Anonymous Publication of Sensitive Transactional Data

Gabriel Ghinita[1], Panos Kalnis[2] and Yufei Tao[3]
[1]Purdue University, email: gghinita@cs.purdue.edu
[2]KAUST University, email: panos.kalnis@kaust.edu.sa
[3]Chinese University of Hong Kong, email: taoyf@cse.cuhk.edu.hk

Abstract—Existing research on privacy-preserving data publishing focuses on relational data: in this context, the objective is to enforce privacy-preserving paradigms, such as k-anonymity and ℓ-diversity, while minimizing the information loss incurred in the anonymizing process (i.e., maximize data utility). Existing techniques work well for fixed-schema data, with low dimensionality. Nevertheless, certain applications require privacy-preserving publishing of transactional data (or basket data), which involve hundreds or even thousands of dimensions, rendering existing methods unusable.

We propose two categories of novel anonymization methods for sparse high-dimensional data. The first category is based on approximate nearest-neighbor (NN) search in high-dimensional spaces, which is efficiently performed through locality-sensitive hashing (LSH). In the second category, we propose two data transformations that capture the correlation in the underlying data: *(i)* reduction to a band matrix and *(ii)* Gray encoding-based sorting. These representations facilitate the formation of anonymized groups with low information loss, through an efficient linear-time heuristic. We show experimentally, using real-life datasets, that all our methods clearly outperform existing state-of-the-art. Among the proposed techniques, NN-search yields superior data utility compared to the band matrix transformation, but incurs higher computational overhead. The data transformation based on Gray code sorting performs best in terms of both data utility and execution time.

Index Terms—Privacy, Anonymity, Transactional Data

1 Introduction

Organizations, such as hospitals, publish detailed data (*micro-data*) about individuals (e.g., medical records) for research or statistical purposes. However, sensitive personal information may be disclosed in this process, due to the existence in the data of quasi-identifying attributes (QID), such as age, zipcode, etc. An attacker can join the QID with external information, such as voting registration lists, to re-identify individual records.

Existing privacy-preserving techniques focus on anonymizing personal data, which have a fixed schema with a small number of dimensions. Through *generalization* or *suppression*, existing methods prevent attackers from re-identifying individual records.

However, anonymization of personal data is not sufficient in some applications. Consider, for instance, the example of a large retail company which sells thousands of different products, and has numerous daily purchase transactions. The large amount of transactional data may contain customer spending patterns and trends that are essential for marketing and planning purposes. The company may wish to make the data available to a third party which can process the data and extract interesting patterns (e.g., perform data mining tasks). Since the most likely purpose of the data is to infer certain purchasing trends, characterized by *correlations* among purchased products, the personal details of the customers are not relevant, and are altogether suppressed. Instead, only the contents of the shopping cart are published for each transaction. Still, there may be particular purchasing habits that disclose customer identity and expose sensitive customer information.

1.1 Motivation

Consider the example in Fig. 1a, which shows the contents of five purchase transactions (the customer name is not disclosed, we include it just for ease of presentation). The sensitive products (items), which are considered to be a privacy breach if associated with a certain individual, are shown shaded. The rest of the items, which are non-sensitive, can be used by an attacker to re-identify individual transactions, similarly to a quasi-identifier, with the distinctive characteristic that the number of potentially identifying items is very large in practice (hence, the QID has very high dimensionality). Consider the transaction of Claire, who bought a pregnancy test. An attacker (Eve) may easily learn about some of the items purchased by Claire on a certain day, possibly from a conversation with her, or from knowing some of her personal preferences. For instance, Claire may treat her guests, including Eve, with fresh cream and strawberries, and Eve can therefore infer that Claire must have purchased these items recently. By joining this information with the purchase transaction log, Eve can re-identify Claire's transaction, and find out that Claire may be pregnant.

The privacy breach occurs because Eve was able to identify with certainty the purchase transaction of Claire, and hence associate her with the sensitive item *pregnancy test*. As we will show later in Section 6, for real-life datasets an attacker can re-identify the transaction of a particular individual with 20% probability based on knowledge on two purchased items. The probability increases to 40% with three known items, and to over 90% with four items. To protect Claire's privacy, we must prevent the association of her QID items with a particular sensitive item with probability larger than a certain threshold.

To address this privacy threat, one solution would be to employ ℓ-diversity [2]: a well-established paradigm in relational data privacy, which prevents sensitive attribute (i.e.,

	Wine	Strawberries	Meat	Cream	Pregnancy Test	Viagra
Bob	X		X			X
David	X		X			
Claire		X		X	X	
Andrea		X	X			
Ellen	X		X	X		

(a) Original Data

	Wine	Meat	Cream	Strawberries	Pregnancy Test	Viagra
Bob	X	X				X
David	X	X				
Ellen	X	X	X			
Andrea		X		X		
Claire			X	X	X	

(b) Band Matrix Transformation

	Wine	Strawberries	Meat	Cream	Pregnancy Test	Viagra
Andrea		X	X			
Claire		X		X	X	
Bob	X		X			X
David	X		X			
Ellen	X		X	X		

(c) Gray Encoding Transformation

	Wine	Meat	Cream	Strawberries	Sensitive Items
Bob	X	X			Viagra: 1
David	X	X			
Ellen	X	X	X		
Andrea		X		X	Pregnancy Test: 1
Claire			X	X	

(d) Published Groups

Fig. 1. Purchase Transaction Log Example; Viagra and Pregnancy Test are sensitive items

item) disclosure. Currently, there exist two broad categories of ℓ-diversity techniques: *generalization*- and *permutation*-based. Both categories assume fixed-schema data, with a relatively low number of QID items. In our case, the transactional data are represented as a table with one row for each transaction t, and one column for each possible item. For each transaction t, a column has value **1** if the corresponding item belongs to t, and **0** otherwise. An existing generalization method [3], [4] would partition the data into disjoint groups of transactions, such that each group contains sufficient records with ℓ distinct, well-represented sensitive items. Then, all quasi-identifier values in a group would be generalized to the entire group extent in the QID space. If at least two transactions in a group have distinct values in a certain column (i.e., one contains an item and the other does not), then all information about that item in the current group is lost. The QID used in this process includes all possible items in the log. Due to the high-dimensionality of the quasi-identifier, with the number of possible items in the order of thousands, it is likely that any generalization method would incur extremely high information loss, rendering the data useless [5].

In contrast, a permutation method such as Anatomy [6] would randomly pick groups of transactions with distinct sensitive items, and permute these items among transactions, to reduce the association probability between an individual transaction and a particular sensitive item. However, the group formation phase does not consider similarity among QID values of transactions, hence correlations between QID and sensitive items may be lost. This may prevent the extraction of useful information from the data, compromising utility.

1.2 Contributions

We propose two categories of anonymization techniques that combine the advantages of both generalization and permutation, and also address the difficult challenge of high dimensionality: *(i)* methods that rely on efficient nearest-neighbor (NN) search in high-dimensional spaces, using Locality-Sensitive Hashing (LSH) [7] and *(ii)* methods that employ data re-organization. Techniques in the latter category consist of two stages: in the first stage, the data representation is changed (without altering the contents), such that transactions with similar QID are placed in proximity to each other. In the second stage, transactions are assembled into anonymized groups.

We propose two re-organization methods: the first method transforms the data into a *band matrix* by performing permutations of rows and columns in the original table. The outcome of the transformation is shown in Fig. 1b. This representation takes advantage of data sparseness and places non-zero entries near the main diagonal. The advantage is that neighboring rows have high correlation, i.e., share a large number of common items. The second data transformation technique relies on sorting with respect to Gray encoding [8]: the QID items in each transaction t are interpreted as the Gray code of t. The transaction set is then sorted according to the rank in the Gray sequence. Fig. 1c shows the transformed dataset. In this representation, transactions that are consecutive in the sequence have low Hamming distance, hence their QID are similar.

The output of any of the two methods is then fed to an efficient linear-time heuristic which groups together nearby transactions, therefore reducing the search space of the solution. Since both data transformations capture correlation well, groups contain transactions with similar QID, leading to increased data utility.

Regardless of the group formation technique, we use a publishing format similar to Anatomy. In each anonymized group, sensitive items are separated from the QID, and published in a separate summary table, as shown in Fig. 1d. Two groups with high intra-group correlation are formed: {*Bob, David, Ellen*}, all with probability $1/3$ of buying *viagra*, and {*Andrea, Claire*} with probability of buying *pregnancy test* $1/2$. Note that, for our particular example, both the band matrix (Fig. 1b), as well as the Gray code re-organization (Fig. 1c) determine the same anonymous groups.

Our techniques address the concern of high data dimensionality by anonymizing each group of transactions according to a *relevant* quasi-identifier, consisting of items that exist in the group. The underlying assumption is that each transaction can be re-identified based on its items, which are a small subset of the entire item space. This has the potential of circumventing the dimensionality curse, by not using a unique, high-dimensional QID for all groups. Although each transaction has a low number of items, they are distributed in the entire item space, so the challenge is how to effectively group together transactions with similar QID.

Our specific contributions are:

- we devise an anonymized group formation strategy which relies on efficient nearest-neighbor search in high-dimensional spaces. This method outputs directly the anonymized groups, without a need for an additional data re-organization step
- we introduce two representations for transactional data which take advantage of data sparseness, preserve correlations among items and arrange transactions with similar QID in close proximity to each other. The first method relies on transformation to a band matrix format, whereas the second employs sorting with respect to binary reflected Gray codes
- we devise an efficient linear-time heuristic which creates anonymized groups based on the two proposed data organization methods
- we evaluate experimentally our methods with real datasets, and show that they clearly outperform existing state-of-the-art in terms of both data utility and computational overhead

The rest of the paper is organized as follows: Section 2 surveys related work. Section 3 discusses fundamental concepts and definitions. Section 4 introduces the first category of anonymization techniques that rely on nearest-neighbor search, whereas Section 5 presents the second category which is based on reduction to band matrix and Gray code sorting. Section 6 includes the results of our experimental evaluation. Finally, Section 7 concludes the paper with directions for future work.

2 Related Work

Privacy-preserving data publishing has received considerable attention in recent years, especially in the context of relational data. The work in [10] employs random perturbation to prevent re-identification of records, by adding noise to the data. In [11], it is shown that an attacker could filter the random noise, and hence breach data privacy, unless the noise is correlated with the data. However, randomly perturbed data is not "truthful" [12], in the sense that it contains records which do not exist in the original data. Furthermore, random perturbation may expose privacy of outliers when an attacker has access to external knowledge.

Published data about individuals (*microdata*) contain *quasi-identifier* attributes (QID), such as age, or zipcode, which can be joined with public databases (e.g., voting registration lists) to re-identify individual records. To address this threat, Samarati [13] introduced k-anonymity, a privacy-preserving paradigm which requires each record to be indistinguishable among at least $k-1$ other records with respect to the set of QID attributes. Records with identical QID values form an *anonymized group*. k-anonymity can be achieved through *generalization*, which maps detailed attribute values to value ranges, and *suppression*, which removes certain attribute values or records from the microdata. The process of data anonymization is called *recoding*, and it inadvertently results in information loss. Several privacy-preserving techniques have been proposed, which attempt to minimize information loss, i.e., maximize utility of the data. LeFevre et al. [3] proposed optimal k-anonymity solutions for *single-dimensional* recoding, which performs value mapping independently for each attribute. In [4], the same authors introduced *Mondrian*, an heuristic solution for *multi-dimensional* recoding, which maps the Cartesian product of multiple attributes. Mondrian outperforms optimal single-dimensional solutions, due to its increased flexibility in forming anonymized groups. Methods discussed so far perform *global* recoding, where a particular detailed value is always mapped to the same generalized value. In contrast, *local* recoding allows distinct mappings across different groups. Clustering-based local recoding methods are proposed in [14].

k-anonymity prevents re-identification of individual records, but it is vulnerable to *homogeneity* attacks, where many of the records in an anonymized group share the same sensitive attribute (SA) value. ℓ-diversity [2] addresses this vulnerability, and creates anonymized groups in which at least ℓ SA values are well-represented. Any k-anonymity technique can be adapted for ℓ-diversity; however, this approach typically causes high information loss. The work in [15] proposes a framework based on dimensionality mapping, which can be tailored for k-anonymity and ℓ-diversity, and outperforms other generalization techniques. However, dimensionality mapping is only effective for low-dimensional QIDs, hence the method is not suitable for transactional data. Furthermore, existing ℓ-diversity methods work for a single sensitive attribute, whereas in our problem, we need to consider a larger number of sensitive items. The work in [9] considers that external knowledge is available to an adversary, in the form of logical constraints on data records. However, the solution proposed targets relational (i.e., low-dimensional) data.

Anatomy [6] introduced a novel approach to achieve ℓ-diversity: instead of generalizing QID values, it decouples the SA from its associated QID, and *permutes* the SA values among records. Since QIDs are published directly, the information loss is reduced. A similar approach is taken in [16]. However, neither of these methods account for correlation between the QID and the SA when forming anonymized groups. We also adopt a permutation approach for transactional data, but we create anonymized groups in a QID-centric fashion, therefore preserving correlation and increasing data utility. Furthermore, our novel data representation helps us tackle the challenge of high-dimensional QID.

Privacy-preservation of transactional data has been acknowledged as an important problem in the data mining literature. However, existing work [17], [18] focuses on publishing *patterns*, and not data. The patterns (or rules), are mined directly from the original data, and the resulting set of rules is sanitized to prevent privacy breaches. Such an approach has two limitations: *(i)* the usability of the data is constrained by the rules that the owner decides to disclose and *(ii)* it is assumed that the data owner has the resources and expertise to perform advanced data mining tasks, which may not be the case in practice. We choose to publish the *data*, instead of patterns; this gives the recipient flexibility in choosing what rules to mine, and also allows for other types of data analysis, such as clustering. Furthermore, the processing cost need not be bared by the data owner.

Similar to our approach, the work in [19], [20], [21] addresses publication of anonymized transactional data. However, only the re-identification of individual transactions is prevented (i.e., the equivalent of the k-anonymity paradigm), whereas the privacy threat of associating sensitive items with

quasi-identifiers is not addressed.

In our work, we employ Locality-sensitive Hashing (LSH). Gionis et al. [7] proposed a LSH technique for similarity search in high-dimensional spaces (e.g., multimedia databases) that relies on the mapping of the data space to a Hamming space of higher dimensionality. A hashing algorithm for binary data, the same that we use in our anonymization method, is then applied.

We also use the Gray (or binary-reflected) code [8] which has benefited a large variety of applications. Initially introduced for error-correction in digital communications, the Gray code has found applications in data compression [22]. Similar to the anonymization of transactions, data compression techniques require low Hamming distance between consecutive data elements, and several effective compression techniques based on Gray sorting have been devised [22], [23].

3 BACKGROUND

Our objective is to anonymize data consisting of a set of transactions $T = \{t_1, t_2, \ldots, t_n\}$, $n = |T|$. Each transaction $t \in T$ contains items from an item set $I = \{i_1, i_2, \ldots, i_d\}$, $d = |I|$. We represent the data as a binary matrix A with n rows and d columns, where $A[i][j] = 1$ if $i_j \in t_i$, and 0 otherwise.

Among the set of items I, some are sensitive, such as *pregnancy test* or *viagra* in our example.

Definition 1 (Sensitive Items): The set $S \in I$ of items that represent a privacy threat if associated to a certain transaction, constitutes the *sensitive items* set, $S = \{s_1 \ldots, s_m\}$, $m = |S|$.

The rest of the items in I, such as *wine, cream* etc, are not sensitive, in the sense that their association with a certain individual is not detrimental. On the other hand, these innocuous items can be used by an attacker to re-identify individual transactions. We denote these items by *quasi-identifier (QID)* items.

Definition 2 (Quasi-identifier Items): The set of items in I that an attacker can gain knowledge on in order to re-identify individual transactions constitute the set of *quasi-identifiers*. Potentially, any non-sensitive item is a quasi-identifier, hence $Q = I \diagdown S = \{q_1, \ldots, q_{d-m}\}$.

We denote a transaction which contains items from S as *sensitive transaction*, and one which contains only items from Q as *non-sensitive*.

In previous work on privacy preservation of relational data, the underlying assumption is that a single, fixed schema exists, and all records abide the schema, therefore a single QID is used for all records. However, such an approach is not suitable due to the high dimensionality of the data. In our problem, the data are sparse, and each transaction can be re-identified by a small number of items. For a given transaction $t \in T$, we denote its quasi-identifier as $t[Q]$. For instance, in the example in Fig. 1, $Bob[Q] = David[Q] = \{Wine, Meat\}$ and $Claire[Q] = \{Cream, Strawberries\}$. In Section 5, we will show two methods that re-organize the data effectively, such that transactions with similar QID are placed in close proximity to each other.

3.1 Privacy Requirements

Two main privacy-preserving paradigms have been established for relational data: k-anonymity [13], which prevents identification of individual records in the data, and ℓ-diversity [2], which prevents the association of an individual record with a sensitive attribute value. Observe that, it is the association of individuals with sensitive information that ultimately threats privacy. Similarly, in the case of transactional data, the privacy threat is defined as the association of an individual transaction to a sensitive item. Nevertheless, the privacy preservation of transactional data is different from its relational database counterpart, where all records have the same number (usually one) of sensitive attributes. In our case, some transactions may not contain any sensitive item, hence are completely innocuous, while others may have multiple sensitive items.

Definition 3 (Privacy): A privacy-preserving transformation of transaction set T has *privacy degree p* if the probability of associating any transaction $t \in T$ with a particular sensitive item $s \in S$ does not exceed $1/p$.

This is equivalent to saying that the transaction of an individual can be associated to a certain sensitive item with probability at most $1/p$ among $p - 1$ other transactions.

Note that, the association of an individual with an item in Q does *not* represent a privacy breach: there is no detrimental information in the fact that Bob, for instance, has purchased *meat*. For this reason, the items in Q can be released directly, and we can employ a permutation-based approach, similar to [6]. This has a considerable impact in reducing information loss, compared to generalization-based approaches.

We enforce the privacy requirement by partitioning the set T into disjoint sets of transactions, which we refer to as *anonymized groups*. For each group G, we publish the exact QID items, together with a summary of the frequencies of sensitive items contained in G. In the example of Fig. 1d the second group contains two transactions, corresponding to Andrea and Claire, and one occurrence for sensitive item *pregnancy test*. The probability of associating any transaction in G to that item is $1/2$. In general, let $f_1^G \ldots f_m^G$ be the number of occurrences for sensitive items $s_1 \ldots s_m$ in group G. Then, group G offers privacy degree

$$degree(G) = \min_{i=1\ldots m} |G|/f_i^G \tag{1}$$

The privacy degree of an entire partitioning $\mathcal{P}$ of T is:

$$degree(\mathcal{P}) = \min_{G \in \mathcal{P}} degree(G) \tag{2}$$

Next, we discuss aspects related to the utility of the data transformed according to a given privacy degree p.

3.2 Utility Requirements

It is well-understood [24] that publishing privacy-sensitive data is caught between the conflicting requirements of privacy and utility. To preserve privacy, a certain amount of information loss is inherent. Nevertheless, the data should maintain a reasonable degree of utility.

Transactional data are mainly utilized to derive certain patterns, such as consumer purchasing habits. Returning to the running example, observe that there are two categories of patterns: those that involve items from Q alone, and those that involve at least one item in S. For the former category, there is no information loss. For instance, we can derive that half of the customers that bought *strawberries* have also bought *cream*.

On the other hand, when sensitive items are involved, the information derived from the anonymized data is only an

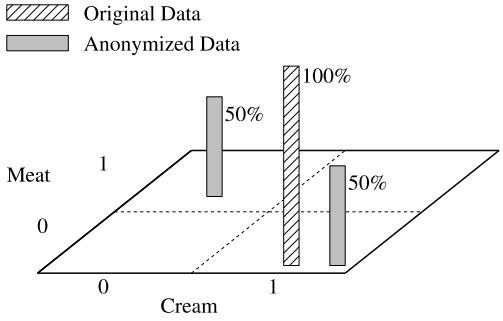

Fig. 2. Data reconstruction for transactions with sensitive item *pregnancy test*

estimation of the real one. For instance, by inspecting the second anonymized group (Fig. 1d) we can infer with 50% probability that whoever buys *cream* and *strawberries* also buys a *pregnancy test*, whereas from the original data we can infer this with 100% probability. A pattern can be expressed as a query:

$$\begin{aligned} &SELECT\ COUNT(*)\ FROM\ T \\ &WHERE\ (SensitiveItem\ s\ is\ present) \\ &AND\ (q_1 = val_1) \wedge \ldots \wedge (q_r = val_r) \end{aligned} \tag{3}$$

The process of estimating the result of the query for each anonymized group G is referred to as *data reconstruction*. Denote the number of occurrences of item s in G by a, and the number of transactions that match the QID selection predicate (last line of (3)) by b. Then the estimated result of the query, assuming each permutation of sensitive items to each QID combination in G is equally likely, is:

$$a \cdot b/|G| \tag{4}$$

If all transactions in G have identical QID, then either $b = |G|$ or $b = 0$, and the reconstruction error is 0. Ideally, to minimize the error, we need to minimize $|G| - b$, hence include in each group transactions with similar, and when possible identical, QID.

A meaningful way of modeling such queries that involve sensitive items is to use a *probability distribution function (pdf)* of an item $s \in S$ over the space defined by a number of r items in Q. For example, assume that a data analyst wishes to find the correlation between *pregnancy test* and quasi-identifier items *cream* and *meat*. Fig. 2 represents this scenario: every cell corresponds to a combination of the QID items; for instance $(1, 0)$ corresponds to transactions that contain *cream* but not *meat*, whereas $(1, 1)$ to transactions with both *cream* and *meat*. From the original data, we can infer that all customers who bought *cream* but not *meat* have also bought a *pregnancy test*. However, from the anonymized data, we can only say that half of such customers have bought a *pregnancy test*.

If the query includes r QID items, the total number of cells is 2^r, corresponding to all combinations when an item is or is not present in a transaction (this is the same as having a "group-by" query on items $q_1 \ldots q_r$). The actual pdf value of sensitive item s for a cell C is

$$Act^s_C = \frac{Occurrences\ of\ s\ in\ C}{Total\ Occurrences\ of\ s\ in\ T}$$

The estimated pdf Est^s_C is computed similarly, except that the numerator consists of Eq.(4) summed over all groups that intersect cell C. We determine the utility of the anonymized data as the distance between the real and estimated pdf over all cells, measured by ***KL-divergence***, already established [24] as a meaningful metric to evaluate the information loss incurred by anonymization:

$$KL\_Divergence(Act^s, Est^s) = \sum_{\forall cell\ C} Act^s_C \log \frac{Act^s_C}{Est^s_C}$$

where Act^s and Est^s represent the actual and estimated *pdf* of s over all cells. If Act^s is identical to Est^s, $KL\_Divergence = 0$. KL-divergence captures the amount of correlation preserved between quasi-identifiers and sensitive items, which is relevant for many use scenarios of anonymized data. For instance, if the correlation between items $q_1 \ldots q_r$ and sensitive item s is preserved, then the estimated count of transactions that contain $q_1 \ldots q_r, s$ is close to the actual count. Therefore, the confidence of association rules such as

$$Conf((q_1 \ldots q_r) \Rightarrow s) = \frac{Count(q_1 \ldots q_r, s)}{Count(q_1 \ldots q_r)}$$

and

$$Conf(s \Rightarrow (q_1 \ldots q_r)) = \frac{Count(q_1 \ldots q_r, s)}{Count(s)}$$

can be accurately estimated, since $Count(q_1 \ldots q_r)$ and $Count(s)$ can be answered exactly using our publishing format. We define the reconstruction error as follows:

Definition 4 (Reconstruction Error): The reconstruction error for partitioning $\mathcal{P}$ of transaction set T is:

$$RE(\mathcal{P}) = \sum_{i=1}^{m} KL\_Divergence(Act^{s_i}, Est^{s_i}) \tag{5}$$

We wish to determine the optimal partitioning that satisfies the privacy degree p and minimizes RE. Formally:

Problem Statement. Given a set of transactions T containing items from I, a subset $S \subset I$ of sensitive items, and privacy degree p, determine a partitioning $\mathcal{P}$ of T such that:

(i) $degree(\mathcal{P}) \geq p$ (i.e., privacy requirement is met)
(ii) $\forall \mathcal{P}'$ s.t. $degree(\mathcal{P}') \geq p$, $RE(\mathcal{P}) \leq RE(\mathcal{P}')$

4 Nearest-Neighbor Search Anonymization

In this section, we study the anonymization of transactional data using techniques based on nearest-neighbor (NN) search. As discussed in Section 3.1, anonymization is achieved by creating groups of transactions, such that the privacy requirement is satisfied for all groups (i.e., $\forall G, degree(G) \geq p$). To fulfill the privacy requirement, each sensitive transaction needs to be grouped either with non-sensitive transactions, or with sensitive ones with different sensitive items, such that the frequency of each sensitive item is reduced below $1/p$ in every group. We say that two transactions are *conflicting* if

NNGroup(T)
Input: trans. set T, sens. trans. set $ST \subset T$, privacy degree p
1. initialize histogram $H[1 \dots m]$ (for every $s_i \in S$)
2. $remaining = |T|$
3. **while** ($\neg empty(ST)$) **do**
4. $t = pop\_front(ST)$ /* removes t from ST */
5. $G = \{t\} \cup NonConflictNN(t, T)$
6. update H for each sensitive item in G
7. **if** ($\nexists s | H[s] \cdot p > remaining$)
8. $remaining = remaining - |G|$
9. $T = T \setminus G$, $ST = ST \setminus G$, output G
10. **else**
11. undo modifications to H
12. output not-assigned transactions as a single group

NonConflictNN(t, CL)
/* finds $p-1$ NN of t from candidate list CL */
13. $Chosen = \emptyset$ /* list of neighbors */
14. **for** $i = 1$ **to** $p - 1$
15. $MinDist = \infty$
16. **forall** $t' \in CL \setminus \{t\}$
17. **if** ($\exists t'' \in Chosen \cup \{t\}$ s.t. t' conflicts with t'')
18. **skip**
19. **if** ($dist(t', t) < MinDist$))
20. $MinDist = dist(t', t)$; $CrtNeighbor = t'$
21. $Chosen = Chosen \cup \{CrtNeighbor\}$
22. **return** $Chosen$

Fig. 3. $NNGroup$ anonymization algorithm

they have at least one common sensitive item. To obtain low reconstruction error, we need to group each sensitive transaction with other non-conflicting ones having similar quasi-identifier items. With the binary representation introduced in Section 3, similarity is captured by the Hamming distance among quasi-identifiers. Similar to other techniques for data anonymization, we adopt a *"one-occurrence-per-group"* approach, that allows only one occurrence of each sensitive item in a group. This approach has two advantages: *(i)* it reduces the search space of the problem, and *(ii)* as shown in previous work [6], [15], it reduces information loss.

A natural anonymization strategy is to group each sensitive transaction with its $p-1$ nearest neighbors. In Section 4.1, we present an anonymization technique which relies on exact NN search. Since data are high-dimensional, no efficient indexing can be employed [5], so the NN search requires a linear scan of the dataset. To avoid this performance bottleneck, we propose in Section 4.2 an approximate NN search method based on Locality-Sensitive Hashing (LSH). LSH incurs a significantly lower overhead, and still achieves good data utility in practice.

4.1 Exact NN Search

The $NNGroup$ algorithm (Fig. 3) takes as input dataset T, subset $ST \subset T$ of sensitive transactions, and privacy degree p. For each sensitive transaction t, $NNGroup$ determines $p-1$ transactions that do not conflict with t (or with each other), and are closest to t in terms of quasi-identifier Hamming distance.

The actual NN search is performed by routine $NonConflictNN$ (line 5). $NonConflictNN$ determines the NN of transaction t (specified as parameter) among the transactions in candidate list CL. For $NNGroup$, $CL = T$ because we perform NN search through linear scan. $NonConflictNN$ loops through all transactions in CL in lines $16-20$. Note that, this loop is executed for each one of the $p-1$ neighbors in sequence. We cannot determine all $p-1$ neighbors in one scan of CL, because of the additional condition on conflicts: if all $p-1$ neighbors are computed in one pass, it is possible that some of these transactions conflict with t, or with each other, and a group cannot be formed.

Since $NNGroup$ is just a greedy heuristic for group formation, we must ensure that it always finds a solution, i.e., at any time, forming a group will not yield a remaining set of transactions that can not be further anonymized (for instance, all remaining transactions may be in conflict with each other). For this reason, we maintain a histogram H with the number of remaining occurrences for each sensitive item. H is initialized (line 1) and updated every time a new candidate group is formed (line 6). Upon validating the current candidate group G, we check (line 7) that the remaining set of transactions satisfies the privacy requirement (i.e., $degree(T \setminus G) \geq p$). G is only validated if the requirement is met (line 9), otherwise G is discarded, and the modifications to H are reversed. The algorithm stops when there are no more ungrouped sensitive transactions remaining, or when no new groups can be formed. Note that, if at any point in the $NNGroup$ execution all remaining transactions are non-sensitive, they are published as a single group (no information loss is incurred for this group, since quasi-identifier items are released directly). Furthermore, each $t \in ST$ is used as seed for $NonConflictNN$ at most once, hence at most $|ST|$ executions of the while loop (lines 3-11) occur.

The next theorem proves that $NNGroup$ finds a solution with privacy requirement p, provided that the input T satisfies the privacy requirement (i.e., $degree(T) \geq p$).

Theorem 1: If $degree(T) \geq p$ for a given privacy degree p, then $NNGroup$ will find a solution.

Proof: The proof is by induction: at any point in the execution of $NNGroup$, denote by G_r the group of remaining transactions. Initially, $G_r = T$, which satisfies the privacy requirement. Every new candidate group G created by $NNGroup$ satisfies the privacy requirement by construction. G is only validated if $G'_r = G_r \setminus G$ satisfies the privacy condition $degree(G'_r) \geq p$. If it does, we publish G and continue the algorithm with $G_r = G'_r$. Otherwise, G_r is published directly and $NNGroup$ terminates. In either case, all published groups abide the privacy requirement. Furthermore, the while loop (lines 3-11) is executed at most once per sensitive transaction, therefore $NNGroup$ always terminates, hence a solution is found. □

4.2 Approximate NN Search with LSH

In the previous formulation of $NNGroup$, the entire set T is searched for nearest-neighbor transactions, which is an expensive process. To address this drawback, we consider approximate NN search based on Locality-Sensitive Hashing (LSH) [7]. LSH is a novel approximate similarity search paradigm which has gained widespread acceptance for a variety of applications, such as multimedia databases, gene expression similarity, etc.

For transactional data anonymization, we employ an LSH technique originally introduced in [7], which is specifically tailored for the high-dimensional Hamming space $\mathcal{H}^{|Q|}$, where $|Q|$ is the dimensionality of the quasi-identifier. Consider a query point $q \in \mathcal{H}^{|Q|}$, and denote its exact NN by p. Then, a

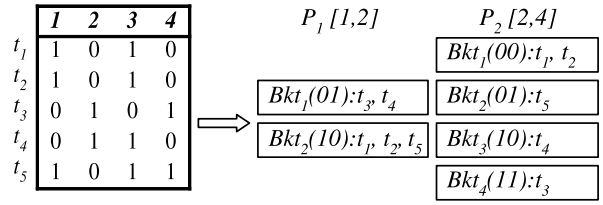

Fig. 4. LSH Example

c-approximate ($c > 1$) nearest-neighbor of q is a point p' such that:

$$dist(q, p') < c \cdot dist(q, p)$$

Nevertheless, the distance to the exact NN is not known in practice. Therefore, a slightly different version of approximate NN search is formally defined in [25] as:

Problem 1 (Randomized Approximate Near Neighbor): Let P be a set of points in $\mathcal{H}^{|Q|}$, and consider query point q, radius $r > 0$, a confidence threshold $0 < \delta < 1$ and approximation bound $c > 1$. If $\exists p \in P$ s.t. $dist(q, p) < r$, then return a point $p' \in P$ s.t. $dist(q, p') < c \cdot r$ with probability at least $1 - \delta$.

LSH solves Problem 1 by employing *locality-sensitive* hash functions:

Definition 5: A hash function $\mathcal{F}$ is called *(r_1, r_2, p_1, p_2)-sensitive* iff $\forall p, q$:

(i) $dist(p, q) \leq r_1 \Rightarrow Prob[\mathcal{F}(p) = \mathcal{F}(q)] > p_1$
(ii) $dist(p, q) > r_2 \Rightarrow Prob[\mathcal{F}(p) = \mathcal{F}(q)] < p_2$

For the transaction anonymization problem, consider the space $\mathcal{H}^{|Q|}$, and the Hamming (i.e., L_1) distance. Denote by $\mathcal{F}_i$ the hash function that projects each transaction on dimension i, and assume we wish to find c-approximate near neighbors for $r = 1$ (i.e., pairs of transactions that differ in one item). The probability for a pair of transactions at distance 1 to be hashed to the same value is $p_1 = \frac{|Q|-1}{|Q|}$ (the only case when they hash to distinct values occurs when they differ in bit i, which happens with probability $1/|Q|$). Similarly, $p_2 = 1 - \frac{c}{|Q|}$. Therefore, the family of hash functions $\mathcal{F}_i$ ($1 \leq i \leq |Q|$) is $(1, c, 1 - \frac{1}{|Q|}, 1 - \frac{c}{|Q|})$-sensitive [7].

To support arbitrary p_1 and p_2, a probabilistic dimensionality reduction can be performed by projecting data along a random set of w dimensions, where $w < |Q|$. Each projection has the form $\{\mathcal{F}_{i_1}, \ldots, \mathcal{F}_{i_w}\}$ with $i_1, \ldots, i_w \in \{1, \ldots, |Q|\}$. By selecting a number of B such projections, each with its own associated hash table, the search accuracy can be tuned. In particular, condition *(i)* of Def. 5 relates to the accuracy of the search, whereas condition *(ii)* bounds the space requirements for the stored hash-tables. For instance, it is shown in [25] that to obtain a r-near neighbor with probability at least $1 - \delta$ for any $c > 1$, the number of distinct hash tables must be

$$B \geq \frac{\log(1/\delta)}{-\log(1 - p_1^w)}, \tag{6}$$

where p_1 (Def. 5) is the probability of hashing two points to the same bucket in each individual hash table.

The transaction set T in Fig. 4 corresponds to the example in Fig. 1a. The quasi-identifier dimensionality is $|Q| = 4$. Consider a search for a near-neighbor transaction of $t_2 = 1010$, $r = 1$ and $c = 1.1$. The result will be any of t_1 or t_5. On the other hand, no near-neighbor exists for transaction t_3 and the same r and c parameter settings, since the closest transaction to t_3 is t_4 at distance 2. Assume there are $B = 2$ different projections, each on a subset of $w = 2$ dimensions. The first projection, P_1, projects onto dimensions 1 and 2, whereas P_2 projects onto dimensions 2 and 4. As a result, two hash tables are obtained, one with two buckets, and the second with four. For instance, bucket Bkt_1 on projection P_1 contains all transactions that have projection 0 on the first dimension and 1 on the second.

Build_LSH_HashTable($T, P_1, \cdots, P_B$)
1. **for** ($i = 1$ **to** B) initialize hash table $HT[i]$
2. **forall** $t \in T$
3. **for** ($i = 1$ **to** B)
4. key = projection of t on P_i
5. insert($HT[i], key, t$)

ApproxNonConflictNN(t, T)
6. $LSHResult = \emptyset$
7. **for** ($i = 1$ **to** B)
8. key = projection of t on P_i
9. $LSHResult = LSHResult \cup find(HT[i], key)$
10. $G = \{t\} \cup NonConflictNN(t, LSHResult)$
11. **if** ($|G| < p$)
12. $G = \{t\} \cup NonConflictNN(t, T)$
13. **return** G

Fig. 5. LSH Nearest Neighbor Search

Fig. 5 outlines the steps required to use LSH in the anonymization process. In a pre-processing stage, the $Build\_LSH\_HashTable$ routine is invoked, which hashes the data into a set of B hash tables $HT[1], \cdots, HT[B]$, corresponding to projections $P_1, \ldots, P_B$. The index key for transaction t in hash table $HT[i]$ is the projection of $t[Q]$ on P_i (recall that $t[Q]$ denotes the quasi-identifier of t). Next, the $NNGroup$ algorithm in Fig. 3 is modified by replacing line 5 with

$$G = \{t\} \cup ApproxNonConflictNN(t, T)$$

The $ApproxNonConflictNN(t, T)$ routine (Fig. 5) performs an approximate search for the NN of t. First, it narrows down the search set (lines $7 - 9$) only to those transactions that are hashed in the same bucket as t in at least one of the B hash tables HT. Due to the good hashing properties of LSH [7], $|LSHResult|$ is considerably smaller than $|T|$, and the search performance is considerably improved. Next, (line 10) the exact NN routine $NonConflictNN$ (Fig. 3) is invoked, but with $LSHResult$ as argument. Note that, the approximate search may not find enough non-conflicting neighbors within $LSHResult$. If such a situation occurs (line 12), $NonConflictNN$ is invoked again, on the entire transaction set T. In other words, the algorithm reverts back to exact NN search within the entire T (in time linear to $|T|$). Nevertheless, this case rarely occurs in practice, as LSH has high probability of hashing transactions with close-by quasi-identifiers in the same bucket, in at least one of the B hash tables.

Note that, in the transaction anonymization setting, NN-search is just a heuristic to create groups. Therefore, obtaining strict bounds on the approximate search is not an essential requirement. Furthermore, due to data sparseness, it may

Reverse Cuthill-McKee (RCM) Algorithm
Input: graph $\mathcal{AG}(V, E)$ with adjacency matrix A
1. pick peripheral vertex $v \in V$
2. $R = \{v\}$
3. $PrevLevel = R$
4. **while** $|R| < |V|$ **do**
5. $CrtLevel = \emptyset$
6. **for** $i = 1$ *to* $|PrevLevel|$ **do**
7. $Tmp = \{v \in V | v \notin R \wedge dist(PrevLevel[i], v) = 1\}$
8. sort Tmp in increasing order of vertex degree
9. append Tmp to $CrtLevel$
10. append $CrtLevel$ to R
11. $PrevLevel = CrtLevel$
12. output R in reverse order

Fig. 6. Reverse Cuthill-McKee Algorithm

be difficult to find r and c values that work well for all query points. Fortunately, as acknowledged in [7], the LSH search is robust to these parameters. Even though theoretical bounds may not be met for all query points, adequate search performance can be achieved in practice with a choice of parameters w and B that depends on the data characteristics, but is independent of the query point. We defer the choice of w and B for the experimental evaluation in Section 6.

4.3 Discussion of NN-based Anonymization Methods

The NN-based techniques introduced in this section create anonymized groups by searching for the nearest neighbors of sensitive transactions. NN search has the advantage of maximizing the quality of each individual group. On the other hand, no global objective is considered, in the sense that the quality of groups other than the current group is completely disregarded. Since the NN methods are localized in nature, their performance depends on the order of processing sensitive transactions, and poor results may be obtained if a particularly unfavorable ordering is chosen. In the next section, we present another category of anonymization methods based on global optimization.

5 Data Re-organization Methods

In this section, we study methods that re-organize the data by placing transactions with similar quasi-identifier in close proximity to each other. We propose two such methods: *(i)* in Section 5.1 we consider a technique that interprets the transactional data as a sparse matrix, which is then transformed in order to reduce its bandwidth *(ii)* in Section 5.2, we devise a method based on the binary reflected Gray encoding, which sorts the data such that the Hamming distance between successive transactions is reduced. Finally, in Section 5.3 we propose *Correlation-aware Anonymization of High-dimensional Data (CAHD)* - a heuristic that takes as input the data organization produced by any of the methods in Sections 5.1 or 5.2, and greedily creates groups with transactions close to each other in the sequence.

5.1 Reduction to Band Matrix Representation

As discussed in Section 3.2, in order to minimize the reconstruction error, it is necessary to group together transactions

Asymmetric Matrix Bandwidth Reduction
Input: asymmetric matrix A
1. compute symmetric matrix $A \times A^T$
2. δ = Reverse Cuthill-McKee($A \times A^T$)
3. A' = permutation δ applied to A
4. output A'

Fig. 7. Asymmetric Matrix Bandwidth Reduction

with similar QID. We organize the data (i.e., matrix A) as a *band matrix*, so that consecutive rows are likely to share a large number of common items. Band matrix organization has been acknowledged as a beneficial mode to represent sparse data in various scientific applications [26]. A band matrix has all elements **0**, except for the main diagonal d_0, U upper diagonals ($d_1 \ldots d_U$), and L lower diagonals ($d_{-1} \ldots d_{-L}$). Our objective is to minimize the total bandwidth $\mathcal{BW} = U + L + 1$. A simple Gaussian elimination algorithm can be employed to obtain an upper or lower *triangular* matrix, where $L = 0$ or $U = 0$, respectively. However, finding an optimal band matrix (i.e., with minimum $\mathcal{BW}$) is an NP-complete problem [27].

The Reverse Cuthill-McKee Algorithm. A general matrix can be transformed into a band matrix by performing permutations of rows and columns. Multiple heuristics have been proposed to obtain band matrices with low bandwidth. The most prominent is the *Reverse Cuthill-McKee (RCM)* algorithm [26]. RCM works for square, symmetric matrices. Given sparse matrix A, it builds adjacency graph $\mathcal{AG} = (V, E)$, where V contains one vertex for each matrix row, and there is an edge from vertex v_i to vertex v_j for every non-zero element $A[i][j]$. If A is symmetric, then $\mathcal{AG}$ is undirected. RCM is based on the observation that a permutation of rows of A corresponds to a re-labeling of vertices for $\mathcal{AG}$. Given permutation δ of V (i.e., a bijective application $\delta : \{1 \ldots |V|\} \rightarrow \{1 \ldots |V|\}$), the bandwidth of $\mathcal{AG}$ (and of matrix A with rows permuted according to δ) is

$$\mathcal{BW}(\mathcal{AG}) = max\{|\delta(v_1) - \delta(v_2)| : (v_1, v_2) \in E\}$$

To determine a permutation that reduces $\mathcal{BW}$, RCM performs a breadth-first (BFS) traversal starting from an initially chosen *root* node. All nodes (i.e., rows) at the same distance from the root in the traversal constitute a *level set*. At each step, the vertices in the same level sharing the same parent are sorted increasingly according to vertex degree. By reversing the obtained order, we find the permutation that needs to be applied to the rows of matrix A. The choice of the root node is crucial for the effectiveness of the transformation; usually, the root is determined by finding a pseudo-diameter of the graph (through an iterative process linear in the number of vertices) and choosing one of the ends. Fig. 6 shows the RCM algorithm. The computational complexity of RCM is $O(|V| D \log D)$, where D is the maximum degree of any vertex in the adjacency list.

Bandwidth Reduction for Asymmetric Matrices. RCM addresses the case of symmetric matrices only. The work in [26] investigates approaches to reduce the bandwidth of asymmetric matrices, based on the RCM algorithm. Two methods can be employed to achieve this. Given matrix A, they apply RCM to one of the following symmetric matrices: *(i)* $A + A^T$ and *(ii)* $A \times A^T$. The obtained permutation δ is then applied to A.

Binary Code	Gray Code	Gray Code Bits		
		Bit 3	Bit 2	Bit 1
000	000	0	0	0
001	001	0	0	1
010	011	0	1	1
011	010	0	1	0
100	110	1	1	0
101	111	1	1	1
110	101	1	0	1
111	100	1	0	0

Cycle Length 4 (Bit 2, rows 010–101); Cycle Length 2 (Bit 1, rows 011–100); Cycle Length 4 (Bit 3, rows 100–111)

Fig. 8. Gray Encoding of a three-dimensional Hamming space

GraySort(T)
Input: transaction set T
1. **foreach** $t \in T$
2. $\quad t'[Q] = compute\_\mathcal{G}^{-1}(t[Q])$
3. sort T increasingly on key $t'[Q]$
4. output T

compute $\mathcal{G}(t')$
Input: transaction t' in natural binary code
5. $t[|Q|] = t'[|Q|]$
6. **for** $i = |Q| - 1$ **downto** 1 **do** $t[i] = t'[i+1] \oplus t'[i]$
7. **return** t

compute $\mathcal{G}^{-1}(t)$
Input: transaction t in Gray binary code
8. $t'[|Q|] = t[|Q|]$
9. **for** $i = |Q| - 1$ **downto** 1 **do** $t'[i] = t'[i+1] \oplus t[i]$
10. **return** t'

Fig. 9. GraySort Algorithm

Method *(i)* is suitable in cases where A is almost symmetric. Method *(ii)* can be applied to any arbitrary matrix; the trade-off is higher execution time. Computing $A \times A^T$ incurs an additional overhead, but the quality of the solution is much better (i.e., the resulting bandwidth is considerably smaller). Fig. 7 shows the pseudocode of the band reduction algorithm for asymmetric matrices. Note that, we are not performing a full matrix multiplication since we are only interested in non-zero entries.

Although A is sparse, $A \times A^T$ can have a large number of non-zeros, leading to prohibitive memory requirements. Storing the matrix on disk may incur a high I/O overhead, limiting the applicability of RCM.

5.2 Gray Code Sorting

We consider another data re-organization technique that preserves data correlation, and relies on the reflected binary Gray code [8]. The Gray code is a binary encoding system widely used for error correction in digital communications. The most important property of the encoding is that successive values differ in only one bit.

Consider the $|Q|$-dimensional Hamming space $\mathcal{H}^{|Q|} = \{0, 1, \ldots 2^{|Q|}-1\}$. The Gray encoding is a bijective application $\mathcal{G} : \mathcal{H}^{|Q|} \rightarrow \mathcal{H}^{|Q|}$ with the following property:

$$\forall x, y \in \mathcal{H}^{|Q|} \; s.t \; (y = x + 1) \mod 2^{|Q|} \Rightarrow dist(\mathcal{G}(x), \mathcal{G}(y)) = 1, \quad (7)$$

where $dist$ signifies the Hamming distance (the modulo operation handles the wrap-around of the encoding space). Fig. 8 gives an example Gray encoding for a 3-bit Hamming space. For clarity, we refer to the conventional encoding of $\mathcal{H}^{|Q|}$ as *natural* binary encoding, as opposed to the Gray binary encoding.

Given any two distinct elements in $\mathcal{H}^{|Q|}$, the distance between them is at least 1. Therefore, the Gray encoding provides an optimal ordering of $\mathcal{H}^{|Q|}$, if the objective is to minimize the distance between any two consecutive elements in the order. For this reason, the Gray encoding is very suitable for our problem.

The projection of transaction set T on quasi-identifier set Q is a subspace of $\mathcal{H}^{|Q|}$. If every element in $\mathcal{H}^{|Q|}$ belongs to T, then all consecutive transactions in the sequence will have high correlation of items. Furthermore, since $\mathcal{G}$ is bijective, identical transactions will always be consecutive in the Gray ordering[1], facilitating the inclusion of transactions with identical QID in the same anonymized group.

For each transaction $t \in T$, we interpret its quasi-identifier $t[Q]$ as the Gray-encoded value of another $|Q|$-bit binary number $t'[Q]$. We can obtain $t'[Q]$, the natural binary encoding of $t[Q]$, by applying the inverse Gray encoding transformation[2], i.e., $t'[Q] = compute\_\mathcal{G}^{-1}(t[Q])$. It results from Eq.(7) that, if we sort the transactions in T according to the natural binary encoding value $t'[Q]$, then the Hamming distance between the Gray representations $t[Q]$ of consecutive transactions will be low. This facilitates the creation of anonymized groups with low reconstruction error.

In the example of Fig. 1a, we have $|Q| = 4$ and $Bob[Q] = 1010$, $David[Q] = 1010$, $Claire[Q] = 0101$, $Andrea[Q] = 0110$ and $Ellen[Q] = 1011$. Using $compute\_\mathcal{G}^{-1}$, we determine that $Bob'[Q] = 1100$, $David'[Q] = 1100$, $Claire'[Q] = 0110$, $Andrea'[Q] = 0100$ and $Ellen'[Q] = 1101$. After sorting, the new order of transactions is $Andrea$, $Claire$, Bob, $David$, $Ellen$, as shown in Fig. 1c.

Fig. 9 presents the pseudocode of the Gray data re-organization process, called $GraySort$: first (lines 1-2), the sort key $t'[Q]$ of each transaction is determined, using the $compute\_\mathcal{G}^{-1}$ routine (note that, since $\mathcal{G}$ is bijective, there exists its inverse $\mathcal{G}^{-1}$). Next, the set T is sorted increasingly according to the value of $t'[Q]$. The $compute\_\mathcal{G}^{-1}$ routine has complexity $O(|Q|)$. The overall complexity of the $GraySort$ algorithm is $O(n \log n + n|Q|)$.

If the set T coincides with $\mathcal{H}^{|Q|}$, then, according to Eq.(7), all pairs of consecutive transactions will have Hamming distance 1. However, in practice, the high-dimensional data space $\mathcal{H}^{|Q|}$ is sparsely populated, and there will be gaps in the sorted order of Gray-encoded transactions. Therefore, the Hamming distance between consecutive Gray-encoded values may be larger than 1. In other applications of Gray codes, such as data compression [22], it has been shown that the effectiveness of the encoding relies on certain properties such as *cycle length*. The Gray code bits exhibit a cyclical behavior, with a cycle length that increases with the bit position. For instance, in Fig. 8 the least-significant bit (bit 1) has a cyclical pattern of

1. Note that, the RCM algorithm does not guarantee this property
2. Details about the direct ($compute\_\mathcal{G}$) and inverse ($compute\_\mathcal{G}^{-1}$) Gray encoding routines can be found in [8]

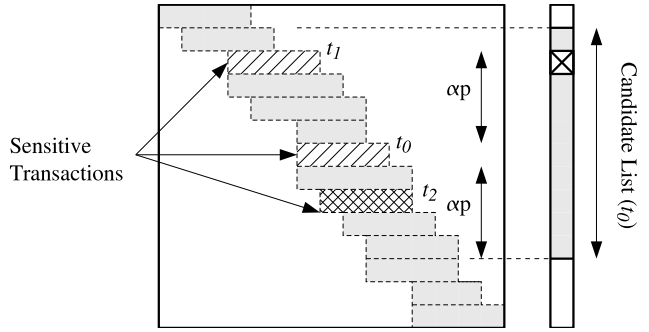

Fig. 10. Group Formation Heuristic

('0','0',1','1', $\cdots$), with length 2. On the other hand, the cycle length of the bit 2 is 4. In general, the cycle length doubles with each bit position (except for the most significant bit which has the same cycle length as the second-most significant bit).

The fact that the transaction set does not completely cover the $\mathcal{H}^{|Q|}$ space determines a change in the cyclical patterns of the encoding. In particular, the ordering of dimensions (i.e., items) in the data influences considerably the bit patterns across consecutive transactions, hence the quality of data grouping. To obtain a sequence with high correlation among consecutive transactions, we sort the transaction bits (i.e., items) such that the most frequently occurring '1' bits correspond to the most significant bits in the binary representation. This procedure compacts the gaps that are left in the Gray ordering of transactions. In our implementation, we initially arrange items in decreasing order of frequency of occurrence, before applying the $GraySort$ algorithm.

5.3 Anonymized Group Formation

Once the data are re-organized, the next step is to create anonymized groups of transactions. We propose *CAHD (Correlation-aware Anonymization of High-dimensional Data)*, a greedy heuristic that capitalizes on the data correlation, and groups together transactions that are close-by in the data sequence.

Consider the example in Fig. 10, where sensitive transactions are marked by hatching. CAHD scans transaction set T in row order, finds the first sensitive transaction in the sequence, and attempts to form an anonymizing group for it. In our example, t_0 and t_1 are conflicting (see Section 4 for the definition of conflicting transactions), and are represented with identical hatching. t_2 is not in conflict with any of t_0 or t_1. Assume that we want to anonymize t_0 with privacy degree p. In this case, we need to create group G containing t_0 and at least other $p-1$ transactions, such that $degree(G) \geq p$.

CAHD works as follows: given sensitive transaction t_0, a *candidate list (CL)* is formed, with the αp transactions that precede, respectively follow t_0, and are not in conflict with t_0 or with each other (conflicting transactions are "skipped" when building CL). $\alpha \in \mathbb{N}$ is a system parameter which restricts the range of the search. Intuitively, the larger α, the better the chance to include in CL transactions with similar items, but at increased execution time. Nevertheless, as we show in Section 6, even a low α can yield good results, because of the effective data organization. In the example, t_1 is excluded from $CL(t_0)$, and its predecessor (which is non-sensitive, hence not in conflict with t_0) is included. Then, out of the $2\alpha p$ transactions in $CL(t_0)$, the $p-1$ of them that have the lowest Hamming distance to t_0 (i.e., the largest number of QID items in common with t_0) are chosen to form an anonymized group. The intuition is that, if more transactions share the same QID, the reconstruction error becomes smaller (see eq. (4)). All selected transactions are then removed from T and the process continues with the next sensitive transaction in the order. Fig. 11 gives the pseudocode of CAHD.

As in the case of $NNGroup$ (Section 4.1), a histogram H with the number of occurrences of each sensitive item is maintained for the remaining transactions, to ensure that a solution is found. Theorem 1 applies to CAHD as well.

To implement efficiently the assembly of CL (i.e., find αp non-conflicting transactions) we can employ a linked-list data representation, where each transaction (list entry) points to its predecessor and successor with a particular sensitive item. The space requirement is $O(m)$ (where $m = |S|$, i.e., constant) per transaction. The computational complexity of CAHD is $O(\alpha pn)$, since the while loop (lines 3-12) is executed at most n times (once per sensitive transaction), and each iteration considers at most $2\alpha p$ other transactions.

6 Experimental Evaluation

The proposed techniques are evaluated using a workload that consists of two representative real-world datasets, introduced in [28]. *BMS-WebView-1 (BMS1)* and *BMS-WebView-2 (BMS2)* represent several months of transaction logs corresponding to two on-line retailers[3]. Their characteristics are presented in Table 1. All experiments were executed on a Pentium 4 2.0GHz machine with 1GB of RAM running Linux 2.6 OS.

In the experiments, the degree of privacy p is varied in the range $4-20$. The number of sensitive items m (i.e., cardinality of S) is randomly chosen between 5 and 20, out of the entire set of items I. We consider group-by queries as discussed in Section 3.2, with the number r of QID items in the group-by clause varying between 2 and 8. For each (p, m, r) triplet setting, 10 group-by queries are generated by randomly selecting s and $q_1 \ldots q_r$, and the average reconstruction error is measured (i.e., KL-divergence, with ideal value 0). All results are averaged over 10 distinct random seeds. For CAHD,

3. These datasets have been used as benchmarks in KDD-Cup 2000

CAHD Group Formation Heuristic

Input: trans. set T, sens. trans. set $ST \subset T$, privacy degree p
1. initialize histogram $H[1 \ldots m]$ (for every $s_i \in S$)
2. $remaining = |T|$
3. **while** $(\neg empty(ST))$ **do**
4. $t = pop\_front(ST)$ /* removes t from ST */
5. $CL(t)$ = non-confl. αp pred. and αp succ. of t
6. $G = \{t\} \cup p-1$ trans. in $CL(t)$ closest to t
7. update H for each sensitive item in G
8. **if** $(\nexists s | H[s] \cdot p > remaining)$
9. $remaining = remaining - |G|$
10. $T = T \setminus G$, $ST = ST \setminus G$, output G
11. **else**
12. undo modifications to H
13. output not-assigned transactions as a single group

Fig. 11. CAHD Pseudocode

TABLE 1
Dataset Characteristics

	# Trans.	# Items	Max. length	Avg. length
BMS1	59,602	497	267	2.5
BMS2	77,512	3,340	161	5.0

TABLE 2
Experimental Parameter Values

Notation	Description	Values
p	privacy degree	4,6,8,**10**,15,20
m	sensitive items	5,**10**,15,20
r	QID items in group-by clause	2,**4**,6,8

parameter α is set to 1 (see Section 5.3). Table 2 summarizes the parameters used in the experiments; default values are shown in bold face.

TABLE 3
Percentage of Re-identified Transactions

	Number of Known QID Items			
Dataset	**1**	**2**	**3**	**4**
BMS1	0.3%	9.5%	24.3%	50.0%
BMS2	0.8%	18.8%	41.6%	91.1%

6.1 Measurement of Re-identification Risk

We reinforce our motivation from Section 1 by measuring the percentage of transactions that can be uniquely identified based on a number of known QID items. Table 3 shows the percentage of re-identified transactions based on a randomly chosen set of QID items, with cardinality varying between 1 and 4. For the sparser BMS2 dataset, 91% of transactions can be uniquely identified based on four known items.

6.2 Evaluation of Anonymization Methods

We evaluate the reconstruction error and execution time of the proposed methods: CAHD in conjunction with RCM (label RCM), CAHD in conjunction with GraySort (label Gray), as well as the nearest-neighbor search method, for which we consider both the exact NN method (label NN) and the faster LSH variant (label LSH). As observed in [7], the LSH parameter settings are robust over different datasets: we choose a value of $w = 32$ projection dimensions (i.e., the hash signature has 32 bits and can fit in one machine word). Considering a conservative probability $p_1 = 0.9$ (see Def. 5)[4] and an accuracy of search of 90% (i.e., $1 - \delta$), the required number of distinct hash tables is $B = 32$.

As a competitor method, we consider a hybrid approach which combines the strengths of the current state-of-the-art: Mondrian [4] and Anatomy [6]. Mondrian is a generalization method: it recursively divides the dataset, based on QID values, until the privacy requirement does not allow any more splits. Mondrian preserves locality in the QID space, but because it generalizes the QID values within each group, it may incur considerable information loss. On the other hand, *Anatomy* is a permutation approach, which publishes directly QID values. This is beneficial for data utility, as it reduces reconstruction error; however, in the process of group formation, Anatomy does not account for QID proximity, therefore it may not preserve correlations well. We compare against a combined method we denote by *PermMondrian (PM)*: similar to Mondrian, PM partitions the dataset according

4. For the $|Q|$ values of the dataset, p_1 will always exceed this threshold for most choices of r_1

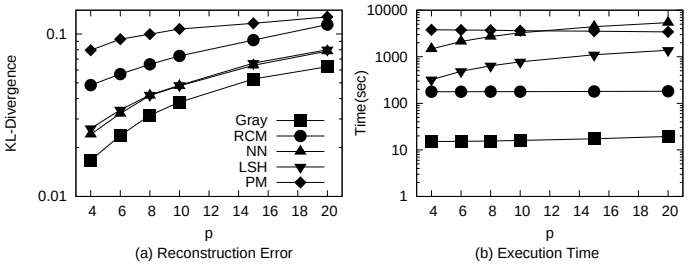

Fig. 12. BMS1 dataset, variable p ($r = 4$, $m = 10$)

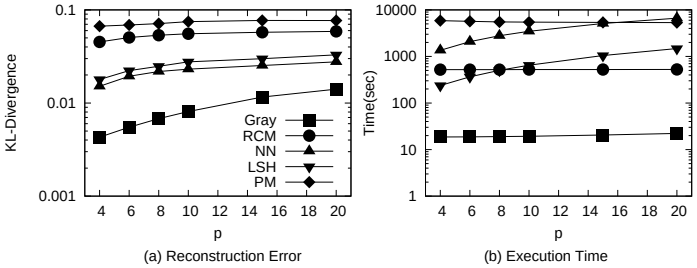

Fig. 13. BMS2 dataset, variable p ($r = 4$, $m = 10$)

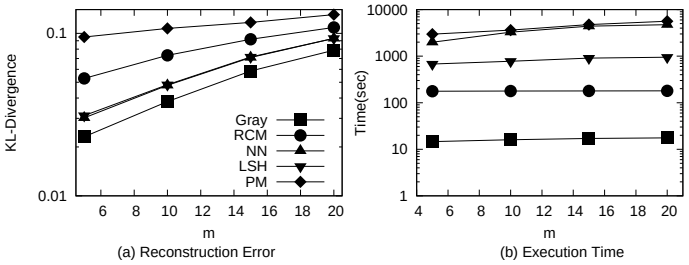

Fig. 14. BMS1 dataset, variable m ($p = 10$, $r = 4$)

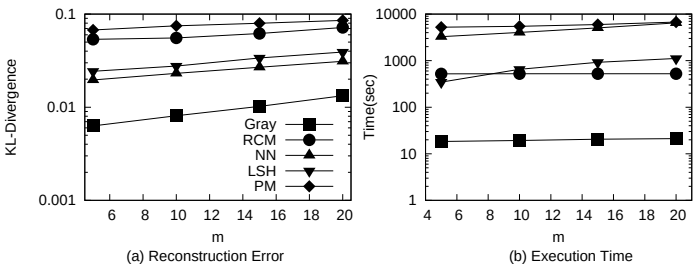

Fig. 15. BMS2 dataset, variable m ($p = 10$, $r = 4$)

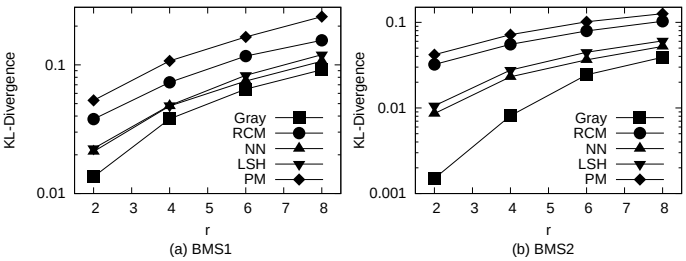

Fig. 16. Reconstruction Error vs r ($p = 10$, $m = 10$)

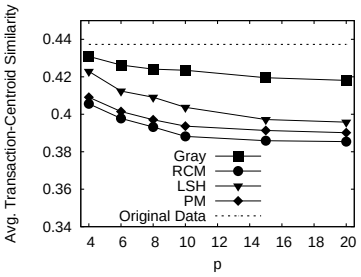

Fig. 17. Clustering Accuracy vs p, BMS1 dataset

to QID proximity. Nevertheless, it publishes exact QID values, instead of generalizing them. Furthermore, an enhanced split heuristic is used, compared to the original one in [4] which considered only group cardinality. To allow PM to obtain fine-grained groups with low reconstruction error, the distribution of sensitive items in the resulting groups is also taken into account, and splits with a balanced distribution of sensitive items are favored. Doing so increases the success probability of subsequent split attempts. Otherwise, many transactions with common sensitive items could be grouped together, increasing the frequency of a single item in a group, hence disallowing further splits.

First, the privacy degree p is varied. For the BMS1 dataset, Fig. 12a shows that Gray outperforms all other methods in terms of reconstruction error. NN and LSH obtain similar reconstruction error, which is lower than that of RCM. PM is the worst of all methods, since it can not handle high-dimensional data. As expected, a higher privacy degree p increases the reconstruction error for all methods, i.e., reduces data utility. Fig. 12b shows the execution time of the studied methods. The reported time includes the data organization phase for Gray and RCM. In addition to being the best in terms of reconstruction error, Gray is also the fastest method. RCM is slower due to the overhead of the transformation to band matrix. For both RCM and Gray, the data re-organization phase is the most costly, whereas the actual group formation phase (CAHD) requires at most 5 seconds. The nearest-neighbor methods LSH and NN are slower than Gray by roughly one and two orders of magnitude, respectively. LSH improves on NN in execution time by a considerable factor of up to 6 times. The execution time of PM is very large, since the algorithm needs to evaluate the benefit of a potential split in each dimension.

Fig. 13 shows similar reconstruction error and execution time trends for the BMS2 dataset. The advantage of Gray compared to other methods is more pronounced in terms of reconstruction error, which suggests that Gray is more resilient as data dimensionality increases. In terms of execution time, LSH manages to outperform RCM for low p values. Recall from Section 5.1 that, in addition to execution time, RCM may also incur a high memory overhead: for instance, applying RCM to BMS2 requires close to 1GB of memory. The memory requirements of the other methods are not significant.

In the next experiment, the number of sensitive items m is varied. Fig. 14 shows the results obtained for the BMS1 dataset. Gray maintains its superiority compared to the other methods, in terms of both reconstruction error and execution time. The relative performance of the other methods is preserved, with RCM being faster than the nearest-neighbor techniques, but worse in terms of reconstruction cost. Fig. 15 shows similar trends for the BMS2 dataset. Note that, the execution time of LSH is not considerably higher than RCM. Furthermore, the gap between RCM and LSH increases only slightly with m, since the overhead of the nearest-neighbor search depends mainly on the privacy parameter p. PM is the slowest method, and it also yields the highest reconstruction error.

In Fig. 16c, we vary parameter r, the number of quasi-identifier items in the query group-by clause. Since query processing is performed outside the anonymization process, r has no effect on execution time. Therefore, only reconstruction error is reported. The results show that Gray maintains its advantage in this case as well. Note that, the reconstruction error gap decreases considerably as r increases, since all methods are penalized by the higher group-by dimensionality.

Finally, we measure the accuracy of the studied anonymization methods for a clustering task, in a similar manner to [19]. Specifically, we perform a k-means clustering ($k = 20$) and we report the average similarity (measured by cosine distance) between a transaction and its cluster center. Fig. 17 shows the results (we omit NN results which are virtually identical to LSH). As expected, accuracy decreases as p increases since anonymization introduces more data distortion. Nevertheless, the obtained accuracy is high: in the worst case, the accuracy of Gray is 95% of that obtained with the original data. Interestingly, PM performs marginally better than RCM. Recall that, PM partitions the dataspace of transactions, effectively limiting the extent of anonymized groups. As a result, the average cluster extent may be smaller than for RCM, which only factors Hamming distance in group formation.

7 Conclusions

In this paper, the problem of anonymizing sparse, high-dimensional transactional data is solved through methods based on *(i)* local NN-search and *(ii)* global data re-organization. All proposed approaches outperform the existing state-of-the-art, which does not handle well high data dimensionality. LSH-based anonymization outperforms the RCM method in terms of data utility, but incurs slightly higher computational overhead, whereas Gray-code sorting is superior to all other methods with respect to both data utility and anonymization overhead.

The necessity of anonymizing transactional data has been recently emphasized with the release of the "Netflix Prize" data, containing movie ratings of $500,000$ subscribers. A

recent study [29] shows that an attacker can re-identify 80% of the subscribers based on knowledge about 6 reviewed movies. In future work, we plan to address the problem of anonymization of high-dimensional data for non-binary databases. Another direction is to employ dimensionality-reduction techniques for more effective anonymization.

ACKNOWLEDGMENTS

This article is an extended version of [1]. The research of Yufei Tao was supported by grants GRF 1202/06, 4161/07, 4173/08, 4169/09 from the RGC of HKSAR, and a grant with project code 2050395 from CUHK.

REFERENCES

[1] G. Ghinita, Y. Tao, and P. Kalnis, "On the Anonymization of Sparse, High-Dimensional Data," in *Proc. of ICDE*, 2008, pp. 715–724.
[2] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-Diversity: Privacy Beyond k-Anonymity," in *Proc. of ICDE*, 2006.
[3] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-domain k-Anonymity," in *Proc. of ACM SIGMOD*, 2005, pp. 49–60.
[4] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-Anonymity," in *Proc. of ICDE*, 2006.
[5] C. C. Aggarwal, "On k-Anonymity and the Curse of Dimensionality," in *Proc. of VLDB*, 2005, pp. 901–909.
[6] X. Xiao and Y. Tao, "Anatomy: Simple and Effective Privacy Preservation," in *Proc. of VLDB*, 2006, pp. 139–150.
[7] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing," in *Proc. of VLDB*, 1999, pp. 518–529.
[8] S. Skiena, *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Addison-Wesley, 1990.
[9] B.-C. Chen, K. LeFevre, and R. Ramakrishnan, "Privacy skyline: privacy with multidimensional adversarial knowledge," in *Proc. of VLDB*, 2007, pp. 770–781.
[10] R. Agrawal and R. Srikant, "Privacy Preserving Data Mining," in *Proc. of ACM SIGMOD*, 2000, pp. 439–450.
[11] Z. Huang, W. Du, and B. Chen, "Deriving Private Information from Randomized Data," in *Proc. of ACM SIGMOD*, 2005, pp. 37–48.
[12] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Workload-aware Anonymization," in *Proc. of KDD*, 2006, pp. 277–286.
[13] P. Samarati, "Protecting Respondents' Identities in Microdata Release." *IEEE TKDE*, vol. 13, no. 6, pp. 1010–1027, 2001.
[14] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu, "Achieving Anonymity via Clustering," in *Proc. of ACM PODS*, 2006, pp. 153–162.
[15] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis, "Fast Data Anonymization with Low Information Loss," in *Proc. of VLDB*, 2007, pp. 758–769.
[16] Q. Zhang, N. Koudas, D. Srivastava, and T. Yu, "Aggregate Query Answering on Anonymized Tables," in *Proc. of ICDE*, 2007, pp. 116–125.
[17] M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi, "Anonymity Preserving Pattern Discovery," *VLDB Journal*, pp. 703–727, 2008.
[18] V. Verykios, A. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, "Association Rule Hiding," *IEEE TKDE*, vol. 16, no. 4, pp. 434–447, 2004.
[19] C. C. Aggarwal and P. S. Yu, "On Privacy-Preservation of Text and Sparse Binary Data with Sketches," in *SIAM Conference on Data Mining*, 2007.
[20] M. Terrovitis, N. Mamoulis, and P. Kalnis, "Privacy-preserving Anonymization of Set-valued Data," in *Proc. of VLDB*, 2008.
[21] Y. Xu, K. Wang, A. W.-C. Fu, and P. S. Yu, "Anonymizing Transaction Databases for Publication," in *Proc. of SIGKDD*, 2008, pp. 767–775.
[22] D. Richards, "Data Compression and Gray-code Sorting," *Information Processing Letters*, vol. 22, pp. 201–205, 1986.
[23] A. Pinar, T. Tao, and H. Ferhatosmanoglu, "Compressing Bitmap Indices by Data Reorganization," in *Proc. of ICDE*, 2005, pp. 310–321.
[24] D. Kifer and J. Gehrke, "Injecting Utility into Anonymized Datasets," in *Proc. of ACM SIGMOD*, 2006, pp. 217–228.
[25] A. Andoni, M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, 2006.
[26] J. K. Reid and J. A. Scott, "Reducing the Total Bandwidth of a Sparse Unsymmetric Matrix," *SIAM J. Matrix Anal. Appl.*, vol. 28, no. 3, pp. 805–821, 2006.
[27] C. Papadimitriou, "The NP-Completeness of the Bandwidth Minimization Problem," *Computing*, vol. 16, pp. 263–270, 1976.
[28] Z. Zheng, R. Kohavi, and L. Mason, "Real World Performance of Association Rule Algorithms," in *Proc. of KDD*, 2001, pp. 401–406.
[29] A. Narayanan and V. Shmatikov, "How To Break Anonymity of the Netflix Prize Dataset," http://arxiv.org/abs/cs/0610105.