Anonymizing data with relational and transaction attributes

Giorgos Poulis[1], Grigorios Loukides[2], Aris Gkoulalas-Divanis[3], and Spiros Skiadopoulos[1]

[1] University of Peloponnese {poulis, spiros}@uop.gr
[2] Cardiff University g.loukides@cs.cf.ac.uk
[3] IBM Research - Ireland arisdiva@ie.ibm.com

Abstract. Publishing datasets about individuals that contain both relational and transaction (i.e., set-valued) attributes is essential to support many applications, ranging from healthcare to marketing. However, preserving the privacy and utility of these datasets is challenging, as it requires *(i)* guarding against attackers, whose knowledge spans both attribute types, and *(ii)* minimizing the overall information loss. Existing anonymization techniques are not applicable to such datasets, and the problem cannot be tackled based on popular, multi-objective optimization strategies. This work proposes the first approach to address this problem. Based on this approach, we develop two frameworks to offer privacy, with bounded information loss in one attribute type and minimal information loss in the other. To realize each framework, we propose privacy algorithms that effectively preserve data utility, as verified by extensive experiments.

1 Introduction

Privacy-preserving data mining has emerged to address privacy concerns related to the collection, analysis, and sharing of data and aims at preventing the disclosure of individuals' private and sensitive information from the published data. Publishing datasets containing both relational and transaction attributes, *RT-datasets* for short, is essential in many real-world applications. Several marketing studies, for example, need to find product combinations that appeal to specific types of customers. Consider the RT-dataset in Fig. 1a, where each record corresponds to a customer. Age, Origin and Gender are relational attributes, whereas Purchased-products is a transaction attribute that contains a *set* of *items*, representing commercial transactions. Such studies may require finding all customers below 30 years old who purchased products E and F. Another application is in healthcare, where several medical studies require analyzing patient demographics and diagnosis information together. In such RT-datasets, patients features (e.g., demographics) are modeled as relational attributes and diagnosis as a transaction attribute. In all these applications, the privacy protection of data needs to performed without adding *fake* or removing *truthful* information [5,16]. This precludes the application of ϵ-*differential privacy* [3], which only allows releasing noisy answers to user queries or noisy summary statistics, as well as *suppression* [19], which deletes values prior to data release.

		Relational attributes			Transaction attribute
Id	Name	Age	Origin	Gender	Purchased-products
0	John	19	France	Male	E F B G
1	Steve	22	Greece	Male	E F D H
2	Mary	28	Germany	Female	B C E G
3	Zoe	39	Spain	Female	F D H
4	Ann	70	Algeria	Female	E G
5	Jim	55	Nigeria	Male	A F H

(a)

	Relational attributes			Transaction attribute
Id	Age	Origin	Gender	Purchased-products
0	[19:22]	Europe	Male	E F (A,B,C,D) G
1	[19:22]	Europe	Male	E F (A,B,C,D) H
2	[28:39]	Europe	Female	E (A B,C D) G
3	[28:39]	Europe	Female	F (A,B,C,D) H
4	[55:70]	Africa	All	E G
5	[55:70]	Africa	All	F (A,B,C,D) H

(b)

Fig. 1: (a) An RT-dataset with patient demographics and IDs of purchased products, and (b) a 2-anonymous dataset with respect to relational attributes and 2^2-anonymous with respect to the transaction attribute. Identifiers Id and Name are not published.

A plethora of methods can be used to preserve the privacy of datasets containing only relational or only transaction attributes [9,12,15,18]. However, there are currently no methods for anonymizing RT-datasets, and simply anonymizing each attribute type separately, using existing methods (e.g., [9,12,15,18]), is not enough. This is because information concerning *both* relational and transaction attributes may lead to *identity disclosure* (i.e., the association of an individual to their record) [15]. Consider, for example, the dataset in Fig. 1a which is anonymized by applying the methods of [18] and [8] to the relational and transaction attributes, as shown in Fig. 1b. An attacker, who knows that Jim is a 55-year-old Male from Nigeria who purchased F, can associate Jim with record 5 in Fig. 1b. Thwarting identity disclosure is essential to comply with legislation, e.g., HIPAA, and to help future data collection. At the same time, many applications require preventing *attribute disclosure* (i.e., the association of an individual with sensitive information). In medical data publishing, for example, this ensures that patients are not associated with sensitive diagnoses [17].

Furthermore, anonymized RT-datasets need to have minimal information loss in relational and in transaction attributes. However, these two requirements are conflicting, and the problem is difficult to address using multi-objective optimization strategies [4]. In fact, these strategies are either inapplicable or incur excessive information loss, as we show in Section 3.

Contributions. Our work makes the following specific contributions:

- We introduce the problem of anonymizing RT-datasets and propose the first approach to tackle it. Our privacy model prevents an attacker, who knows the set of an individual's values in the relational attributes and up to m items in the transaction attribute, from linking the individual to their record.
- We develop an approach for producing (k, k^m)-anonymous RT-datasets with bounded information loss in one attribute type and minimal information loss in the other. Following this approach, we propose two frameworks which employ *generalization* [15] and are based on a three-phase process: *(i)* creating k-anonymous clusters with respect to the relational attributes, *(ii)* merging these clusters in a way that helps anonymizing RT-datasets with low information loss, and *(iii)* enforcing (k, k^m)-anonymity to each merged cluster.
- We propose a family of algorithms to implement the second phase in each framework. These algorithms operate by building clusters, which can be made (k, k^m)-anonymous with minimal information loss, and preserve different aspects of data utility.

Id	Relational attributes: Age	Origin	Gender	Transaction attribute: Purchased-products
0	[19:22]	Europe	Male	D E (B,D) G
1	[19:22]	Europe	Male	E E (B,D) H
2	[28:39]	Europe	Female	(B,C,F) (D,E) G
3	[28:39]	Europe	Female	(B,C,F) (D,E) H
4	[55:70)	Africa	All	(A,E,F) G
5	[55:70)	Africa	All	(A,E,F) H

(a)

Id	Relational attributes: Age	Origin	Gender	Transaction attribute: Purchased-products
0	[19:70]	All	All	E F (A,B,C,D) G
1	[19:70]	All	All	E F (A,B,C,D) H
2	[19:70]	All	All	E (A,B,C,D) G
3	[19:70]	All	All	F (A,B,C,D) H
4	[19:70]	All	All	E G
5	[19:70]	All	All	F (A,B,C,D) H

(b)

Id	Relational attributes: Age	Origin	Gender	Transaction attribute: Purchased-products
0	[19:39]	Europe	All	E F (B,C,D) G
1	[19:39]	Europe	All	E F (B,C,D) H
2	[19:39]	Europe	All	E (B,C,D) G
3	[19:39]	Europe	All	F (B,C,D) H
4	[55:70)	Africa	All	(A,E,F) G
5	[55:70)	Africa	All	(A,E,F) H

(c)

Id	Relational attributes: Age	Origin	Gender	Transaction attribute: Purchased-products
0	[19:70]	All	All	E F (A,B,D) G
1	[19:70]	All	All	E F (A,B,D) H
4	[19:70]	All	All	E G
5	[19:70]	All	All	F (A,B,D) H
2	[28:39]	Europe	Female	(B,C,F) (D,E) G
3	[28:39]	Europe	Female	(B,C,F) (D,E) H

(d)

Fig. 2: The $(2, 2^2)$-anonymous datasets from applying (a) **R**FIRST, and (b) **T**FIRST to the dataset of Fig. 1a, and (c) **R**MERGE$_R$, and (d) **R**MERGE$_T$, to the clusters of Fig. 2a

– We investigate the effectiveness of our approach by conducting experiments on two real-world RT-datasets. Our results verify that the proposed approach is effective at preserving data utility.

PAPER ORGANIZATION. Section 2 defines concepts used in this work. Section 3 clarifies why popular multi-objective optimization strategies are unsuited for enforcing (k, k^m)-anonymity and formulates the target problems. Sections 4 and 5 present our approach and an instance of it. Sections 6 and 7 present experiments and discuss related work, and Section 8 concludes the paper.

2 *RT*-datasets and their anonymity

RT-DATASETS. An *RT-dataset* D consists of records containing relational attributes $R_1, \ldots, R_v$, which are single-valued, and a transaction attribute T, which is set-valued. For convenience, we consider that: *(i)* identifiers have been removed from D, and *(ii)* a single transaction attribute T is contained in D[1].

(k, k^m)-ANONYMITY. We propose (k, k^m)-anonymity to guard against *identity disclosure*. To prevent both identity and attribute disclosure, (k, k^m)-anonymity can be extended, as we explain in Section 5.

Before defining (k, k^m)-anonymity, we associate each record r in an RT-dataset D with a *group* of records $G(r)$, as shown below.

Definition 1. *For a record $r \in D$, its* group $G(r)$ *is a set of records that contains r and each record $q \in D$, such that $q[R_1, \ldots, R_v] = r[R_1, \ldots, R_v]$ and $q[T] \cap I = r[T] \cap I$, where I is any set of m or fewer items of $r[T]$*[2].

Group $G(r)$ contains r and all records that are indistinguishable from r to an attacker, who knows the values of r in relational attributes *and* up to m items

[1] Multiple transaction attributes $T_1, \ldots, T_u$ can be transformed to a single transaction attribute T, whose domain contains every item in the domain of $T_1, \ldots, T_u$, preceded by the domain name, i.e., $dom(T) = \{d.t \mid d = T_i \text{ and } t \in dom(T_i), i \in [1, u]\}$.

[2] Expression $r[A]$ is a shortcut for the projection $\pi_A(r)$.

in the transaction attribute. The size of $G(r)$, denoted with $|G(r)|$, represents the risk of associating an individual with a record r. Thus, to provide privacy, we may lower-bound $|G(r)|$. This idea is captured by (k, k^m)-anonymity.

Definition 2. *A group of records $G(r)$ is* (k, k^m)-anonymous*, if and only if $|G(r)| \geq k$, for each record r in $G(r)$. An RT-dataset D is (k, k^m)-anonymous, if and only if the group $G(r)$ of each record $r \in D$ is (k, k^m)-anonymous.*

For example, in Fig. 2a groups {0,1} (=G(0)=G(1)), {2,3} (=G(2)=G(3)) and {4,5} (=G(4)=G(5)) are $(2, 2^2)$-anonymous, rendering the whole dataset $(2, 2^2)$-anonymous. Note that in each group, all records have the same values in the relational attributes, as required by Definition 1, but do not necessarily have the same items in the transaction attribute Purchased-products (see Fig. 2b).

The notion of (k, k^m)-anonymity for RT-datasets extends and combines relational k-anonymity [15] and transactional k^m-anonymity [17].

Proposition 1. *Let $D[R_1, \ldots, R_v]$ and $D[T]$ be the relational and transaction part of an RT-dataset D, respectively. If D is (k, k^m)-anonymous, then $D[R_1, \ldots, R_v]$ is k-anonymous and $D[T]$ is k^m-anonymous.*

Proposition 1 shows that (k, k^m)-anonymity provides the same protection as k-anonymity [15], for relational attributes, and as k^m-anonymity [17], for transaction attributes. *Unfortunately, the inverse does not hold.* That is, an RT-dataset may be k and k^m but not (k, k^m)-anonymous. For instance, let D be the dataset of Fig. 1b. Note that D[Age, Origin, Gender] is 2-anonymous and D[Purchased-products] is 2^2-anonymous, but D is not $(2, 2^2)$-anonymous.

Generalization. We employ the generalization functions defined below.

Definition 3. *A* relational generalization function $\mathcal{R}$ *maps a value v in a relational attribute R to a generalized value $\tilde{v}$, which is a range of values, if R is numerical, or a collection of values, if R is categorical.*

Definition 4. *A* transaction generalization function $\mathcal{T}$ *maps an item u in the transaction attribute T to a generalized item $\tilde{u}$. The generalized item $\tilde{u}$ is a non-empty subset of items in T that contains u.*

The way relational values and transactional items are generalized is fundamentally different, as they have different semantics [19]. Specifically, a generalized value bears *atomic* semantics and is interpreted as a *single value* in a range or a collection of values, whereas a generalized item bears *set* semantics and is interpreted as *any non-empty subset* of the items mapped to it [12]. For instance, the generalized value [19:22] in Age, in the record 0 in Fig. 2a, means that the actual Age is in $[19, 22]$. Contrary, the generalized item (B, D) in Purchased-products means that B, or D, or both products were bought. Given a record r, the function $\mathcal{R}$ is applied to a single value $v \in R$, and all records in the k-anonymous group $G(r)$ must have the same generalized value in R. On the other hand, the function $\mathcal{T}$ is applied to one of the potentially many items in T, and the records in the k^m-anonymous $G(r)$ may not have the same generalized items.

DATA UTILITY MEASURES. In this work, we consider two general data utility measures; **Rum**, for relational attributes, and **Tum**, for the transaction attribute. These measures satisfy Properties 1, 2 and 3.

Property 1. Lower values in **Rum** and **Tum** imply better data utility.

Property 2. **Rum** is *monotonic* to subset relationships. More formally, given two groups G and G' having at least k records, and a relational generalization function $\mathcal{R}$, it holds that $\mathbf{Rum}(\mathcal{R}(G) \cup \mathcal{R}(G')) \leq \mathbf{Rum}(\mathcal{R}(G \cup G'))$.

Property 2 suggests that data utility is preserved better, when we generalize the relational values of small groups, and is consistent with prior work on relational data anonymization [2,6]. Intuitively, this is because the group $G \cup G'$ contains more distinct values in a relational attribute R than G or G', and thus more generalization is needed to make its values indistinguishable.

A broad class of measures, such as *NCP*, the measures expressed as Minkowski norms [6], *Discernability* [1], and the *Normalized average equivalence class size metric* [9], satisfy Property 2 [6], and can be used as **Rum**.

Property 3. **Tum** is *anti-monotonic* to subset relationships. More formally, given two groups G and G' having at least k records, and a transaction generalization function $\mathcal{T}$ that satisfies Definition 4 and *(i)* maps each item in the group it is applied to a generalized item that is not necessarily unique, and *(ii)* constructs the mapping with the minimum **Tum**, it holds that $\mathbf{Tum}(\mathcal{T}(G) \cup \mathcal{T}(G')) \geq \mathbf{Tum}(\mathcal{T}(G \cup G'))$.

Property 3 suggests that generalizing large groups can preserve transaction data utility better, and is consistent with earlier works [12,17]. Intuitively, this is because, all mappings between items and generalized items constructed by $\mathcal{T}$ when applied to G and G' separately (Case I) can also be constructed when $\mathcal{T}$ is applied to $G \cup G'$ (Case II), but there can be mappings that can only be considered in Case II. Thus, the mapping with the minimum **Tum** in Case I cannot have lower **Tum** than the corresponding mapping in Case II.

3 Challenges of enforcing (k, k^m)-anonymity

LACK OF OPTIMAL SOLUTION. Constructing a (k, k^m)-anonymous RT-dataset D with minimum information loss is far from trivial. Lemma 1 follows from Theorem 1 and shows that there is no (k, k^m)-anonymous version of D with minimum (i.e., optimal) **Rum** and **Tum**, for any D of realistic size.

Theorem 1. *Let $\mathcal{D}_R$ and $\mathcal{D}_T$ be the optimal (k, k^m)-anonymous version of an RT-dataset D with respect to* **Rum** *and* **Tum***, respectively. Then, no group in $\mathcal{D}_R$ contains more than $2k-1$ records, and $\mathcal{D}_T$ is comprised of a single group.*

Proof. (Sketch) The proof that no group in $\mathcal{D}_R$ contains more than $2k-1$ records is based on Property 2, and has been given in [6]. The proof that $\mathcal{D}_T$ is comprised of a single group is similar and, it is based on Property 3.

Lemma 1. *There is no optimal (k, k^m)-anonymous version $\mathcal{D}$ of an RT-dataset D with respect to both* **Rum** *and* **Tum**, *unless* $|D| \in [k, 2k-1]$.

Inadequacy of popular optimization strategies. Constructing useful (k, k^m)-anonymous RT-datasets requires minimizing information loss with respect to both **Rum** and **Tum**. Such multi-objective optimization problems are typically solved using the *lexicographical*, the *conventional weighted-formula*, or the *Pareto optimal* approach [4]. We will highlight why these approaches are not adequate for our problem.

Lexicographical. In this approach, the optimization objectives are ranked and optimized in order of priority. In our case, we can prioritize the lowering of information loss in *(i)* the relational attributes (i.e., minimal **Rum**), or *(ii)* the transaction attribute (i.e., minimal **Tum**).

Given an RT-dataset D and anonymization parameters k and m, an algorithm that implements strategy *(i)* is **R**first. This algorithm partitions D into a set of k-anonymous groups $\mathcal{C}$, with respect to the relational attributes (e.g., using [18]), and applies $\mathcal{T}$ to generalize items in each group of records in $\mathcal{C}$, separately (e.g., using [17]). Symmetrically, to implement strategy *(ii)*, we may use an algorithm **T**first, which first partitions D into a set of k^m-anonymous groups (e.g., using the LRA algorithm [17]), and then applies a relational generalization function (see Definition 3) to each relational attribute, in each group.

Both **R**first and **T**first enforce (k, k^m)-anonymity, but produce vastly different results. For instance, Figs. 2a and 2b show $(2, 2^2)$-anonymous versions of the dataset in Fig. 1a, produced by **R**first and **T**first, repectively. Observe that **R**first did not generalize the relational attributes as heavily as **T**first but applied more generalization to the transaction attribute. This is because, **R**first constructs small groups, and does not control the grouping of items. Contrary, the groups created by **T**first contain records, whose items are not heavily generalized, unlike their values in the relational attributes. In either case, the purpose of producing anonymized RT-datasets that allow meaningful analysis of relational and transaction attributes together, is defeated.

Conventional weighted-formula. In this approach, all objectives are combined into a single one, using a weighted formula. The combined objective is then optimized by a single-objective optimization algorithm. For example, a clustering-based algorithm [13] would aim to minimize the weighted sum of **Rum** and **Tum**. However, this approach works only for *commensurable* objectives [4]. This is not the case for **Rum** and **Tum**, which are fundamentally different and have different properties (see Section 2). Therefore, this approach is not suitable.

Pareto optimal. This approach finds a set of solutions that are *non-dominated* [4], from which the most appropriate solution is selected by the data publisher, according to their preferences. However, the very large number of non-dominated solutions that can be constructed by flexible generalization functions, such as those in Definitions 3 and 4, render this approach impractical.

Problem formulation. To construct a (k, k^m)-anonymous version of an RT-dataset, we *either* upper-bound the information loss in relational attributes and

Algorithm: **Rum**-BOUND

// Initial cluster formation
1 $\{C_1, \ldots, C_n\} :=$ ClusterFormation(D, k)
2 $\mathcal{D} := \{C_1, \ldots, C_n\}$
3 **if Rum**$(\mathcal{D}) > \delta$ **then return** false
// Cluster merging
4 $\mathcal{D} :=$ **R**Merge$(\mathcal{D}, \mathcal{T}, \delta)$
// (k, k^m)-anonymization
5 **for** *each cluster* $C \in \mathcal{D}$ **do**
6 $\quad \mathcal{D} := (\mathcal{D} \setminus C) \cup \mathcal{T}(C)$
7 **return** $\mathcal{D}$

Algorithm: **Tum**-BOUND

// Initial cluster formation
1 $\{C_1, \ldots, C_n\} :=$ ClusterFormation(D, k)
2 $\mathcal{D} := \{C_1, \cdots, C_n\}$
3 **if Tum**$(\mathcal{T}(\mathcal{D})) \leq \delta$ **then return** $\mathcal{D}$
// Cluster merging
4 $\mathcal{D} :=$ **T**Merge$(\mathcal{D}, \mathcal{T}, \delta)$
// (k, k^m)-anonymization
5 **for** *each cluster* $C \in \mathcal{D}$ **do**
6 $\quad \mathcal{D} := (\mathcal{D} \setminus C) \cup \mathcal{T}(C)$
7 **if Tum**$(\mathcal{D}) > \delta$ **then return** false
8 **return** $\mathcal{D}$

seek to minimize the information loss in the transaction attribute (Problem 1), *or* upper-bound the information loss in the transaction attribute and seek to minimize the information loss in relational attributes (Problem 2).

Problem 1. Given an RT-dataset D, data utility measures **Rum** and **Tum**, parameters k and m, and a threshold δ, construct a (k, k^m)-anonymous version $\mathcal{D}$ of D, such that **Rum**$(\mathcal{D}) \leq \delta$ and **Tum**$(\mathcal{D})$ is minimized.

Problem 2. Given an RT-dataset D, data utility measures **Rum** and **Tum**, parameters k and m, and a threshold δ, construct a (k, k^m)-anonymous version $\mathcal{D}$ of D, such that **Tum**$(\mathcal{D}) \leq \delta$ and **Rum**$(\mathcal{D})$ is minimized.

Threshold δ must be specified by data publishers, as in [6]. Thus, constructing $\mathcal{D}$ might be infeasible for an arbitrary δ. Solving Problem 1 or Problem 2 ensures that $\mathcal{D}$ preserves privacy and utility, but it is NP-hard (proof follows from [12]).

4 Anonymization approach

We propose an approach that overcomes the deficiencies of the aforementioned optimization approaches and works in three phases:

Initial cluster formation: k-anonymous clusters with respect to relational attributes, which incur low information loss, are formed.

Cluster merging: Clusters are merged until the conditions set by Problems 1 or 2 are met.

(k, k^m)-*anonymization*: Each cluster becomes (k, k^m)-anonymous, by generalizing the its items with low **Tum**.

Based on our approach, we developed two anonymization frameworks, **Rum**-BOUND and **Tum**-BOUND, which address Problems 1 and 2, respectively. **Rum**-BOUND seeks to produce a dataset with minimal **Tum** and acceptable **Rum**, and implements the phases of our approach, as follows.

Initial cluster formation (Steps 1–3): Algorithm **Rum**-BOUND clusters D, using a function ClusterFormation, which can be implemented by any generalization-based k-anonymity algorithm [9,18,2]. This function produces a set of k-

anonymous clusters $C_1, \ldots, C_n$, from which a dataset $\mathcal{D}$ containing $C_1, \ldots, C_n$, is created (Step 2). The dataset $\mathcal{D}$ must have a lower **Rum** than δ, since subsequent steps of the algorithm cannot decrease **Rum** (see Property 2). If the dataset $\mathcal{D}$ does not satisfy this condition, it cannot be a solution to Problem 1, and false is returned (Step 3).

Cluster merging (Step 4): This phase is the crux of our framework. It is performed by a function **R**MERGE, which merges the clusters of $\mathcal{D}$ to produce a version that can be (k, k^m)-anonymized with minimal **Tum** and without violating δ. To implement **R**MERGE we propose three algorithms, namely **R**MERGE$_R$, **R**MERGE$_T$ and **R**MERGE$_{RT}$, which aim at minimizing **Tum** using different heuristics.

(k, k^m)*-anonymization (Steps 5–7):* In this phase, $\mathcal{D}$ is made (k, k^m)-anonymous, by applying a transaction generalization function $\mathcal{T}$ to each of its clusters.

Tum-BOUND, on the other hand, focuses on Problem 2 and aims at creating a dataset with minimal **Rum** and acceptable **Tum**. This framework has the following major differences from **Rum**-BOUND.

- At Step 3, after the formation of $\mathcal{D}$, **Tum**-BOUND checks if $\mathcal{D}$ has lower **Tum** than the threshold δ. In such case, $\mathcal{D}$ is a solution to Problem 2.
- At Step 4, function **T**MERGE merges clusters until the **Tum** threshold is reached, or no more merging is possible. To implement **T**MERGE we propose three algorithms: **T**MERGE$_R$, **T**MERGE$_R$ and **T**MERGE$_{RT}$, which aim at minimizing **Rum** using different heuristics.
- At Step 7, **Tum**-BOUND checks if **Tum**$(\mathcal{D}) > \delta$; in this case, we cannot satisfy Problem 2 conditions and, thus, return false.

CLUSTER-MERGING ALGORITHMS. We now present three algorithms that implement function **R**MERGE, which is responsible for the merging phase of **Rum**-BOUND (Step 4). Our algorithms are based on different merging heuristics. Specifically, **R**MERGE$_R$ merges clusters with similar relational values, **R**MERGE$_T$ with similar transaction items and **R**MERGE$_{RT}$ takes a middle line between these two algorithms. In all cases, relational generalization is performed by a set of functions $\mathcal{G} = \{\mathcal{L}_1, \ldots, \mathcal{L}_v\}$, one for each relational attribute (Definition 3) and transaction generalization is performed by function $\mathcal{T}$ (Definition 4).

RMERGE$_R$ selects the cluster C with the minimum **Rum**(C) as a seed (Step 2). Cluster C contains relational values that are not highly generalized and is expected to be merged with a low relational utility loss. The algorithm locates the cluster C' with the most similar relational values to C (Step 3) and constructs a temporary dataset $\mathcal{D}_{tmp}$ that reflects the merging of C and C' (Step 4). If $\mathcal{D}_{tmp}$ does not violate the **Rum** threshold, it is assigned to $\mathcal{D}$ (Step 5).

RMERGE$_T$ starts by selecting the same seed C as **R**MERGE$_R$ (Step 2) and seeks a cluster C' that contains similar transaction items to C and, when merged with C, results in a dataset with **Rum** no higher than δ. To this end, **R**MERGE$_T$ merges C with every other cluster C_i in $\mathcal{D} \backslash C$ and orders the clusters by increasing **Tum**$(\mathcal{T}(C \cup C_i))$ (Step 3). This allows efficiently finding the best merging for minimizing **Tum** that does not violate **Rum**$(\mathcal{D}) \leq \delta$. The algorithm considers

Algorithm: **R**MERGE$_R$

```
1 while 𝒟 changes do
2     Select, as a seed, the cluster C ∈ 𝒟 with minimum Rum(C)
3     Find the cluster C′ ∈ 𝒟 that minimizes Rum(𝒢(C ∪ C′)) .
4     𝒟_tmp := ((𝒟 \ C) \ C′) ∪ 𝒢(C ∪ C′)
5     if Rum(𝒟_tmp) ≤ δ then
          𝒟 := 𝒟_tmp
6 return 𝒟
```

Algorithm: **R**MERGE$_T$

```
1 while 𝒟 changes do
2     Select, as a seed, the cluster C ∈ 𝒟 with minimum Rum(C)
      // Find the appropriate cluster C′ to be merged with C
3     Let {C_1, ..., C_t} be the set of clusters in 𝒟 \ C ordered by increasing
      Tum(𝒯(C ∪ C_i)), i ∈ [1, t)
4     for i := 1 to t do          // Test if C′ = C_i
5         𝒟_tmp := ((𝒟 \ C) \ C_i) ∪ 𝒢(C ∪ C_i)
6         if Rum(𝒟_tmp) ≤ δ then   // C′ is C_i
7             𝒟 := 𝒟_tmp
8             exit the for loop
9 return 𝒟
```

Algorithm: **R**MERGE$_{RT}$

```
1  while 𝒟 changes do
2      Select, as a seed, the cluster C ∈ 𝒟 with minimum Rum(C)
3      Let {C_1, ..., C_t} (resp. {Ĉ_1, ..., Ĉ_t}) be the set of clusters in 𝒟 \ C ordered by
       increasing Rum(𝒢(C ∪ C_i)) (resp. Tum(𝒯(C ∪ Ĉ_i))), i ∈ [1, t)
       // Find the appropriate cluster C′ to be merged with C
4      for i := 1 to t do
5          Find cluster C′, that has the i-th minimum sum of indices u + v s.t.
           C_u ∈ {C_1, ..., C_t} and C_v ∈ {Ĉ_1, ..., Ĉ_t}
6          𝒟_tmp := ((𝒟 \ C) \ C_i) ∪ 𝒢(C ∪ C_i)
7          if Rum(𝒟_tmp) ≤ δ then
8              𝒟 := 𝒟_tmp
9              exit the for loop
10 return 𝒟
```

the clusters with increasing **Tum**$(\mathcal{T}(C \cup C_i))$ scores. The first cluster that gives a dataset with acceptable **Rum** is used for merging (Steps 4–5).

RMERGE$_{RT}$ combines the benefits of **R**MERGE$_R$ and **R**MERGE$_T$. It selects the same seed cluster C as **R**MERGE$_T$, and constructs two orderings, which sort the generalized merged clusters in ascending order of **Rum** and **Tum**, respectively (Step 3). Then, a cluster C' that is as close as possible to C, based on both orderings (i.e., it has the i-th minimum sum $(u + v)$, where u and v are the indices of C' in the $\{C_1, \ldots, C_t\}$ and orderings $\{\hat{C}_1, \ldots, \hat{C}_t\}$ repsectively), is found (Step 5). The next steps of **R**MERGE$_{RT}$ are the same as in **R**MERGE$_T$.

We now discuss **T**MERGE$_R$, **T**MERGE$_R$, and **T**MERGE$_{RT}$, used in **Tum**-BOUND. These algorithms perform cluster merging, until $\mathcal{D}$ satisfies the **Tum** threshold, or all possible mergings have been considered. The pseudocode of **R**MERGE$_R$ is the same as that of **T**MERGE$_R$, except that Step 5 in **R**MERGE$_R$ is replaced by the following steps. Note that D is returned if it satisfies the **Tum** threshold, because **Rum** cannot be improved by further cluster merging (Property 2).

```
5 if Tum(𝒟_tmp) ≤ δ then
■     𝒟 := 𝒟_tmp
■     return 𝒟
```

The pseudocode of **T**MERGE$_R$ and **T**MERGE$_{RT}$ can be derived by replacing the same steps with Steps 5 and 7 in **T**MERGE$_R$ and **T**MERGE$_{RT}$, respectively.

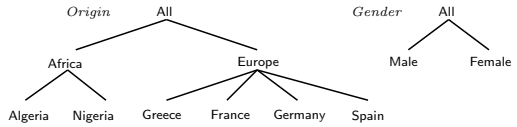

Fig. 3: Hierarchies for the dataset of Fig. 1a

The runtime cost of anonymization is $O(\mathcal{F} + |\mathcal{C}|^2 \cdot (\mathcal{K}_\mathcal{R} + \mathcal{K}_\mathcal{T}))$, where $\mathcal{F}$ is the cost for initial cluster formation, $|\mathcal{C}|$ the number of clusters in $\mathcal{D}$, and $\mathcal{K}_\mathcal{R}$ and $\mathcal{K}_\mathcal{T}$ the cost of generalizing the relational and transaction part of a cluster.

5 Instantiating and extending the frameworks

Our frameworks can be parameterized by generalization functions, data utility measures, and initial cluster formation algorithms. This section presents such instantiations and strategies to improve their efficiency, as well as extensions of our frameworks to prevent both identity and attribute disclosure.

GENERALIZATION FUNCTIONS. We employ the *local recoding* [18] and *set-based generalization* [8,12]. As an example, the dataset in Fig. 1b has been created by applying these functions to the dataset in Fig. 1a, using the hierarchies in Fig. 3.

DATA UTILITY MEASURES. To measure data utility in relational and transaction attributes, we used *Normalized Certainty Penalty* (*NCP*) [18] and *Utility Loss* (*UL*) [12], respectively. The *NCP* for a generalized value $\tilde{v}$, a record r, and an RT-dataset D, is defined as: $NCP_R(\tilde{v}) = \begin{cases} 0, |\tilde{v}| = 1 \\ |\tilde{v}|/|R|, \text{otherwise} \end{cases}$, $NCP(r) = \sum_{i \in [1,v]} w_i \cdot NCP_{R_i}(r[R_i])$ and $NCP(D) = \frac{\sum_{r \in D} NCP(r)}{|D|}$ resp., where $|R|$ denotes the number of leaves in the hierarchy for a categorical attribute R (or domain size for a numerical attribute R), $|\tilde{v}|$ denotes the number of leaves of the subtree rooted at $\tilde{v}$ in the hierarchy for a categorical R (or the length of the range for a numerical R), and $w_i \in [0, 1]$ is a weight that measures the importance of an attribute. The *UL* for a generalized item $\tilde{u}$, a record r, and an RT-dataset D, is defined as: $UL(\tilde{u}) = (2^{|\tilde{u}|} - 1) \cdot w(\tilde{u})$, $UL(r) = \frac{\sum_{\forall \tilde{u} \in r} UL(\tilde{u})}{2^{\sigma(r)} - 1}$ and $UL(D) = \frac{\sum_{\forall r \in D} UL(r)}{|D|}$ resp., where $|\tilde{u}|$ is the number of items mapped to $\tilde{u}$, $w(\tilde{u}) \in [0, 1]$ a weight reflecting the importance of $\tilde{u}$ [12], and $\sigma(r)$ the sum of sizes of all generalized items in r.

INITIAL CLUSTER FORMATION WITH CLUSTER. The initial cluster formation phase should be implemented using algorithms that create many small clusters, with low **Rum**, because this increases the chance of constructing a (k, k^m)-anonymous dataset with good data utility. Thus, we employ CLUSTER, an algorithm that is instantiated with *NCP* and local recoding, and it is inspired by the algorithm in [2]. The time complexity of CLUSTER is $O(\frac{|D|^2}{k} \cdot \log(|D|))$.

EFFICIENCY OPTIMIZATION STRATEGIES. To improve the efficiency of cluster-merging algorithms, we compute $\mathbf{Rum}(\mathcal{D}_{tmp})$ incrementally, thereby avoiding to access all records in $\mathcal{D}_{tmp}$, after a cluster merging. This can be performed for all measures in Section 2, but we illustrate it for *NCP*. We use a list λ of tuples $<|C|, NCP(r_c))>$, for each cluster C in $\mathcal{D}_{tmp}$ and any record r_c in C, which is initialized based on $\mathcal{D}$. Observe that $NCP(\mathcal{D}_{tmp}) = \frac{\sum_{\forall C \in \mathcal{D}_{tmp}} (|C| \cdot NCP(r_c))}{|D|}$, and

Algorithm: CLUSTER

$\mathcal{C} := \emptyset$
// Create clusters of size k
while $|D| \geq k$ **do**
  Select, as a seed, a random record s from D
  Add s and each record $r \in D$ having one of the lowest $k-1$ values in $NCP(\mathcal{G}(\{s, r\}))$ to cluster C
  Add cluster C to $\mathcal{C}$ and remove its records from D
// Accommodate the remaining $|D|$ mod k records
for *each record* $r \in D$ **do**
  Add r to the cluster $C \in \mathcal{C}$ that minimizes $NCP(\mathcal{G}(C \cup r))$
Apply $\mathcal{G}$ to the relational values of each cluster in $\mathcal{C}$
// Extend clusters
for *each cluster* $C \in \mathcal{C}$ **do**
  Let S be the set of clusters in $\mathcal{C}$ with the same values in relational attributes as C.
  Extend C with the records of S and remove each cluster in S from $\mathcal{C}$.
return $\mathcal{C}$

it can be updated, after C and C' are merged, by adding: $\frac{(|C|+|C'|)\cdot NCP(r_{c\cup c'})}{|D|} - \frac{|C|\cdot NCP(r_c)-|C'|\cdot NCP(r_{c'})}{|D|}$. This requires accessing only the records in $C \cup C'$.

The efficiency of $\mathbf{R}$MERGE$_T$, $\mathbf{R}$MERGE$_{RT}$, $\mathbf{T}$MERGE$_R$, and $\mathbf{T}$MERGE$_{RT}$ can be further improved by avoiding computing $\mathbf{Tum}(\mathcal{T}(C \cup C_1)), \ldots, \mathbf{Tum}(\mathcal{T}(C \cup C_t))$. For this purpose, we merge clusters using *Bit-vector Transaction Distance* (*BTD*). The *BTD* for records r_1, r_2 is defined as $BTD(r_1, r_2) = \frac{ones(b_1 \veebar b_2)+1}{ones(b_1 \wedge b_2)+1}$. $ones(b_1 \vee b_2)$, where b_1 and b_2 are the bit-vector based representations of $r_1[T]$ and $r_2[T]$, $\veebar$, $\wedge$ and $\vee$ are the Boolean operators, for XOR, AND, and OR, and the function *ones* counts the number of 1 bits in a bit-vector. The *BTD* of a cluster C is defined as $BTD(C) = \max\{BTD(r_1, r_2)|$ for all $r_1, r_2 \in C\}$. *BTD* helps enforcing (k, k^m)-anonymity with minimal **Tum**, as it favors the grouping of records with a small number of items, many of which are common.

PREVENTING BOTH IDENTITY AND ATTRIBUTE DISCLOSURE. To prevent both types of disclosure, we propose the concept of (k, ℓ^m)-diversity, defined below.

Let $G(r)$ be a group of records and $G(r')$ be a group with the same records as $G(r)$ projected over $\{R_1, \ldots, R_v, T'\}$, where T' contains only the nonsensitive items in T. $G(r)$ is (k, ℓ^m)-*diverse*, if and only if $G(r')$ is (k, k^m)-anonymous, and an attacker, who knows up to m nonsensitive items about an individual, cannot associate any record in $G(r)$ to any combination of sensitive items, with a probability greater than $\frac{1}{\ell}$. An RT-dataset D is (k, ℓ^m)-diverse, if and only if the group $G(r)$ of each record $r \in D$ is (k, ℓ^m)-diverse.

(k, ℓ^m)-diversity forestalls identity disclosure, and, additionally, the inference of any combination of sensitive items, based on ℓ^m-diversity [17]. Extending our anonymization frameworks to enforce (k, ℓ^m)-diversity requires: *(i)* applying **Tum** to nonsensitive items, and *(ii)* replacing the transaction generalization function $\mathcal{T}$, which enforces k^m-anonymity to each cluster, with one that applies ℓ^m-diversity. The ℓ^m-diversity version of AA [17] was used as such a function.

6 Experimental evaluation

In this section, we evaluate our algorithms in terms of data utility and efficiency, and demonstrate the benefit of choices made in their design.

Dataset	$\|D\|$	Rel. att.	$\|dom(T)\|$	Max, Avg # items/record
INFORMS	36553	5	619	17, 4.27
YOUTUBE	131780	6	936	37, 6.51

Table 1: Description of the datasets

EXPERIMENTAL SETUP. We implemented all algorithms in C++ and applied them to INFORMS (https://sites.google.com/site/informsdataminingcontest) and YOUTUBE (http://netsg.cs.sfu.ca/youtubedata) datasets, whose characteristics are shown in Table 1[4]. The default parameters were k=25, m=2, and δ=0.65, and hierarchies were created as in [17]. Our algorithms are referred to in abbreviated form (e.g., **R**M$_R$ for **R**MERGE$_R$) and were not compared against prior works, since they cannot (k, k^m)-anonymize RT-datasets. The algorithms that enforce (k, ℓ^m)-diversity are named after those based on (k, k^m)-anonymity. All experiments ran on an Intel i5 at 2.7 GHz with 8 GB of RAM.

DATA UTILITY. We evaluated data utility on INFORMS and YOUTUBE using k=25 and k=100, respectively, and varied δ in $[X, 1)$, where X is the NCP of the dataset produced by CLUSTER, for **Rum**-BOUND, or the UL, for **Tum**-BOUND. Data utility is captured using ARE [9,12,16], which is invariant of the way our algorithms work and reflects the average number of records that are retrieved incorrectly, as part of query answers. We used workloads of 100 queries, involving relational, transaction, or both attribute types, which retrieve random values and/or sets of 2 items by default [9,12]. Low ARE scores imply that anonymized data can be used to accurately estimate the number of co-occurrences of relational values and items. This statistic is an important building block of several data mining models.

Figs. 4a to 4g demonstrate the conflicting objectives of minimizing information loss in relational and transaction attributes, and that **Rum**-BOUND can produce useful data. By comparing Fig. 4a with 4c, and Fig. 4d with 4g, it can be seen that a small δ forces all algorithms to incur low information loss in the relational attributes, whereas a large δ favors the transaction attribute. Also, NCP is at most δ, in all tested cases, and data remain useful for queries involving both attribute types (see Figs. 4b, 4e, and 4f). We performed the same experiments for the **Tum**-BOUND and present a subset of them in Fig. 4h. Note that, increasing δ (i.e., the bound for UL), favors relational data, and that the information loss in the transaction attribute is low. Similar observations can be made for the (k, l^m)-diversity algorithms (see Fig. 5).

Next, we compared **R**M$_R$, **R**M$_T$, and **R**M$_{RT}$. As shown in Fig. 4, **R**M$_R$ incurred the lowest information loss in the transaction attribute, and the highest in the relational attributes, and **R**M$_T$ had opposite trends. **R**M$_{RT}$ allows more accurate query answering than **R**M$_R$, in relational attributes, and than **R**M$_T$, in the transaction attribute, as it merges clusters, based on both attribute types. Similar results were obtained for YOUTUBE (see Figs. 4d-4g), from comparing **T**M$_T$,

[4] INFORMS contains the relational attributes {*month of birth, year of birth, race, years of education, income*}, and the transaction attribute *diagnosis_codes*. YOUTUBE contains the relational attributes {*age, category, length, rate, #ratings, #comments*}, and the transaction attribute *related_videos*.

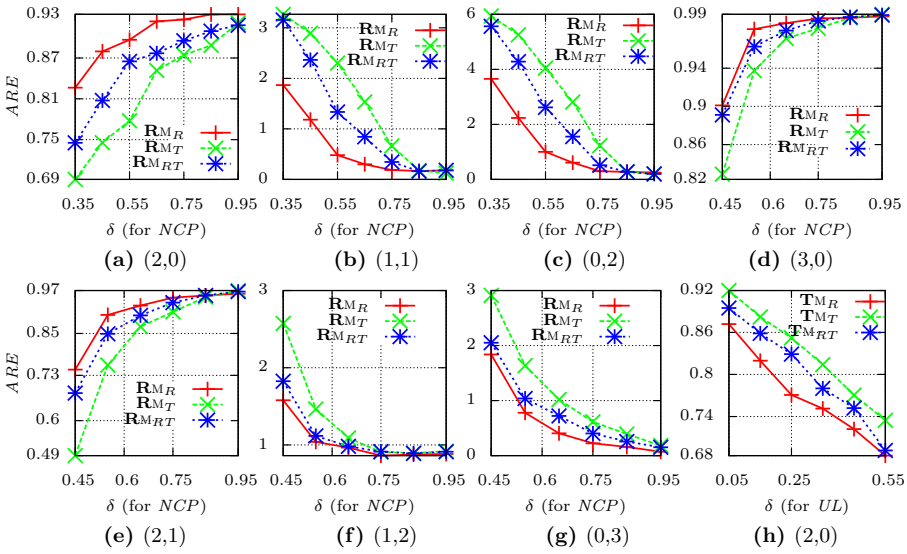

Fig. 4: ARE for queries involving (x, y) relational values and items. Figs. (a)-(c) are for INFORMS; (d)-(g) for YOUTUBE (**Rum**-BOUND). Fig. (h) is for INFORMS (**Tum**-BOUND)

$\mathbf{T}\text{M}_R$, and $\mathbf{T}\text{M}_{RT}$ (see e.g., Fig. 4h), and from comparing the (k, l^m)-diversity algorithms (see Figs. 5). Figs. 6a and 6b show the size of the largest cluster created by $\mathbf{R}\text{M}_R$, $\mathbf{R}\text{M}_T$, and $\mathbf{R}\text{M}_{RT}$, for varying δ. $\mathbf{R}\text{M}_R$ created the largest clusters, as it merges many clusters with similar relational values. These clusters have low *UL*, as shown in Figs. 6c and 6d. Furthermore, Figs. 6a and 6c, show that $\mathbf{R}\text{M}_{RT}$ created slightly larger clusters than $\mathbf{R}\text{M}_T$, which have lower *UL* scores. The results for $\mathbf{T}\text{M}_T$, $\mathbf{T}\text{M}_R$, and $\mathbf{T}\text{M}_{RT}$ and the (k, l^m)-diversity algorithms were similar.

EFFICIENCY. We studied the impact of dataset size using random subsets of INFORMS, whose records were contained in all larger sets. As can be seen in Fig. 7a, $\mathbf{R}\text{M}_T$ outperformed $\mathbf{R}\text{M}_R$ and $\mathbf{R}\text{M}_{RT}$, and it was more scalable, due to the use of the *BTD* measure. $\mathbf{R}\text{M}_{RT}$ was the slowest, because it computes two cluster orderings. $\mathbf{T}\text{M}_T$, $\mathbf{T}\text{M}_R$, and $\mathbf{T}\text{M}_R$ perform similarly to $\mathbf{R}\text{M}_R$, $\mathbf{R}\text{M}_T$, and $\mathbf{R}\text{M}_{RT}$ (their results were omitted). Fig. 7a shows the cost of CL. We also studied the impact of k using the largest dataset of the previous experiment. Fig. 7b shows that the runtime of $\mathbf{R}\text{M}_R$, $\mathbf{R}\text{M}_T$, and $\mathbf{R}\text{M}_{RT}$ improves with k, as fewer clusters are merged. $\mathbf{R}\text{M}_T$ was up to 2.2 times more efficient than $\mathbf{R}\text{M}_R$ and $\mathbf{R}\text{M}_{RT}$ was the least efficient. Fig. 7b shows that the runtime of CL improves with k. The cost of the (k, l^m)-diverse algorithms was similar (omitted).

BENEFITS OF ALGORITHMIC CHOICES. To show that *BTD* helps efficiency without degrading data utility, we developed the baseline algorithms $\mathbf{R}\text{M}_{T\,UL}$, $\mathbf{R}\text{M}_{T\,UL}$, $\mathbf{T}\text{M}_{T\,UL}$, and $\mathbf{T}\text{M}_{RT\,UL}$, which do not perform the optimization of Section 5. Due to their high runtime, a subset of INFORMS with $4K$ records was used. Observe in Figs. 7c and 7e that $\mathbf{R}\text{M}_T$ and $\mathbf{R}\text{M}_{RT}$ have the same *UL* scores with their

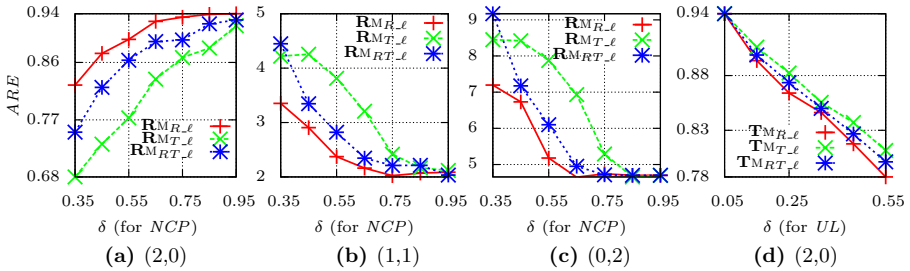

Fig. 5: ARE for queries involving (x, y) relational values and items. Figs. (a)-(c) are for INFORMS (**Rum**-BOUND); Fig. (d) is for INFORMS (**Tum**-BOUND)

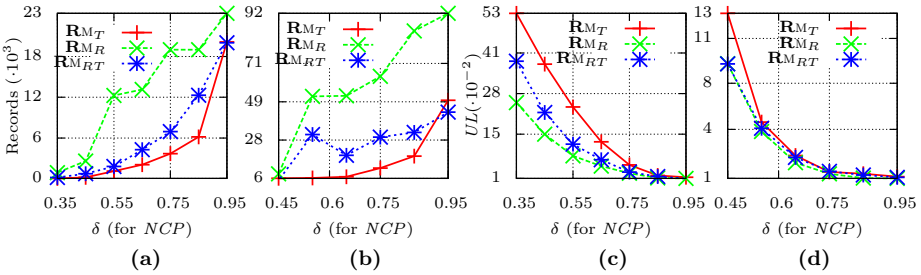

Fig. 6: Max. cluster size for (a) INFORMS, and (b) YOUTUBE, and *UL* for (c) INFORMS and (d) YOUTUBE (**Rum**-BOUND)

corresponding baseline algorithms, but are at least 10 times more efficient and scalable with respect to δ. Similar observations can be made from Figs. 7d and 7f, for $\mathbf{T}\text{M}_R$ and $\mathbf{T}\text{M}_{RT}$. Last, we show that *UL* decreases monotonically, as our algorithms merge clusters. Figs. 7g-7h show the results with $\delta = 1$ for the dataset used in the previous experiment. The fact that *UL* never increases shows that avoiding to compute $UL(\mathcal{T}(\mathcal{D}_{tmp}))$ after a cluster merge does not impact data utility but helps efficiency. The (k, l^m)-diversity algorithms performed similarly.

7 Related work

Preventing identity disclosure is crucial in many real-world applications [5,11] and can be achieved through k-anonymity [15]. This privacy principle can be enforced through various generalization-based algorithms (see [5] for a survey). Thwarting *attribute disclosure* may additionally be needed [14,19,17], and this can be achieved by applying other privacy models, such as l-diversity [14], together with k-anonymity.

Privacy-preserving transaction data publishing requires new privacy models and algorithms, due to the high dimensionality and sparsity of transaction data [19,7,17]. k^m-anonymity is a model for protecting transaction data against attackers, who know up to m items about an individual [17]. Under this condition, which is often satisfied in applications [17,16,11], an individual cannot be associated with fewer than k records in the dataset. k^m-anonymity can be enforced using several algorithms [17,12,8], which can be incorporated into our frameworks. However, k^m-anonymity does not guarantee protection against stronger

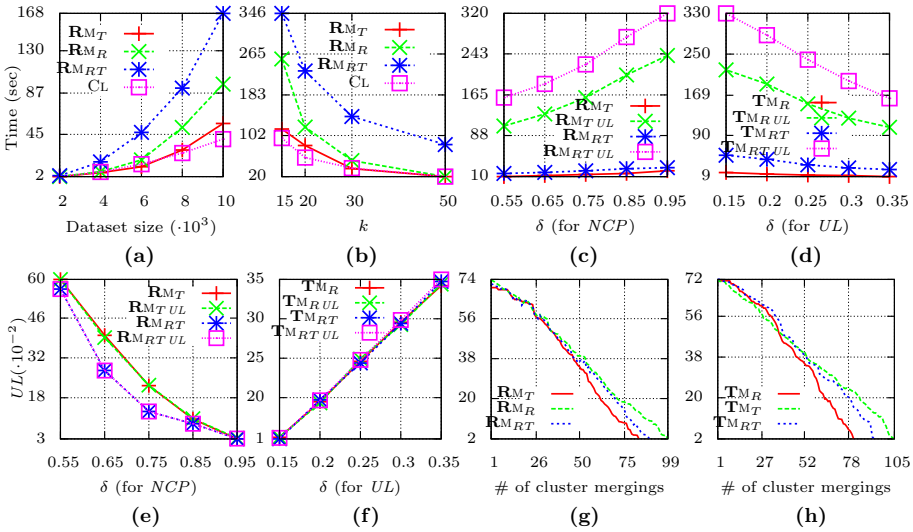

Fig. 7: Runtime (a) vs. $|D|$ and (b) vs. k. Impact of using BTD on runtime for (c) **Rum**-BOUND and (d) (**Tum**-BOUND), and on UL for (e) **Rum**-BOUND and (f) **Tum**-BOUND. UL vs. number of cluster mergings for (g) **Rum**-BOUND, and (h) **Tum**-BOUND

attackers, who know that an individual is associated with exactly certain items [17,16]. This is because, by excluding records that have exactly these items from consideration, the attackers may be able to increase the probability of associating an individual with their record to greater than $\frac{1}{k}$ (although not necessarily 1). A recent method [20] can guard against such attackers while preserving data utility based on a *nonreciprocal recoding* anonymization scheme. To thwart both identity and attribute disclosure in transaction data publishing, [17] proposes ℓ^m-diversity, which we also employ in our frameworks.

Our frameworks employ generalization, which incurs lower information loss than suppression [17] and helps preventing identity disclosure, contrary to bucketization [7]. Also, we seek to publish record-level and truthful data. Thus, we do not employ ϵ-differential privacy [3], nor disassociation [16]. However, the relationship between (k, k^m)-anonymization and relaxed differential privacy definitions is worth investigating to strengthen protection. For instance, Li et al. [10] proved that *safe* k-anonymization algorithms, which perform data grouping and recoding in a differentially private way, can satisfy a relaxed version of differential privacy when preceded by a random sampling step.

8 Conclusions

In this paper, we introduced the problem of anonymizing RT-datasets and proposed the first approach to protect such datasets, along with two frameworks for enforcing it. Three cluster-merging algorithms were developed, for each framework, which preserve different aspects of data utility. Last, we showed how our approach can be extended to prevent both identity and attribute disclosure.

Acknowledgements

G. Poulis is supported by the Research Funding Program: Heraclitus II. G. Loukides is partly supported by a Research Fellowship from the Royal Academy of Engineering. S. Skiadopoulos is partially supported by EU/Greece the Research Funding Program: Thales.

References

1. R.J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, pages 217–228, 2005.
2. J-W. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymization using clustering techniques. In *DASFAA*, pages 188–200, 2007.
3. C. Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.
4. A.A. Freitas. A critical review of multi-objective optimization in data mining: a position paper. *SIGKDD Explorations*, 6(2):77–86, 2004.
5. B.C.M. Fung, K. Wang, R. Chen, and P.S. Yu. Privacy-preserving data publishing: A survey on recent developments. *ACM Comput. Surv.*, 42, 2010.
6. G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis. A framework for efficient data anonymization under privacy and accuracy constraints. *TODS*, 34(2), 2009.
7. G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In *ICDE*, pages 715–724, 2008.
8. A. Gkoulalas-Divanis and G. Loukides. Utility-guided clustering-based transaction data anonymization. *Trans. on Data Privacy*, 5(1):223–251, 2012.
9. K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE*, page 25, 2006.
10. N. Lii, W. Qardaji, and D. Su. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *ASIACCS*, pages 32–33, 2012.
11. G. Loukides, A. Gkoulalas-Divanis, and B. Malin. Anonymization of electronic medical records for validating genome-wide association studies. *Proceedings of the National Academy of Sciences*, 17:7898–7903, 2010.
12. G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. *KAIS*, 28(2):251–282, 2011.
13. G. Loukides and J. Shao. Clustering-based k-anonymisation algorithms. In *DEXA*, pages 761–771, 2007.
14. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *ICDE*, page 24, 2006.
15. P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *PODS*, page 188, 1998.
16. M. Terrovitis, J. Liagouris, N. Mamoulis, and S. Skiadopoulos. Privacy preservation by disassociation. *PVLDB*, 5(10):944–955, 2012.
17. M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *VLDB J.*, 20(1):83–106, 2011.
18. J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A.W-C. Fu. Utility-based anonymization using local recoding. In *KDD*, pages 785–790, 2006.
19. Y. Xu, K. Wang, A.W-C. Fu, and P.S. Yu. Anonymizing transaction databases for publication. In *KDD*, pages 767–775, 2008.
20. M. Xue, P. Karras, C. Raïssi, J. Vaidya, and K. Tan. Anonymizing set-valued data by nonreciprocal recoding. In *KDD*, pages 1050–1058, 2012.