$ $(1 \le p', q' \le d)$ there exists $V_t \in \mathcal{V}$ to satisfy $\{X_{p'}, X_{q'}\} \subset V_t$.

Proof: Given an attribute set $\mathcal{X}$ with size d, in the first step of NOCD, l and n are determined, where $(l-1)\cdot n < d \le l \cdot n \wedge l \le n \wedge n \in \mathbb{P}$ and $\mathbb{P}$ is the set of all primes. In the second step of NOCD, an $l \times n$ matrix A is constructed. Based on A, in the third step of NOCD, n^2 views are generated, denoted by $\mathcal{V}_3 = \{V_{01}, \ldots, V_{0n}, \ldots V_{n-1,1}, \ldots, V_{n-1,n}\}$. In the fourth step of NOCD, l views are generated, denoted by $\mathcal{V}_4 = \{V_1, \ldots V_l\}$. In the following, we first prove that the generated views in $\mathcal{V} = \mathcal{V}_3 \cup \mathcal{V}_4$ are non-overlapping.

(1) For any view $V_{mj} \in \mathcal{V}_3$ $(0 \le m \le n-1,\ 1 \le j \le n)$, any two attributes in V_{mj} are from different rows, while for any view $V_x \in \mathcal{V}_4$, any two attributes in V_x are from the same row. Thus there exists no attribute set $\{X_p, X_q\}$ to be in $V_{mj} \cap V_x$, and we learn that V_{mj} and V_x are non-overlapping.

(2) For any two views V_{l_1} and V_{l_2} in $\mathcal{V}_4$, the attributes of V_{l_1} and the attributes of V_{l_2} are from two different rows, respectively. There exists no attribute to be in $V_{l_1} \cap V_{l_2}$, let alone attribute set. Thus, V_{l_1} and V_{l_2} are non-overlapping.

(3) For any two different views $V_{m_1 j_1}$ and $V_{m_2 j_2}$ in $\mathcal{V}_3$, suppose there exists an attribute set $\{X_p, X_q\}$ to satisfy $\{X_p, X_q\} \subset V_{m_1 j_1} \cap V_{m_1 j_1}$, and X_p and X_q correspond to $a_{i_1 s_1}$ and $a_{i_2 s_2}$ in matrix A, then we have

$$s_1 - s_2 \equiv (i_1 - i_2)\cdot m_1 \pmod{n},$$
$$s_1 - s_2 \equiv (i_1 - i_2)\cdot m_2 \pmod{n}.$$

So we have $(i_1 - i_2)\cdot(m_1 - m_2) \equiv 0 \pmod{n}$. As n is a prime, $i_1 - i_2 \not\equiv 0 \pmod{n}$ and $0 \le m_1, m_2 \le n-1$, we learn that $m_1 = m_2$. Besides, $a_{1j_1} \in V_{m_1 j_1}$ and $a_{1j_2} \in V_{m_2 j_2}$, so

$$s_1 - j_1 \equiv (i_1 - 1)\cdot m_1 \pmod{n},$$
$$s_1 - j_2 \equiv (i_1 - 1)\cdot m_2 \pmod{n}.$$

As $m_1 = m_2$, we learn that $j_1 = j_2$. Thus $V_{m_1 j_1} = V_{m_2 j_2}$. However, this is in conflict with the assumption that $V_{m_1 j_1}$ and $V_{m_2 j_2}$ are two different views. Therefore, any two different views $V_{m_1 j_1}$ and $V_{m_2 j_2}$ in $\mathcal{V}_3$ are non-overlapping.

Thus the generated views are non-overlapping. Then, we prove that any 2-way marginal can be reconstructed by the generated views, i.e., for any attribute set $\{X_{p'}, X_{q'}\}$, there exists $V_t \in \mathcal{V}_3 \cup \mathcal{V}_4$ to satisfy $\{X_{p'}, X_{q'}\} \subset V_t$.

(1) If $X_{p'}$ and $X_{q'}$ are from the same row, there exists one view $V_t \in \mathcal{V}_4$ to satisfy $\{X_{p'}, X_{q'}\} \subset V_t$.

(2) If $X_{p'}$ and $X_{q'}$ are from two different rows, assume that $X_{p'}$ and $X_{q'}$ correspond to $a_{i_3 s_3}$ and $a_{i_4 s_4}$ in matrix A. As n is a prime and $i_3 - i_4 \not\equiv 0 \pmod{n}$, there exists m' to satisfy $(i_3 - i_4)\cdot m' \equiv 1 \pmod{n}$, and we have

$$s_3 - s_4 \equiv (i_3 - i_4)\cdot m' \cdot (s_3 - s_4) \pmod{n}.$$

Let $m' \cdot (s_3 - s_4)$ be denoted by m_3. Since, there exists j_3 to satisfy $s_3 - j_3 \equiv (i_3 - 1)\cdot m_3 \pmod{n}$, there exists $V_t = V_{m_3 j_3} \in \mathcal{V}_3$ to satisfy $\{X_{p'}, X_{q'}\} \subset V_t$. ■

We observe that, for an attribute set with a large size, the value of n will be large too. In the fourth step of NOCD, if we simply let the attributes in the same row of the attribute matrix make up a view, the size of generated views might still be large. Therefore, we need to consider each row as a new attribute set, and generate some small size views from the row according to steps 1-4. This process continues until we generate all the views with appropriate sizes. Based on Theorem 4, we can prove that all the views are non-overlapping, and these views can be used to reconstruct all 2-way marginals.

D. Determining the Optimal Parameters of NOCD

In NOCD, before generating views from the attribute set, we need to determine a set of parameters. All the parameters can be represented by a tree structure, which is referred to as *parameter tree*.

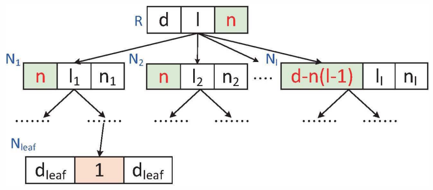


Fig. 3. A parameter tree $\mathcal{T}$ for the attribute set $\mathcal{X}$

Fig. 3 shows a parameter tree $\mathcal{T}$ for the attribute set $\mathcal{X}$. Each node N in $\mathcal{T}$ has three parameters, the size of the attribute set $N.d$, the number of the columns in the matrix $N.l$ and the number of the rows $N.n$. These parameters satisfy $(N.l-1)\cdot N.n < N.d \leq N.l \cdot N.n \wedge N.l \leq N.n \wedge N.n \in \{N.d\} \cup \mathbb{P}$, where $\mathbb{P}$ is the set of all primes. For a node N, if $N.l = 1$, N has no child (i.e., it is a leaf node), and $N.d = N.n$; otherwise, N has $N.l$ children $C_1, C_2, \ldots, C_{N.l}$, where C_i corresponds to the i^{th} row, $C_1.d = \cdots = C_{N.l-1}.d = N.n$, and $C_{N.l}.d = N.d - (N.l-1)\cdot N.n$. Particularly, for the root node R of $\mathcal{T}$, $R.d$ is the size of the entire attribute set $|\mathcal{X}|$ (i.e., $R.d = |\mathcal{X}| = d$).

Adopting the definition of the Expected value of the Sum of squared Errors (ESE) in each item of the marginals [26], we can calculate the amount of noise injected into the marginal distributions of the views generated by NOCD as follows:

$$ESE\left(\mathcal{V}^{d}\right) = (l-1)\cdot ESE\left(\mathcal{V}^{n}\right) + ESE\left(\mathcal{V}^{d-n\cdot(l-1)}\right) + 2^{l}\left(n\cdot d - n\cdot l + 2n^{2}\right)\cdot \frac{\left|\mathcal{V}^{d}\right|}{\varepsilon^{2}}, \tag{6}$$

where $\mathcal{V}^i$ denotes the set of views generated from the attribute set with size i, $|\cdot|$ denotes the size of a set, and l and n denotes the parameters for the original attribute set $\mathcal{X}$ with size d. The right side of the equation denotes the ESE of all the views generated from $\mathcal{X}$. The first two term on the left side denotes the ESE of the views generated from the first l-1 rows and the last row in the fourth step of NOCD, respectively. The third term denotes the ESE of the views generated in third step. From Eq. 6, we can observe that different parameters can result in different amount of noise. Therefore, we must find the optimal *parameter tree* to minimize the amount of noise. In addition, we can see that, to minimize $ESE\left(\mathcal{V}^{d}\right)$, we should first ensure $ESE\left(\mathcal{V}^{n}\right)$ and $ESE\left(\mathcal{V}^{d-n\cdot(l-1)}\right)$ to be minimum. Thus, to find the optimal parameters for the entire attribute set, we must first find the optimal parameters for its relevant subsets. We note that, some subsets are with the same size and the number of all relevant subsets with different sizes are limited. Taking advantage of these properties, we propose a dynamic programming method to find the optimal parameters.

Algorithm 2 Finding the Optimal Parameters for NOCD

```
Input: The size of attribute set $d$
Output: The optimal parameter tree $\mathcal{T}$
1: Initialize $opt\_tree[d] = NULL$, $min\_ese[d] = +\infty$;
2: Initialize $N = NULL$;
3: for ($d' = 1$; $d' \leq d$; $d++$) do
4:     $min\_ese[d'] = 2^{d'}$;
5:     $N.d = d'$, $N.l = 1$, $N.n = d'$;
6:     $opt\_tree[d].root = N$;
7:     for ($n = \lceil\sqrt{d'}\rceil$; $n < d' \wedge n \in \mathbb{P}$; $n++$) do
8:         $l = \lceil d'/n \rceil$;
9:         $min\_ese' = min\_ese[d' - (l-1)\cdot n] + (l-1)\cdot min\_ese[n] + 2^{l}\left(n\cdot d' - n\cdot l + 2n^{2}\right)$;
10:        if $min\_ese' < min\_ese[d']$ then
11:            $min\_ese[d'] = min\_ese'$;
12:            $N.d = d'$, $N.l = \lceil\sqrt{d'}\rceil$, $N.n = n$;
13:        end if
14:    end for
15:    if $N.l \neq 1$ then
16:        for ($l' = 1$; $l' \leq N.l - 1$; $l++$) do
17:            Add $opt\_tree[N.n]$ as a subtree of $N$;
18:        end for
19:        Add $opt\_tree[N.d - (N.l-1)\cdot N.n]$ as a subtree of $N$;
20:    end if
21: end for
22: $\mathcal{T} = opt\_tree[d]$;
23: return $\mathcal{T}$;
```

The detailed process of our dynamic programming method for determining the optimal parameter tree $\mathcal{T}$ is shown in Algorithm 2. Given the number of attributes d, we recursively find $\mathcal{T}$ in the following manner. First, we initialize an array of optimal parameter trees opt_tree with $NULL$, and an array of min_ese with $+\infty$ (Line 1). Then, we determine the optimal parameter tree $\mathcal{T}$ for the given attribute set with size d in a bottom-up manner (Line 3-21). In particular, we start at $d' = 1$ and process all the integers in $\{d' : 1 \leq d' \leq d\}$ sequentially. For each d', we first let all the d' attributes make up a view (Line 4-6). In this case, we calculate the value of ESE as the minimal ESE (i.e., $min\_ese[d']$), and set $N.l = 1$, $N.n = d'$. As determining the optimal parameter tree does not have any privacy risk, to calculate the value of ESE, we simply set the privacy budget $\varepsilon = 1$. Second, we consider each value of n in $\left\{n \middle| \lceil\sqrt{d'}\rceil \leq n < d'\right\}$ (Line 7-14). Among these n, we find the one under which the value of $min\_ese[d']$ can achieve minimum and update the values of $N.l$ and $N.n$. Third, we determine the optimal parameter tree for each row (Line 15-20). As we store the optimal parameter trees of the processed d' in opt_tree, we only need to visit opt_tree to get the optimal parameter tree of the rows and add them as a subtree of N. Finally, we determine the optimal parameter tree $\mathcal{T}$ for the attribute set with size d (Line 22).

E. Privacy Analysis

DP-SUBN consists of five phases. Based on the composability property, we have the following theorem.

Theorem 5: DP-SUBN satisfies ε-differential privacy for each local database.

Proof: In the correlation quantification phase, only the procedure of generating marginal distributions of the views requires the parties to access the local datasets. During this procedure, the multi-party Laplace mechanism is employed to ensure ε_1-differential privacy for each local dataset, and thus this phase satisfies ε_1-differential privacy for each local dataset.

In the structure initialization phase, each party P_k is assigned with d_k attributes to learn over the local dataset D_k. For each party P_k, it needs d_k iterations to learn the parents for the d_k attributes, and each iteration consumes $\frac{\varepsilon_2}{d_k}$ privacy budget. Based on the *sequential composition*, this phase satisfies ε_2-differential privacy for each local dataset.

In the structure updating phase, the update procedure consists of K iterations. In each iteration, the curator needs the party P_k to calculate the local sufficient statistics $\mathbb{S}_k^l$ of the search frontier $\mathbb{F}_{k-1}$ over the local dataset D_k, which consumes ε_3 privacy budget. Thus, this phase satisfies ε_3-differential privacy for each local dataset.

In the parameter learning phase, the curator aggregates d noisy marginal distributions generated by the parties. In this phase, the mulit-party Laplace mechanism is employed to ensure ε_4-differential privacy for each local dataset.

In the final data synthesizing phase, since DP-SUBN does not need to access any input dataset, this phase does not require any privacy budget.

Based on sequential composition, DP-SUBN as a whole satisfies ε-differential privacy for each local dataset, where $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4$. ■

F. Communication Cost

Similar to DP-CCBN, in DP-SUBN, there also exist two phases, structure learning and parameter learning that incur communication costs. The parameter learning phase in DP-SUBN is the same as that in DP-CCBN, and the number of communication rounds is $(d-k)\,2^{k+1}$ and the size of the message sent is $Kt_1\,(d-k)\,2^{k+2}$, where d, k, t_1, K denote the size of the attribute set, the degree of the Bayesina network, the number of bits in a ciphertext element and the number of parties, respectively. In the following, we discuss the number of communication rounds and the size of the message sent in the structure learning phase, which includes structure initializing and structure updating. In structure initializing, the multi-party Laplace mechanism is used to calculate the marginal distributions of the views. When there are d attributes, at least $n_v = C_d^2 \Big/ C_{|v|}^2$ views need to be calculated, where $|v|$ is the average size of the views (in practice, $|v|$ is usually smaller than 10). Thus the number of the communication rounds is $n_v 2^{|v|+1}$ and the size of the message sent is $Kt_1 n_v 2^{|v|+2}$. In structure updating, there are K iterations. In each iteration, the simple secure function evaluation protocol [5] is employed to calculate the sufficient statistics, where the additive homomorphic function $\varsigma(\cdot)$ is used to perform operations on the ciphertext elements. Since there are $d-1$ AP pairs to calculate in the search frontier construction, in each iteration, the number of communication rounds is $(d-1)\,2^{k'+1}$, and the size of the message sent is $t_1\,(d-1)\,2^{k'+2}$, where k' denotes the degree of the search frontier. Therefore, in the structure learning phase, the number of communication rounds is $K\,(d-1)\,2^{k'+1} + n_v 2^{|v|+1}$ and the size of the message sent is $Kt_1\,(d-1)\,2^{k'+2} + Kt_1 n_v 2^{|v|+2}$. Since the values of k, k' and $|v|$ are much smaller than d, both of the number of communication rounds and the size of the message sent in DP-SUBN are smaller than those in DP-CCBN, which are $dC_d^{k+1} 2^{k+1}$ and $dKt_1 C_d^{k+1} 2^{k+2}$, respectively.

VII. Experiments

In this section, we first experimentally evaluate the performance of DP-SUBN in the multi-party setting. We compare DP-SUBN with two benchmark methods. The first method is based on the idea of independent publication (referred to as *Independent*): The parties independently sanitize their local data by employing *PrivBayes* [31] and then integrate the synthetic data released by all parties. The second method is DP-CCBN introduced in Section V. In the two-party setting, we also compare DP-SUBN with *DistDiffGen* [2], which is the state-of-the-art method for differentially private data release for classification tasks in the two-party setting. To evaluate the effectiveness of NOCD, we then compare NOCD with *PriView* [26], which is the state-of-art method for α-way marginal release under differential privacy in the centralized setting. Finally, we evaluate the computation cost of the three differentially private data publishing methods, namely DP-CCBN, DP-SUBN and *DistDiffGen*.

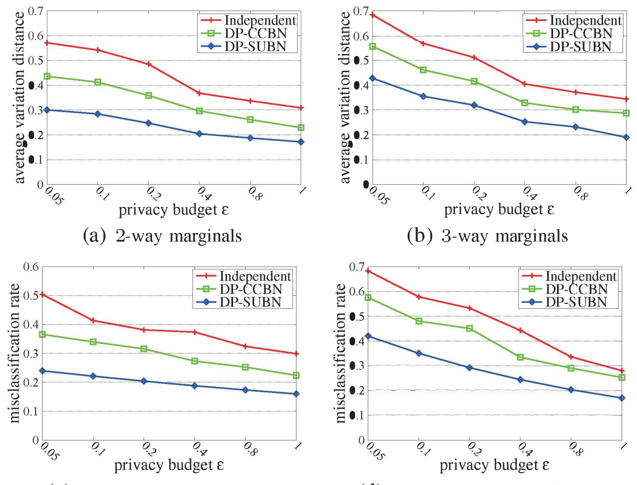
We employ the publicly available dataset *Adult* [3] for evaluation. *Adult* contains the information of 45,222 individuals extracted from the 1994 US Census, where each record has 15 attributes. Since *Adult* contains both continuous and categorical attributes, similar to [31], we adopt the approach proposed in [30] to convert each non-binary attribute into a set of binary attributes. Since this conversion only depends on the attributes' public domains, it does not incur a privacy breach. After this conversion, the number of binary attributes in the dataset is 52. To accommodate the distributed multi-party setting, similarly to [2], we randomly partition the dataset equally among the parties.

We consider two tasks to evaluate the performance of DP-SUBN. The first task is to examine the accuracy of *α-way marginals* of the synthetic integrated dataset. In our experiments, we evaluate 2-way and 3-way marginals on *Adult*. To measure the accuracy of each noisy marginal, we adopt the *total variation distance* [9] between itself and the noise-free marginal. We report the average total variation distance over all α-way marginals. The second task is *SVM classification*. We simultaneously train multiple SVM classifiers on the synthetic integrated dataset. Each classifier predicts one attribute based on all other attributes. For each classification task, we use 80% of the tuples in *Adult* as the training set, and the remaining 20% as the testing set. Specifically, two classifiers are trained on *Adult* to predict whether a person: (1) holds a post-secondary degree and (2) earns more than 50K annually. The prediction accuracy is measured by the *misclassification rate*.

We use the concept of θ-usefulness [31] to determine the parameters k and k' used in the construction of the Bayesian network and the search frontier. By setting a desired value of θ, k and k' can be chosen automatically by DP-SUBN. In our experiments, we set $\theta = 6$ and $\theta = 2$ for k and k', respectively, which usually give good results. All experiments are conducted on an Intel Core i7 2.7Hz PC with 12GB RAM. In our experiments, we run each method 20 times, and report the average results.

A. Performance of DP-SUBN vs. Independent and DP-CCBN

In this section, we compare DP-SUBN with *Independent* and DP-CCBN in the multi-party setting on two different tasks, *α-way marginals* and *SVM classification*.

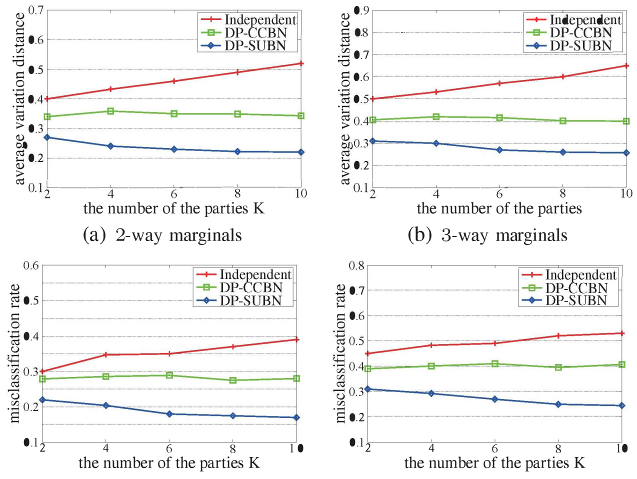


(a) 2-way marginals (b) 3-way marginals

(c) SVM classifier on salary (d) SVM classifier on education

Fig. 4. Performance of Independent, DP-CCBN and DP-SUBN under different privacy budgets

Fig. 4 shows the impact of different privacy budgets on each method. Here we set the number of the parties K=3. In particular, Fig. 4(a) and 4(b) show the average variation distance of each different α values, and Fig. 4(c) and 4(d) show the misclassification rate of each method. In all cases, DP-SUBN clearly outperforms the other two methods. This is because, in DP-SUBN, by employing the sequential update framework, the number of candidate AP pairs is substantially reduced. Thus the amount of injected noise substantially decreases, which results in a Bayesian network $\mathbb{N}$ with a better "fitness". Using $\mathbb{N}$, we can generate the synthetic dataset with a better quality.



(a) 2-way marginals (b) 3-way marginals

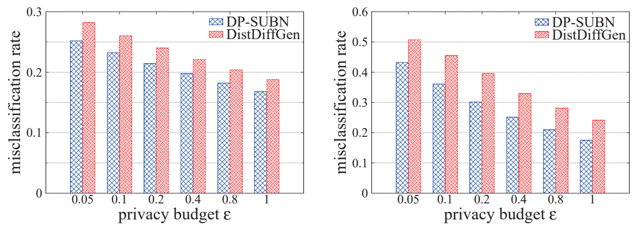
(c) SVM classifier on salary (d) SVM classifier on education

Fig. 5. Performance of Independent, DP-CCBN and DP-SUBN under different numbers of the parties

Fig. 5 shows the impact of increasing number of the parties on each method. Here we set the privacy budget ε=0.2. In particular, Fig. 5(a) and 5(b) show the average variation distance of each method for each different α values, and Fig. 5(c) and 5(d) show the misclassification rate of each method. Again, DP-SUBN outperforms the other two methods. The relative superiority of DP-SUBN is more pronounced when K increases. In particular, as K increases, the performance of DP-SUBN improves slightly, the performance of DP-CCBN remains stable, and the performance of *Independent* shows an apparent decline. The reason is that, as K increases, the number of the update iterations in DP-SUBN increases. After a series of updates, the search frontier becomes more accurate, which will lead to a better result. In contrast, as K increases, the injected noise in *Independent* becomes larger, which will lead a useless result. For DP-CCBN, one share of noise is retained in the aggregated result. The scale of the noise is decided by the sensitivity of the algorithm and the privacy budget ε. Thus the performance of DP-CCBN is insensitive to the number of the parties.

B. Performance of DP-SUBN vs. DistDiffGen

In this section, we compare the performance of DP-SUBN with *DistDiffGen* in the two-party setting. Notice that *DistDiffGen* is designed for classification task, and therefore we do not compare DP-SUBN with *DistDiffGen* on the task of α-way marginals. Since *DistDiffGen* needs to generate a synthetic dataset for each SVM classifier, we evenly divide the privacy budget ε into two portions, and use $\varepsilon/2$ to generate each set of data by using *DistDiffGen* for *salary* and *education*.



(a) SVM classifier on salary (b) SVM classifier on education

Fig. 6. Performance of DP-SUBN and DistDiffGen in the two-party setting

Fig. 6 shows the misclassification rate of each method under different ε values, from which we can see that, DP-SUBN outperforms *DistDiffGen* on both attributes. The reason lies in that *DistDiffGen* first groups the original dataset by specializing the attributes, and then adds Laplace noise to the count of each group to generate the synthetic dataset. For the high-dimensional data *Adult*, *DistDiffGen* generates too many groups. Thus, the noise is relatively larger in *DistDiffGen* than that in DP-SUBN. In addition, we also observe that the relative superiority is more significant on *education* because the attribute domain of *education* is larger than that of *salary*. As a result, more groups are generated in the synthetic data for *education*, and the amount of injected noise will be larger, which reduces the utility of the data.

C. Effect of NOCD

We evaluate the effectiveness of NOCD against *PriView* in the centralized setting. The performance of each method is measured by the *average variation distance* for all 2-way marginals.

Fig. 7 presents the performance of NOCD and *PriView* on *Adult*. We can see that the *average variation distances* of NOCD and *PriView* reduce as the privacy budget increases. In addition, we can observe that NOCD outperforms *PriView* in all cases. This confirms our analysis in Section VI-C that NOCD needs less views than *PriView* to cover all 2-way marginals, which increases the privacy budget used to calculate each marginal, and thus improves the utility of the results.

D. Communication Cost

We evaluate the communication cost of DP-CCBN and DP-SUBN. In addition, we also compare the communication cost of DP-SUBN with *DistDiffGen* in the two-party setting.

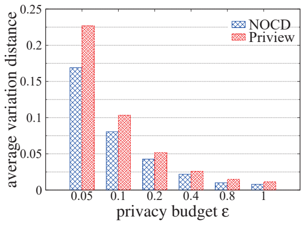


Fig. 7. Effect of NOCD

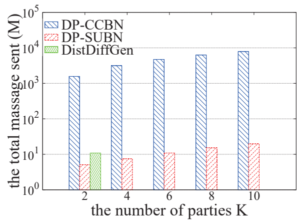


Fig. 8. Communication cost

Fig. 8 shows the total size of the message sent in DP-SUBN and DP-CCBN under different numbers of the parties K, and the total size of the message sent in *DistDiffGen* in the two-party setting. We can observe that the communication cost of DP-SUBN is smaller than that of *DistDiffGen* in the two-party setting. Moreover, the communication cost of DP-SUBN is orders of magnitude smaller than that of DP-CCBN in all cases, and the difference becomes more apparent as K increases. This confirms our communication cost analysis. The reason is explained as follows. In DP-SUBN, the number of candidate AP pairs can be substantially reduced by using the search frontier. For a larger K, the size of the search frontier becomes smaller and smaller during the update process in DP-SUBN, and thus less message needs to be sent. However, the size of message sent for each party remains unchanged in DP-CCBN, and thus the total size of message sent increases proportionally with the value of K.

VIII. Conclusion

In this paper, we have studied the problem of differentially private multi-party high-dimensional data publishing. To solve this problem, we have presented a differentially private sequential update of Bayesian network (*DP-SUBN*) solution, which is accompanied with two novel techniques, namely correlation-aware search frontier construction (*CSFC*) and non-overlapping covering design (*NOCD*). We have formally proven that DP-SUBN guarantees ε-differential privacy for any local dataset. Extensive experiments on a real dataset demonstrate that DP-SUBN offers high data utility with low communication cost.

Acknowledgement

We sincerely thank the anonymous reviewers for their insightful comments. The work was supported by National Natural Science Foundation of China under Grant 61502047 and HK-RGC Grant 12200114.

References

[1] G. Acs and C. Castelluccia. I have a DREAM!: differentially private smart metering. In *IH*, pages 118–132, 2012.

[2] D. Alhadidi, N. Mohammed, B. C. M. Fung, and M. Debbabi. Secure distributed framework for achieving ϵ-differential privacy. In *PETS*, pages 120–139, 2012.

[3] K. Bache and M. Lichman. Uci machine learning repository. In *PODS*, 2013.

[4] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *PODS*, pages 273–282, 2007.

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *STOC*, pages 1–10, 1988.

[6] T.-H. H. Chan, M. Li, E. Shi, and W. Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In *PETS*, pages 140–159, 2012.

[7] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke. Towards statistical queries over distributed private user data. In *NSDI*, pages 169–182, 2012.

[8] R. Chen, Q. Xiao, Y. Zhang, and J. Xu. Differentially private high-dimensional data publication via sampling-based inference. In *SIGKDD*, pages 129–138, 2015.

[9] A. B. Cybakov. Introduction to nonparametric estimation. *Springer*, 2009.

[10] W. Day and N. Li. Differentially private publishing of high-dimensional data using sensitivity control. In *ASIA CCS*, pages 451–462, 2015.

[11] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD*, pages 217–228, 2011.

[12] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[13] A. Friedman, I. Sharfman, D. Keren, and A. Schuster. Privacy-preserving distributed stream monitoring. In *NDSS*, pages 1–12, 2014.

[14] B. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):1–53, 2010.

[15] S. Goryczka and L. Xiong. A comprehensive comparison of multiparty secure additions with differential privacy. *TDSC*, 2015.

[16] S. Goryczka, L. Xiong, and B. C. M. Fung. *m*-privacy for collaborative data publishing. *TKDE*, 26(10):2520–2533, 2014.

[17] M. Hardt and S. Nath. Privacy-aware personalization for mobile advertising. In *CCS*, pages 662–673, 2012.

[18] Y. Hong, J. Vaidya, H. Lu, P. Karras, and S. Goel. Collaborative search log sanitization: Toward differential privacy and boosted utility. *TDSC*, 12(5):504–518, 2015.

[19] W. Jiang and C. Clifton. A secure distributed framework for achieving *k*-anonymity. *VLDB J.*, 15(4):316–333, 2006.

[20] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *ICDE*, page 24, 2006.

[21] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD*, pages 19–30, 2009.

[22] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[23] N. Mohammed, B. C. M. Fung, and M. Debbabi. Anonymity meets game theory: secure data integration with malicious participants. *VLDB J.*, 20(4):567–588, 2011.

[24] N. Mohammed, B. C. M. Fung, P. C. K. Hung, and C. Lee. Centralized and distributed anonymization for high-dimensional healthcare data. *TKDD*, 4(4):18, 2010.

[25] M. A. Pathak, S. Rane, and B. Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *NIPS*, pages 1876–1884, 2010.

[26] W. H. Qardaji, W. Yang, and N. Li. Priview: practical differentially private release of marginal contingency tables. In *SIGMOD*, pages 1435–1446, 2014.

[27] L. Sweeney. k-anonymity: A model for protecting privacy. *IJUFKS*, 10(5):557–570, 2002.

[28] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, pages 225–236, 2010.

[29] K. Yamanishi. Distributed cooperative bayesian learning strategies. *IANDC*, 150(1):22–56, 1999.

[30] G. Yaroslavtsev, G. Cormode, C. M. Procopiuc, and D. Srivastava. Accurate and efficient private release of datacubes and contingency tables. In *ICDE*, pages 745–756, 2013.

[31] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: private data release via bayesian networks. In *SIGMOD*, pages 1423–1434, 2014.

[32] S. Zhong, Z. Yang, and R. N. Wright. Privacy-enhancing *k*-anonymization of customer data. In *PODS*, pages 139–147, 2005.