Distance-based k^m-anonymization of trajectory data

Giorgos Poulis*, Spiros Skiadopoulos*, Grigorios Loukides[†], Aris Gkoulalas-Divanis[‡]
* University of Peloponnese, Email: {poulis,spiros}@uop.gr
[†] Cardiff University, Email: g.loukides@cs.cf.ac.uk
[‡] IBM Research - Ireland, Email: arisdiva@ie.ibm.com

***Abstract*—The publication of trajectory data opens up new directions in studying human behavior, but it is challenging to perform in a privacy-preserving way. This is mainly because, the identities of individuals, whose movement is recorded in the data, can be disclosed, even after removing identifying information. Existing works to anonymize trajectory data offer privacy, but at a high data utility cost. This is because, they either do not produce truthful data, which is important in many applications, or are limited in their privacy specification component. This paper proposes an approach that overcomes these shortcomings by adapting k^m-anonymity to trajectory data and by using distance-based generalization. We also develop an effective and efficient anonymization algorithm, which is based on the apriori principle. Our experiments verify that this algorithm preserves data utility well, and it is fast and scalable.**

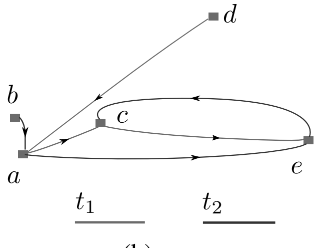
I. Introduction

The widespread use of GPS-enabled smartphones and location-based social networking applications, such as Foursquare (https://foursquare.com), opens up new opportunities in understanding human behavior through the analysis of collected mobility data. However, the publication of these data, which correspond to trajectories of personal movement (i.e., ordered lists of locations visited by individuals), can lead to *identity disclosure*, even if identifying information (ID) is not published [23]. The values that, in combination, may lead to identity disclosure are called *quasi-identifiers* (QI) [24], [22]. For example, let us assume that a location-based social network service, publishes the movement of users during a day in form of checkins in various locations. An example of this data is shown in Fig. 1a. If Mary's colleague, John, knows that sometime that day, Mary checked in at locations a and d, he cannot associate Mary with her record (trajectory), as both trajectories t_1 and t_3 include the locations a and d. But if John knew that Mary first visited d and then a, he can accurately link Mary with her trajectory t_1.

This example highlights not only the need to transform a set of user trajectories $\mathcal{T}$ to prevent identity disclosure, based on partial location knowledge held by adversaries, but also the difference from well-studied set-valued data anonymity models, like k^m-anonymity [26] and privacy-constrained anonymization [17], [11]. In these models, value ordering is not significant; thus records are represented as unordered sets of items. For instance, if an adversary knew that someone visited location c and then e, they could link this individual only to record t_1 (Fig. 1b). On the other hand, if $\mathcal{T}$ was a set-valued dataset, records t_1, t_2 and t_4 would have items c and e, hiding this individual's identity among 3 records. Consequently, for any set of n items in a trajectory, there are $n!$ possible quasi-identifiers.

id	trajectory
t_1	(d, a, c, e)
t_2	(b, a, e, c)
t_3	(a, d, e)
t_4	(b, d, e, c)
t_5	(d, c)
t_6	(d, e)

(a)



(b)

Fig. 1: (a) the original database $\mathcal{T}$ (b) visual representation of trajectories t_1 and t_2

This difference makes anonymizing trajectory datasets more challenging, as it drastically increases the number of potential quasi-identifiers. Existing methods operate either by (i) anonymizing each trajectory as a whole, thereby not assuming any specific background knowledge of attackers [1], [2], [18], [21], or (ii) by anonymizing parts of the user trajectories by considering attackers who can effectively link specific locations to individuals in order to re-identify them [25], [28]. The first category of approaches are based on *clustering and perturbation* [1], [2], [21], while the second category employs *generalization and suppression* of quasi-identifiers [19], [28], [20], [25].

The main drawback of clustering-based approaches is that they may lose information about the direction of movement of co-clustered trajectories, as well as cause excessive information loss, due to space translation. Moreover, applying perturbation to datasets, creates data that are not *truthful* and cannot be used in several applications [9]. Similarly, existing generalization-and-suppression based methods [19], [28], [20], [25] have the following limitations. First, they assume that quasi-identifiers are known to the data publisher prior to anonymization [28], [25] (e.g., defined by users) or that any combination of locations can act as a quasi-identifier [19]. Second, they require a location taxonomy to be specified by data publishers [20], based on locations' semantics. However, this taxonomy may not well reflect the distance between locations and therefore the anonymized data may incur a high amount of information loss. Last, some approaches assume that each location can be classified as either sensitive or non-sensitive [20]. In practice, however, this assumption may not hold, as location sensitivity depends on context (e.g. visiting a hospital may be sensitive for a patient, but not for a doctor).

Our proposed approach addresses the aforementioned shortcomings by adapting k^m-anonymity [26] to trajectory data and by applying generalization in a way that minimizes the distance between the original and the anonymized user trajectories. k^m-anonymity is a privacy model that was proposed to limit the probability of identity disclosure in transaction data publishing. The benefit of this model is that it does not require detailed knowledge of quasi-identifiers, or a distinction between sensitive and non-sensitive information, prior to publication. At the same time, our approach avoids the use for a location taxonomy and generalizes trajectories in a way that preserves data utility.

The rest of the paper is organized as follows. Section II discuss related work. Section III formulates the problem and Section IV presents our anonymization algorithm. Section V presents the experimental evaluation of our algorithm in terms of utility and efficiency. Finally, Section VI concludes the paper.

II. Related work

k-anonymity is a privacy model that prevents identity disclosure, by requiring at least k records of a dataset to have the same values over QI [24], [23], [16], [12], [13], [27]. Thus, a k-anonymous dataset upperbounds the probability of associating an individual to their record by $\frac{1}{k}$. To enforce k-anonymity, most works [22], [14], [15], [10] employ *generalization*, which replaces a QI value with a more general but semantically consistent value, or *suppression*, which removes QI values prior to data publishing.

The k-anonymity principle has been recently considered in the context of publishing user trajectories, leading to several trajectory anonymization methods [4]. These methods operate either by (i) anonymizing each trajectory as a whole [1], [2], [18] or (ii) by anonymizing parts of the user trajectories by considering attackers who can link specific locations to individuals in order to perform identity disclosure [25], [28]. The approaches of the first category operate by grouping original trajectories into clusters of k members in a way that each trajectory within a cluster becomes indistinguishable from the other trajectories in the cluster. One such method, called NWA [1], enforces (k, δ)-anonymity to anonymize user trajectories by generating cylindrical volumes of radius δ that contain at least k trajectories. Each trajectory that belongs to an anonymity group (cylinder), generated by NWA, is protected from identity disclosure, due to the other trajectories that appear in the same group. To produce the cylindrical volumes, the anonymity algorithm proposed in [1] identifies trajectories that lie close to each other in time and employs space translation.

The second category of approaches considers attackers with background knowledge on ordered sequences of places of interest (POIs) visited by specific individuals. Terrovitis et al. [25] proposed an approach to prohibit multiple attackers, each knowing a different set of POIs, from associating these POIs to fewer than k individuals in the released dataset. To achieve this, the authors developed a suppression-based method that aims at removing the least number of POIs from user trajectories so that the remaining trajectories are k-anonymous with respect to the knowledge of each adversary.

Yarovoy et al. [28] proposed a k-anonymity based approach for publishing user trajectories by considering time as a quasi-identifier and supporting privacy personalization. Unlike previous work that assumed that all users share a common quasi-identifier, [28] assumes that each user has a different set of POIs and times for which he or she requires protection, thereby enabling each trajectory to be protected differently. To achieve k-anonymity, this approach uses generalization and creates anonymization groups that are not necessarily disjoint.

A recent approach, proposed by Monreale et al. [19], extends the l-diversity principle to trajectories by assuming that each location is either nonsensitive (acting as a QI) or sensitive. This approach applies c-safety to prevent adversaries from linking sensitive locations to trajectories with a probability greater than c. To enforce c-safety, the proposed algorithm applies generalization to replace original POIs with generalized ones based on a location taxonomy. If generalization alone cannot enforce c-safety, suppression is used.

Contrary to related work on trajectory anonymization approaches operating through data generalization or suppression of quasi-identifiers, our method makes the realistic assumption that an adversary may have knowledge of up to m locations that a user has visited. To protect the trajectories, we employ a distance-based generalization approach that does not depend on a pre-specified location taxonomy. Moreover, in this work we refrain from classifying locations as sensitive or nonsensitive (QI), and prevent identity disclosure, based on any combination of up to m locations.

Recently, differential privacy [8] methods were proposed to anonymize sequential datasets [6], [7]. These methods focus on specific data analytic tasks, such as query answering or frequent pattern mining [3] and work by adding noise to the data. Thus, they harm data truthfulness, which is essential to preserve in many data analysis tasks [17].

III. Problem formulation

Let $\mathcal{L}$ be a set of locations (points of interest, touristic sites, shops, etc.).

Definition 1: A *trajectory* t is an ordered list of locations $(l_1, \ldots, l_n)$, where $l_i \in \mathcal{L}$, $1 \leq i \leq n$. The *size* of the trajectory $t = (l_1, \ldots, l_n)$, denoted by $|t|$, is the number of its locations, i.e., $|t| = n$.

A trajectory represents the locations and the order these locations are visited by a moving object (individual, bus, taxi, etc.). In our setting a location may also model points in space (1D, 2D, 3D, etc.) and even incorporate a temporal dimension.

Definition 2: A trajectory $s = (\lambda_1, \ldots, \lambda_\nu)$ is a *subtrajectory of* or is *contained in* trajectory $t = (l_1, \ldots, l_n)$, denoted by $s \sqsubseteq t$, if and only if $|s| \leq |t|$ and there is a mapping f such that $\lambda_1 = l_{f(1)}, \ldots, \lambda_\nu = l_{f(\nu)}$ and $f(1) < \cdots < f(\nu)$.

Thus, a subtrajectory is formed by removing some locations from the original trajectory, while maintaining the order of the remaining locations. For instance, the trajectory (a, e) is contained in $t_1 = (d, a, c, e)$ (Fig. 1).

Definition 3: Given a set of trajectories $\mathcal{T}$, the *support* of a subtrajectory s, denoted by $\sup(s, \mathcal{T})$, is defined as the number of distinct trajectories in $\mathcal{T}$ that contain s.

In other words, the support of a subtrajectory s measures the number of trajectories in a dataset that s is contained in. For example, for the dataset in Fig. 1a, we have $\sup((a,e),\mathcal{T}) = 3$. Naturally, by considering locations as unary trajectories, the support can also be measured for the locations of a dataset.

In this work, we adapt the notion of k^m-anonymity to trajectory data, as explained below.

Definition 4: A set of trajectories $\mathcal{T}$ is k^m*-anonymous* if and only if every subtrajectory s of every trajectory $t \in \mathcal{T}$, which contains m or fewer locations (i.e., $|s| \leq m$), is contained in at least k distinct trajectories of $\mathcal{T}$.

Definition 4 ensures that an attacker, who knows any subtrajectory s of size m of an individual, cannot associate the individual to fewer than k trajectories (i.e., the probability of identity disclosure, based on s, is at most $\frac{1}{k}$).

Example 1: Consider the dataset of trajectories $\mathcal{T}$ depicted in Fig. 1a. $\mathcal{T}$ is 2^1-anonymous and 1^3-anonymous. However, it is not 2^2-anonymous, as the subtrajectory (d,a) is contained only in the trajectory t_1 of $\mathcal{T}$.

To explain the way we generalize trajectories, we define the notion of generalized location as follows.

Definition 5: A *generalized location* $\{l_1,\ldots,l_v\}$, is defined as a set of at least two locations $l_1,\ldots,l_v \in \mathcal{L}$.

A *generalized location* is interpreted as any of it's locations. Therefore, if a trajectory t in an anonymized version $\mathcal{T}'$ of $\mathcal{T}$ contains a generalized location $\{l_1,\ldots,l_v\}$, then the trajectory t in $\mathcal{T}$ contains exactly one location among $l_1,\ldots,l_v$.

To enforce k^m-anonymity, we either generalize a location l to a generalized location that contains l or leave l intact.

We are interested in generalization transformations that distort as little as possible the initial dataset $\mathcal{T}$. A common way to measure the distortion of a transformation is to measure the *distance* between the original and the transformed dataset [25], [21], [28]. In our case, the distance between the initial and the anonymized dataset is defined as the *average* of the distances of their corresponding trajectories. In turn, the distance between the initial and the anonymized trajectory is defined as the *average* of the distance between their corresponding locations. In more detail, we have.

Definition 6: Let l be a location that will be generalized to the generalized location $\{l_1,\ldots,l_v\}$. The *location distance* between l and $\{l_1,\ldots,l_v\}$, denoted by $\mathcal{D}_{loc}(l,\{l_1,\ldots,l_v\})$, is defined as:

$$\mathcal{D}_{loc}(l,\{l_1,\ldots,l_v\}) = \text{avg}\big\{\mathit{EuclDist}(l,l_i) \mid 1 \leq i \leq v\big\}$$

where *EuclDist* is the Euclidean distance. The *trajectory distance* between $t = (l_1,\ldots,l_n)$ and its generalized counterpart $t' = (l'_1,\ldots,l'_n)$, denoted by $\mathcal{D}_{traj}(t,t')$, is defined as:

$$\mathcal{D}_{traj}(t,t') = \text{avg}\big\{\mathcal{D}_{loc}(l_i,l'_i) \mid 1 \leq i \leq n\big\}$$

Finally, the *trajectory dataset distance* between $\mathcal{T} = \{t_1,\ldots,t_u\}$ and its generalized counterpart $\mathcal{T}' = \{t'_1,\ldots,t'_u\}$ (where the trajectory t_i is generalized to trajectory t'_i, $1 \leq i \leq u$), denoted by $\mathcal{D}(\mathcal{T},\mathcal{T}')$, is defined as:

$$\mathcal{D}(\mathcal{T},\mathcal{T}') = \text{avg}\big\{\mathcal{D}_{traj}(t_i,t'_i) \mid 1 \leq i \leq u\big\}$$

Algorithm: SEQANON

Input: A dataset $\mathcal{T}$ and anonymization parameters k and m
Output: A k^m-anonymous dataset $\mathcal{T}'$ corresponding to $\mathcal{T}$

```
1 $\mathcal{T}' := \mathcal{T}$ // Initialize output
2 for $i := 1$ to $m$ do
3     Let $\mathcal{S}$ be the set of subtrajectories $s$ of $\mathcal{T}$ with size $i$ such
      that $\sup(s,\mathcal{T}') < k$ sorted by increasing support
4     for each $s \in \mathcal{S}$ do
5         while $\sup(s,\mathcal{T}') < k$ do
6             Find the location $l_1$ of $s$ with the minimum
              support in $\mathcal{T}'$
7             Find the location $l_2 \neq l_1$ with the minimum
              distance from $l_1$
8             Replace all occurrences of $l_1$ and $l_2$ in $\mathcal{T}'$ and $s$
              with $\{l_1,l_2\}$
9 return $\mathcal{T}'$
```

For example, let a, a_1, a_2 and b be locations and let $\mathit{EuclDist}(a,a_1) = 1$ and $\mathit{EuclDist}(a,a_2) = 2$. If location a is generalized to the generalized location $\{a,a_1,a_2\}$ the location distance $\mathcal{D}_{loc}(a,\{a,a_1,a_2\}) = (0+1+2)/3 = 1$. Also, if trajectory (a,b) is generalized to $(\{a,a_1,a_2\},b)$ the trajectory distance $\mathcal{D}_{traj}((a,b),(\{a,a_1,a_2\},b)) = (1+0)/2$.

Note that the distances in Definition 6 can be normalized by dividing each of them with the maximum distance between locations in $\mathcal{T}$.

The problem we consider can be expressed as follows.

Problem 1: Given a dataset of trajectories $\mathcal{T}$ construct a k^m-anonymous version $\mathcal{T}'$ of $\mathcal{T}$ such that $\mathcal{D}(\mathcal{T},\mathcal{T}')$ is minimized.

In the rest of the paper, we present and evaluate a method to tackle Problem 1.

IV. ANONYMIZATION ALGORITHM

Given an input set of trajectories $\mathcal{T}$, we will present a method that transforms $\mathcal{T}$ into a k^m-anonymous set of trajectories $\mathcal{T}'$ corresponding to $\mathcal{T}$ by generalizing the locations of the trajectories that do not satisfy the k^m-anonymity metric.

The proposed anonymization method is illustrated in Algorithm SEQANON that takes as input a trajectories dataset $\mathcal{T}$ and the anonymization parameters k and m and returns the k^m-anonymous counterpart $\mathcal{T}'$ of $\mathcal{T}$. The algorithm works in an apriori, bottom up fashion. Initially, it considers and generalizes the subtrajectories in $\mathcal{T}$ of size 1 (i.e., single locations) that have low support. Then, SEQANON continues by progressively increasing the size of the subtrajectories it considers.

In more detail, SEQANON proceeds as follows. First, SEQANON initializes $\mathcal{T}'$ (Step 1). Then, Steps 2 – 2 follow the apriori principle. Step 3 computes set $\mathcal{S}$ containing the subtrajectories s of $\mathcal{T}$ having size i (i.e., having i locations) and lower support than the anomymization parameter k (i.e., $\sup(s,\mathcal{T}') < k$). SEQANON considers the lower support subtrajectories of $\mathcal{S}$ first. This tactic improves the efficiency of

subT.	sup
(d,a)	1
(c,e)	1
(b,a)	1
(a,d)	1
(b,d)	1

(a)

id	trajectory
t'_1	$(d,\{a,b\},c,e)$
t'_2	$(\{a,b\},\{a,b\},e,c)$
t'_3	$(\{a,b\},d,e)$
t'_4	$(\{a,b\},d,e,c)$
t'_5	(d,c)
t'_6	(d,e)

(b)

id	trajectory
t'_1	$(d,\{a,b,c\},\{a,b,c\},e)$
t'_2	$(\{a,b,c\},\{a,b,c\},e,\{a,b,c\})$
t'_3	$(\{a,b,c\},d,e)$
t'_4	$(\{a,b,c\},d,e,\{a,b,c\})$
t'_5	$(d,\{a,b,c\})$
t'_6	(d,e)

(c)

Fig. 2: (a) Set $\mathcal{S}$ for subtrajectories of size $i=2$ and the respective supports, (b) Transformed dataset $\mathcal{T}'$ after the processing of subtrajectory (d,a), and (c) The final 2^2-anonymous result $\mathcal{T}'$

the method and the quality of the results. Remedying the lower support subtrajectories commonly benefits higher support subtrajectories while at the same time, their generalization does not significantly affect the dataset. Continuing, for every such trajectory $s \in \mathcal{S}$, the algorithm finds the location l_1 of s with the minimum support (Step 6). Similarly to subtrajectories, we consider lower support locations first. Then, the algorithm searches the locations of $\mathcal{T}$ to detect the closest location l_2 to l_1 (Step 7). Finally, SeqAnon generalizes l_1 and l_2 by constructing the generalized location $\{l_1,l_2\}$ and replaces every occurrence of l_1 and l_2 with the generalized location $\{l_1,l_2\}$ (Step 8). The algorithm repeats Steps 6 – 8 until the support of the subtrajectory s exceeds the anonymization parameter k.

The following is an example of SeqAnon in operation.

Example 2: We will demonstrate the operation of SeqAnon with input the dataset $\mathcal{T}$ of Fig. 1a and $k=m=2$. The intermediate steps are illustrated in Fig. 2. The first iteration of the for loop (Steps 2 – 2) considers the subtrajectories of size $i=1$. It is not hard to verify that all size 1 locations have support greater than $k=2$, thus, the algorithm proceeds to $i=2$. For this case, Step 3 computes the set of low support subtrajectories $\mathcal{S}$ (illustrated in Fig. 2a). SeqAnon considers subtrajectory $s=(d,a)$, which is the first subtrajectory in $\mathcal{S}$. Then, Step 6 sets $l_1=a$ (since a is the lowest support location of (d,a)) and Step 7 sets $l_2=b$ (since location b is closer to a – see also the map of Fig. 1b). Finally, Step 8 replaces a and b with the generalized location $\{a,b\}$ in s and all the trajectories of $\mathcal{T}'$. After these replacement, we have $s=(d,\{a,b\})$ while $\mathcal{T}'$ is depicted in Fig. 2b. Since, for the changes values of s and $\mathcal{T}'$ we still have $\sup(s,\mathcal{T}')<k$, the while loop (Steps 5 – 8) is executed again. This time $l_1=\{a,b\}$, $l_2=c$ and after the replacements $\mathcal{T}'$ is depicted in Fig. 2c. The remaining steps of the algorithm SeqAnon do not change $\mathcal{T}'$, thus, Fig. 2c illustrates the final output.

Complexity analysis. Algorithm SeqAnon executes the **for** loop (Steps 2 – 8). For each iteration of this loop, set $\mathcal{S}$ is constructed and explored. The size of $\mathcal{S}$ is, in the worst case, $\mathcal{O}(|\mathcal{L}|^i)$, where $|\mathcal{L}|$ is the size of the location set used in $\mathcal{T}$ and i is the loop counter. The above bound is a very crude approximation. $\mathcal{O}(|\mathcal{L}|^i)$ is the size of $\mathcal{S}$ when all size i subtrajectories have support lower than k. This is hardly the case; the actual sutrajectories s with $\sup(s,\mathcal{T}')<k$ are a small fraction of $\mathcal{O}(|\mathcal{L}|^i)$. This number depends heavily on the dataset $\mathcal{T}$ and the value of the anonymization parameter k. To have a more precise bound we multiply with the factor $p(i,k,\mathcal{T})$ which measures the probability of a subtrajectory of size i having support less than k in dataset $\mathcal{T}$. Thus, the size of $\mathcal{S}$ is bounded by $\mathcal{O}(p(i,k,\mathcal{T})\cdot|\mathcal{L}|^i)$. For each element s of $\mathcal{S}$ the **while** loop (Steps 5 – 8) is executed. This loop takes $\mathcal{O}(|s|)=\mathcal{O}(i)$ time in the worst case (i.e., when we are going to generalize all locations of the trajectory). Overall, each iteration of the **for** loop takes $\mathcal{O}(i\cdot p(i,k,\mathcal{T})\cdot|\mathcal{L}|^i)$ time. Thus, in total, the complexity of the SeqAnon algorithm is $\mathcal{O}\big(\sum_{i=1}^{m}(i\cdot p(i,k,\mathcal{T})\cdot|\mathcal{L}|^i)\big)=\mathcal{O}\big(m\cdot p(k,\mathcal{T})\cdot|\mathcal{L}|^m\big)$ where $p(k,\mathcal{T})$ averages $p(1,k,\mathcal{T}),\ldots,p(m,k,\mathcal{T})$.

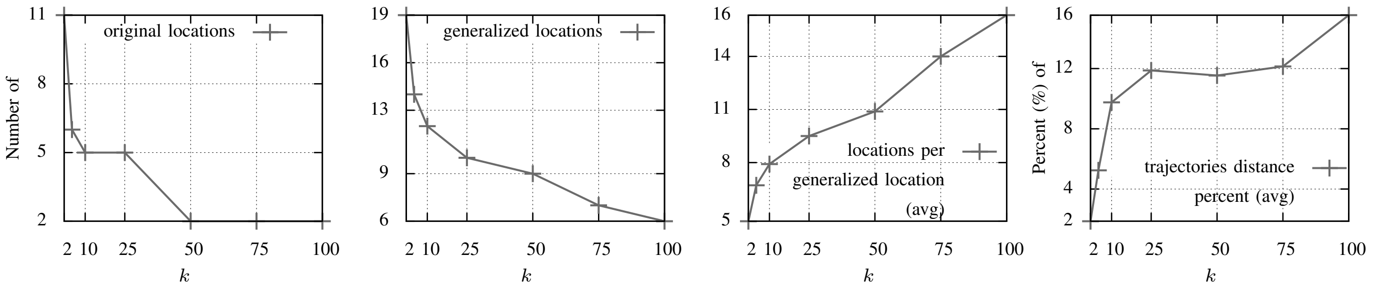
V. Experimental evaluation

In this section, we evaluate our algorithm in terms of data utility and efficiency.

Experimental setup. We implemented our algorithm in C++ and tested it on an Intel Core i7 at 2.2 GHz with 6 GB of RAM. We generated synthetic trajectories of moving objects on Oldenburg city map using Brinkhoff's generator [5]. This setting has been used by many works [1], [21], [28], [25]. We then normalized trajectories so that all coordinates take values in a $10^3\times10^3$ map and we simulated trajectories corresponding to these routes as follows. The map was divided into 100 regions using a uniform grid. An object visits a sequence of regions in a certain order. The centroids of the visited regions model the locations in the trajectories of $\mathcal{T}$. The average length of the generated trajectories is 4.72. The default number of locations of $\mathcal{L}$ and trajectories of $\mathcal{T}$ are 100 and 18,143 respectively. Unless otherwise stated, m is set to 2.

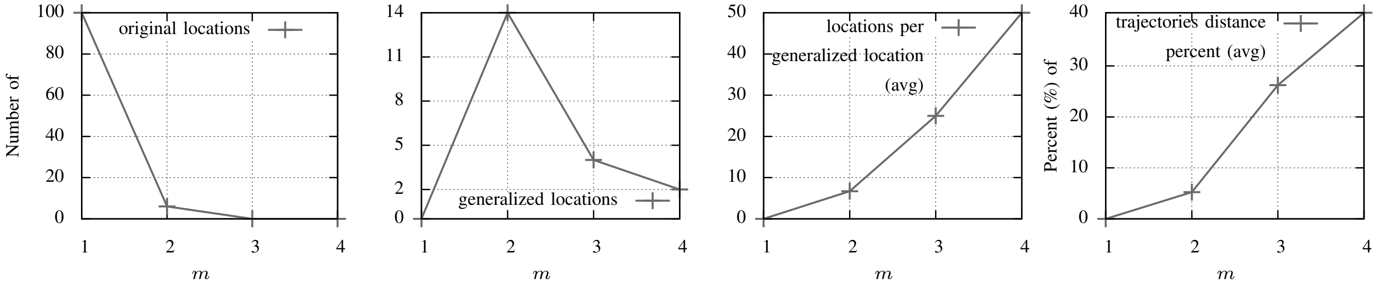
Data utility. To measure data utility, we evaluated the number of published original locations and the number of generalized locations. For the generalized locations, we also measure their average size and distance. Initially, we vary the anonymization parameter k in [2, 100]. Our results are summarized in Fig. 3.

In Fig. 3a, we present the number of the original locations published (i.e., locations that were not generalized) as a function of k. As expected, increasing k led to fewer original locations published. In Fig. 3b, we illustrate the number of generalized locations. When k increases, more locations are grouped together to ensure k^m-anonymity, leading to fewer generalized locations. As an immediate result, the average number of locations in a generalized location increased, as shown in Fig. 3c. Finally, we present the average distance



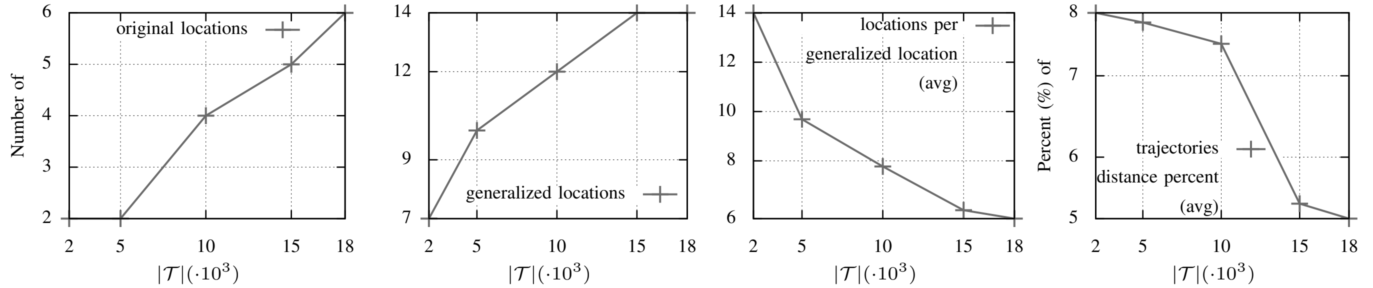
(a) (b) (c) (d)

Fig. 3: number of (a) original (non generalized) locations published and (b) generalized locations published, (c) average number of generalized locations size, (d) average percent of distance in generalized locations



(a) (b) (c) (d)

Fig. 4: number of (a) original (non generalized) locations published and (b) generalized locations published, (c) average number of generalized locations size, (d) average percent of distances in generalized locations



(a) (b) (c) (d)

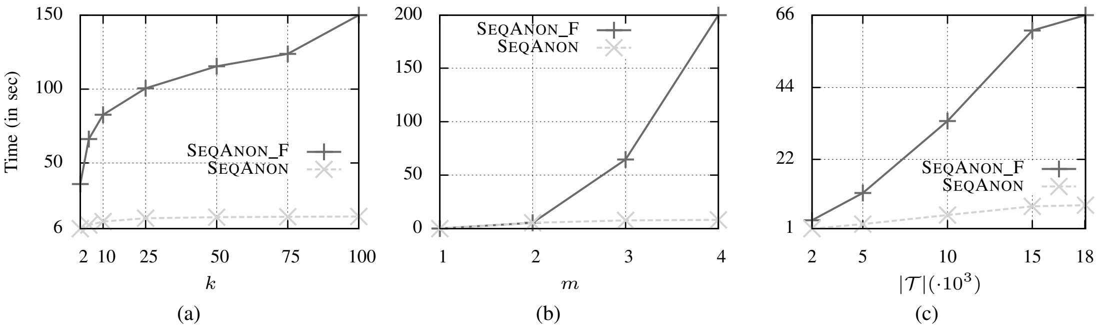
Fig. 5: number of (a) original (non generalized) locations published and (b) generalized locations published, (c) average number of generalized locations size, (d) average percent of distances in generalized locations

of all locations inside each generalized location from original location in $\mathcal{L}$. We normalize this distance as a percentage of the maximum possible distance (i.e., the distance between the furthermost points). This percentage quantifies the distance distortion in a generalized location. In Fig. 3d, we illustrate the distance percentage as a function of k. When k increases, more locations are grouped together in the same generalized location, leading to more distortion. As our algorithm focuses in minimizing the distance of locations in each generalized location, distortion is relatively low and increases slowly.

To show the impact of m on utility, we set $k = 5$ and varied m in [1,4]. Since our dataset has an average of 4.72 locations per trajectory, $m = 3$ (respectively, $m = 4$) means that the adversary knows approximately 65% (respectively, 85%) of a user's locations. So, for $m = 3$ and $m = 4$, we expect significant information loss. On the contrary, for $m = 1$, all locations have support greater than $k = 5$, so no generalization is performed and no generalized locations are created. As m increases, more generalizations are performed, in order to eliminate subtrajectories with low support. This leads to fewer generalized locations with larger sizes. These results are shown in Figs 4a-4d.

Also, we evaluated the impact of dataset size on data utility, using various random subsets of the original dataset containing 2,000, 5,000, 10,000, and 15,000 records. In Fig. 5a, we illustrate the number of original locations published for variable dataset sizes. For larger datasets, this number increases, as the support of single locations is higher. Consequently, the support of subtrajectories increases, and fewer locations are generalized. This leads to more generalized locations, with lower average size, and lower distance, as can be seen in Figs 5b-5d.

Efficiency. We studied the impact of the anonymization parameters k and m, and the dataset size on efficiency. To highlight the impact on efficiency of the apriori principle used by our algorithm, we created a version of Algorithm



(a) (b) (c)

Fig. 6: Runtime of Algorithms SeqAnon and SeqAnon_F for (a) variable k, (b) variable m and (c) variable $|\mathcal{T}|$

SeqAnon, denoted by SeqAnon_F, which does not use the apriori principle. In this version, we removed the **for** loop from Step 2 of SeqAnon and set $i = m$. In other words, Algorithm SeqAnon_F tries to eliminate directly subtrajectories of size m with low support. At first, we evaluated our algorithm using various k values, as in the experiments above. As we illustrate in Fig. 6a, the execution time increases with k. As expected, greater values for k lead to more subtrajectories with a lower support than k, resulting in a $\mathcal{S}$ of increased size (see also Step 3 of SeqAnon). Similarly, m has the same affect on execution time. As m increases, our algorithm performs more iterations (Steps 2-2 of Algorithm SeqAnon). In Fig. 6b, we show the impact of m on efficiency. Our algorithm significantly outperforms SeqAnon_F, verifying that the apriori principle improves the efficiency rate for larger m values. Finally, in Fig. 6c, we illustrate the impact of dataset size in the execution time of SeqAnon. As expected, larger datasets require longer processing time, since SeqAnon needs to process more records.

VI. Conclusions

In this paper, we proposed a new approach to publishing trajectory data in a way that prevents identity disclosure. Our approach makes realistic privacy assumptions, as it adapts k^m-anonymity to trajectory data, and allows the production of truthful data that preserve important data utility characteristics, as it employs distance-based generalization. We also developed an anonymization algorithm that performs well in terms of data utility preservation, and it is fast and scalable, due to the use of apriori principle.

Acknowledgements

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

References

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376–385, 2008.

[2] O. Abul, F. Bonchi, and M. Nanni. Anonymization of moving objects databases by clustering and perturbation. *IS*, 35(8):884–910, 2010.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 3–14, 1995.

[4] F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Trajectory anonymity in publishing personal mobility data. *SIGKDD Explorations*, 13(1):30–42, 2011.

[5] T. Brinkhoff. A framework for generating network-based moving objects. *GeoInformatica*, 6(2):153–180, 2002.

[6] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 638–649, New York, NY, USA, 2012. ACM.

[7] R. Chen, B. C. M. Fung, and B. C. Desai. Differentially private trajectory data publication. *CoRR*, abs/1112.2020, 2011.

[8] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

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

[10] B. C. M. Fung, K. Wang, and P. Yu. Top-down specialization for information and privacy preservation. In *ICDE*, pages 205–216, 2005.

[11] A. Gkoulalas-Divanis and G. Loukides. PCTA: privacy-constrained clustering-based transaction data anonymization. In *PAIS*, pages 1–10, 2011.

[12] A. Gkoulalas-Divanis and G. Loukides. Utility-guided clustering-based transaction data anonymization. *Transactions on Data Privacy*, 5(1):223–251, 2012.

[13] A. Gkoulalas-Divanis, Y. Saygin, and D. Pedreschi. Privacy in mobility data mining. *SIGKDD Explorations*, 13(1):4–5, 2011.

[14] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD*, pages 49–60, 2005.

[15] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional *k*-anonymity. In *ICDE*, page 25, 2006.

[16] G. Loukides and A. Gkoulalas-Divanis. Utility-preserving transaction data anonymization with low information loss. *Expert Syst. Appl.*, 39(10):9764–9777, 2012.

[17] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. *Knowl. Inf. Systems*, 28(2):251–282, 2011.

[18] N. Mohammed, B. C. M. Fung, and M. Debbabi. Walking in the crowd: anonymizing trajectory data for pattern analysis. In *CIKM*, pages 1441–1444, 2009.

[19] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel. Movement data anonymity through generalization. *TDP*, 3(2):91–121, 2010.

[20] A. Monreale, R. Trasarti, D. Pedreschi, C. Renso, and V. Bogorny. C-safety: a framework for the anonymization of semantic trajectories. *TDP*, 4(2):73–101, 2011.

[21] M. E. Nergiz, M. Atzori, and Y. Saygin. Towards trajectory anonymization: a generalization-based approach. In *SPRINGL*, pages 52–61, 2008.

[22] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.

[23] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *PODS*, pages 188–, 1998.

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

[25] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*, pages 65–72, 2008.

[26] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *VLDB J.*, 20(1):83–106, 2011.

[27] Y. Xu, K. Wang, A. W.-C. Fu, and P. S. Yu. Anonymizing transaction databases for publication. In *KDD*, pages 767–775, 2008.

[28] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.

Privacy Preservation by Disassociation

Manolis Terrovitis IMIS, Research Center ‘Athena’
mter@imis.athena-innovation.gr

John Liagouris Dept. of Electrical and Comp. Eng., NTUA
liagos@dblab.ece.ntua.gr

Nikos Mamoulis
Dept. of Comp. Sci., Univ. of Hong Kong
nikos@cs.hku.hk

Spiros Skiadopoulos
Dept. of Comp. Sci., Univ. of Peloponnese
spiros@uop.gr

ABSTRACT

In this work, we focus on protection against identity disclosure in the publication of sparse multidimensional data. Existing multidimensional anonymization techniques *(a)* protect the privacy of users either by altering the set of quasi-identifiers of the original data (e.g., by generalization or suppression) or by adding noise (e.g., using differential privacy) and/or *(b)* assume a clear distinction between sensitive and non-sensitive information and sever the possible linkage. In many real world applications the above techniques are not applicable. For instance, consider web search query logs. Suppressing or generalizing anonymization methods would remove the most valuable information in the dataset: the original query terms. Additionally, web search query logs contain millions of query terms which cannot be categorized as sensitive or non-sensitive since a term may be sensitive for a user and non-sensitive for another. Motivated by this observation, we propose an anonymization technique termed *disassociation* that preserves the original terms but hides the fact that two or more different terms appear in the same record. We protect the users' privacy by disassociating record terms that participate in identifying combinations. This way the adversary cannot associate with high probability a record with a rare combination of terms. To the best of our knowledge, our proposal is the first to employ such a technique to provide protection against *identity disclosure*. We propose an anonymization algorithm based on our approach and evaluate its performance on real and synthetic datasets, comparing it against other state-of-the-art methods based on generalization and differential privacy.

1. INTRODUCTION

The anonymization of sparse multidimensional data in the form of transactional data (e.g., supermarket sales logs, credit card logs, web search query logs) poses significant challenges. Adversaries that have a part of a record as background knowledge are aided by the dataset's sparsity in identifying the original record. Consider, for example, a dataset D which contains records that trace web search query logs. Even without any direct identifier in the data (user's name or ID) the publication of D might lead to privacy breaches, if an attacker has background knowledge that associates some queries to a known user. Assume that John knows that Jane was interested in buying air tickets to New York, so he has a background knowledge consisting of terms New York and air tickets. If D is published without any modification then John can trace all records that contain both terms New York and air tickets. If only one such record exists, then John can easily infer that this record corresponds to Jane.

To counter such privacy leaks, several anonymization techniques have been proposed in the literature [5, 6, 11, 13, 14, 17, 19, 24, 27]. Most of these methods employ *generalization* [5, 13, 19, 27] to reduce the original term domain and eliminate identifying combinations. For example, they would generalize New York to North America, so that the infrequent combination would be replaced by the more frequent {North America, air tickets}. Alternatively, other methods which are based on suppression, simply remove infrequent terms or terms which participate in infrequent combinations. Generalization and suppression have been mostly used to provide protection against *identity* and *attribute* disclosure. There are few works that rely on adding noise (fake records or terms) to offer differential privacy [6, 14] or to hide the user intent in web search engines [24]. The problem with existing methods is that a large part of the the initial terms are usually missing from the anonymized dataset. There are only a few works [11, 18, 30] that preserve all original terms, without adding noise, based on the *Anatomy* [30] idea of separating quasi identifiers from sensitive values. Still, all these methods can only protect against attribute disclosure.

The main contribution of this work is a novel method called *disassociation* that preserves the original terms but hides identifying combinations. The privacy model is based on k^m-anonymity [27]: an adversary, who knows up to m items from any record, will not be able to match his knowledge to less than k records in the published data. Anonymization is achieved, not by hiding their constituent terms (as done by earlier approaches), but instead by suppressing the fact that some terms (like New York and air tickets) appear together in the same record. The *disassociation* transformation extends the idea of *Anatomy* [30] to provide for the first time protection against identity disclosure by separating terms of the original data. We focus on protection against *identity disclosure* for three reasons: *(a)* it is usually explicitly or implicitly required by law in many countries and applications, *(b)* it is often the case that the sensitivity of items cannot be accurately characterized, so protection against attribute disclosure is not an option, and *(c)* differential privacy approaches [6, 14], which offer strong privacy protection, incur a high information loss that is often not an acceptable trade-off. Protecting identities using disassociation has already been identified as a complicated problem even for the case of simple relational

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales. Investing in knowledge society through the European Social Fund.
Proceedings of the VLDB Endowment, Vol. 5, No. 1

data [18], and no previous solution exists for our problem settings. Finally, the proposed anonymization technique is equally capable to existing state-of-the-art methods in providing protection against attribute disclosure if sensitive terms have been identified.

In brief, the main contribution of the paper is the proposal of *disassociation*, a new data transformation for sparse multidimensional data that preserves the original terms of the dataset. We show how this transformation can be used to anonymize a dataset and prove that the resulting data adhere to our privacy guarantee. Moreover, we present an anonymization algorithm that uses disassociation, and we show that it achieves limited information loss, by evaluating it experimentally on real and synthetic datasets.

2. PROBLEM DEFINITION

The proposed anonymization method focuses on sparse multidimensional data and provides protection against identity disclosure. This section formally presents our assumptions about the data and the attack model. In addition, we define the anonymity guarantee our method targets to. Figure 1 summarizes our notation.

Data. We assume a collection D of *records*; each record is a set of *terms* from a huge *domain* T. For example, a term can be a query posed by a user in the context of web search logs, or a product bought by a customer in the context of supermarket logs. As a motivating example, consider a web search query log that traces the queries posed by users over a period of time. Each record models a different user and contains the set of queries posed by the user. Figure 2a presents an exemplary web search query log consisting of 10 records (each being the web search history of a different user). We do not distinguish between sensitive and non-sensitive terms; we consider the general case, where any term might reveal sensitive information for the user and any term can be used as part of a quasi-identifier for a user. As we discuss in Section 5, having a clear distinction between sensitive and identifying terms simplifies the problem and our proposed technique can guarantee, in this case, protection against attribute disclosure.

Attack model. The identification of a user in D is made possible by tracing records that contain unique combinations of terms. For example, if the database of Figure 2a is published and an adversary A knows that a user U has searched for terms madonna and viagra, he can infer that only record r_2 contains both of them; therefore A is certain that r_2 is associated to U. We assume that the adversary A may have background knowledge of up to m terms (i.e., queries) for any record (i.e., user) and that A does not have negative background knowledge (i.e., the adversary does not know whether a user *did not* pose a specific query). Moreover, we assume that the adversary A does not have background knowledge for so many individuals that it will allow her to infer negative knowledge about U (see Section 5 for details).

Anonymity guarantee. The most popular guarantee for protection against identity disclosure is *k-anonymity* [26]. k-anonymity makes each published record indistinguishable from other $k-1$ published records. Applying k-anonymity on sparse multidimensional data can result to a huge information loss, since groups of identical records must be created in a sparse data space [1, 13, 28]. For this reason, we opt for *k^m-anonymity* [27], a conditional form of k-anonymity, which guarantees that an adversary, who has partial knowledge of a record (up to m items, according to our assumption above), will not be able to distinguish any record from other $k-1$ records. More formally:

DEFINITION 1. *An anonymized dataset D^A is k^m-anonymous if no adversary that has a background knowledge of up to m terms of a record can use these terms to identify less than k candidate records in D^A.*

Symbol	Explanation	Symbol	Explanation
D, D^A	Original, anonymized dataset	T	Domain
$\mathcal{A}, \mathcal{I}$	Anonymization, inverse transf.	$s(a)$	Support of a
P / J ...	Clusters / Joint cluster	T^P	cluster domain
$C, C_1, \ldots$	Record Chunks	T^C	Chunk domain
$SC, SC_1, \ldots$	Shared chunks	C_T	Term chunk

Figure 1: Notation

For the original dataset D and its anonymized counterpart D^A, we define two transformations $\mathcal{A}$ and $\mathcal{I}$. The anonymization transformation $\mathcal{A}$ takes as input dataset D and results in the anonymized dataset D^A. The inverse transformation $\mathcal{I}$ takes as input the anonymized dataset D^A and outputs all possible (non-anonymized) datasets D' that could produce D^A, i.e., $\mathcal{I}(D^A) = \{D' \mid D^A = \mathcal{A}(D')\}$. Obviously, $D \in \mathcal{I}(\mathcal{A}(D))$. For example, consider the dataset

$$D(\mathit{age}, \mathit{zip}) = \{(32, 14122), (39, 14122)\}$$

and its corresponding anonymized dataset (using generalization)

$$D^A(\mathit{age}, \mathit{zip}) = \{(3x, 14xxx), (3x, 14xxx)\},$$

we have: $\mathcal{A}(D) = D^A$ and

$$\mathcal{I}(D^A) = \left\{ \begin{array}{l} \{(30, 14000), (30, 14000)\}, \ldots \\ \{(30, 14000), (31, 14000)\}, \ldots \\ \{(32, 14122), (39, 14122)\}, \ldots \end{array} \right\}$$

In this paper, to achieve k^m-anonymity (Definition 1), we enforce the following privacy guarantee.

GUARANTEE 1. *Consider an anonymized dataset D^A and a set $\mathcal{S}$ of up to m terms. Applying $\mathcal{I}(D^A)$ will always produce at least one dataset $D' \in \mathcal{I}(D^A)$, for which there are at least k records that contain all terms in $\mathcal{S}$.*

Intuitively, an adversary, who has limited background knowledge (consisting of a set $\mathcal{S}$ of up to m terms) about a person, will have to consider k distinct candidate records in a possible original dataset.

3. ANONYMIZATION BY DISASSOCIATION

In this paper, we propose an anonymization transformation $\mathcal{A}$ that is based on disassociation. The proposed transformation partitions the original records into smaller and disassociated subrecords. The objective is to hide infrequent term combinations in the original records by scattering terms in disassociated subrecords. To illustrate the crux of the disassociation idea, we will use Figure 2. We have already seen that the dataset of Figure 2a is prone to identity attacks (e.g., r_2 can be identified by madonna and viagra). The corresponding disassociated anonymized dataset is illustrated in Figure 2b. Our approach, initially, divides the table into two *clusters* P_1 and P_2 containing records $r_1 - r_5$ and $r_6 - r_{10}$ respectively. In each cluster P_i, records are partitioned to smaller subrecords, after dividing the terms in P_i into subsets. For example, in P_1, the terms are divided into subsets $T_1 =$\{itunes, flu, madonna\}, $T_2 =$\{audi a4, sony tv\}, and $T_T =$\{ikea, viagra, ruby\}. Then, each record is split into subrecords according to these subsets. The collection of all subrecords of different records that correspond to the same subset of terms is called a *chunk*. For example, r_1 is split into subrecords \{itunes, flu, madonna\}, which goes into chunk C_1 (corresponding to T_1), \{\}, which goes into chunk C_2, and \{ikea, ruby\}, which goes into chunk C_T. C_T is a special, *term chunk*; the

ID	Records
r_1	{itunes, flu, madonna, ikea, ruby}
r_2	{madonna, flu, viagra, ruby, audi a4, sony tv}
r_3	{itunes, madonna, audi a4, ikea, sony tv}
r_4	{itunes, flu, viagra}
r_5	{itunes, flu, madonna, audi a4, sony tv}
r_6	{madonna, digital camera, panic disorder, playboy}
r_7	{iphone sdk, madonna, ikea, ruby}
r_8	{iphone sdk, digital camera, madonna, playboy}
r_9	{iphone sdk, digital camera, panic disorder}
r_{10}	{iphone sdk, digital camera, madonna, ikea, ruby}

(a) Original dataset D

		Record chunks		Term chunk
		C_1	C_2	C_T
Cluster P_1 $\lvert P_1\rvert = 5$	r_1	{itunes, flu, madonna}		ikea, viagra, ruby
	r_2	{madonna, flu}	{audi a4, sony tv}	
	r_3	{itunes, madonna}	{audi a4, sony tv}	
	r_4	{itunes, flu}		
	r_5	{itunes, flu, madonna}	{audi a4, sony tv}	

		Record chunk	Term chunk
		C_1	C_T
Cluster P_2 $\lvert P_2\rvert = 5$	r_6	{madonna, digital camera}	panic disorder, playboy, ikea, ruby
	r_7	{iphone sdk, madonna}	
	r_8	{iphone sdk, digital camera, madonna}	
	r_9	{iphone sdk, digital camera}	
	r_{10}	{iphone sdk, digital camera, madonna}	

(b) Anonymized dataset D^A

Figure 2: Disassociation example

subrecords that fall into the last chunk (C_T) are merged to a single set of terms. In our example, C_T contains set {ikea, viagra, ruby}, which represents the subrecords from all r_1–r_5 containing these terms. In addition, the order of the subrecords that fall in a chunk is randomized; i.e., the association between subrecords in different chunks is hidden. According to this representation, the original dataset could contain any record that could be *reconstructed* by a combination of subrecords from the different chunks plus *any subset of terms* from C_T. For example, {itunes, madonna, viagra, ruby} is a reconstructed record, which takes {itunes, madonna} from C_1, {} from C_2, and {viagra, ruby} from C_T. Observe that this record does not appear in the original dataset.

Similarly to the generalization based techniques, the disassociated dataset D^A models a set of possible original datasets $\mathcal{I}(D^A)$. However, in our case the possible datasets are not described in a closed form captured by the generalization ranges, but by the possible combinations of subrecords. In other words, the original dataset is hidden amongst the multiple possible datasets in $\mathcal{I}(D^A)$ that can be reconstructed by combining the subrecords and terms taken from the disassociated dataset.

Overall, the anonymized dataset in Figure 2b satisfies Guarantee 1 for $k = 3$ and $m = 2$. We see in detail how this happens in Section 5, but we can observe that an attacker who knows up to $m = 2$ terms from a record r of the original database is not able to reconstruct less that $k = 3$ records (by combining appropriate subrecords) that might have existed in the original data.

In the following, we present the details of our technique, which performs 3 steps: a horizontal partitioning, a vertical partitioning and a refining. The *horizontal partitioning* brings similar records together into clusters. The heart of the anonymization procedure lies in the *vertical partitioning* which disassociates infrequent combinations of terms. Finally, to reduce information loss and improve the quality of the anonymized dataset a *refining* step is executed.

Horizontal partitioning. Records of the original dataset D are grouped into *clusters* according to the similarity of their contents (e.g., Jaccard similarity). For instance, cluster P_1 is formed by records r_1–r_5 (Figure 2b). Horizontal partitioning reduces the anonymization of the original dataset to the anonymization of multiple small and independent clusters. The benefits of this approach are threefold. First, it limits the scope of the term disassociation to the records that are contained in the cluster; two terms may be disassociated only within the local scope of a partition, limiting this way the negative effect in the information quality of the published dataset. Second, since clustering brings similar records together in the same partition, the anonymity guarantee can be achieved with reduced disassociation. Third, the anonymization process can be done more efficiently and even in parallel.

Vertical partitioning. Intuitively, vertical partitioning leaves term combinations that appear many times intact and disassociates terms that create infrequent and, thus, identifying combinations. The disassociation is achieved by concealing the fact that these terms appear together in a single record. Vertical partitioning applies on each cluster and divides it into *chunks*. There are two types of chunks: record and term chunks. *Record chunks* contain subrecords of the original dataset; i.e., each chunk is a collection (with bag semantics) of sets of terms, and they are k^m-anonymous. That is, every m-sized combination of terms that appears in a chunk, appears at least k times. *Term chunks* do not contain subrecords; they contain the terms that appear in the cluster, but have not been placed to record chunks. A term chunk is a simple collection of terms with set semantics. Each cluster may contain an arbitrary number of record chunks (≥ 0) and exactly one term chunk (which might be empty). In Section 5, we explain how the term chunk can be used to provide l-diversity some terms have been designated as sensitive.

Vertical partitioning is applied to each cluster independently. Let us consider a cluster P and let T^P be the set of terms that appear in P. To partition P into v record chunks $C_1, \ldots, C_v$ and a term chunk C_T, we divide T^P into $v+1$ subsets $T_1, \ldots, T_v, T_T$ that are pairwise disjoint (i.e., $T_i \cap T_j = \emptyset$, $i \neq j$) and jointly exhaustive (i.e., $\bigcup T_i = T^P$). Subsets $T_1, \ldots, T_v$ are used to define record chunks $C_1, \ldots, C_v$ while subset T_T, is used to define term chunk C_T. Specifically, $C_T = T_T$ and record chunks C_i, $1 \leq i \leq v$ are defined as $C_i = \{\!\!\{\ T_i \cap r \mid \text{for every record } r \in P \}\!\!\}$ where $\{\!\!\{\cdot\}\!\!\}$ denotes a collection with bag semantics (i.e., duplicate records are allowed in C_i). Thus, chunks $C_1, \ldots, C_v$ are collections of records while chunk C_T is a set of terms. The partitioning of T^P to $T_1, \ldots, T_v, T_T$ is performed in a way which ensures that all resulting record chunks $C_1, \ldots, C_v$ are k^m-anonymous. In Figure 2b, two 3^2-anonymous record chunks C_1 and C_2 are formed for P_1, by projecting the records of P_1 to sets $T_1 = $ {itunes,flu,madonna} and $T_2 = $ {audi a4, sony tv} respectively; the remaining terms {ikea, viagra,ruby} of P_1 form the term cluster C_T.

Note that, for each published cluster, we explicitly show the number of original records in it. Without this explicit information, a data analyst may only infer that the cluster has at least as many records as the cardinality of the chunk with the greatest number of subrecords. Not knowing the cardinality of a cluster introduces significant information loss; for instance, it is not feasible to estimate the co-existence of terms in different chunks.

Finally, we remark that horizontal and vertical partitioning are applied in reverse order from what is followed by approaches that employ similar data transformations [11, 18, 30]. Thus, since vertical partitioning is applied *independently* in each horizontal partition (i.e., cluster), our method follows a *local* anonymization approach. This constitutes a significant difference from previous works that anonymize datasets by performing a global partitioning between terms (usually between sensitive terms and quasi-identifiers).

Record		Term	Shared
P_1 cluster			
{itunes, flu, madonna}		viagra	{ikea,ruby}
{madonna, flu}	{audi a4, sony tv}		{ruby}
{itunes, madonna}	{audi a4, sony tv}		{ikea}
{itunes, flu}			
{itunes, flu, madonna}	{audi a4, sony tv}		
P_2 cluster			
{madonna, digital camera}			
{iphone sdk, madonna}		panic disorder, playboy	{ikea,ruby}
{iphone sdk, digital camera, madonna}			
{iphone sdk, digital camera}			
{iphone sdk, digital camera, madonna}			{ikea,ruby}

Figure 3: Disassociation with a shared chunk.

Refining. At this final step of the method, we focus on improving the quality of the published result while maintaining the anonymization guarantee. To this end, we examine the terms that reside in term chunks. Consider the example of Figure 2b. Terms ikea and ruby are in the term chunk of P_1 because their support in P_1 is low (each term appears in only 2 records). For similar reasons these terms are also in the term chunk of P_2. However, the support of these terms considering both clusters P_1 and P_2 is not small enough to endanger user privacy (ikea and ruby appear as many times as itunes and iphone that are in record chunks).

To address such situations, we introduce the notion of *joint clusters* that offer greater flexibility in our partitioning scheme by allowing different clusters to *share* record chunks. Given a set T^s of *refining terms* (e.g., ikea and ruby), which commonly appear in the term chunks of two or more clusters (e.g., P_1 and P_2), we can define a joint cluster by *(a)* constructing one or more *shared chunks* after projecting the original records of the initial clusters to T^s and *(b)* removing all T^s terms from the term chunks of the initial clusters. Figure 3 shows a joint cluster, created by combining clusters P_1 and P_2 of Figure 2b, based on T^s={ikea,ruby}.

The idea of a joint cluster can be recursively generalized. We may form higher-level joint clusters by combining simple and joint clusters of a lower level (for example see Figure 5). In the general case a joint cluster J, has as children the joint clusters $J_1, \ldots, J_n$ and at its leaves the simple clusters $P_1, \ldots, P_m$. Moreover it contains the k^m-anonymous shared chunks $SC_1, \ldots, SC_w$, which are created over a domain T^s. All terms of T^s come from the term chunks $C_{T_1}, \ldots, C_{T_m}$ of $P_1, \ldots, P_m$. If $T_1, \ldots, T_w$ are the domains of $SC_1, \ldots, SC_w$, $T_1 \cup \cdots \cup T_w = T^s$ and $T_i \cap T_j = \emptyset$ for $i \neq j$, then each shared chunk SC_i is created by projecting the records of every P_j to $C_{T_j} \cap T_i$. Shared chunks are defined in this way, in order to avoid having a record contributing the same projection to shared or simple record chunks more than once.

Reconstruction of datasets. A disassociated dataset D^A has the original records of D partitioned into subrecords (residing in record or shared chunks) and terms (residing in term chunks). An adversary A can combine record, shared and term chunks in an effort to reconstruct the world of all possible original datasets $\mathcal{I}(D^A)$. Possible original datasets may be reconstructed by combining the subrecords of record and term clusters padded with some terms from the term chunks. Such datasets D' are called *reconstructed datasets* and by construction belong to $\mathcal{I}(D^A)$. The adversary A may consider only the reconstructed datasets that abide to his background knowledge. Guarantee 1 requires that for every m terms that exist in a record of D, there will be a D' that contains k records with these terms. Thus, an adversary will always have k candidate records that will match her background knowledge.

Reconstructed datasets are also useful to data analysts, since they have similar statistical properties to the original one. We experimentally evaluate this similarity in Section 7. The benefit of providing the disassociated form, instead of a reconstruction directly, is threefold: *(a)* an analyst can work directly on the disassociated dataset. The disassociated dataset reveals some information, i.e., itemset supports, that is *certain* to exist on the original data, *(b)* the reconstruction procedure is transparent; an adversary cannot draw incorrect conclusions about the identity of a user by considering the reconstructed dataset as original or as ineffectively perturbated and *(c)* an analyst can create an arbitrary set of reconstructed datasets and average query results from all of them.

4. THE ANONYMIZATION ALGORITHM

The proposed algorithm uses heuristics to perform the partitioning (horizontal and vertical) and the refining step of Section 3.

Horizontal partitioning. Horizontal partitioning should bring together similar records that contain many common terms and few uncommon ones. Similarity may be assessed using measures from Information Theory (e.g., Jaccard coefficient). Related clustering algorithms exist in the literature for set-valued data [29], but unfortunately they are not appropriate for our setting since: *(a)* they are not efficient on large datasets and *(b)* they do not explicit control the size of the clusters. We employ Algorithm HORPART, a lightweight heuristic that does not have these problems. The key idea is to split the dataset into two parts: one with the records that contain the most frequent term a in the dataset and another with the remaining records. This procedure is recursively applied to the new datasets until the final datasets are small enough to become clusters. Terms that have been previously used for partitioning are recorded in set *ignore* and are not used in subsequent splitting (Line 3).

Vertical partitioning. To vertically partition the clusters, we follow a *greedy* strategy (Algorithm VERPART), executed independently for each cluster. VERPART takes as input a cluster P and integers k and m; the result is a set of k^m-anonymous record chunks $C_1, \ldots, C_v$ and the term chunk C_T of P.

Let T^P be the set of terms of P. Initially, the algorithm computes the number of appearances (support) $s(t)$ of every term t and orders T^P with decreasing $s(t)$. All terms that appear less than k times are moved from T^P to the term chunk T_T. Since all the remaining terms have support at least k, they will participate in some record chunk. Next, the algorithm computes sets $T_1, \ldots, T_v$ (while loop). To this end, the algorithm uses set T_{remain} that contains the non-assigned terms (ordered by decreasing support s) and T_{cur} (that contains the terms that will be assigned to the current set). To compute T_i ($1 \leq i \leq v$), Algorithm VERPART considers all terms of set T_{remain}. A term t is inserted into T_{cur} only if the C_{test} chunk constructed from $T_{cur} \cup \{t\}$ remains k^m-anonymous (Line 12). Note that the first execution of the for loop (Line 10) will always add a term t to T_{cur} since $C_{test} = \{t\}$ is k^m-anonymous ($s(t) \geq k$). If the insertion of a term t to T_{cur} renders $T_{cur} \cup \{t\}$ non k^m-anonymous, t is skipped and the algorithm continues with the next term. After having assigned to T_{cur} as many terms from T_{remain} as possible, the algorithm *(a)* assigns T_{cur} to T_i, *(b)* removes the terms of T_{cur} from T_{remain} and *(c)* continues to the next

```
Algorithm: HORPART
Input  : Dataset $D$, set of terms $ignore$ (initially empty)
Output : A HORizontal PARTitioning of $D$
Param. : The maximum cluster size $maxClusterSize$
1 if $|D| < maxClusterSize$ then return $\{\!\{D\}\!\}$;
2 Let $T$ be the set of terms of $D$;
3 Find the most frequent term a in $T - ignore$;
4 $D_1 =$ all records of $D$ having term a;
5 $D_2 = D - D_1$;
6 return HORPART($D_1, ignore \cup$ a)$\cup$HORPART($D_2, ignore$)
```

```
Algorithm: VERPART
Input   : A cluster $P$, integers $k$ and $m$
Output  : A $k^m$-anonymous VERtical PARTitioning of $P$
1  Let $T^P$ be the set of terms of $P$;
2  for every term $t \in T^P$ do
3      Compute the number of appearances $s(t)$;
4  Sort $T^P$ with decreasing $s(t)$;
5  Move all terms with $s(t) < k$ into $T_T$;                // $T_T$ is finalized
6  $i = 0$;
7  $T_{remain} = T^P - T_T$;                 // $T_{remain}$ has the ordering of $T^P$
8  while $T_{remain} \neq \emptyset$ do
9      $T_{cur} = \emptyset$;
10     for every term $t \in T_{remain}$ do
11         Create a chunk $C_{test}$ by projecting to $T_{cur} \cup \{t\}$ ;
12         if $C_{test}$ is $k^m$-anonymous then $T_{cur} = T_{cur} \cup \{t\}$;
13     $i$++;
14     $T_i = T_{cur}$;
15     $T_{remain} = T_{remain} - T_{cur}$;
16 Create record chunks $C_1, \ldots, C_v$ by projecting to $T_1, \ldots, T_v$;
17 Create term chunk $C_T$ using $T_T$;
18 return $C_1, \ldots, C_v, C_T$
```

set T_{i+1}. Finally, Algorithm VERPART constructs record chunks $C_1, \ldots, C_v$ using $T_1, \ldots, T_v$ and the term chunk C_T using T_T.

Refining. The result of the vertical partitioning is a set $\mathcal{P}$ of k^m-anonymous clusters. The refining step improves the quality of the anonymized dataset by iteratively creating joint clusters until no further improvement is possible. A naïve method to perform this step consists of computing the information loss (e.g., using a metric of Section 6) for all possible refinement scenarios and choosing the one with the best effect on data quality. Since such an option is very inefficient, we define a refining criterion. Let us consider two clusters J_1 and J_2. These cluster are joined into cluster J_{new} if the following inequality holds:

$$\frac{s(t_1) + \cdots + s(t_n)}{|J_{new}|} \geq \frac{u_1 + \cdots + u_m}{|P_1| + \cdots + |P_m|} \tag{1}$$

where *(a)* $t_1, \ldots, t_n$ are the refining terms T^s (Section 3), *(b)* $s(t_1)$, $\ldots$, $s(t_n)$ are the supports of $t_1, \ldots, t_n$ respectively in the shared chunks of J_{new}, *(c)* $P_1, \ldots, P_m$ are the simple clusters of J_1 and J_2 that contain $t_1, \ldots, t_n$ and *(d)* $v_1, \ldots, v_m$ are the number of terms $t_1, \ldots, t_n$ that appear in the term chunk of each of $P_1, \ldots, P_m$ respectively. For instance, if J_1 and J_2 are clusters P_1 and P_2 of Figure 2b and J_{new} is the joint cluster of Figure 3 then the refining terms are ruby and ikea and we have: $\frac{s(\text{ruby})+s(\text{ikea})}{|J_{new}|} = \frac{4+4}{10} \geq \frac{2+2}{10} = \frac{u_1+u_2}{|P_1|+|P_2|}$. Thus, J_1 and J_2 are replaced by J_{new}.

Note that the left part of Equation 1 estimates the probability of attributing one of $t_1, \ldots, t_n$ to the records of the joint J_{new} while the its right part expresses the probability of attributing one of $t_1, \ldots, t_n$ to the initial records of J_1 and J_2.

Even with the criterion of Equation 1, we still need to exhaustively explore all the combinations of clusters (simple or joint) in order to choose the best ones. This is computationally infeasible. Thus, we have opted for a heuristic that merges each time only two existing clusters (simple or joint) to form a new joint cluster. The sketch of this method is illustrated in Algorithm REFINE. The algorithm takes as input a collection of simple clusters $\mathcal{P}$ and transforms it to a collection of joint clusters. The algorithm orders the clusters of $\mathcal{P}$ as follows: a) each term t is given a *term chunks support* tcs(t); i.e., the number of term chunks in clusters of $\mathcal{P}$ where t appears; b) the terms in term chunks are ordered in descending order of their tcs; and c) clusters are ordered by comparing lexicographically their term chunks. After the first iteration, joint clusters are introduced in $\mathcal{P}$. To each joint cluster J, we add a *virtual term chunk*, which is the union of the term chunks of its simple clusters, and we use it in the ordering step. REFINE modifies $\mathcal{P}$ by merging adjacent pairs of clusters and repeats the process until $\mathcal{P}$ does not change. The merging is done only if the criterion of Equation 1 is satisfied, and produces a joint cluster as defined in Section 3.

```
Algorithm: REFINE
Input   : A set $\mathcal{P}$ of $k^m$-anonymous clusters
Output  : A refinement of $\mathcal{P}$
repeat
    Add to every joint cluster a virtual term chunk as the union of the
    term chunks of its simple clusters;
    Order (joint) clusters in $\mathcal{P}$ according to the contents of their
    (virtual) term chunks;
    Modify $\mathcal{P}$ by joining adjacent pairs of clusters (simple or joint)
    based on Equation 1;
until there are no modifications in $\mathcal{P}$;
return $\mathcal{P}$
```

Correctness of the algorithm. Disassociation performs the partitioning (vertical and horizontal) and refining steps detailed in the previous sections. The proposed method is correct; it succeeds for any input and it always produces a disassociated k^m-anonymous dataset. It is not hard to verify that the algorithm terminates and produces a disassociated result. The proof that a disassociated dataset is k^m-anonymous is provided in Section 5. In a nutshell notice that *(a)* horizontal partitioning does not alter the original dataset and always produces clusters, *(b)* vertical partitioning creates k^m-anonymous clusters since Algorithm VERPART will put every term that has support over k to the record chunks (Lines 10-17) and the rest of the terms in the term chunk (Lines 6 and 18) and *(c)* refining has the trivial solution of not merging any clusters and if a joint cluster is created (i.e., if shared chunks are added), then k^m-anonymity is preserved as we prove in Section 5.

Complexity. The most expensive part of disassociation is the horizontal partitioning that has a worst case complexity of $O(|D|^2)$ time. The horizontal partitioning can be seen as a version of quicksort, which instead of a pivot uses the most frequent term to split each partition; in the worst case it will do $|D|$ partitionings and at each partitioning it has to re-order $|D|$ records. The complexity of vertical partitioning depends on the domain T^P of the input cluster P, and not on the characteristics of the complete dataset. The most expensive operation in the vertical partitioning is to establish whether a clunk is k^m-anonymous or not. This task requires examining $\binom{|T^P|}{m}$ combinations, thus it takes $O(|T^P|!)$ time. Since we regulate the size of clusters, the behavior of the overall algorithm, as the dataset size grows, is dominated by the behavior of the horizontal partitioning. Finally, the refining algorithm has again a $O(|D|^2)$ time complexity, since in the worst case it will perform as many passes over the clusters as the number of the clusters. Note that this a worst case analysis; in practice, the behavior of our algorithm is significantly better; this is also verified by the experimental evaluation of Section 7, which shows a linear increase of the computational cost with the input dataset size $|D|$.

5. ANONYMIZATION PROPERTIES

In Section 3, we described our disassociation transformation technique, which is implemented by the algorithm presented in Section 4. In this section, we prove how the disassociated result can guarantee k^m-anonymity, by showing how the transformed data can be used to reconstruct a possible initial dataset that contains k times any combination of m terms. In this proof we define two additional properties that must be preserved in a disassociated dataset.

Cluster anonymity. First, we prove that each disassociated cluster

Records
a
a
b, c
b, c
a, b, c

(a) Original dataset

Record chunks		Term chunk
C_1	C_2	C_T
a		
a		
a	b, c	
	b, c	
	b, c	

(b) Anonymized dataset

Figure 4: Illustration of Example 1, Original cluster size = 5

is k^m-anonymous, by constructing an initial cluster that contains k times any m terms of the disassociated cluster.

Let P be an arbitrary cluster of the anonymized dataset which is vertically partitioned into k^m anonymous record chunks $C_1, \ldots, C_v$ and a term chunk C_T. Then the following Lemma holds:

LEMMA 1. *For any m terms $\mathcal{S} = t_1, \ldots, t_m$ that appear in P, at least k distinct records that contain $\mathcal{S}$ can be reconstructed by combining subrecords from the chunks $C_1, \ldots, C_v$ and terms from C_T, or no record can be reconstructed that contains $\mathcal{S}$.*

PROOF. We first prove that Lemma 1 holds if all m terms fall inside the record chunks. In this case the m terms $\mathcal{S} = t_1, \ldots, t_m$ are scattered in n, $(n \leq m, n \leq v)$ record chunks $C_1, \ldots, C_n$. Let $S_1, \ldots, S_n$ be the subsets of $\mathcal{S}$ that appear in each of $C_1, \ldots, C_n$. Since each chunk is k^m anonymous, S_i will appear in the respective record chunk C_i at least k times together or none at all. The latter case happens if the S_i terms exist in disjoint groups of subrecords inside C_i. If there is even one of $S_1, \ldots, S_n$ whose terms do not appear together at all in the respective chunk, then the $\mathcal{S}$ terms cannot appear together in any reconstructed record. If every set of $S_1, \ldots, S_n$ appears together in the respective chunk, then it has to appear in at least k subrecords in each chunk. Let $SR_1, \ldots, SR_n$ be these sets of subrecords, one from each record chunk. We can then create a record by combining 1 subrecord from each of $SR_1, \ldots, SR_n$, i.e., $r = sr_1 \cup \cdots \cup sr_n$, where sr_i is a subrecord, $sr_i \in SR_i$. Since each SR_i contains at least k subrecords, we remove the used subrecord and repeat the process at least $k-1$ more times. This results to at least k distinct records that contain all $\mathcal{S}$ terms. Assume now that only g, $g < m$ terms fall inside the record chunks and $m - g$ terms fall in the term chunk. The previous proof holds for the g terms too, since $g < m$, thus k records can be reconstructed from the record chunks that contain the g items. We can then directly pad these k records with the rest of $m - g$ terms from the term chunk. We are free to do so, since the multiplicity and the correlations of these terms are not disclosed in the disassociated cluster. □

Lemma 1 shows that k records can be constructed from a disassociated cluster; still, this is not sufficient for providing k^m-anonymity as defined in Guarantee 1 as the following example illustrates.

EXAMPLE 1. *Let us consider the dataset of Figure 4a. Assume that we want to publish it as 3^2-anonymous and that we create two record chunks C_1 and C_2 with domains $T_1 = \{\mathsf{a}\}$, $T_2 = \{\mathsf{b,c}\}$ and $T_T = \emptyset$, as illustrated in Figure 4b. It is not hard to verify that all chunks are 3^2-anonymous and that Lemma 1 holds.*

Let us now consider an adversary A that knows: (a) *the anonymized dataset of Figure 4b,* (b) *that the size of the original cluster is 5 and* (c) *that a user had used terms* a *and* b*, i.e.,* $\{\mathsf{a}, \mathsf{b}\}$ *is a subrecord of the original dataset. Adversary A also knows that the original dataset is composed by a combination of the records stored in chunks C_1 and C_2.*

While the subrecords from C_1 and C_2 can be combined to create $k = 3$ records that contain a *and* b*, these records cannot appear in any original dataset, which must contain 5 records. It is not hard to verify that the only combination that results in a dataset with 5 records is the one presented in Figure 4a. Thus, no dataset that contains* $\{\mathsf{a,b}\}$ *3 times can be the initial dataset of the example of Figure 4a. This way, the user's record* $\{\mathsf{a,b,c}\}$ *is revealed.*

Example 1 demonstrates that Lemma 1 is not sufficient to guarantee k^m-anonymity. Lemma 1 guarantees that k records that contain any m terms can be constructed, but it does not guarantee that these records can appear in a valid dataset of a predefined size. The sparsity of the original data, often leads to empty subrecords inside different chunks. Since there cannot be empty records, a record that is created as a result of combining empty subrecords is not valid. An initial dataset that contains an empty record is also not valid, thus and adversary can discard it. To enforce Guarantee 1, we must require not only that Lemma 1 holds, but also that these records can appear in a valid initial dataset. Fortunately, we do no need to reconstruct all possible original datasets to see if this condition is satisfied. It suffices to enforce the condition of the following lemma.

LEMMA 2. *Let $C_1, \ldots, C_v$ be the record chunks that correspond to the anonymization of a cluster P with size s. If* (a) *chunks $C_1, \ldots, C_v$ are k^m-anonymous and* (b) *the total number of subrecords in all chunks $\sum(|C_i|)$ is greater than or equal to $s + k \cdot (h-1)$, $h = min(m, v)$ or the term chunk is not empty, then Guarantee 1 holds.*

PROOF. To prove this lemma, it suffices to show that for every different combination of m items: (a) no record that contains the m terms can be constructed or (b) a valid initial cluster P_r of size s where the m terms appear in at least k records can be reconstructed.

Assume m random terms $t_1, \ldots, t_m$ from T^P. According to Lemma 1, given a disassociated cluster P_a, no record that contains these m terms can be created or at least k records can be reconstructed. In the former case, the k^m anonymity trivially holds (this case corresponds to a combination of m terms that did not appear in the initial dataset[1]) and it covers case (a). In the latter case, to prove (b) we need to show that these k records can appear in at least one valid reconstruction of the disassociated cluster P_a. A valid reconstruction of cluster P_a is a possible initial cluster that has s non-empty records. We construct a cluster that contains s records in total, where at least k of them contain $t_1, \ldots, t_m$ as follows. We first construct the k records, denoted as R_k that contain the m terms as described in Lemma 1. If the m terms are scattered in h chunks, then to construct each of these records we need h subrecords; one form each chunk, thus $k \cdot h$ subrecords. To create a valid initial dataset of size s that contains the R_k records we only need to populate it with $s - k$ additional records R_o that are valid i.e., non-empty. If the term chunk is non-empty then the $s - k$ records can be populated by randomly combining terms from the term chunk. If the term chunk is empty, we can create such records by assigning 1 subrecord that has not been used in the construction of R_k, from any of the $C_1, \ldots, C_v$ chunks,. The total number of subrecords needed is $h \cdot k + s - k = s + k \cdot (h-1)$. The worst case, i.e., the maximum number of subrecords that are required for constructing a valid cluster, is when we need to combine one different subrecord for each of $t_1, \ldots, t_m$ to create a record of R_k. In this case, $h = m$ or if the cluster has less than m record chunks $h = v$. Thus, having $s + k \cdot (h-1)$, $h = min(m, v)$ subrecords is sufficient to create a valid initial cluster. □

[1] If a combination of terms cannot be created by combining subrecords, it holds that it did not appear in the original data. The reverse is not true; if a combination can be created, it does not mean that it existed in the original data.

Joint cluster anonymity. An example which demonstrates that careless creation of shared chunks can lead to cases where combinations of m terms might not appear k times in any reconstructed dataset is depicted in Figure 5a. Although every chunk (i.e., vertical partition) in the illustrated dataset is 3^2-anonymous, the overall dataset is not. Since each record has set semantics, an adversary can discard initial datasets that contain records with two identical terms. An attacker A knowing that a user U asked for terms x and o can only find one matching record in every possible original dataset using the following reasoning. Term x appears only in the 1^{st} cluster (always together with a) and o appears in the shared chunk. Thus, to construct U's record, A has to combine {a,x} with any of {a,x}, {a} and {o}; but, by the semantics of shared chunks, the only allowed combination is {a,x,o} which appears just once. In order to avoid these conflicts we enforce the following property.

PROPERTY 1. *Let J be a joint cluster and T^r be the set of terms that appear in the record and shared chunks of the clusters (joint or not) forming J. A shared chunk of J that does not contain terms from T^r must be k^m-anonymous; if it does, it must be k-anonymous.*

For example in Figure 5a, Property 1 does not hold since $T^r =$ {a,b,c,d,e,f,x}, term a appears on the shared chunk, a $\in T^r$ and the shared chunk is not k-anonymous. On the other hand, the property holds for Figure 5b. T^r contains all terms that appear in J except those that are placed in term chunks and those that appear only in J's shared chunks (only o in the previous example). Let us now consider the following lemma.

LEMMA 3. *A joint cluster for which Property 1 holds, is k^m-anonymous.*

PROOF. We will prove the Lemma by induction. Lemma 2 shows that simple clusters are k^m-anonymous. It is also easy to see why joint clusters who contain only simple clusters are k^m-anonymous, since no conflicts between the terms of the record and shared chunks appear there. In the following we will prove the inductive step; a joint cluster J that is formed by existing k^m-anonymous joint clusters is k^m-anonymous too.

Let J be a joint cluster with domain T^J, the k^m-anonymous joint clusters $J_1, \ldots, J_q$ be its children and the simple clusters k^m-anonymous $P_1, \ldots, P_w$ be its leaves. Let $\mathcal{SC}$ be the set of the shared chunks of J that all satisfy Property 1. Moreover, let T^r be the set of terms that appear in the record and shared chunks of $J_1, \ldots, J_q$. Since $J_1, \ldots, J_q$ are k^m anonymous we only have to check how the introduction of the shared chunks $\mathcal{SC}$ affects anonymity. Because Lemma 2 holds for each cluster independently, there is no need to set a new bound for the number of subrecords contained in $\mathcal{SC}$. We only have to show that the addition of $\mathcal{SC}$ allows the creation of k records (or no record at all) that contain any m-sized combination of terms from T^J.

Assume a random combination of m terms $t_1, \ldots, t_m$ from T^J where terms $t_1, \ldots, t_i$ appear in $J_1, \ldots, J_q$ (in either record or term chunks) and $t_{i+1}, \ldots, t_m$ appear in the shared chunks $\mathcal{SC}$. If $i = m$, i.e., all terms belong to $J_1, \ldots, J_q$, then k^m anonymity holds since we assumed that $J_1, \ldots, J_q$ are k^m-anonymous. If $i = 0$, i.e., all terms belong to the shared chunks, then by following the same constructive proof as we did in Lemma 1 we can create k records that contain $t_1, \ldots, t_m$. This is sufficient for proving k^m-anonymity since there is no requirement for the number of subrecords in the shared chunks. Finally, if some of the m terms cannot appear together by any combination of subrecords, i.e., they did not appear together in the original data at all, then the k^m-anonymity trivially holds. It remains to prove that J is k^m-anonymous for $0 < i < m$.

Let $SC_1, \ldots, SC_n$, $n \leq m$, with domains $T^1, \ldots, T^n$ be the shared chunks of $\mathcal{SC}$ that contain $t_{i+1}, \ldots, t_m$. Using the reconstructed clusters of $J_1, \ldots, J_q$ we partially reconstruct a joint cluster J_r that contains at least k records with the terms $t_1, \ldots, t_i$. Let PR be the partially reconstructed records of J that contain $t_1, \ldots, t_i$, $|PR| \geq k$. We expand the PR with subrecords from each SC_i of $SC_1, \ldots, SC_n$ to create records that contain all m terms. For each of SC_i with domain T^i we have two cases:

$T^r \cap T^i = \emptyset$ **holds:** In this case, SC_i is k^m-anonymous and no term from SC_i appears in any of the PR records. We can then select k subrecords that contain the terms from $t_{i+1}, \ldots, t_m$ that fall in T^i and concatenate them to k records of PR.

$T^r \cap T^i \neq \emptyset$ **holds:** In this case, SC_i is k-anonymous. Let SR_i be the records of SC_i that contain the terms of $t_1, \ldots, t_m$ that fall in T^i, $|SR_i| \geq k$. We want to append k subrecords from SR_i to k records of PR to create records that contain all m terms. Still, not all combinations of $PR \times SR_i$ are valid due to conflicts. The conflicts are caused by terms that appear both in the subrecords of SC_i and the records of J_r that have partially been constructed until now. Assume that the conflict is based on a term a. a is independent of $t_1, \ldots, t_m$. Assume that a appears in the record or shared chunks of the simple or joint clusters $\mathcal{J}_a$, which are descendants of J. The existence of a in these record chunks means that SR_a did not exist in any of $\mathcal{J}_a$, thus the records of SR_a cannot be combined with any of the records of $\mathcal{J}_a$. Let JR_a be the partially reconstructed records of $\mathcal{J}_a$. Because of the conflict, the adversary knows that if any record of PR belongs to JR_a too, then it cannot be combined with SR_a to create the k records we need. To guarantee k^m anonymity, we must be able to combine at least k records from $PR' = PR \setminus JR_a$ and $SR'_i = SR_i \setminus SR_a$ or none at all. We will prove this by showing that either all records of PR' and all subrecords of SR'_i are disqualified, or that at least k remain in each set. Since each joint cluster is anonymized independently, it contributes at least k records to PR. Any conflict with even one record of a cluster from $\mathcal{J}_a$ disqualifies all the records from the same cluster, thus JR_a will be equal to PR or they will differ at least by k records, i.e., all the records contributed by a cluster that has no conflict. Thus $|PR'| = 0 \vee |PR'| \geq k$. Moreover, since we required that S_i is k-anonymous, there will be at least k duplicates of each record. A conflict over term a will disqualify at least k records, and if records without a exist in SR' there will be at least k of them. So, after eliminating conflicts, $|SR'| = 0 \vee |SR'| \geq k$. Since both PR' and SR' will have either more than k records or none after eliminating conflicting records, we can either create k records that contain all $t_1, \ldots, t_m$ or no such record. □

The proof is similar for conflicts based on more than one item.

Since a disassociated dataset consists of either joint or simple clusters, Lemmas 2 and 3 are sufficient to prove that the whole dataset is k^m-anonymous. We only have to show that the properties required by the previous Lemmas can be guaranteed by the algorithm of Section 4. To guarantee the property required by Lemma 2 we only need to add a check at the end of VERPART that verifies that the cluster contains enough subrecords. If the size limit is not met, then by moving the least frequent item of the record chunks to the term chunk, we guarantee that the conditions set by Lemma 2 are satisfied. This solution is always feasible; at least one term will exist to populate the term chunk in each cluster. To satisfy Lemma 3 the refining algorithm has to check in the creation of a shared chunk, if any of its terms appears in the record chunk of any descendant joint or simple cluster. If this holds, then the chunk

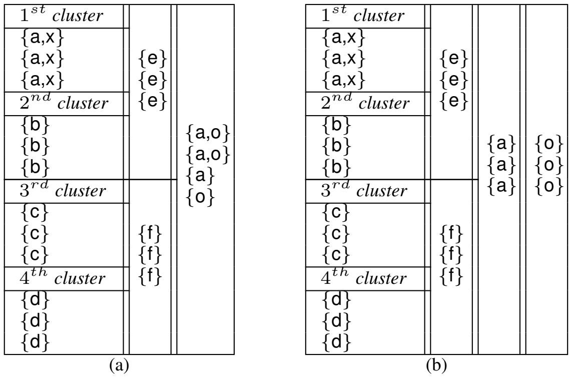


Figure 5: Unsafe (a) vs. safe (b) creation of a shared chunk

must be k-anonymous, else it can be k^m-anonymous. Since there is always the trivial solution of a record chunk that contains only 1 term, which is both k-anonymous and k^m-anonymous, the refining algorithm always produces a k^m-anonymous dataset.

Protection against stronger adversaries. k^m-anonymity is a conditional guarantee and the protection it offers is reduced against adversaries with background knowledge that exceeds the attack model assumptions. The most common case is to have adversaries that have more knowledge than m terms about a user or adversaries that have background knowledge about all users that contain certain m terms. In both cases, the adversary's background knowledge consists of enough information to accurately associate some records to a known group of users $\mathcal{U}$. This allows the adversary to remove these records from the groups of candidate records that match her background knowledge for any user who does not belong to $\mathcal{U}$. Still, this attack does not lead automatically to complete re-identification of the additional users, but reduces the number of candidates according to their overlap with the records that are associated with $\mathcal{U}$. This type of attacks has been studied in other contexts [2, 32] and their effect on disassociation and generalization based methods is similar. Disassociation has an additional weakness that is related to Lemma 2; if a record of a user is identified and the remaining terms violate Lemma 2, then the probability of identifying additional records might be reduced to less than k-1.

Diversity. So far we have discussed an anonymization framework offering protection against identity disclosure. In this section, we discuss how the proposed framework may also offer protection against attribute disclosure and achieve l-diversity.

Former works that guarantee l-diversity, separate sensitive attributes from quasi identifiers [11, 18, 30]. Following the same idea, we can enforce l-diversity in our framework by *(a)* ignoring all sensitive values in the horizontal partitioning and *(b)* placing all sensitive values in the term chunk at the vertical partitioning stage. In the resulting data, all sensitive values will reside at the term chunk and no association between them and any other subrecord or value can be done with probability over than $1/|C|$, where $|C|$ is the size of the cluster. By adjusting the size of the clusters, the anonymization method achieves the desired degree of l-diversity.

The proposed anonymization framework offers protection against both identity and attribute disclosure. We focus on the former because to the best of our knowledge there is no other work that employs a similar to disassociation transformation to guarantee protection against identity disclosure (works enforcing l-diversity do not consider re-identification dangers [11, 18, 30]). We expect that in practice both protection against identity and attribute disclosure (for the recognized sensitive values) are needed.

6. INFORMATION LOSS

By definition, disassociation incurs a different information loss compared to classic anonymization methods. The disassociated dataset preserves all the initial terms and many of the initial itemsets. An analyst can work directly on the disassociated dataset or reconstruct a possible initial one. In the former case, the analyst can compute lower bounds of the supports of all terms and itemsets. These bounds can be computed by counting all the appearances of terms and itemsets in the record chunks of the simple and joint clusters and by adding one to the support of each term that appears in a term chunk. Moreover, the analyst can employ models for answering queries in probabilistic databases to directly query the anonymization result [9]. Using such a model, one can assume that the contents of each record chunk are possible assignments to every record of the cluster with probability $(1/|P|)$. Still, existing work on uncertain data management is not tailored to the disassociated dataset and does not take advantage of the constraints in the reconstruction procedure that we detailed in Section 3 to increase the quality of the estimations. Moreover, working directly on the disassociated dataset requires adjusting existing tools and models for analyzing data. Because of this, we believe that it is easier to apply most analysis tasks on a reconstructed dataset. During horizontal partitioning, clusters are created by bringing similar records together; thus, the statistical properties of a reconstructed dataset are quite close to the original one. A way to further increase the accuracy of the analysis on reconstructed data is to create more than one reconstructed datasets and average the query results on them. We experimentally evaluate the similarity between the reconstructed datasets and the original one in Section 7.

Disassociation hides infrequent term combinations, therefore the incurred information loss is related to term combinations that exist in the original dataset but are lost in the disassociated dataset. To assess the impact of the information loss, we examine the behavior of common mining and querying operations on the transformed data. We employ metrics that are generic and can be used as a comparison basis with anonymization methods that employ different data transformations (such as generalization, suppression and differential privacy). More specifically, we examine how many of the frequent itemsets that exist in the original data are preserved in the published data, and we also measure the relative error in the estimation of the supports of pairs of items.

Top-K deviation (tKd). The tKd metric measures how the top-K frequent itemsets of the original dataset change in the published anonymized data. Let FI (respectively, FI') be the top-K frequent itemsets in the original dataset (respectively, the anonymized dataset); tKd is defined as follows:

$$tKd = 1 - \frac{|FI \cap FI'|}{|FI|} \tag{2}$$

Intuitively, tKd expresses the ratio of the top-K frequent itemsets of the original dataset that appear in the top-K frequent itemsets of the anonymized data.

To compare disassociation with generalization-based methods, we define an appropriate version of tKd, called the *top-k multiple level mining loss* tKd-ML^2, which is based on the ML^2 metric defined in [27]. Mining a dataset at multiple levels of a generalization hierarchy is an established technique [12], which allows detecting frequent association rules and frequent itemsets that might not appear in the most detailed level of the data. If a generalization hierarchy that allows the anonymization of the data exists, then we can assume that the same hierarchy can be used to mine frequent itemsets from the published (and the original) data at different lev-

Dataset	$\|D\|$	$\|T\|$	max rec. size	avg rec. size
POS	515,597	1,657	164	6.5
*WV*1	59,602	497	267	2.5
*WV*2	77,512	3,340	161	5.0

Figure 6: Experimental datasets

els of abstraction. $tKd\text{-}ML^2$ is given again by Equation 2, but in this case FI and FI' are the sets of generalized frequent itemsets that can be traced in the original and anonymized data, respectively. In the case of generalized datasets, a generalized frequent itemset is lost if it contains terms that have been generalized at a higher level during the anonymization process. Reconstructed datasets do not contain any generalized items, but given a generalization hierarchy generalized frequent itemsets can be mined.

Relative error (re). This metric (used also in [6]) is used to measure the relative error in the support of term combinations in the published data. Since there is a huge number of possible combinations, we limit ourselves to combinations of size two as an indication of the dataset quality. Larger combinations are usually infrequent, and the case of very frequent ones is already covered by tKd. The relative error is defined as follows:

$$re = \frac{|s_o(a,b) - s_p(a,b)|}{AVG(s_o(a,b),\ s_p(a,b))}, \qquad (3)$$

where $s_o(a,b)$ and $s_p(a,b)$ is the support of the combination of terms (a,b) in the original and in the published data, respectively. Reconstructing anonymized datasets might introduce new item combinations in the published data, which did not exist in the original data. In order to take them into account in the definition of the relative error, we use the average of the two supports as denominator, instead of using the original support $s_o(a,b)$. The average has a smoothing effect on the metric, since it normalizes re to $[0,2]$, and avoids divisions by 0.

7. EXPERIMENTAL EVALUATION

The goal of the experimental evaluation is to demonstrate the advantages that disassociation in preserving data quality and to show that disassociation has a robust behavior in different settings.

7.1 Experimental Settings

Datasets. In the experiments, we use the 3 real datasets described in Figure 6, which were introduced in [33]. Dataset *POS* is a transaction log from an electronics retailer. Datasets *WV*1 and *WV*2 contain click-stream data from two e-commerce web sites, collected over a period of several months. Synthetic datasets were created with IBM's Quest market-basket synthetic data generator (http://www.almaden.ibm.com/cs/quest/syndata.html). Unless otherwise stated, the default characteristics for the synthetic datasets are $1M$ records, $5k$ term domain and 10 average record length.

Evaluation metrics. We measure the information loss incurred by our method with respect to the following: *(a)* the tKd, $tKd\text{-}ML^2$, and re measures defined in Section 6 and *(b)* the percentage of terms $tlost$ that have support more than k in the original dataset D but they are placed in term chunks by our method. We report tKd and re for the disassociated datasets calculated in two different ways: *(a)* one on a single random reconstructed dataset, labeled as tKd and re, and *(b)* one calculated only by taking into account the subrecords that appear inside the record and shared chunks, labeled as $tKd\text{-}a$ and $re\text{-}a$. In the latter case, we do not take into account the probability that an itemset can be created by combining subrecords. $tKd\text{-}a$ and $re\text{-}a$ trace the itemsets that would exist in *any* reconstructed dataset, thus they are based on lower bounds of itemset supports in the original dataset. tKd and $tKd\text{-}ML^2$ are measured for the 1000 most frequent itemsets. Finally, computing an average re on all combinations of size 2 is not very informative in cases of skewed distributions and large domains. The majority of combinations would be rare or would not exist at all in the original data, but they would dominate the result. To avoid this, we ordered the domain of each dataset by descending term support and we used a small range of consecutive terms to trace their re. After some testing we chose the 200th-220th most frequent terms. re in this case is an indicator of how well less frequent but not utterly rare combinations are preserved in the anonymized dataset.

Evaluation parameters. We compared performance by varying the following parameters: *(a)* k, *(b)* the size of the dataset, *(c)* the size of the dataset's domain, *(d)* the average size of the records, *(e)* the terms we use to calculate re and *(f)* the number of reconstructed datasets we use to calculate re and tKd. We do not present a detailed evaluation for m, because in all the available datasets its effect for values $m > 2$ is negligible. The explanation for this is that most record clusters are k^m-anonymous for any m either because they have gathered very frequent terms or because they contain small subrecords. The experiments are all performed with $k = 5$, $m = 2$ unless explicitly stated otherwise.

Comparison to state-of-the-art. Comparing disassociation to other methods is not straightforward; no other method offers the same privacy guarantee while introducing the same type of information loss. We chose to compare disassociation to the generalization-based *Apriori* approach [27], since it offers the same privacy guarantee and it is the most closely related method. This comparison allows us to see how the different data transformations, generalization and disassociation, affect the quality of the anonymized dataset in a similar privacy framework. Furthermore, we compare disassociation to *DiffPart* [6], which offers differential privacy for set-valued data by suppressing infrequent terms and adding noise. The comparison with *DiffPart* demonstrates the gains disassociation offers in terms of information quality, when a more relaxed guarantee like k^m-anonymity is chosen over differential privacy. All methods were implemented in C++ (g++ 4.3.2).

7.2 Experimental Results

The first experiment (Figures 7a-d) investigates the information loss by our method on the real datasets. In Figure 7a, we see the result of disassociation in the quality of all datasets and in Figures 7b-d just for the *POS* dataset. The $tKd\text{-}a$ in Figure 7a is similar for all datasets, showing that the most frequent combinations are preserved for different data characteristics. Still, when we trace tKd on the reconstructed datasets, the results significantly improve only for the *POS* dataset, which is the largest of the 3 and its records have the longest average length. This reflects the fact that disassociation managed to create multiple record chunks for *POS*. The combinations of their contents results to a significantly better reconstructed dataset. Disassociation produces significantly different results for the 3 datasets, when looking to the re and $re\text{-}a$ metrics. The supports of the combinations traced by re are preserved better when the ratio of the dataset size to the dataset domain is high. This ratio is higher for *POS* and *WV1*, where re has significantly superior results to $re\text{-}a$. This indicates the gains from combining terms from different record chunks in the reconstructed datasets. Finally, the same ratio affects how many terms are placed in the record chunk, as reported by $tlost$, but to a lesser degree. In Figures 7b and 7c, we see how information loss escalates as the power of the guarantee, expressed by the k parameter, grows. The measures that depend on the most frequent items and itemsets are only slightly affected

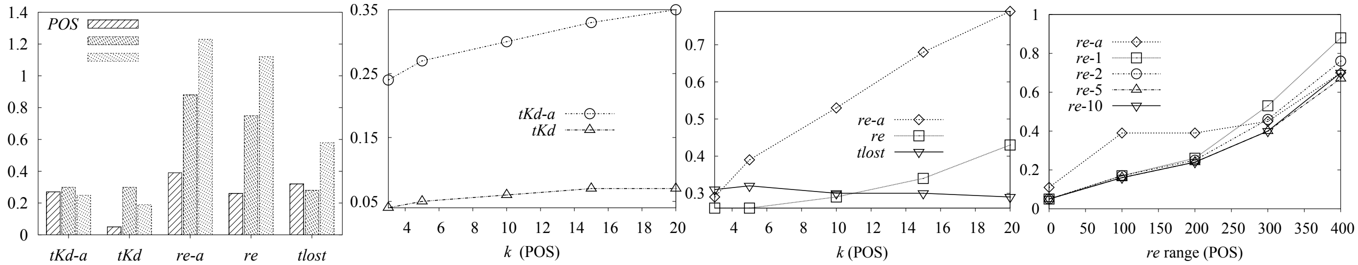


Figure 7: Information loss on real data (a-d)

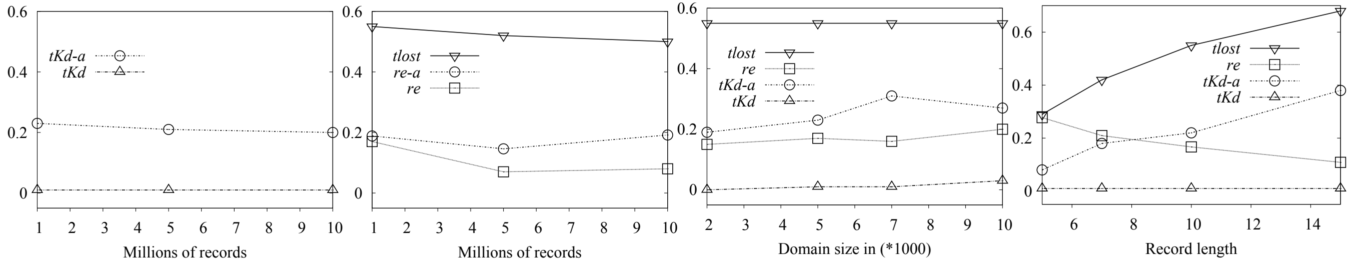
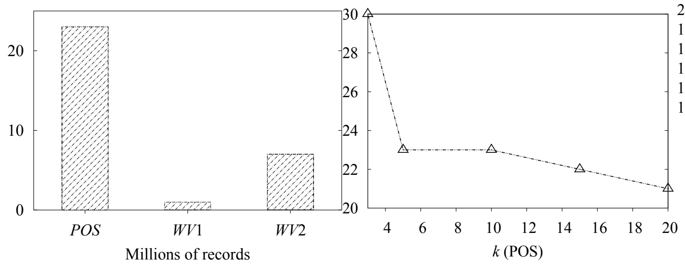


Figure 8: Information loss on synthetic data (a-d)



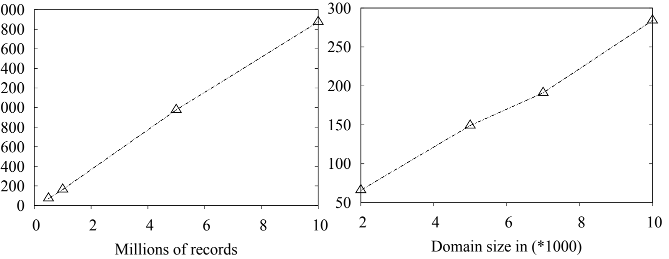


Figure 9: Performance on real data (a-b)

Figure 10: Performance on synthetic data (a-b)

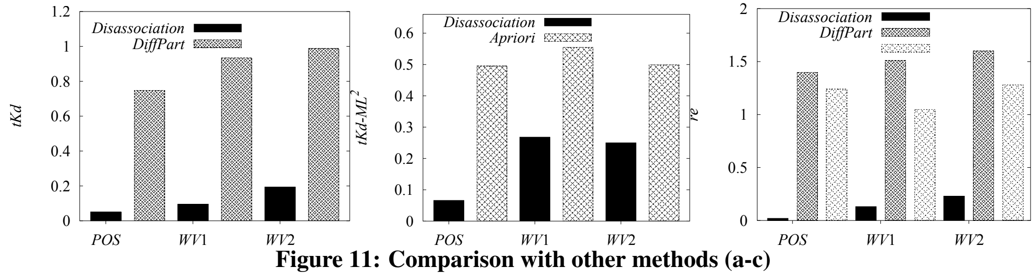


Figure 11: Comparison with other methods (a-c)

(Figure 7b), since the disassociation algorithm preserves them in record chunks. On the other hand, re, which does not depend on the most frequent items, increases linearly with k, but with a low rate (Figure 7c). In Figure 7d we explore the gain in information quality by creating several reconstructed datasets and averaging the itemset supports on them. We created 10 random reconstructions of the anonymized *POS* dataset, and we traced re taking this time into account the average supports of the itemsets in 2 (re-$r2$), 5 (re-$r5$) and 10 (re-$r5$) of the reconstructed datasets. We do not report results for tKd since they were already close to 0 and did not benefit substantially from multiple reconstructions. We measure the re on the combinations of the 0-20, 100-120, 200-220, 300-320 and 400-420 most frequent terms in *POS*. In the x-axis of Figure 7d, we depict the frequency order of the terms; e.g. a point over 100 refers to the re of the combinations of the 100th-120th most frequent terms in *POS*. When the terms are frequent, the support of their combinations is reported accurately in any reconstructed dataset, so taking the average does not provide any benefit. As the combinations become less frequent, using more than one reconstructed datasets allows for more accurate estimations. In the previous experiments we also examined separately how frequent itemsets of size less or equal to m and of size greater than m ($m = 2$) are preserved. We did not notice any systematic behavior; depending on the dataset, any of the aforementioned frequent itemset classes may be preserved better. For example, frequent itemsets smaller than m were preserved better in *POS* and in *WV2* and worse in *WV1*. We do not report detailed results due to space limitations.

In the experiments of Figure 8 we used synthetic data to see how the information loss is affected, when the dataset characteristics variate. Since the anonymization is applied independently on each cluster, the database size does not have a significant effect on the quality of the results as demonstrated in Figures 8a and 8b. Only the re and re-a are positively affected, because the terms it traces become more frequent and they end up in record chunks more often. Moreover, in Figure 8c we see that increasing the domain when the distribution is skewed, basically affects the tail of the distribution, thus it does not have a significant effect on frequent combinations of terms traced by tKd, whereas re slightly deteriorates. The effect of record length is depicted in Figure 8d. Having larger records results in more record chunks and more rare terms in each cluster, thus tKd-a and $tlost$ increase. On the other hand, when we keep the dataset and domain size constant and we only increase the record size, the support of the terms in the dataset increases and this explains how re benefits from larger records. Finally, tKd remains close to 0 for all record sizes, since the multiple record chunks reconstruct most of the frequent itemsets in the reconstructed dataset.

Figures 9 and 10 illustrate the performance of the proposed algorithm in terms of CPU time (results in seconds). Disassociation is not significantly affected by the value of k, and at the same time it scales linearly to the dataset and the domain size.

Figure 11 shows how disassociation performs compared to *DiffPart* and the *Apriori* algorithm. The graphs illustrate the impact of all algorithms on the quality of the anonymized dataset for $k = 5$ and $m = 2$ (*DiffPart* is unaffected by this parameter). For the *DiffPart* algorithm we used privacy budgets ranging from 0.5 to 1.25, using a step 0.25 with the same parameters as in [6] and we report the best results. In Figure 11a we see how disassociation compares to *DiffPart* in terms of tKd. Since in both cases the anonymized datasets contain only original terms (the differential private one has only a subset of them) tKd is computed in exactly the same way. The trade-off for using a stronger privacy guarantee like differential privacy is quite important; in the best case 75% of the top frequent items have been lost, whereas disassociation loses only 5% in the same experiment. In Figure 11b we see how disassociation compares to *Apriori* in terms of tKd-ML^2, since no original frequent itemset appears in the generalized dataset. Disassociation performs again significantly better than *Apriori* especially for *POS* which is the largest dataset and has more frequent terms than *WV1* and *WV2*. A problem of *Apriori* is that few uncommon terms cause the generalization of several common ones. Finally, Figure 11c shows how all algorithms compare in terms of re. re in the generalized dataset is calculated by uniformly dividing the support of a generalized term to the original terms that map to it. *DiffPart* has suppressed all the 200-220th most frequent terms in *POS* (less that 100 of the original 1657 terms are left), so in order to make the comparison meaningful we report the re for the (0-20th) most frequent terms. The re for both *DiffPart* and *Apriori* is over 1, which indicates that the supports of the term combinations have limited usefulness for analysis, whereas disassociation provides 0.18 re in the worst case.

In summary, the experiments on both real and synthetic datasets demonstrate that disassociation offers an anonymized dataset of significantly superior quality compared to other state-of-the-art methods. Moreover, the information loss does not increase aggressively as k increases. Finally, disassociation is not computationally expensive and it is practical for large datasets.

8. RELATED WORK

Privacy preservation was first studied in the relational context and focused on protection against identity disclosure. In [25, 26] the authors introduce k-anonymity and use generalization and suppression as their two basic tools for anonymizing a dataset. *Incognito* [15] and *Mondrian* [16] are two well known algorithms that guarantee k-anonymity for a relation table by transforming the original data using global (full-domain) and local recoding, respectively. [21] demonstrates that the information loss, when providing k-anonymity, can be reduced by using *natural* domain generalization hierarchies (as opposed to user-defined ones).

To address the problem of attribute disclosure, where a person can be associated with a sensitive value, the concept of ℓ-diversity [20] was introduced. *Anatomy* [30] provides ℓ-diversity and lies closer to our work, in the sense that it does not generalize or suppress the data, but instead it disassociates them by publishing them separately. Still, the anonymization approach is restricted to relational data and it does not protect against identity disclosure. *Slicing*, a more flexible version of *Anatomy* appears in [18]. *Slicing* guarantees l-diversity as *Anatomy*, but instead of completely separating sensitive attributes from quasi-identifies, it might publish some quasi-identifiers without disassociating them from sensitive values, if the diversity guarantee is not violated. Moreover, *Slicing*, disassociates quasi-identifiers to increase protection from membership disclosure. By disassociating quasi identifiers, an adversary is faced with several options for reconstructing each record, thus she cannot be certain that a specific record existed in the original data. The data transformation is similar to the approach of our work, but there are significant differences: a) there is no protection against identity disclosure and b) the disassociation between quasi-identifiers does not provide any privacy guarantee, and it takes place only if the impact on information loss is limited. Protection against membership disclosure is facilitated, but not guaranteed; it is roughly estimated using the number of attribute combinations, and not guaranteed by considering the possible initial datasets as in our work. The issues of empty and duplicate records are not addressed. Our work differs from *Slicing* mainly because it uses the disassociation of quasi-identifiers to provide a guarantee against identity, and because it addresses *sparse* multidimensional values.[2].

A similar idea, the vertical fragmentation of relational tables, is employed in a different context to guarantee user anonymity in [7]. The proposed technique distributes a relational table to different servers. In each server, only a subset of the relation's attributes are available unencrypted. The subsets that are available without encryption are chosen so that sensitive associations between attributes, captured by *confidentiality constraints*, are broken. Fragmentation is similar to the basic idea in our work and in [30, 18], but the anonymization model is very different since it focuses on known confidentiality constraints; attacks based on background knowledge are not considered.

More recently, a stronger privacy preservation paradigm, differential privacy, has been proposed [10]. Differential privacy is independent of adversary's background knowledge and it roughly requires that the existence of every single record in the data does not have a significant impact in any query. Finally, the work of [8], although focusing at the protection of associations in sparse bipartite graphs, is related to our work because of the way they define their semantics. The anonymization technique of [8] replaces each node of the graph with a safe group of labels, allowing in this way the anonymized graph to be matched to multiple possible initial graphs.

Privacy on set-valued data. The works that lie closer to this paper are those for privacy on set-valued data. Most works that provide protection against identity disclosure rely on generalization. An efficient algorithm for classical k-anonymity in a set-value context appears in [13]. [27, 28] introduce the k^m anonymity guarantee, which is used and extended in this paper. The authors provide algorithms for anonymizing the data that, unlike our approach, are based on generalization, employing both local and global recoding. In [4] an algorithm for providing k^m-anonymity using only suppression is proposed. The authors have a similar motivation to our work and focus on web search query logs, which they anonymize by removing terms that violate k^m-anonymity. The proposed method preserves original terms, but due to the large tail of the term support distribution in such logs, it removes 90% of the terms even for low k and m values. In a different setting, [22] studied multire-

[2] In [18] there is an application of *Slicing* to the Netflix data [23], which are sparse. This is achieved by padding all null values with the average of the corresponding attribute values. This technique works only for specific types of data processing and cannot address of sparse data in general.

lational k-anonymity, which can be translated to a problem similar to the one studied here, but the anonymization procedure still relies on generalization. [31] provide protection both against identity and attribute disclosure by relying on suppression.

Protection against attribute disclosure is provided both by generalization and disassociation transformations. The work of [11] extends [30] to provide ℓ-diversity for transactional datasets with a large number of items per transaction, but it does not depart from the anonymization framework of [30]; it still has a separate set of quasi-identifiers and sensitive values. The basic idea of [11] is to create equivalence classes where the quasi-identifiers are published separately from the sensitive values and their supports. [5] provides a more elaborate ℓ-diversity guarantee for sparse multidimensional data, termed ρ-uncertainty, where sensitive items can act as quasi-identifiers too. Still, unlike our approach, generalization and suppression are employed to anonymize the data.

There have been few works that investigate the publication of set-valued data under differential privacy guaranties. [14] focuses on the anonymization of web search logs, using the AOL data [3]. The proposed method that guarantees differential privacy but it only publishes query terms and not records. Moreover, the anonymization procedure completely hides all terms that are infrequent, which are the majority of terms in AOL data. In [6] a method for publishing itemsets instead of isolated terms from a set-valued collection of data is proposed. The *DiffPart* algorithm follows a top down approach, which starts from the unification of the whole domain and refines it by partitioning it to subdomains, if the item combinations can be published without breaching differential privacy.

Our work lies closer to [11, 30, 18] in the sense that it does not suppress or generalize the data but instead it severs the links between values attributed to the same entity. Unlike [11, 30, 18] we focus on identity protection, and not simply on separating sensitive values from quasi-identifiers. The work of [18] has the most similar data transformation, but it solves a different problem and does not address the peculiarities of sparse multidimensional data. Our privacy guarantee comes from [27], but we follow a completely different path with respect to the data transformation and the type of targeted data utility.

9. CONCLUSIONS

In this paper, we proposed a novel anonymization method for sparse multidimensional data. Our method guarantees k^m-anonymity, for the published dataset using a novel data transformation called *disassociation*. Instead of eliminating identifying information by not publishing many original terms, either by suppressing or generalizing them, we partition the records so that the existence of certain terms in a record is obscured. This transformation introduces a different type of information loss from existing methods, making it a valuable alternative when the original terms are important.

10. REFERENCES

[1] C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pp. 901-909, 2005.

[2] M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi. Anonymity preserving pattern discovery. *VLDB Journal*, 17(4):703-727, 2008.

[3] M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. New York Times, 2006.

[4] T. Burghardt, K. Böhm, A. Guttmann, and C. Clifton. Anonymous search histories featuring personalized advertisement - balancing privacy with economic interests. *TDP*, 4(1):31-50, 2011.

[5] J. Cao, P. Karras, C. Raissi, and K.-L. Tan. ρ-uncertainty: inference-proof transaction anonymization. *PVLDB*, 3(1-2):1033-1044, 2010.

[6] R. Chen, M. Noman, B. C. Fung, B. C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. *PVLDB*, 4(11):1087-1098, 2011.

[7] V. Ciriani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Combining fragmentation and encryption to protect privacy in data storage. *TISSEC*, 13(3):1-33, 2010.

[8] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. *PVLDB*, 1(1):833-844, 2008.

[9] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, pp. 864-875, 2004.

[10] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *TCC*, pp. 265-284, 2006.

[11] G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In *ICDE*, pp. 715-724, 2008.

[12] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *VLDB*, pp. 420-431, 1995.

[13] Y. He and J. F. Naughton. Anonymization of set-valued data via top-down, local generalization. *PVLDB*, 2(1):934-945, 2009.

[14] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *WWW*, pp. 171-180, 2009.

[15] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD*, pp. 49-60, 2005.

[16] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE*, pp. 25, 2006.

[17] J. Li, R. C.-W. Wong, A. W.-C. Fu, and J. Pei. Anonymization by local recoding in data with attribute hierarchical taxonomies. *TKDE*, 20(9):1181-1194, 2008.

[18] T. Li, N. Li, J. Zhang, and I. Molloy. Slicing: a new approach to privacy preserving data publishing. *TKDE*, 24(3):561-574, 2012.

[19] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. Anonymization of electronic medical records for validating genome-wide association studies. *PNAS*, 17:7898-7903, 2010.

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

[21] M. Nergiz and C. Clifton. Thoughts on k-anonymization. *DKE*, 63(3):622-645, 2007.

[22] M. Nergiz, C. Clifton, and A. Nergiz. Multirelational k-anonymity. In *ICDE*, pp. 1417-1421, 2007.

[23] Netflix Prize FAQ. http://www.netflixprize.com/faq, 2009.

[24] H. Pang, X. Ding, and X. Xiao. Embellishing text search queries to protect user privacy. *PVLDB*, 3(1-2):598-607, 2010.

[25] P. Samarati. Protecting respondents' identities in microdata release. *TKDE*, 13(6):1010-1027, 2001.

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

[27] M. Terrovitis, N. Mamoulis, and P. Kalnis. Privacy-preserving anonymization of set-valued data. *PVLDB*, 1(1):115-125, 2008.

[28] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *VLDB Journal*, 20(1):83-106, 2010.

[29] K. Wang, C. Xu, and B. Liu. Clustering transactions using large items. In *CIKM*, pp. 483-490, 1999.

[30] X. Xiao and Y. Tao. Anatomy: simple and effective privacy preservation. In *VLDB*, pp. 139-150, 2006.

[31] Y. Xu, K. Wang, A. W.-C. Fu, and P. S. Yu. Anonymizing transaction databases for publication. In *KDD*, pp. 767-775, 2008.

[32] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pp. 72-83, 2009.

[33] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In *KDD*, pp. 401-406, 2001.

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.

	Relational attributes			Transaction attribute
Id	Age	Origin	Gender	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)

	Relational attributes			Transaction attribute
Id	Age	Origin	Gender	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)

	Relational attributes			Transaction attribute
Id	Age	Origin	Gender	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)

	Relational attributes			Transaction attribute
Id	Age	Origin	Gender	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 **Rfirst**. 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 **Tfirst**, 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 **Rfirst** and **Tfirst** 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 **Rfirst** and **Tfirst**, repectively. Observe that **Rfirst** did not generalize the relational attributes as heavily as **Tfirst** but applied more generalization to the transaction attribute. This is because, **Rfirst** constructs small groups, and does not control the grouping of items. Contrary, the groups created by **Tfirst** 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   $\mathcal{D}$ := Rmerge($\mathcal{D}, \mathcal{T}, \delta$)
    // $(k, k^m)$-anonymization
5   for each cluster $C \in \mathcal{D}$ do
6       $\mathcal{D} := (\mathcal{D} \setminus C) \cup \mathcal{T}(C)$
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   $\mathcal{D}$ := Tmerge($\mathcal{D}, \mathcal{T}, \delta$)
    // $(k, k^m)$-anonymization
5   for each cluster $C \in \mathcal{D}$ do
6       $\mathcal{D} := (\mathcal{D} \setminus C) \cup \mathcal{T}(C)$
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 $\mathbf{Rum}(\mathcal{D}) \leq \delta$ and $\mathbf{Tum}(\mathcal{D})$ 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 $\mathbf{Tum}(\mathcal{D}) \leq \delta$ and $\mathbf{Rum}(\mathcal{D})$ 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 **RMERGE**, 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 **RMERGE** we propose three algorithms, namely **RMERGE**$_R$, **RMERGE**$_T$ and **RMERGE**$_{RT}$, 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 **TMERGE** merges clusters until the **Tum** threshold is reached, or no more merging is possible. To implement **TMERGE** we propose three algorithms: **TMERGE**$_R$, **TMERGE**$_R$ and **TMERGE**$_{RT}$, 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 **RMERGE**, which is responsible for the merging phase of **Rum**-BOUND (Step 4). Our algorithms are based on different merging heuristics. Specifically, **RMERGE**$_R$ merges clusters with similar relational values, **RMERGE**$_T$ with similar transaction items and **RMERGE**$_{RT}$ 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 **RMERGE**$_R$ (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, **RMERGE**$_T$ 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: RMERGE$_R$
1  while $\mathcal{D}$ changes do
2      Select, as a seed, the cluster $C \in \mathcal{D}$ with minimum $\mathbf{Rum}(C)$
3      Find the cluster $C' \in \mathcal{D}$ that minimizes $\mathbf{Rum}(\mathcal{G}(C \cup C'))$ .
4      $\mathcal{D}_{tmp} := ((\mathcal{D} \setminus C) \setminus C') \cup \mathcal{G}(C \cup C')$
5      if $\mathbf{Rum}(\mathcal{D}_{tmp}) \leq \delta$ then
           $\mathcal{D} := \mathcal{D}_{tmp}$
6  return $\mathcal{D}$
```

```
Algorithm: RMERGE$_T$
1  while $\mathcal{D}$ changes do
2      Select, as a seed, the cluster $C \in \mathcal{D}$ with minimum $\mathbf{Rum}(C)$
       // Find the appropriate cluster $C'$ to be merged with $C$
3      Let $\{C_1, \ldots, C_t\}$ be the set of clusters in $\mathcal{D} \setminus C$ ordered by increasing $\mathbf{Tum}(\mathcal{T}(C \cup C_i))$, $i \in [1, t)$
4      for $i := 1$ to $t$ do        // Test if $C' = C_i$
5          $\mathcal{D}_{tmp} := ((\mathcal{D} \setminus C) \setminus C_i) \cup \mathcal{G}(C \cup C_i)$
6          if $\mathbf{Rum}(\mathcal{D}_{tmp}) \leq \delta$ then   // $C'$ is $C_i$
7              $\mathcal{D} := \mathcal{D}_{tmp}$
8              exit the for loop
9  return $\mathcal{D}$
```

```
Algorithm: RMERGE$_{RT}$
1  while $\mathcal{D}$ changes do
2      Select, as a seed, the cluster $C \in \mathcal{D}$ with minimum $\mathbf{Rum}(C)$
3      Let $\{C_1, \ldots, C_t\}$ (resp. $\{\hat{C}_1, \ldots, \hat{C}_t\}$) be the set of clusters in $\mathcal{D} \setminus C$ ordered by increasing $\mathbf{Rum}(\mathcal{G}(C \cup C_i))$ (resp. $\mathbf{Tum}(\mathcal{T}(C \cup \hat{C}_i))$), $i \in [1, t)$
       // 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. $C_u \in \{C_1, \ldots, C_t\}$ and $C_v \in \{\hat{C}_1, \ldots, \hat{C}_t\}$
6          $\mathcal{D}_{tmp} := ((\mathcal{D} \setminus C) \setminus C_i) \cup \mathcal{G}(C \cup C_i)$
7          if $\mathbf{Rum}(\mathcal{D}_{tmp}) \leq \delta$ then
8              $\mathcal{D} := \mathcal{D}_{tmp}$
9              exit the for loop
10 return $\mathcal{D}$
```

the clusters with increasing $\mathbf{Tum}(\mathcal{T}(C \cup C_i))$ 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 $\mathbf{Tum}(\mathcal{D}_{tmp}) \leq \delta$ then
■      $\mathcal{D} := \mathcal{D}_{tmp}$
■      return $\mathcal{D}$
```

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.

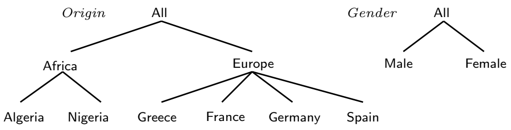


Fig. 3: Hierarchies for the dataset of Fig. 1a

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: $NCP_R(\tilde{v}) = \begin{cases} 0, |\tilde{v}| = 1 \\ |\tilde{v}|/|R|, \text{otherwise} \end{cases}$, $NCP(r) = \sum\limits_{i \in [1,v]} w_i \cdot NCP_{R_i}(r[R_i])$ 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 **R**merge$_T$, **R**merge$_{RT}$, **T**merge$_R$, and **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 $BTD(r_1, r_2) = \frac{ones(b_1 \veebar b_2)+1}{ones(b_1 \wedge b_2)+1} \cdot ones(b_1 \vee b_2)$, 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	$\lvert D \rvert$	Rel. att.	$\lvert dom(T) \rvert$	Max, Avg # items/record
Informs	36553	5	619	17, 4.27
YouTube	131780	6	936	37, 6.51

Table 1: Description of the datasets

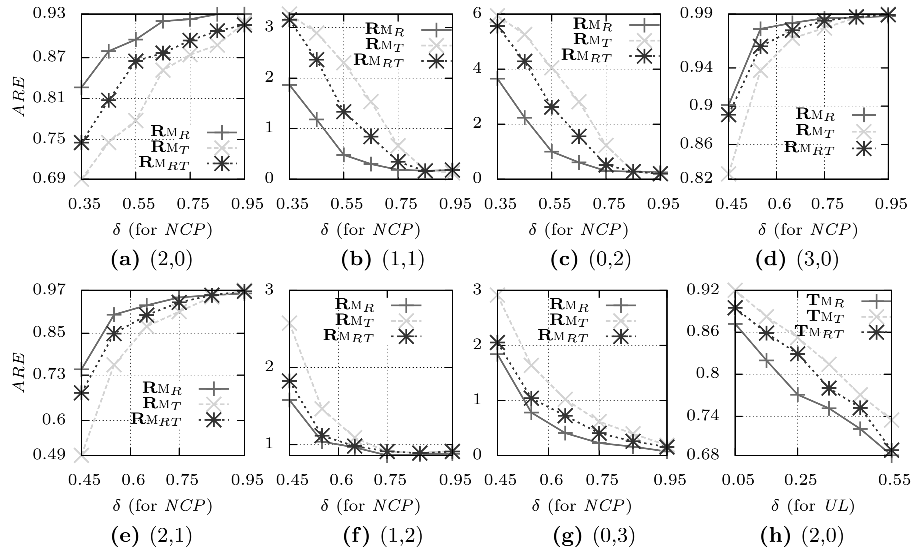
Experimental setup. We implemented all algorithms in C++ and applied them to Informs (`https://sites.google.com/site/informsdataminingcontest`) and YouTube (`http://netsg.cs.sfu.ca/youtubedata`) datasets, whose characteristics are shown in Table 1[4]. The default parameters were k=25, m=2, and δ=0.65, 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*.



(a) (2,0) **(b)** (1,1) **(c)** (0,2) **(d)** (3,0)

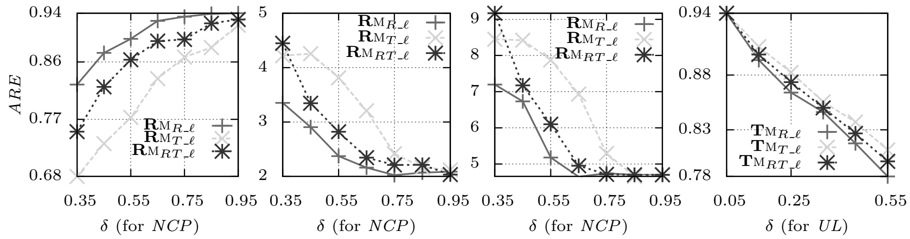
(e) (2,1) **(f)** (1,2) **(g)** (0,3) **(h)** (2,0)

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)

$\mathbf{T}\mathrm{M}_R$, and $\mathbf{T}\mathrm{M}_{RT}$ (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 $\mathbf{R}\mathrm{M}_R$, $\mathbf{R}\mathrm{M}_T$, and $\mathbf{R}\mathrm{M}_{RT}$, for varying δ. $\mathbf{R}\mathrm{M}_R$ 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 $\mathbf{R}\mathrm{M}_{RT}$ created slightly larger clusters than $\mathbf{R}\mathrm{M}_T$, which have lower UL scores. The results for $\mathbf{T}\mathrm{M}_T$, $\mathbf{T}\mathrm{M}_R$, and $\mathbf{T}\mathrm{M}_{RT}$ and the (k, l^m)-diversity algorithms were similar.

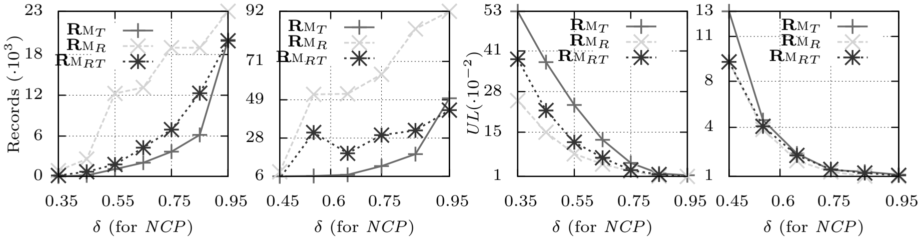
Efficiency. We studied the impact of dataset size using random subsets of Informs, whose records were contained in all larger sets. As can be seen in Fig. 7a, $\mathbf{R}\mathrm{M}_T$ outperformed $\mathbf{R}\mathrm{M}_R$ and $\mathbf{R}\mathrm{M}_{RT}$, and it was more scalable, due to the use of the BTD measure. $\mathbf{R}\mathrm{M}_{RT}$ was the slowest, because it computes two cluster orderings. $\mathbf{T}\mathrm{M}_T$, $\mathbf{T}\mathrm{M}_R$, and $\mathbf{T}\mathrm{M}_R$ perform similarly to $\mathbf{R}\mathrm{M}_R$, $\mathbf{R}\mathrm{M}_T$, and $\mathbf{R}\mathrm{M}_{RT}$ (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 $\mathbf{R}\mathrm{M}_R$, $\mathbf{R}\mathrm{M}_T$, and $\mathbf{R}\mathrm{M}_{RT}$ improves with k, as fewer clusters are merged. $\mathbf{R}\mathrm{M}_T$ was up to 2.2 times more efficient than $\mathbf{R}\mathrm{M}_R$ and $\mathbf{R}\mathrm{M}_{RT}$ 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 $\mathbf{R}\mathrm{M}_{T\,UL}$, $\mathbf{R}\mathrm{M}_{T\,UL}$, $\mathbf{T}\mathrm{M}_{T\,UL}$, and $\mathbf{T}\mathrm{M}_{RT\,UL}$, 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 $\mathbf{R}\mathrm{M}_T$ and $\mathbf{R}\mathrm{M}_{RT}$ have the same UL scores with their



(a) (2,0) **(b)** (1,1) **(c)** (0,2) **(d)** (2,0)

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)



(a) **(b)** **(c)** **(d)**

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 $\mathbf{T}\mathrm{M}_R$ and $\mathbf{T}\mathrm{M}_{RT}$. 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

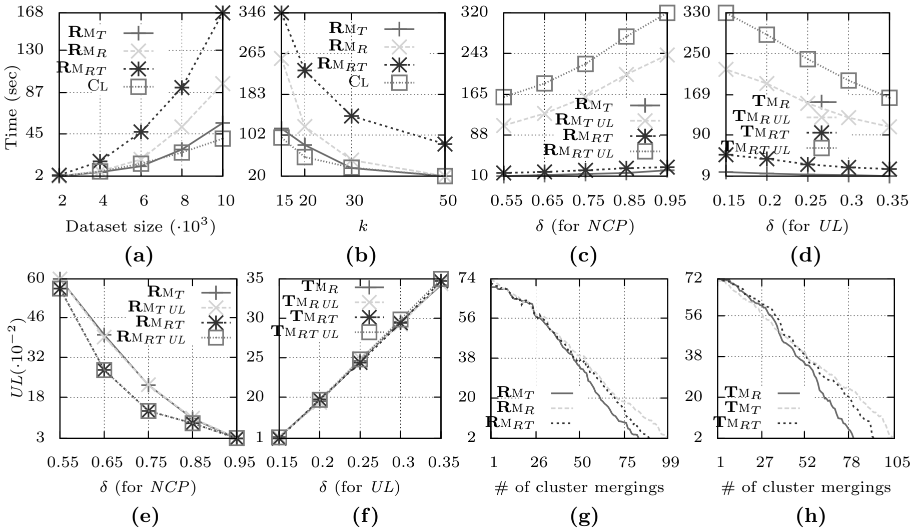


Fig. 7: Runtime (a) vs. $|D|$ and (b) vs. k. Impact of using *BTD* on runtime for (c) **Rum**-BOUND and (d) (**Tum**-BOUND), and on *UL* for (e) **Rum**-BOUND and (f) **Tum**-BOUND. *UL* vs. number of cluster mergings for (g) **Rum**-BOUND, and (h) **Tum**-BOUND

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. *Knowledge and Information Systems*, 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.

Apriori-based algorithms for k^m-anonymizing trajectory data

Giorgos Poulis*, **Spiros Skiadopoulos***, **Grigorios Loukides****, **Aris Gkoulalas-Divanis*****

*University of Peloponnese, Email: {poulis,spiros}@uop.gr

**Cardiff University, Email: g.loukides@cs.cf.ac.uk

***IBM Research - Ireland, Email: arisdiva@ie.ibm.com

Abstract. The proliferation of GPS-enabled devices (e.g., smartphones and tablets) and location-based social networks has resulted in the abundance of trajectory data. The publication of such data opens up new directions in analyzing, studying and understanding human behavior. However, it should be performed in a privacy-preserving way, because the identities of individuals, whose movement is recorded in trajectories, can be disclosed even after removing identifying information. Existing trajectory data anonymization approaches offer privacy but at a high data utility cost, since they either do not produce truthful data (an important requirement of several applications), or are limited in their privacy specification component. In this work, we propose a novel approach that overcomes these shortcomings by adapting k^m-anonymity to trajectory data. To realize our approach, we develop three efficient and effective anonymization algorithms that are based on the apriori principle. These algorithms aim at preserving different data characteristics, including location distance and semantic similarity, as well as user-specified utility requirements, which must be satisfied to ensure that the released data can be meaningfully analyzed. Our extensive experiments using synthetic and real datasets verify that the proposed algorithms are efficient and effective at preserving data utility.

Keywords. privacy, anonymity, trajectories, spatial data, k^m-anonymity, utility constraints

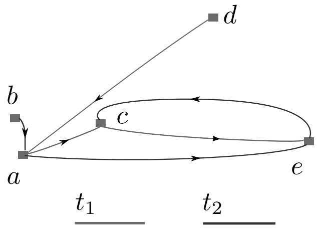
1 Introduction

The widespread adoption of GPS-enabled smartphones and location-based social networking applications, such as Foursquare (`https://foursquare.com`), opens up new opportunities in understanding human behaviour through the analysis of collected mobility data. However, the publication of these data, which correspond to trajectories of personal movement (i.e., ordered lists of locations visited by individuals), can lead to *identity disclosure*, even if directly identifying information, such as names or SSN of individuals, is not published [33].

The values that, in combination, may lead to identity disclosure are called *quasi-identifiers* (QI) [32, 34]. For example, let us assume that a location-based social network service publishes the movement of users during a day in the form of checkins in various locations. An example of these data is shown in Figure 1a. If Mary's colleague, John, knows that Mary

id	trajectory
t_1	(d, a, c, e)
t_2	(b, a, e, c)
t_3	(a, d, e)
t_4	(b, d, e, c)
t_5	(d, c)
t_6	(d, e)

(a)



(b)

Figure 1: (a) the original database $\mathcal{T}$ (b) visual representation of trajectories t_1 and t_2

checked in at locations a and d, he cannot associate Mary with her record (trajectory), as both trajectories t_1 and t_3 include the locations a and d. But if John knew that Mary first checked in at location d and then at a, he can uniquely associate Mary with the trajectory t_1.

This example highlights not only the need to transform a set of user trajectories $\mathcal{T}$ to prevent identity disclosure based on partial location knowledge held by attackers, but also the difference from well-studied set-valued data anonymity models, like k^m-anonymity [36] and privacy-constrained anonymization [18, 24]. In these models, value ordering is not significant; thus, records are represented as unordered sets of items. For instance, if an attacker knows that someone checked in first at the location c and then at e, they could uniquely associate this individual with the record t_1 (Figure 1b). On the other hand, if $\mathcal{T}$ was a set-valued dataset, three records, namely t_1, t_2 and t_4, would have the items c and e. Thus, the individual's identity is "hidden" among 3 records. Consequently, for any set of n items in a trajectory, there are $n!$ possible quasi-identifiers.

This difference makes preventing identity disclosure in trajectory data publishing more challenging, as the number of potential quasi-identifiers is drastically increased. Existing methods operate either by anonymizing *(i)* each trajectory as a whole, thereby not assuming any specific background knowledge of attackers [1, 2, 26, 29], or *(ii)* parts of trajectories, thereby considering attackers who aim to re-identify individuals based on specific locations [35, 39]. The first category of methods are based on *clustering and perturbation* [1, 2, 29], while the second category employs *generalization and suppression* of quasi-identifiers [27, 39, 28, 35]. The main drawback of clustering-based methods is that they may lose information about the direction of movement of co-clustered trajectories and cause excessive information loss, due to space translation. Moreover, applying perturbation may create data that are not *truthful* and cannot be used in several applications [14]. Similarly, existing generalization-and-suppression based methods [27, 39, 28, 35] have the following limitations. First, they assume that quasi-identifiers are known to the data publisher prior to anonymization [35, 39], or that any combination of locations can act as a quasi-identifier [27]. Second, they require a location taxonomy to be specified by data publishers [28] based on location semantics. However, such a taxonomy may not exist, or may not accurately reflect the distance between locations. In both cases, the anonymized data will be highly distorted. Last, some approaches assume that each location can be classified as either sensitive or non-sensitive [28]. In practice, however, this assumption may not hold, as location sensitivity usually depends on context (e.g., visiting a hospital may be considered as sensitive for a patient, but not for a doctor).

Recently, another class of approaches that aims at limiting the amount of information about the presence or absence of any individual trajectory has been proposed [5, 7, 9]. These approaches enforce a well-established privacy model, called *differential privacy* [12], by employing *perturbation*. Specifically, they release a noisy summary of the original data that can be used in specific analytic tasks, such as frequent sequential pattern mining [3]. While being able to offer strong privacy guarantees, these approaches do not preserve data truthfulness, since they rely on perturbation.

1.1 Contributions

In this work, we propose a novel approach for publishing trajectory data, in a way that prevents identity disclosure, and three effective and efficient algorithms to realize it. Specifically, our work makes the following contributions.

First, we adapt k^m-anonymity [35, 36] to trajectory data. k^m-anonymity is a privacy model that was proposed to limit the probability of identity disclosure in transaction data publishing. The benefit of this model is that it does not require detailed knowledge of quasi-identifiers, or a distinction between sensitive and non-sensitive information, prior to data publishing.

Second, we develop three algorithms for enforcing k^m-anonymity on trajectory data. These algorithms generalize data in an apriori-like fashion (i.e., apply generalization to increasingly larger parts of trajectories) and aim at preserving different aspects of data utility. Our first algorithm, called SEQANON, applies *distance-based* generalization, effectively creating generalized trajectories with locations that are close in proximity. For instance, SEQANON would favor generalizing a together with b, because b is the closest location to a, as can be seen in Fig. 1b. SEQANON does not require a location taxonomy and aims at preserving the distance between original locations. Thus, it should be used when accurate semantic information about locations is not available[1]. Clearly, however, the presence of accurate, semantic location information should also be exploited, as it can help the preservation of data utility. For example, assume that a and c represent the locations of restaurants, whereas b represents the location of a coffee shop. In this case, generalizing a together with c would be preferred, because c is a restaurant that is also not very far from a. To take into account both the distance and the semantic similarity of locations, we propose an algorithm, called SD-SEQANON. This algorithm produces generalized trajectories, whose locations are typically slightly more distant but much more semantically similar than those created by SEQANON. Both SEQANON and SD-SEQANON allow generalizing any locations together, as they aim to minimize information loss. In several applications, however, data publishers have specific utility requirements, which dictate how locations must be generalized to ensure that the anonymized dataset is practically useful [24]. For instance, assume that the anonymized version of the dataset in Fig. 1a needs to be used to enable the accurate counting of the number of restaurants, in which individuals checked in. To satisfy this requirement, generalizing together a and b must be prohibited, because the resultant generalized location $\{a, b\}$ can be interpreted as either a restaurant or a coffee shop. On the other hand, the generalization of a together with any other restaurant is allowable, and the generalization that incurs the minimum information loss should be preferred. To account for such utility requirements, we propose a third algorithm, called U-SEQANON. This algorithm aims at satisfying utility constraints and uses both generalization and suppression.

Third, we investigate the effectiveness and efficiency of our approach through experiments on a synthetic dataset, generated using the Brinkhoff's generator [6], and on a real dataset, derived from a location-based social networking website [10]. The results of these

[1] A preliminary version of this work that discusses the SEQANON algorithm appeared in the PriSMO workshop, which was held in conjunction with IEEE MDM 2013.

experiments verify that our approach is able to anonymize trajectory data, under various privacy and utility requirements, with a low level of information loss. In addition, they show that our algorithms are fast and scalable, due to the use of the apriori principle.

1.2 Organization

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 presents some preliminary concepts related to trajectory data anonymization, as well as the privacy and utility objectives of our algorithms. Section 4 presents our anonymization algorithms, and Section 5 an experimental evaluation of them. Last, we conclude the paper in Section 6.

2 Related work

Privacy-preserving trajectory data publishing has attracted significant attention, due to the pervasive use of location-aware devices and location-based social networks, which led to a tremendous increase in the volume of collected data about individuals [4]. One of the main concerns in trajectory data publishing is the prevention of identity disclosure, which is the objective of the k-anonymity privacy model [33, 34]. k-anonymity prevents identity disclosure by requiring at least k records of a dataset to have the same values over QI. Thus, a k-anonymous dataset upperbounds the probability of associating an individual with their record by $\frac{1}{k}$. To enforce k-anonymity most works [15, 20, 21, 23, 32] employ *generalization*, which replaces a QI value with a more general but semantically consistent value, or *suppression*, which removes QI values prior to data publishing.

k-anonymity has been considered in the context of publishing user trajectories, leading to several trajectory anonymization methods [4]. As mentioned in Section 1, these methods operate by anonymizing either entire trajectories [1, 2, 26], or parts of trajectories (i.e., sequences of locations) that may lead to identity disclosure [35, 39]. In the following, we discuss the main categories of trajectory anonymization works, as well as how our approach differs from them.

2.1 Clustering and perturbation

Methods based on clustering and perturbation are applied to time-stamped trajectories. They operate by grouping original trajectories into clusters (cylindrical tubes) of at least k trajectories, in a way that each trajectory within a cluster becomes indistinguishable from the other trajectories in the cluster. One such method, called *NWA* [1], enforces (k, δ)-anonymity to anonymize user trajectories by generating cylindrical volumes of radius δ that contain at least k trajectories. Each trajectory that belongs to an anonymity group (*cylinder*), generated by NWA, is protected from identity disclosure, due to the other trajectories that appear in the same group. To produce the cylindrical volumes, the algorithm in [1] identifies trajectories that lie close to each other in time and employs space translation. Trujillo-Rasua and Domingo-Ferrer [37] performed a rigorous analysis of the (k, δ)-anonymity model, which shows that this model is not able to hide an original trajectory within a set of k-indistinguishable, anonymized trajectories. Thus, the algorithms in [1, 2] may not provide meaningful privacy guarantees, in practice. An effective algorithm for enforcing k-anonymity on trajectory data was recently proposed by Domingo-Ferrer et al. [11]. The algorithm, called *SwapLocations*, creates trajectory clusters using *microaggregation* and then permutes the locations in each cluster to enforce privacy. The experimental evaluation in [11] demonstrates that *SwapLocations* is significantly more effective at preserving data utility than *NWA* [1]. Finally, Lin et al. [22] guarantees k-anonymity of published data, under the assumption that road-network data are public information. Their method uses clustering-based anonymization, protecting from identity disclosure.

Contrary to the methods of [1, 2, 11, 22], our work *(a)* does not consider time-stamped trajectories, and *(b)* applies generalization to derive an anonymized dataset.

2.2 Generalization and suppression

Differently to the methods of Section 2.1, this category of methods considers attackers with background knowledge on ordered sequences of places of interest (POIs) visited by specific individuals. Terrovitis et al. [35] proposed an approach to prohibit multiple attackers, each knowing a different set of POIs, from associating these POIs to fewer than k individuals in the published dataset. To achieve this, the authors developed a suppression-based method that aims at removing the least number of POIs from user trajectories, so that the remaining trajectories are k-anonymous with respect to the knowledge of each attacker.

Yarovoy et al. [39] proposed a k-anonymity based approach for publishing user trajectories by considering time as a quasi-identifier and supporting privacy personalization. Unlike previous approaches that assumed that all users share a common quasi-identifier, [39] assumes that each user has a different set of POIs and times requiring protection, thereby enabling each trajectory to be protected differently. To achieve k-anonymity, this approach uses generalization and creates anonymization groups that are not necessarily disjoint.

A recent approach, proposed by Monreale et al. [27], extends the l-diversity principle to trajectories by assuming that each location is either nonsensitive (acting as a QI) or sensitive. This approach applies c-safety to prevent attackers from linking sensitive locations to trajectories with a probability greater than c. To enforce c-safety, the proposed algorithm applies generalization to replace original POIs with generalized ones based on a location taxonomy. If generalization alone cannot enforce c-safety, suppression is used.

Assuming that each record in a dataset is comprised of a user's trajectory and user's sensitive attributes, Chen et al. [8] propose the $(K, C)_L$-privacy model. This model protects from identity and attribute linkage by employing local suppression. In this paper, the authors assume that an adversary knows at most L locations of a user's trajectory. Their model guarantees that a user is indistinguishable from at least $K - 1$ users, while the probability of linking a user to his/her sensitive values is at most C.

Contrary to the methods of [8, 27, 35, 39], our work *(a)* assumes that an attacker may know up to m user locations, which is a realistic assumption in many applications, and *(b)* does not classify locations as sensitive or nonsensitive, which may be difficult in some domains [36].

2.3 Differential privacy

Recently, methods for enforcing differential privacy [12] on trajectory data have been proposed [5, 7, 9]. The objective of these methods is to release noisy data summaries that are effective at supporting specific data analytic tasks, such as count query answering [7, 9] and frequent pattern mining [5]. To achieve this, the method in [9] uses a *context-free, taxonomy* tree, for identifying the set of counting queries that should be supported by the noisy summary, while the method in [5] employs a prefix-tree to generate candidate patterns, used in the construction of the data summary.

The method proposed in [7] was shown to be able to generate summaries that permit highly accurate count query answering. This method, referred to as NGRAMS, works in three steps. First, it truncates the original trajectory dataset by keeping only the first ℓ_{max} locations of each trajectory, where ℓ_{max} is a parameter specified by data publishers. Larger ℓ_{max} values improve efficiency but deteriorate the quality of the frequencies, calculated during the next step. Second, it uses the truncated dataset to compute the frequency of *n-grams* (i.e., all possible contiguous parts of trajectories that are comprised of 1, or 2, ... , or n locations). Third, this method constructs a differentially private summary by inserting calibrated Laplace noise [12] to the frequencies of n-grams.

Contrary to the methods of [5, 7, 9], our work publishes truthful data at a record (individual user) level, which is required by many data analysis tasks [24]. That is, our work retains the number of locations in each published trajectory and the number of published trajectories in the anonymized dataset. Furthermore, our method is able to preserve data utility significantly better than these methods, as shown in our extensive experiments. Thus, our approach can be used to offer a better privacy/utility trade-off than the methods of [5, 7, 9].

3 Privacy and utility objectives

In this section, we first define some preliminary concepts that are necessary to present our approach, and then discuss the privacy and utility objectives of our anonymization algorithms.

3.1 Preliminaries

Let $\mathcal{L}$ be a set of locations (e.g., points of interest, touristic sites, shops). A trajectory represents one or more locations in $\mathcal{L}$ and the order in which these locations are visited by a moving object (e.g., individual, bus, taxi), as explained in the following definition.

Definition 1. *A* trajectory *t is an ordered list of locations $(l_1, \ldots, l_n)$, where $l_i \in \mathcal{L}$, $1 \leq i \leq n$. The* size *of the trajectory $t = (l_1, \ldots, l_n)$, denoted by $|t|$, is the number of its locations, i.e., $|t| = n$.*

Note that, in our setting, a location may model points in space. A part of a trajectory, which is formed by removing some locations while maintaining the order of the remaining locations, is a *subtrajectory* of the trajectory, as explained below.

Definition 2. *A trajectory $s = (\lambda_1, \ldots, \lambda_\nu)$ is a* subtrajectory of *or is* contained in *trajectory $t = (l_1, \ldots, l_n)$, denoted by $s \sqsubseteq t$, if and only if $|s| \leq |t|$ and there is a mapping f such that $\lambda_1 = l_{f(1)}, \ldots, \lambda_\nu = l_{f(\nu)}$ and $f(1) < \cdots < f(\nu)$.*

For instance, the trajectory (a, e) is a subtrajectory of (or contained in) the trajectory $t_1 = (d, a, c, e)$ in Figure 1. Clearly, (a, e) can be obtained from t_1 by removing d and c.

Definition 3. *Given a set of trajectories $\mathcal{T}$, the* support *of a subtrajectory s, denoted by* $\sup(s, \mathcal{T})$, *is defined as the number of distinct trajectories in $\mathcal{T}$ that contain s.*

In other words, the support of a subtrajectory s measures the number of trajectories in a dataset that s is contained in. For example, for the dataset in Figure 1a, we have $\sup((a, e), \mathcal{T}) = 3$. Note that the support does not increase when a subtrajectory is contained multiple times in a trajectory. For instance, $\sup((a, e), \{(a, e, b, a, e)\}) = 1$. Naturally, by considering locations as unary trajectories, the support can also be measured for the locations of a dataset.

In this work, we adapt the notion of k^m-anonymity [35, 36] to trajectory data, as explained below.

Definition 4. *A set of trajectories $\mathcal{T}$ is* k^m-anonymous *if and only if every subtrajectory s of every trajectory $t \in \mathcal{T}$, which contains m or fewer locations (i.e., $|s| \leq m$), is contained in at least k distinct trajectories of $\mathcal{T}$.*

Definition 4 ensures that an attacker who knows any subtrajectory s of size m of an individual, cannot associate the individual to fewer than k trajectories (i.e., the probability of identity disclosure, based on s, is at most $\frac{1}{k}$). The privacy parameters k and m are specified by data publishers, according to their expectations about adversarial background knowledge, or certain data privacy policies [18, 35, 36].

The following example illustrates a dataset that satisfies k^m-anonymity.

Example 1. *Consider the trajectory dataset that is shown in Figure 1a. This dataset is 2^1-anonymous, because every location (i.e., subtrajectory of size 1) appears at least 2 times in it. This dataset is also 1^3-anonymous, because every subtrajectory of size 3 appears only once in it. However, the dataset is not 2^2-anonymous, as the subtrajectory (d, a) is contained only in the trajectory t_1.*

Note that, unlike k-anonymity, the k^m-anonymity model assumes that an attacker possesses background knowledge about subtrajectories, which are comprised of at most m locations. That is, an attacker knows at most m locations that are visited by an individual, in a certain order. Clearly, m can be set to any integer in $[0, \max\{|t| \mid t \in \mathcal{T}\}]$. Setting m to 0 corresponds to the trivial case, in which an attacker has no background knowledge. On the other hand, setting m to $\max\{|t| \mid t \in \mathcal{T}\}$, can be used to guard against an attacker who knows the maximum possible subtrajectory about an individual (i.e., that an individual has visited all the locations in their trajectory, and the order in which they visited these locations). In this case, k^m-anonymity "approximates" k-anonymity, but it does not provide the same protection guarantees against identity disclosure. This is because k^m-anonymity does not guarantee protection from attackers who know that an individual has visited *exactly* the locations, contained in a subtrajectory of size m. For example, assume that a dataset is comprised of the trajectories $\{(a, d, e), (a, d, e), (a, d)\}$. The dataset satisfies 2^3-anonymity, hence it prevents an attacker from associating an individual with any of the subtrajectories (a, d), (a, e), and (d, e). However, the dataset is not 2-anonymous, hence an attacker who knows that an individual has visited exactly the locations a and d, in this order, can uniquely associate the individual with the trajectory (a, d).

The k^m-anonymity model is practical in several applications, in which it is extremely difficult for attackers to learn a very large number of user locations [35]. However, k^m-anonymity does not guarantee that all possible attacks, based on background knowledge, will be prevented. For example, k^m-anonymity is not designed to prevent *collaborative attacks*, in which *(i)* two or more attackers combine their knowledge in order to re-identify an individual, or *(ii)* an attacker possesses background knowledge about multiple trajectories in $\mathcal{T}$. Such powerful attack schemes can only be handled within stronger privacy principles, such as differential privacy (see Section 2). However, applying these principles usually results in significantly lower data utility, compared to the output of our algorithms, as shown in our experiments. In addition, as we do not deal with time-stamped trajectories, time information is not part of our privacy model. In the case of time-stamped trajectory data publishing, time information can be used by attackers to perform identity disclosure, and privacy models to prevent this are the focus of [8, 39]. For the same reason, we do not deal with attacks that are based on both time and road-network information (e.g., the *inference route problem* [22]). These attacks can be thwarted using privacy models, such as *strict k*-anonymity [22].

To explain the way we generalize trajectories, we define the notion of *generalized location*, as explained below.

Definition 5. *A* generalized location $\{l_1, \ldots, l_v\}$ *is defined as a set of at least two locations* $l_1, \ldots, l_v \in \mathcal{L}$.

Thus, a generalized location is the result of replacing at least two locations in $\mathcal{L}$ with their set. A *generalized location* is interpreted as exactly one of the locations mapped to it. For example, the generalized location $\{a, b\}$ may be used to replace the locations a and b in Figure 1a. This generalized location will be interpreted as a or b. Therefore, if a trajectory t' in an anonymized version $\mathcal{T}'$ of $\mathcal{T}$ contains a generalized location $\{l_1, \ldots, l_v\}$, then the trajectory t in $\mathcal{T}$ contains exactly one of the locations $l_1, \ldots, l_v$.

To enforce k^m-anonymity, we either replace a location l with a generalized location that contains l, or leave l intact. Thus, a *generalized trajectory* t' is an ordered list of locations and/or generalized locations. The *size* of t', denoted by $|t'|$, is the number of elements of t'. For instance, a generalized trajectory $t' = (\{a, b\}, c)$ is comprised of one generalized location $\{a, b\}$ and a location c, and it has a size of 2.

We are interested in generalization transformations that produce a transformed dataset $\mathcal{T}'$ by distorting the original dataset $\mathcal{T}$ as little as possible, A common way to measure the distortion of a transformation is to measure the *distance* between the original and the transformed dataset [29, 35, 39]. In our case, the distance between the original and the anonymized dataset is defined as the *average* of the distances of their corresponding trajectories. In turn, the distance between the initial and the anonymized trajectory is defined as the *average* of the distance between their corresponding locations.

The following definition illustrates how the distance between locations, trajectories, and datasets $\mathcal{T}$ and $\mathcal{T}'$ can be computed.

Definition 6. *Let l be a location that will be generalized to the generalized location $\{l_1, \ldots, l_v\}$. The* location distance *between l and $\{l_1, \ldots, l_v\}$, denoted by $\mathcal{D}_{loc}(l, \{l_1, \ldots, l_v\})$, is defined as:*

$$\mathcal{D}_{loc}(l, \{l_1, \ldots, l_v\}) = \text{avg}\{\, d(l, l_i) \mid 1 \leq i \leq v\}$$

where d is the Euclidean distance. The trajectory distance *between $t = (l_1, \ldots, l_n)$ and its generalized counterpart $t' = (l'_1, \ldots, l'_n)$, denoted by $\mathcal{D}_{traj}(t, t')$, is defined as:*

$$\mathcal{D}_{traj}(t, t') = \text{avg}\{\mathcal{D}_{loc}(l_i, l'_i) \mid 1 \leq i \leq n\}$$

Finally, the trajectory dataset distance *between $\mathcal{T} = \{t_1, \ldots, t_u\}$ and its generalized counterpart $\mathcal{T}' = \{t'_1, \ldots, t'_u\}$ (where the trajectory t_i is generalized to trajectory t'_i, $1 \leq i \leq u$), denoted by $\mathcal{D}(\mathcal{T}, \mathcal{T}')$, is defined as:*

$$\mathcal{D}(\mathcal{T}, \mathcal{T}') = \text{avg}\{\mathcal{D}_{traj}(t_i, t'_i) \mid 1 \leq i \leq u\}$$

For example, let a, a_1, a_2 and b be locations and let $d(a, a_1) = 1$ and $d(a, a_2) = 2$. If location a is generalized to the generalized location $\{a, a_1, a_2\}$ the location distance $\mathcal{D}_{loc}(a, \{a, a_1, a_2\}) = (0 + 1 + 2)/3 = 1$. Also, if trajectory (a, b) is generalized to $(\{a, a_1, a_2\}, b)$ the trajectory distance $\mathcal{D}_{traj}((a, b), (\{a, a_1, a_2\}, b)) = (1 + 0)/2 = 1/2$.

Note that the distances in Definition 6 can be normalized by dividing each of them with the maximum distance between locations in $\mathcal{T}$.

3.2 Problem statement

As discussed in Introduction, the objective of our approach is to enforce k^m-anonymity to a trajectory dataset, while preserving data utility. However, there are different aspects of data utility that data publishers may want to preserve. To account for this, we have developed three anonymization algorithms, namely SEQANON, SD-SEQANON, and U-SEQANON, which generalize locations in different ways.

The SEQANON algorithm aims at generalizing together locations that are close in proximity. The distance between locations, in this case, is expressed based on Definition 6. Thus, the problem that SEQANON aims to solve can be formalized as follows.

Problem 1. *Given an original trajectory dataset $\mathcal{T}$, construct a k^m-anonymous version $\mathcal{T}'$ of $\mathcal{T}$ such that $\mathcal{D}(\mathcal{T}, \mathcal{T}')$ is minimized.*

Note that Problem 1 is NP-hard (the proof follows easily from observing that Problem 1 contains the NP-hard problem in [35] as a special case), and that SD-SEQANON is a heuristic algorithm that may not find an optimal solution to this problem.

The SD-SEQANON algorithm considers both the distance and the semantic similarity of locations, when constructing generalized locations. Thus, it exploits the availability of semantic information about locations to better preserve data utility. Following [27], we assume that the semantic information of locations is provided by data publishers, using a location taxonomy. The leaf-level nodes in the taxonomy correspond to each of the locations of the original dataset, while the non-leaf nodes represent more general (abstract) location information.

Given a location taxonomy, we define the notion of *semantic dissimilarity* for a generalized location, as explained in the following definition. A similar notion of semantic dissimilarity, for relational values, was proposed in [38].

Definition 7. *Let $l' = \{l_1, \ldots, l_v\}$ be a generalized location and $\mathcal{H}$ be a location taxonomy. The* semantic dissimilarity *for l' is defined as:*

$$\mathcal{SD}(l') = \frac{CCA(\{l_1, \ldots, l_v\})}{|\mathcal{H}|}$$

where $CCA(\{l_1, \ldots, l_j\})$ is the number of leaf-level nodes in the subtree rooted at the closest common ancestor of the locations $\{l_1, \ldots, l_v\}$ in the location taxonomy $\mathcal{H}$, and $|\mathcal{H}|$ is the total number of leaf-level nodes in $\mathcal{H}$.

Thus, locations that belong to subtrees with a small number of leaves are more semantically similar. Clearly, the $\mathcal{SD}$ scores for generalized locations that contain more semantically similar locations are lower.

Example 2. *An example location taxonomy is illustrated in Figure 2. The leaf-level nodes a to e represent the locations (i.e., specific restaurants and coffee houses), while the non-leaf nodes represent the general concepts* Restaurants *and* Coffee shops. *We also have $CCA(\{a, c, e\}) = 3$, as the subtree rooted at* Restaurants *has three leaf-level nodes, and $|\mathcal{H}| = 5$, as the taxonomy in Figure 2 has 5 leaf-level nodes. Thus, the semantic dissimilarity for the generalized location $\{a, c, e\}$, denoted with $\mathcal{SD}(\{a, c, e\})$, is $\frac{3}{5} = 0.6$. Similarly, we can compute $SD(\{a, d\}) = \frac{5}{5} = 1$, which is greater than $SD(\{a, c, e\})$ because a and d are more semantically dissimilar (i.e., restaurants and coffee shops, instead of just restaurants).*

We now define the criteria that are used by the SD-SEQANON algorithm to capture both the distance and semantic similarity between locations, trajectories, and datasets $\mathcal{T}$ and $\mathcal{T}'$. The computation of these criteria is similar to those in Definition 6.

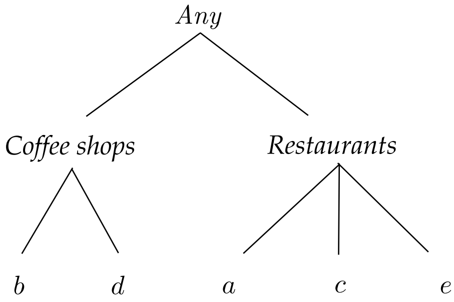


Figure 2: A location taxonomy

Definition 8. *Let l be a location that will be generalized to $l' = \{l_1, \ldots, l_v\}$. The* combined location distance *between l and l', denoted by $\mathcal{C}_{loc}(l, l')$, is defined as:*

$$\mathcal{C}_{loc}(l, l') = \text{avg}\big\{d(l, l_i) \cdot SD(l') \mid 1 \le i \le v\big\}, \textit{where } SD(l') \textit{ takes values in } (0, 1]$$

where d is the Euclidean distance and $\mathcal{SD}$ is the semantic dissimilarity. Note that the above formula is a conventional weighted-formula, where similarity and distance are combined into the $\mathcal{C}_{loc}(l, l')$. Thus, the combined objective is then optimized by the single-objective optimization metric $\mathcal{C}_{loc}(l, l')$. Using conventional weighted-formulas is an effective approach for addressing multi-objective optimization problems, as discussed in [13]. The combined trajectory distance *between $t = (l_1, \ldots, l_n)$ and its generalized counterpart $t' = (l'_1, \ldots, l'_n)$, denoted by $\mathcal{C}_{traj}(t, t')$, is defined as:*

$$\mathcal{C}_{traj}(t, t') = \text{avg}\big\{\mathcal{C}_{loc}(l_i, l'_i) \mid 1 \le i \le n\big\}$$

Finally, the combined trajectory dataset distance *between $\mathcal{T} = \{t_1, \ldots, t_u\}$ and its generalized counterpart $\mathcal{T}' = \{t'_1, \ldots, t'_u\}$ (where the trajectory t_i is generalized to trajectory t'_i, $1 \le i \le u$), denoted by $\mathcal{C}(\mathcal{T}, \mathcal{T}')$, is defined as:*

$$\mathcal{C}(\mathcal{T}, \mathcal{T}') = \text{avg}\big\{\mathcal{C}_{traj}(t_i, t'_i) \mid 1 \le i \le u\big\}$$

We now define the problem that the SD-SEQANON algorithm aims to solve, as follows.

Problem 2. *Given an original trajectory dataset $\mathcal{T}$, construct a k^m-anonymous version $\mathcal{T}'$ of $\mathcal{T}$ such that $\mathcal{C}(\mathcal{T}, \mathcal{T}')$ is minimized.*

Note that Problem 2 can be restricted to Problem 1, by allowing only instances where $SD(l') = 1$, for each generalized location l' that is contained in a trajectory of $\mathcal{T}'$. Thus, Problem 2 is also NP-hard. The SD-SEQANON algorithm aims to derive a (possibly sub-optimal) solution to Problem 2 by taking into account both the distance and the semantic similarity of locations, when constructing generalized locations.

However, in several applications, there are specific utility requirements that must be taken into account to ensure that the published dataset is practically useful. In what follows, we explain our notion of utility constraints, which is used to capture the utility requirements of data publishers, and explain when the utility constraints are satisfied. Subsequently, we discuss the practical importance of utility constraints in applications. Our definitions are similar to those proposed in [24] for transaction data.

utility constraints	*locations*
u_1	$\{b, c\}$
u_2	$\{a, d, e\}$

(a)

id	trajectory
t'_1	$(d, \{a, b, c\}, \{a, b, c\}, e)$
t'_2	$(\{a, b, c\}, \{a, b, c\}, e, \{a, b, c\})$
t'_3	$(\{a, b, c\}, d, e)$
t'_4	$(\{a, b, c\}, d, e, \{a, b, c\})$
t'_5	$(d, \{a, b, c\})$
t'_6	(d, e)

(b)

id	trajectory
t'_1	$(\{a, d, e\}, \{a, d, e\}, \{b, c\}, \{a, d, e\})$
t'_2	$(\{b, c\}, \{a, d, e\}, \{a, d, e\}, \{b, c\})$
t'_3	$(\{a, d, e\}, \{a, d, e\}, \{a, d, e\})$
t'_4	$(\{b, c\}, \{a, d, e\}, \{a, d, e\}, \{b, c\})$
t'_5	$(\{a, d, e\}, \{b, c\})$
t'_6	$(\{a, d, e\}, \{a, d, e\})$

(c)

Figure 3: *(a)* An example utility constraint set $\mathcal{U}$. *(b)* A 2^2-anonymous dataset $\mathcal{T}'$ that does not satisfy the utility constraint set $\mathcal{U}$. *(c)* A 2^2-anonymous dataset $\mathcal{T}'$ that satisfies $\mathcal{U}$.

Definition 9. *A* utility constraint *u is a set of locations $\{l_1, \ldots, l_v\}$, specified by data publishers. A* utility constraint set *$\mathcal{U} = \{u_1, \ldots, u_p\}$ is a partition of the set of locations $\mathcal{L}$, which contains all the specified utility constraints $u_1, \ldots, u_p$.*

Definition 10. *Given a utility constraint set $\mathcal{U} = \{u_1, \ldots, u_p\}$, a generalized dataset $\mathcal{T}'$ that contains a set of generalized locations $\{l'_1, \ldots, l'_n\}$, and a parameter δ, $\mathcal{U}$ is satisfied if and only if* (i) *for each generalized location $l' \in \{l'_1, \ldots, l'_n\}$, and for each utility constraint $u \in \mathcal{U}$, $l' \subseteq u$ or $l' \cap u = \varnothing$, and* (ii) *at most $\delta\%$ of the locations in $\mathcal{L}$ have been suppressed to produce $\mathcal{T}'$, where δ is a parameter specified by data publishers.*

The first condition of Definition 10 limits the maximum amount of generalization each location is allowed to receive, by prohibiting the construction of generalized locations whose elements (locations) span multiple utility constraints. The second condition ensures that the number of suppressed locations is controlled by a threshold. When both of these conditions hold, the utility constraint set $\mathcal{U}$ is satisfied. Note that we assume that the utility constraint set is provided by data publishers, e.g., using the method in [24]. The example below illustrates Definitions 9 and 10.

Example 3. *Consider the utility constraint set $\mathcal{U} = \{u_1, u_2\}$, shown in Figure 3a, and assume that $\delta = 5$. The dataset, shown in Figure 3b, does not satisfy $\mathcal{U}$, because the locations in the generalized location $\{a, b, c\}$ are contained in both u_1 and u_2. On the other hand, the dataset in Figure 3c satisfies $\mathcal{U}$, because the locations of every generalized location are all contained in u_1.*

In the following, we provide two examples of real-life applications to justify the importance of utility constraints. The first example comes from the business domain, and it is related to the *Octopus* card, used for payment at various sites (e.g., at convenience stores and service stations) in Hong Kong. Data published by the Octopus card company must preserve privacy and, at the same time, allow meaningful analysis by data recipients, such as owners of specific shops or certain public authorities [35]. The analysis of data recipients often involves counting the number of trajectories that are associated with a particular type of locations, such as *coffee shops* and *bus stations* when data recipients are coffee shops owners and public transport authorities, respectively. The second example comes from the healthcare domain, and it is related to publishing the locations (e.g., healthcare institutions, clinics, and pharmacies) visited by patients [25]. To support medical research studies, it is important that the published data permit data recipients to accurately count the number of trajectories (or equivalently the number of patients) that are associated with specific locations, or types of locations.

Observe that the number of trajectories that contain a generalized location $l' = (l_1, \ldots, l_n)$, in the generalized dataset $\mathcal{T}'$, is equal to the number of trajectories that contain *at least* one of the locations $l_1, \ldots, l_n$, in the original dataset $\mathcal{T}$. This is because a trajectory $t \in \mathcal{T}$ that contains at least one of these locations corresponds to a trajectory $t' \in \mathcal{T}'$ that contains l'. Thus, the number of trajectories in $\mathcal{T}$ that contain any location in a utility constraint $u \in \mathcal{U}$ can be accurately computed from the generalized dataset $\mathcal{T}'$, when $\mathcal{U}$ is satisfied, as no other location will be generalized together with the locations in u. Therefore, the generalized data that satisfy $\mathcal{U}$ will be practically useful in the aforementioned applications.

We now define the problem that U-SeqAnon aims to solve, as follows.

Problem 3. *Given an original trajectory dataset $\mathcal{T}$, a utility constraint set $\mathcal{U}$, and parameters k, m and δ, construct a k^m-anonymous version $\mathcal{T}'$ of $\mathcal{T}$ such that $\mathcal{D}(\mathcal{T}, \mathcal{T}')$ is minimized and $\mathcal{U}$ is satisfied with at most $\delta\%$ of the locations of $\mathcal{T}$ being suppressed.*

Thus, a solution to Problem 3 needs to satisfy the specified utility constraints, without suppressing more than $\delta\%$ of locations, and additionally incur minimum information loss. Note that Problem 3 is NP-hard (it can be restricted to Problem 1, by allowing only instances where $\mathcal{U}$ contains an single utility constrained with all locations in $\mathcal{L}$ and $\delta = 100$) and that U-SeqAnon is a heuristic algorithm that may not solve Problem 3 optimally.

4 Anonymization algorithms

In this section, we present our SeqAnon, SD-SeqAnon, and U-SeqAnon anonymization algorithms, which aim at solving Problems 1, 2, and 3, respectively.

4.1 The SeqAnon algorithm

The pseudocode of the SeqAnon algorithm is illustrated in Algorithm SeqAnon. The algorithm takes as input a trajectory dataset $\mathcal{T}$, and the anonymization parameters k and m, and returns the k^m-anonymous counterpart $\mathcal{T}'$ of $\mathcal{T}$. The algorithm employs the apriori principle and works in a bottom up fashion. Initially, it considers and generalizes the subtrajectories of size 1 (i.e., single locations) in $\mathcal{T}$ which have low support. Then, the algorithm continues by progressively increasing the size of the subtrajectories it considers.

In more detail, SeqAnon proceeds as follows. First, it initializes $\mathcal{T}'$ (Step 1). Then, in Steps 2 – 8, it considers subtrajectories of size up to m, iteratively. Specifically, in Step 3, it computes the set $\mathcal{S}$, which contains all subtrajectories in $\mathcal{T}$ that have size i (i.e., that have i locations) and support lower than k (i.e., $\sup(s, \mathcal{T}') < k$). SeqAnon considers the subtrajectories of $\mathcal{S}$ that have lower support first. This heuristic improves both the efficiency and the effectiveness of the algorithm. This is because remedying such subtrajectories does not require a large amount of generalization, while it contributes to protecting trajectories with higher support. Continuing, for every such trajectory $s \in \mathcal{S}$, the algorithm finds the location l_1 of s with the minimum support (Step 6). SeqAnon considers locations with

```
Algorithm: SEQANON
Input: A dataset $\mathcal{T}$ and anonymization parameters $k$ and $m$
Output: A $k^m$-anonymous dataset $\mathcal{T}'$ corresponding to $\mathcal{T}$
1 $\mathcal{T}' := \mathcal{T}$ // Initialize output
2 for $i := 1$ to $m$ do
3     Let $\mathcal{S}$ be the set of subtrajectories $s$ of $\mathcal{T}$ with size $i$ such that $\sup(s, \mathcal{T}') < k$ sorted by
      increasing support
4     for each $s \in \mathcal{S}$ do
5         while $\sup(s, \mathcal{T}') < k$ do
6             Find the location $l_1$ of $s$ with the minimum support in $\mathcal{T}'$
7             Find the location $l_2 \neq l_1$ with the minimum $d(l_1, l_2)$
8             Replace all occurrences of $l_1$ and $l_2$ in $\mathcal{T}'$ and $s$ with $\{l_1, l_2\}$
9 return $\mathcal{T}'$
```

low support first, as they can be generalized with low information loss. Then, in Step 7, the algorithm searches the locations of $\mathcal{T}$ to detect the location l_2 that has the minimum Euclidean distance from l_1. Finally, SEQANON generalizes l_1 and l_2 by constructing the generalized location $\{l_1, l_2\}$ and replaces every occurrence of l_1 and l_2 with the generalized location $\{l_1, l_2\}$ (Step 8). The algorithm repeats Steps 6 – 8, until the support of the subtrajectory s exceeds the value of the anonymization parameter k.

The following is an example of SEQANON in operation.

Example 4. *We will demonstrate the operation of* SEQANON *using dataset $\mathcal{T}$ of Figure 1a and $k = m = 2$. The intermediate steps are illustrated in Figure 4. The first iteration of the for loop (Steps 2 – 8) considers the subtrajectories of size $i = 1$. It is not hard to verify that all size 1 locations have support greater (or equal) than $k = 2$, thus the algorithm proceeds to $i = 2$. For this case, Step 3 computes the set of subtrajectories $\mathcal{S}$ (illustrated in Figure 4a).* SEQANON *considers subtrajectory $s = (d, a)$, which is the first subtrajectory in $\mathcal{S}$. Then, the algorithm sets $l_1 = a$, because a is the location with the lowest support in (d, a) (Step 6), and $l_2 = b$, because $d(a, b)$ is minimum, according to Figure 1b (Step 7). Finally, in Step 8* SEQANON *replaces a and b with the generalized location $\{a, b\}$ in s and in all the trajectories of $\mathcal{T}'$. After these replacements, we have $s = (d, \{a, b\})$ and the resultant $\mathcal{T}'$ shown in Figure 4b. Since, we still have $\sup(s, \mathcal{T}') < k$, the while loop (Steps 5 – 8) is executed again. This time, $l_1 = \{a, b\}$ and $l_2 = c$, and the algorithm constructs the generalized dataset $\mathcal{T}'$, shown in Figure 4c. The remaining steps of the algorithm* SEQANON *do not change $\mathcal{T}'$. Thus, the final output of* SEQANON *is shown in Figure 4c.*

Time complexity analysis. We first compute the time needed by SEQANON to execute the **for** loop (Steps 2 – 8). For each iteration of this loop, the set $\mathcal{S}$ is constructed, sorted, and explored. The cost of creating and sorting this set is $\mathcal{O}(|\mathcal{L}|^i)$ and $\mathcal{O}(|\mathcal{L}|^i \cdot log(|\mathcal{L}|^i))$, respectively, where $|\mathcal{L}|$ is the size of the location set used in $\mathcal{T}$ and i is the loop counter. These bounds are very crude approximations, which correspond to the case in which all size i subtrajectories have support lower than k. In practice, however, the number of the subtrajectories s with $\sup(s, \mathcal{T}') < k$ is a small fraction of $\mathcal{O}(|\mathcal{L}|^i)$, which depends heavily on the dataset $\mathcal{T}$ and the value of the anonymization parameter k. The cost of exploring the set $\mathcal{S}$ (Steps 4 – 8) is $\mathcal{O}(|\mathcal{L}|^i \cdot (|\mathcal{L}| + |\mathcal{T}'|))$ because, for each element of $\mathcal{S}$, the algorithm needs to consider at most $\mathcal{O}(|\mathcal{L}|)$ locations and access all trajectories in $\mathcal{T}'$. Thus, each iteration of the **for** loop, in Steps 2 – 8, takes $\mathcal{O}(|\mathcal{L}|^i \cdot (log(|\mathcal{L}|^i) + |\mathcal{L}| + |\mathcal{T}'|))$ time, and the time

subT.	sup
(d,a)	1
(c,e)	1
(b,a)	1
(a,d)	1
(b,d)	1

(a)

id	trajectory
t'_1	$(d,\{a,b\},c,e)$
t'_2	$(\{a,b\},\{a,b\},e,c)$
t'_3	$(\{a,b\},d,e)$
t'_4	$(\{a,b\},d,e,c)$
t'_5	(d,c)
t'_6	(d,e)

(b)

id	trajectory
t'_1	$(d,\{a,b,c\},\{a,b,c\},e)$
t'_2	$(\{a,b,c\},\{a,b,c\},e,\{a,b,c\})$
t'_3	$(\{a,b,c\},d,e)$
t'_4	$(\{a,b,c\},d,e,\{a,b,c\})$
t'_5	$(d,\{a,b,c\})$
t'_6	(d,e)

(c)

Figure 4: *(a)* Set $\mathcal{S}$ for subtrajectories of size $i = 2$ and the respective supports, *(b)* Transformed dataset $\mathcal{T}'$ after SEQANON has processed the subtrajectory (d,a), and *(c)* The final 2^2-anonymous result $\mathcal{T}'$, produced by SEQANON.

complexity of SEQANON is $\mathcal{O}(\sum_{i=1}^{m}(|\mathcal{L}|^i \cdot (log(|\mathcal{L}|^i) + |\mathcal{L}| + |\mathcal{T}'|)))$.

4.2 The SD-SEQANON algorithm

SD-SEQANON takes as input an original trajectory dataset $\mathcal{T}$, the anonymization parameters k and m, and a location taxonomy, and returns the k^m-anonymous counterpart $\mathcal{T}'$ of $\mathcal{T}$. The algorithm operates similarly to SEQANON, but it takes into account both the Euclidean distance and the semantic similarity of locations, when it applies generalization to them.

The pseudocode of SD-SEQANON is provided in Algorithm SD-SEQANON. Notice that SD-SEQANON and SEQANON differ in Step 7. This is because SD-SEQANON calculates the product of the Euclidean distance for the locations l_1 and l_2, and the $\mathcal{SD}$ measure for the generalized location $\{l_1, l_2\}$ (see Definition 7). Thus, it aims at creating a generalized location, which consists of locations that are close in proximity and are semantically similar. The time complexity of SD-SEQANON is the same as that of SEQANON, because the computation in Step 7 does not affect the complexity.

Algorithm: SD-SEQANON

Input: A dataset $\mathcal{T}$, a locations hierarchy and anonymization parameters k and m
Output: A k^m-anonymous dataset $\mathcal{T}'$ corresponding to $\mathcal{T}$

```
1 $\mathcal{T}' := \mathcal{T}$ // Initialize output
2 for $i := 1$ to $m$ do
3     Let $\mathcal{S}$ be the set of subtrajectories $s$ of $\mathcal{T}$ with size $i$ such that $\sup(s,\mathcal{T}') < k$ sorted by
      increasing support
4     for each $s \in \mathcal{S}$ do
5         while $\sup(s,\mathcal{T}') < k$ do
6             Find the location $l_1$ of $s$ with the minimum support in $\mathcal{T}'$
7             Find the location $l_2 \neq l_1$ with the minimum $d(l_1,l_2) \cdot \mathcal{SD}(\{l_1,l_2\})$
8             Replace all occurrences of $l_1$ and $l_2$ in $\mathcal{T}'$ and $s$ with $\{l_1,l_2\}$
9 return $\mathcal{T}'$
```

4.3 The U-SeqAnon algorithm

The U-SeqAnon algorithm takes as input an original trajectory dataset $\mathcal{T}$, anonymization parameters k, m and δ, as well as a utility constraint set $\mathcal{U}$. The algorithm differs from SeqAnon and SD-SeqAnon along two dimensions. First, the k^m-anonymous dataset it produces satisfies $\mathcal{U}$, hence it meets the data publishers' utility requirements. Second, it additionally employs suppression (i.e., removes locations from the resultant dataset), when generalization alone is not sufficient to enforce k^m-anonymity.

Algorithm: U-SeqAnon

Input: A dataset $\mathcal{T}$, utility constraint set $\mathcal{U}$, and anonymization parameters k, m, and δ
Output: A k^m-anonymous dataset $\mathcal{T}'$ corresponding to $\mathcal{T}$

```
1  $\mathcal{T}' := \mathcal{T}$ // Initialize output
2  for $i := 1$ to $m$ do
3      Let $\mathcal{S}$ be the set of subtrajectories $s$ of $\mathcal{T}$ with size $i$ such that $\sup(s, \mathcal{T}') < k$ sorted by
       increasing support
4      for each $s \in \mathcal{S}$ do
5          while $\sup(s, \mathcal{T}') > 0$ or $\sup(s, \mathcal{T}') < k$ do
6              Find the location $l_1$ of $s$ with the minimum support in $\mathcal{T}'$
7              Find the utility constraint $u \in \mathcal{U}$ that contains the location $l_1$
8              Find the location $l_2 \neq l_1, l_2 \in u$ with the minimum $d(l_1, l_2)$
9              if Cannot find location $l_2$ then
10                 Suppress location $l_1$ from $\mathcal{T}'$
11                 if More than $\delta\%$ of locations have been suppressed then
12                     Exit: $\mathcal{U}$ is not satisfied
13             else
14                 Replace all occurrences of $l_1$ and $l_2$ in $\mathcal{T}'$ and $s$ with $\{l_1, l_2\}$
15 return $\mathcal{T}'$
```

The pseudocode of U-SeqAnon is provided in Algorithm U-SeqAnon. As can be seen, the algorithm initializes $\mathcal{T}'$ (Step 1) and then follows the apriori principle (Steps 2 – 14). After constructing and sorting $\mathcal{S}$, U-SeqAnon iterates over each subtrajectory in $\mathcal{S}$ and applies generalization and/or suppression, until its support is either at least k or 0 (Steps 3 – 5). Notice that $sup(s, \mathcal{T}') = 0$ corresponds to an empty subtrajectory s (i.e., the result of suppressing all locations in s), which does not require protection. Next, U-SeqAnon finds the location l_1 with the minimum support in $\mathcal{T}'$ and the utility constraint that contains it (Steps 6 – 7). Then, the algorithm finds a different location l_2, which also belongs to u and is as close to l_1 as possible, according to the Euclidean distance (Step 8). In case such a location cannot be found (i.e., when there is a single generalized location that contains all locations in u), U-SeqAnon suppresses l_2 from $\mathcal{T}'$ (Steps 9 – 10). If more than $\delta\%$ of locations have been suppressed, $\mathcal{U}$ cannot be satisfied and the algorithm terminates (Steps 11 – 12). Otherwise, U-SeqAnon generalizes l_1 and l_2 together and replaces every occurrence of either of these locations with the generalized location $\{l_1, l_2\}$ (Step 14). The algorithm repeats Steps 2 – 14 as long as the size of the considered subtrajectories does not exceed m. After considering the subtrajectories of size m, U-SeqAnon returns the k^m-anonymous dataset $\mathcal{T}'$ that satisfies $\mathcal{U}$, in Step 15.

The following is an example of U-SeqAnon in operation.

Example 5. *We will demonstrate the operation of* U-SeqAnon *with input the original dataset*

$\#$	UC
1	$\{b,c\}$
2	$\{a,d,e\}$

(a)

id	trajectory
t'_1	$(\{a,d\},\{a,d\},c,e)$
t'_2	$(b,\{a,d\},e,c)$
t'_3	$(\{a,d\},\{a,d\},e)$
t'_4	$(b,\{a,d\},e,c)$
t'_5	$(\{a,d\},c)$
t'_6	$(\{a,d\},e)$

(b)

id	trajectory
t'_1	$(\{a,d,e\},\{a,d,e\},\{b,c\},\{a,d,e\})$
t'_2	$(\{b,c\},\{a,d,e\},\{a,d,e\},\{b,c\})$
t'_3	$(\{a,d,e\},\{a,d,e\},\{a,d,e\})$
t'_4	$(\{b,c\},\{a,d,e\},\{a,d,e\},\{b,c\})$
t'_5	$(\{a,d,e\},\{b,c\})$
t'_6	$(\{a,d,e\},\{a,d,e\})$

(c)

Figure 5: *(a)* Set of Utility Constraints *(b)* Transformed dataset $\mathcal{T}'$ after the processing of subtrajectory (d,a), and *(c)* The final 2^2-anonymous result $\mathcal{T}'$ meeting the provided set of UC

$\mathcal{T}$, shown in Figure 1a, the utility constraint set in Figure 5a, $k = m = 2$, and $\delta = 5\%$. During the first iteration of the for loop (Steps 2 – 14), U-SeqAnon *considers the subtrajectories of size $i = 1$, which all have a support of at least 2. Thus, the algorithm considers subtrajectories of size $i = 2$, and constructs the set $\mathcal{S}$ shown in Figure 4a (Steps 4 – 3). Then,* U-SeqAnon *considers the subtrajectory $s = (d,a)$ in $\mathcal{S}$, which has the lowest support in $\mathcal{T}'$ (Step 6). Next, in Steps 6 – 8, the algorithm finds the location $l_1 = a$, which has the lowest support in $\mathcal{T}'$, and the location $l_2 = d$, which belongs to the same utility constraint as a and is the closest to it – see also the map in Figure 1b. After that, the algorithm replaces a and d with the generalized location $\{a,d\}$ in s and all the trajectories of $\mathcal{T}'$ (Step 14). After these replacements, $s = (\{a,d\},\{a,d\})$ and the trajectory dataset $\mathcal{T}'$ is as shown in Figure 4b. Since the support of s in $\mathcal{T}'$ is at least k, the while loop in Step 5 terminates and the algorithm checks the next problematic subtrajectory, $s = (c,e)$. After considering all problematic subtrajectories of size 2,* U-SeqAnon *produces the 2^2-anonymous dataset in Figure 5c, which satisfies $\mathcal{U}$.*

The time complexity of U-SeqAnon is the same as that of SeqAnon, in the worst case when $\mathcal{U}$ is comprised of a single utility constraint that contains all locations in $\mathcal{L}$, $\mathcal{S}$ contains $O(|\mathcal{L}|^i)$ subtrajectories with support in $(0,k)$, and $\delta = 100$. Note that the cost of suppressing a location l_1 is $O(|\mathcal{T}'|)$ (i.e., the same as that of replacing the locations l_1 and l_2 with the generalized location (l_1,l_2) in all trajectories in $\mathcal{T}'$).

5 Experimental evaluation

In this section, we provide a thorough experimental evaluation of our approach, in terms of data utility and efficiency.

Experimental setup. We implemented our algorithms in C++ and tested them on an Intel Core i7 at 2.2 GHz with 6 GB of RAM. In our experiments, we used both synthetic and real datasets. The synthetic dataset, referred to as Oldenburg, was generated using the Brinkhoff's data generator [6] and contains synthetic trajectories of objects moving on the Oldenburg city map. This setting has been used by many works [1, 29, 35, 39]. We normalized the trajectories, so that all coordinates take values in a $10^3 \times 10^3$ map, and simulated trajectories corresponding to these routes, as follows. The map was divided into 100 regions using a uniform grid. A moving object visits a sequence of regions in a certain order. The centroids of the visited regions model the locations in the trajectories of $\mathcal{T}$. The Oldenburg dataset contains $18,143$ trajectories, whose average length is 4.72, and 100 locations.

In addition, we used a dataset that has been derived from the Gowalla location-based social networking website and contains the check-ins (locations) of users between February 2009 and October 2010 [10]. In our experiments, we used aggregate locations of $86,061$ users, in the state of New York and nearby areas. This dataset is referred to as Gowalla and contains $86,061$ trajectories, whose average length is 3.92, and 662 locations.

To study the effectiveness of our approach, we compare it against the NGRAMS method [7], discussed in Section 2, using the implementation provided by the authors of [7]. Contrary to [5], the NGRAMS method was developed for count query answering. Thus, a comparison between the NGRAMS method and ours allows us to evaluate the effectiveness of our approach with respect to count query answering. For this comparison, we set the parameters l_{max}, n_{max}, and e to the values 20, 5, and 0.1, respectively, which were suggested in [7]. Unless otherwise stated, k is set to 5 and m is set to 2. The location taxonomies were created as in [35]. Specifically, each non-leaf node in the taxonomy for the Oldenburg (respectively, Gowalla) dataset has 5 (respectively, 6) descendants.

Measures. To measure data utility we used the *Average Relative Error (ARE)* measure [21], which has become a defacto data utility indicator [24, 31]. ARE estimates the average number of trajectories that are retrieved incorrectly, as part of the answers to a workload of COUNT queries. Low ARE scores imply that anonymized data can be used to accurately estimate the number of co-occurrences of locations. We used workloads of 100 COUNT queries, involving randomly selected subtrajectories with size in $[1, 2]$, because the output of the NGRAMS method contained very few longer trajectories (see Figure 10b).

In addition, we used *Kullback–Leibler divergence* (KL-divergence), an information-theoretic measure that quantifies information loss and is used widely in the anonymization literature [16, 19]. In our context, KL-divergence measures support estimation accuracy based on the difference between the distribution of the support of a set of subtrajectories S, in the original and in the anonymized data[2]. Let P_S (respectively, Q_S) be the distribution of the support of the subtrajectories in S in the dataset $\mathcal{T}$ (respectively, generalized dataset $\mathcal{T}'$). The KL-divergence between P_S and Q_S is defined as:

$$D_{KL}(P_S \parallel Q_S) = \sum_{s \in S} P_S \ln\left(\frac{P_S}{Q_S}\right)$$

Clearly, low values in KL-divergence are preferred, as they imply that a small amount of information loss was incurred to generalize the subtrajectories in S. Furthermore, we used statistics on the number, size, and distance, for the locations in generalized data.

To evaluate the extent to which SD-SEQANON generalizes semantically close locations together, we compute a *semantic similarity penalty* $\mathcal{P}$ for the generalized dataset, as follows:

$$\mathcal{P}(\mathcal{T}') = \frac{\sum_{t' \in \mathcal{T}'} \left(\frac{1}{|t'|} \cdot \sum_{l' \in t'} \mathcal{SD}(l')\right)}{|\mathcal{T}'|}$$

where t' is a trajectory in the generalized dataset $\mathcal{T}'$, with size $|t'|$, and $|\mathcal{T}'|$ is the number of trajectories in $\mathcal{T}'$. $\mathcal{P}$ reflects how semantically dissimilar are (on average) the locations in the trajectories in the generalized dataset. The values in $\mathcal{P}(\mathcal{T}')$ are in $[0, 1]$ and lower values imply that the generalized locations in $\mathcal{T}'$ contain more semantically similar locations.

[2] The support of a subtrajectory $s \in S$ in $\mathcal{T}'$ is computed as the support of its generalized counterpart (i.e., the subtrajectory induced by the generalized locations of each location in s).

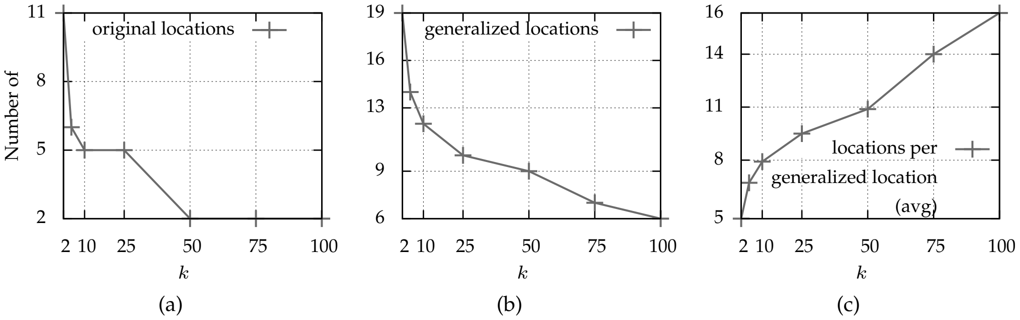
To evaluate the ability of the SEQANON and NGRAMS algorithms to permit sequential pattern mining, we employ a testing framework similar to that of Gionis et al. [17]. In more detail, we construct random projections of the datasets produced by our algorithms, by replacing every generalized location in it with a random location, selected from that generalized location. Then, we use Prefixspan [30], to obtain the frequent sequential patterns (i.e., the sequential patterns having support larger than a threshold) in the original, the output of NGRAMS and the projected datasets. Next, we calculate the percentage of the frequent sequential patterns of the original dataset that are preserved in the output of NGRAMS and in the projected datasets. We also calculate the percentage of the frequent sequential patterns in the output of NGRAMS and in the projected datasets that are not frequent in the original dataset. Clearly, an anonymized dataset (produced by either NGRAMS or by our algorithms) allows accurate mining, when: *(i)* a high percentage of its frequent sequential patterns are frequent in the original dataset, and *(ii)* a low percentage of its frequent sequential are not frequent in the original dataset.

5.1 Data utility

In this section, we evaluate the effectiveness of the SEQANON, SD-SEQANON, and U-SEQANON algorithms at preserving data utility.

SEQANON. We begin by evaluating the data utility offered by the SEQANON algorithm, using some general statistics, computed for the output of this algorithm on the Oldenburg dataset. Specifically, we measured the number of the locations that were released intact (referred to as original locations) and the number of the locations that were generalized. For the generalized locations, we also measured their average size and distance. Initially, we varied the anonymization parameter k in [2, 100]. Our results are summarized in Figures 6a-7a.

In Figure 6a, we present the number of the original locations, as a function of k. As expected, increasing k led to fewer original locations. In Figure 6b, we report the number of generalized locations. When k increases, more locations are grouped together to ensure k^m-anonymity, leading to fewer generalized locations. As an immediate result, the average number of locations in a generalized location increased, as shown in Figure 6c. In addition, we report the average Euclidean distance of all locations contained in each generalized location. We normalize this distance as a percentage of the maximum possible distance (i.e., the distance between the two furthermost points). This percentage quantifies the distortion



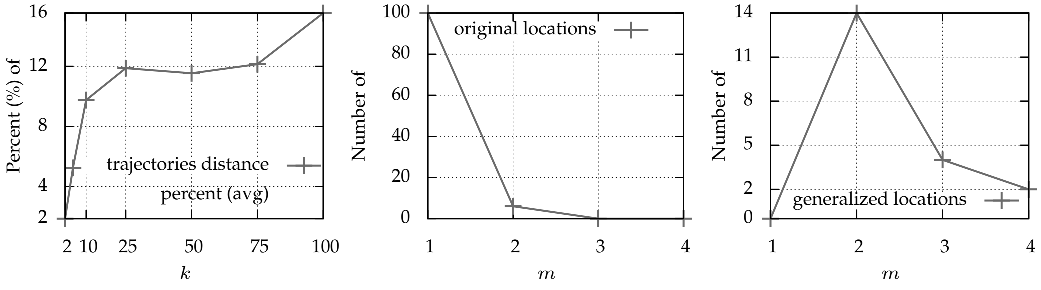
(a) (b) (c)

Figure 6: number of *(a)* original (non generalized) locations published, *(b)* generalized locations published and *(c)* average generalized location size

in a generalized location. In Figure 7a, we illustrate the distance percentage as a function of k. When k increases, more locations are grouped together in the same generalized location, leading to more distortion. As SEQANON focuses on minimizing the Euclidean distance of locations in each generalized location, the distortion is relatively low and increases slowly.

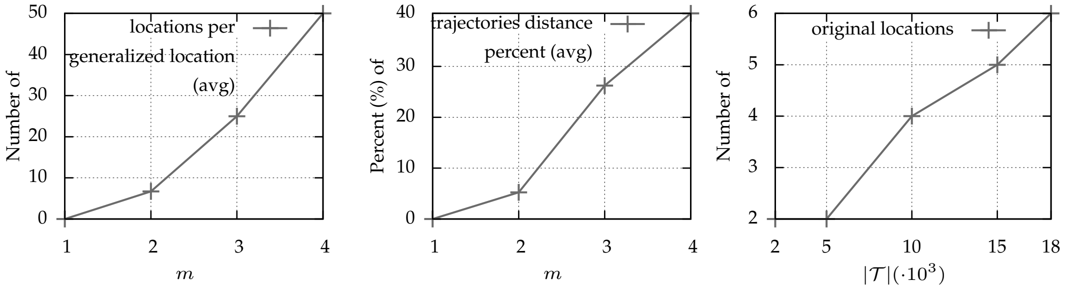
To show the impact of m on data utility, we set $k = 5$ and varied m in [1,4]. Since our dataset has an average of 4.72 locations per trajectory, $m = 3$ (respectively, $m = 4$) means that the adversary knows approximately 65% (respectively, 85%) of a user's locations. So, for $m = 3$ and $m = 4$, we expect significant information loss. On the contrary, for $m = 1$, all locations have support greater than $k = 5$, so no generalization is performed and no generalized locations are created. As m increases, more generalizations are performed in order to eliminate subtrajectories with low support. This leads to fewer generalized locations with larger sizes. These results are shown in Figures 7b-8b.

Also, we evaluated the impact of dataset size on data utility, using various random subsets of the original dataset, containing $2,000$, $5,000$, $10,000$, and $15,000$ records. In Figure 8c, we illustrate the number of original locations for variable dataset sizes. For larger datasets, this number increases, as the support of single locations is higher. Consequently, the support of subtrajectories increases, and fewer locations are generalized. This leads to more generalized locations, with lower average size, and lower distance, as can be seen in Figures 9a-9c.



(a) (b) (c)

Figure 7: number of *(a)* average percent of distance in generalized locations, *(b)* original (non generalized) locations published and *(c)* generalized locations published



(a) (b) (c)

Figure 8: *(a)* average generalized location size, *(b)* average percent of distances in generalized locations and *(c)* number of original (non generalized) locations published

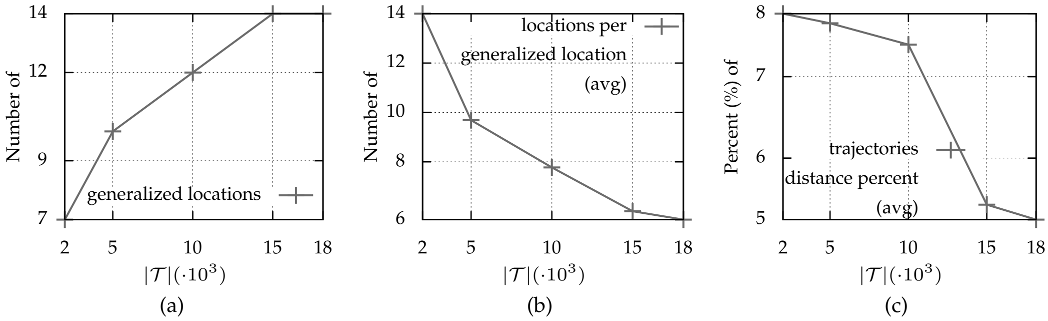


Figure 9: *(a)* number of generalized locations published, *(b)* average generalized location size and *(c)* average percent of distances in generalized locations

SEQANON vs. NGRAMS. In this section, we report the count of the subtrajectories of different sizes that are created by the NGRAMS method. In addition, we report the ARE and KL-divergence scores for the SEQANON and NGRAMS methods. Finally, we report the percentage of the frequent sequential patterns of the original dataset that are preserved in the anonymous dataset and the percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original dataset. The results of comparing NGRAMS to SD-SEQANON and to U-SEQANON were quantitatively similar (omitted for brevity).

Fig. 10a reports the number of subtrajectories of different sizes that are contained in the Oldenburg dataset, denoted with $\mathcal{T}$, and the output of NGRAMS, denoted with $\mathcal{T}'_{\text{SEQANON}}$. As can be seen, the output of NGRAMS contains noisy versions of only a small percentage (0.29%) of short subtrajectories. Thus, the information of the subtrajectories of length greater than 4, which correspond to 72.62% of the subtrajectories in the dataset, is lost. The results of the same experiment on the Gowalla dataset, reported in Fig. 10b, are quantitatively similar. That is, NGRAMS created noisy versions of 0.11% of the subtrajectories with 3 or fewer locations, and the information of all longer subtrajectories, which correspond to 98.1% of the subtrajectories in the dataset, is lost. On the other hand, SEQANON employs generalization, which preserves the information of all subtrajectories, although in a more aggregate form.

Figure 11a reports the ARE scores for SEQANON and NGRAMS, as a function of k, for the Oldenburg dataset and for 100 queries involving subtrajectories of sizes in $[1, 2]$. In this experiment, we set m to 3, assuming that an attacker knows about 75% of the locations visited by a user. As can be seen, the ARE scores for SEQANON are at least 4.45 and up to 7.3 times lower (better) than those of NGRAMS. Furthermore, the ARE scores for our method increase with k, which is expected due to the utility/privacy trade-off. On the contrary, the ARE scores for NGRAMS remained the same, as this method does not use the parameter k. Next, we studied the impact of m on ARE, by varying this parameter in $[1, 4]$ (recall that $m = 4$ implies that the attacker knows almost all of the locations, visited by a user). Figure 11b reports the result for $k = 5$, on the Oldenburg dataset. The ARE scores for our algorithm were at least 6.3 and up to 11.9 times better than those of NGRAMS. Also, it can be seen that the ARE scores for SEQANON increase with m, as the algorithm has to incur more generalization to protect from stronger attackers. The ARE scores for NGRAMS are not affected by m, as this method does not use this parameter. We also studied the impact of k and m on ARE, using the Gowalla dataset, and obtained similar results, which

size	# in $\mathcal{T}$	# in $\mathcal{T}'_{\text{NGRAMS}}$
1	100	73
2	6955	220
3	48268	222
4	124070	2
5	177054	–
6	158684	–
7	93479	–
8	36328	–
9	8989	–
≥ 10	1366	–

(a)

size	# in $\mathcal{T}$	# in $\mathcal{T}'_{\text{NGRAMS}}$
1	662	427
2	107811	577
3	803093	6
4	1757607	–
5	2959148	–
6	4478016	–
7	6033559	–
8	7138181	–
9	7336036	–
≥ 10	17281056	–

(b)

Figure 10: Number of distinct subtrajectories of different sizes, for *(a)* the Oldenburg, and *(b)* the Gowalla dataset.

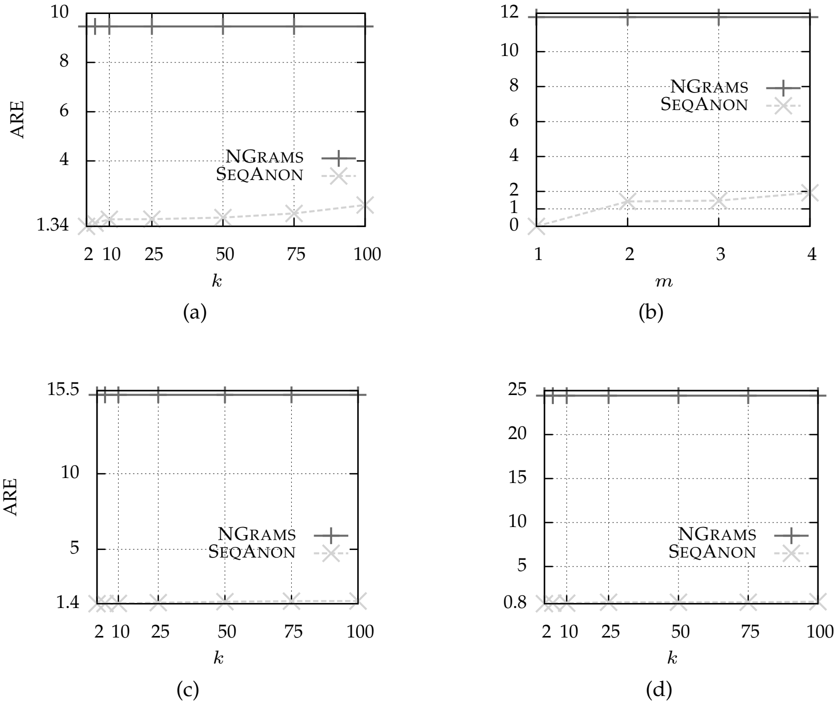
are reported in Figures 11c and 11d.

The results with respect to KL-divergence, as a function of m, are shown in Figure 14. Specifically, Figure 14a reports the result for the set S of all subtrajectories with size 1 (i.e., all locations) in the Oldenburg dataset, and for $k = 5$. As can be seen, the information loss for SEQANON was significantly lower than that of NGRAMS, particularly for smaller values of m. Increasing m led to fewer, larger generalized locations. Thus, the KL-divergence scores of SEQANON increase with m, while those of NGRAMS are not affected by m, for the reason explained before. Figure 14b (respectively, Figure 14c) reports the KL-divergence scores for 100 randomly selected locations (respectively, subtrajectories with size 2) in the Gowalla dataset. As noted previously, we did not consider longer subtrajectories, as all but 6 of the subtrajectories in the output of NGRAMS have size at most 2. Again, SEQANON outperformed NGRAMS by a large margin, which demonstrates that our method can preserve the distribution of the support of subtrajectories better. Specifically, the KL-divergence scores for our method were at least 20% and up to 4.3 times lower (better) than those for NGRAMS. Similar results were obtained for larger k values (omitted for brevity).

Next, we present the results of experiments, in which we evaluated the ability of the algorithms to support frequent sequential pattern mining. Specifically, we report the percentage of the frequent sequential patterns of the original dataset that are preserved in the anonymous dataset and the percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original dataset. In order to get a more accurate statistical distribution we used 2000 randomly projected anonymous datasets[3]. In our experiments, we mined the Oldenburg and Gowalla dataset, using a support threshold of 0.83% and 0.14%, respectively.

Figures 12a and 12b present the median and standard deviation of the percentage of frequent sequential patterns that were preserved in the anonymous dataset, when SEQANON was applied with a k in $[2, 100]$ and NGRAMS with the default parameters, on Oldenburg and Gowalla datasets respectively. The results for SEQANON are significantly better than those of NGRAMS. Specifically, SEQANON reported at least 48% and up to 192% more frequent patterns than NGRAMS. Note also that SEQANON performs better when k is smaller, because it applies a lower amount of generalization.

[3] *Creating more projected datasets allows estimating the dataset quality more accurately. However, the increase in the accuracy of estimation was negligible, when we used more than* 2000 *projected datasets, in our experiments.*



(a)

(b)

(c)

(d)

Figure 11: ARE for queries involving 100 randomly selected subtrajectories *(a)* with size in $[1, 2]$, in the Oldenburg dataset (varying k), *(b)* with size in $[1, 2]$, in the Oldenburg dataset (varying m), *(c)* with size 1, in the Gowalla dataset (varying k), and *(d)* with size 2 in the Gowalla dataset (varying k).

Figures 12c and 12d report the median and standard deviation of the percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original dataset, when SEQANON was applied with a k in $[2, 100]$ and NGRAMS with the default parameters, on Oldenburg and Gowalla datasets respectively. Again, the results for SEQANON are better than those of NGRAMS. In more detail, the percentage of patterns that are not frequent in the original dataset but are frequent in the anonymized dataset was up to 1.2 times lower (on average 45% lower) for SEQANON compared to NGRAMS. Note that as k increases, the percentage of such patterns for SEQANON decreases, because fewer patterns are frequent in the anonymized dataset.

Figures 13a and 13b report the percentage of frequent sequential patterns preserved in anonymous dataset and the percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original dataset. We set $k = 5$ and vary m in $[1, 4]$ for SEQANON, while NGRAMS was executed with the default parameters. Larger values of m result in more generalization. Thus, SEQANON preserves at least 20% and up to 650% more frequent patterns frequent than NGRAMS, while the percentage of patterns that are incorrectly identified as frequent is lower by at least 50% and up to 500% compared to that for NGRAMS. The corresponding results for Gowalla were qualitatively similar (omitted for brevity).

In summary, our results show that the SEQANON algorithm permits more effective query answering, more accurate pattern mining, and incurs lower information loss than NGRAMS. Thus, it offers a different trade-off between utility and privacy, which is important in ap-

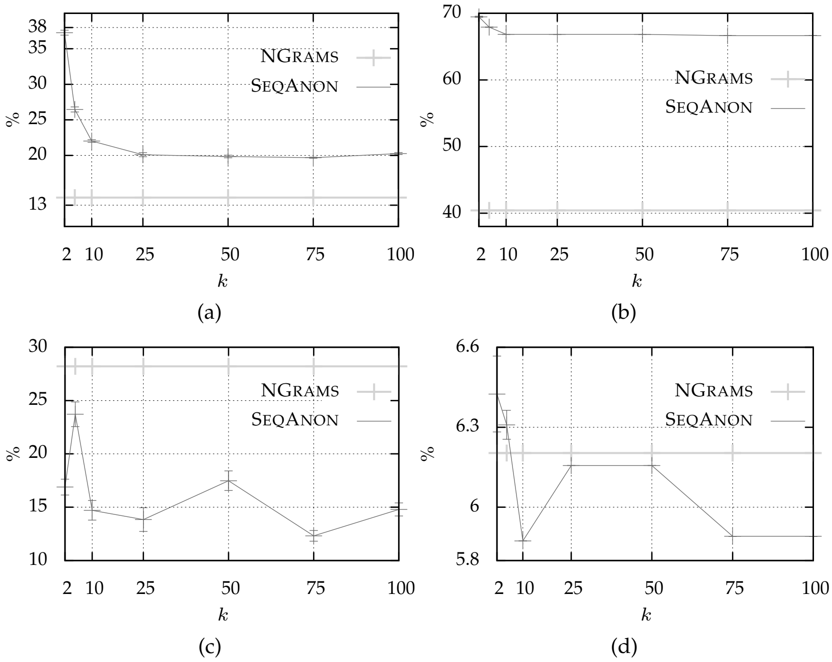


Figure 12: Percentage of frequent sequential patterns preserved in anonymous dataset (median) for varying k on *(a)* Oldenburg dataset, *(b)* Gowalla dataset, and percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original (median) for varying k on *(c)* Oldenburg dataset and *(d)* Gowalla dataset.

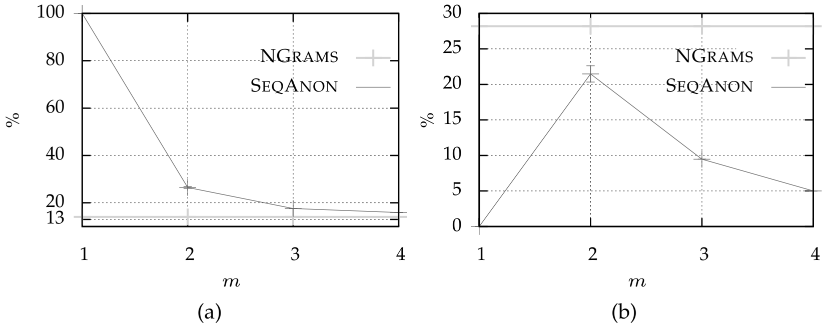


Figure 13: *(a)* Percentage of frequent sequential patterns preserved in anonymous dataset (median) and *(b)* percentage of frequent sequential patterns of the anonymous dataset that are not frequent on the original (median) on Oldenburg dataset for varying m.

plications that require preserving data truthfulness.

SD-SEQANON. In this section, we evaluate the data utility offered by SD-SEQANON, using statistics computed for the output of this algorithm. Figure 15a (respectively, 15b) reports the average distance of all locations that are mapped to each generalized location, as a function of k, for the Oldenburg (respectively, the Gowalla) dataset. Increasing k leads SD-SEQANON to create more, larger generalized locations, which results in larger average location distance. Note that the results for SD-SEQANON are slightly worse than those of SEQANON and may also decrease as k gets larger. This is expected because SD-

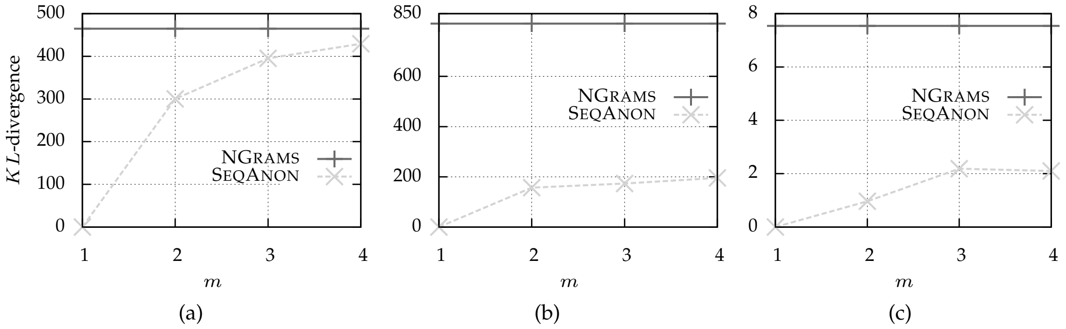


Figure 14: KL-divergence for the distribution of the support of *(a)* all locations in the Oldenburg dataset, *(b)* 100 randomly selected locations in the Gowalla dataset, and *(c)* 100 randomly selected subtrajectories of size 2 in the Gowalla dataset.

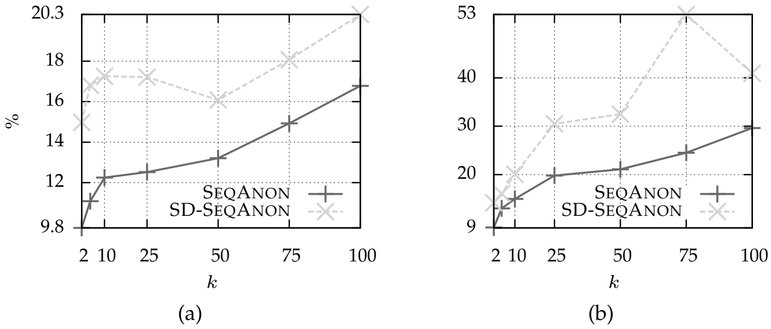


Figure 15: Average percent of distance in generalized locations for *(a)* Oldenburg and *(b)* Gowalla dataset.

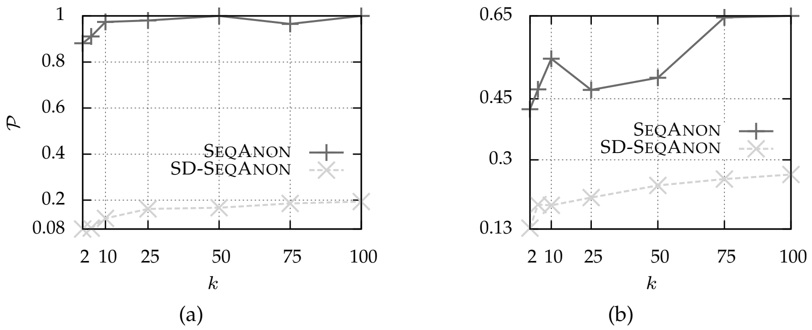
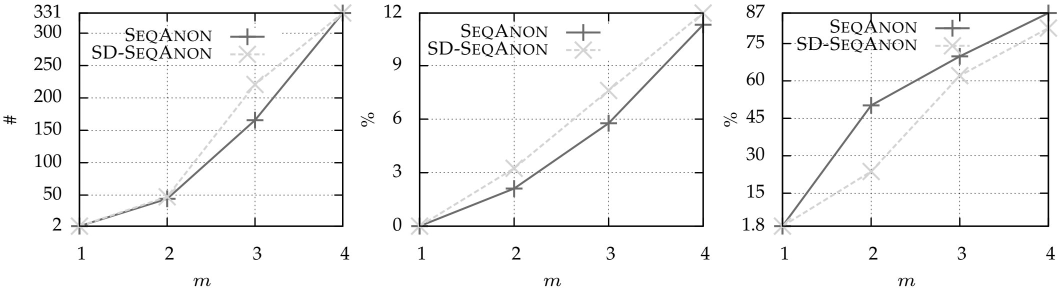


Figure 16: Semantic similarity penalty $\mathcal{P}(\mathcal{T}')$ for *(a)* Oldenburg and *(b)* Gowalla dataset.

SeqAnon takes into account not only the distance but also the semantic similarity of locations, when performing generalization. Thus, as can be seen in Figures 16a and 16b, the SD-SeqAnon algorithm performs much better than SeqAnon with respect to the *semantic similarity penalty* (the scores for SD-SeqAnon are at least 2.5 and up to 4.6 times better than those for SeqAnon). This demonstrates that SD-SeqAnon generalizes more semantically close locations together.

Figure 17 presents the average size of generalized locations, the average percent of distance in generalized locations, and the semantic similarity penalty $\mathcal{P}$, as a function of m. In these experiments k was set to 50. As can be seen, increasing m leads the algorithms to construct fewer, larger generalized locations, which are comprised of more distant and less semantically close locations. As expected, SeqAnon generalized together locations that are closer in proximity (see Figure 17b) but more semantically distant (see Figure 17c).



(a) (b) (c)

Figure 17: *(a)* average generalized location size, *(b)* average percent of distance in generalized locations and *(c)* average percent of similarity in generalized locations for $k = 50$ (Gowalla dataset).

U-SeqAnon. This section reports results for the U-SeqAnon algorithm, which was configured with three different utility constraint sets, namely $\mathcal{U}_1$, $\mathcal{U}_2$ and $\mathcal{U}_3$, and a suppression threshold $\delta = 10$. The utility constraints in each of these sets contain a certain number of semantically close locations (i.e., sibling nodes in the location taxonomy), as shown in Figure 18. Note that $\mathcal{U}_3$ is more difficult to satisfy than $\mathcal{U}_1$, as the number of allowable ways to generalize locations is smaller.

$\mathcal{U}_1$	$\mathcal{U}_2$	$\mathcal{U}_3$
$\lvert u_1\rvert = 50$	$\lvert u_1\rvert = 25$	$\lvert u_1\rvert = 20$
$\lvert u_2\rvert = 50$	$\lvert u_2\rvert = 25$	$\lvert u_2\rvert = 14$
	$\lvert u_3\rvert = 25$	$\lvert u_3\rvert = 16$
	$\lvert u_4\rvert = 25$	$\lvert u_4\rvert = 14$
		$\lvert u_5\rvert = 18$
		$\lvert u_6\rvert = 18$

Figure 18: The size of the utility constraints in each utility constraint set $\mathcal{U}_1$, $\mathcal{U}_2$, and $\mathcal{U}_3$.

Figure 19a reports the average size of generalized locations, for various values of k in $[2, 100]$. Note that all configurations of U-SeqAnon created smaller generalized locations than those constructed by SeqAnon, and the smallest generalized locations were created when $\mathcal{U}_3$ was used. This is because the presence of utility constraints that contain a small number of locations reduces the number of available generalizations. For this reason, all configurations of U-SeqAnon generalized together more distant locations than SeqAnon, as can be seen in Figure 19b. Also, observe that the use of less restrictive utility constraints (e.g., those in $\mathcal{U}_1$) leads U-SeqAnon to generalize together locations that are close in proximity.

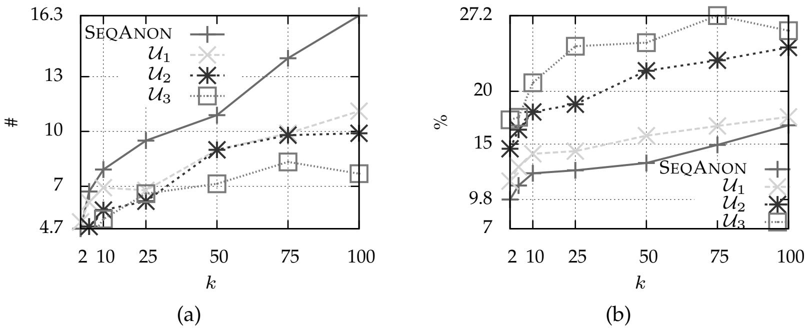


Figure 19: *(a)* average generalized location size and *(b)* average percent of distance in generalized locations, for the Oldenburg dataset.

5.2 Efficiency

In this section, we evaluate the impact of the anonymization parameters k and m, the dataset size, and the location size, on the efficiency of our approach.

SEQANON. To highlight the benefit of employing the apriori principle on efficiency, we created a version of SEQANON, called SEQANON_F, which does not use the apriori principle. In this version, we removed the **for** loop from Step 2 of SEQANON and set $i = m$. In other words, SEQANON_F tries to deal directly with subtrajectories of size m. First, we studied the impact of the anonymization parameter k on efficiency. As illustrated in Figure 20a, the execution time of both algorithms increases with k. This is expected because there are more subtrajectories with a lower support than k, when k is larger. However, SEQANON is significantly more efficient and scalable than SEQANON_F. Specifically, the SEQANON algorithm was at least 6.5 and up to 10.85 times more efficient than SEQANON_F. Then, we studied the impact of m on efficiency and report the results in Figure 20b. As can be seen, the use of the apriori principle by SEQANON enables it to scale much better than SEQANON_F, as m gets larger. In addition, we studied the effect of dataset size on the execution time of SEQANON. The results in Figure 20c demonstrate that the SEQANON outperforms SEQANON_F, being up to 20 times more efficient. We then studied the impact of m, dataset size, and number of locations on the larger Gowalla dataset, and report the results in Figure 21. Due to the fact that this dataset is more sparse than the Oldenburg dataset and contains a larger number of distinct locations, SEQANON needed more time to anonymize it. Again, SEQANON was more efficient than SEQANON_F, which needed more than 12 hours to anonymize the entire dataset. Of note, SEQANON is less efficient than NGRAMS, mainly because generalization requires accessing all trajectories in the dataset more times.

SD-SEQANON and U-SEQANON. In this section, we evaluate the impact of k on the efficiency of SD-SEQANON and U-SEQANON. In this set of experiments, we set $m = 2$ and configured U-SEQANON using the utility constraint sets $\mathcal{U}_1$, $\mathcal{U}_2$, and $\mathcal{U}_3$, and $\delta = 10$. Figure 22a reports the runtime of SD-SEQANON, for varying k, and for the Oldenburg dataset. As can be seen, the runtime of SD-SEQANON is similar to that of SEQANON. The small differences between these algorithms are attributed to the fact that they use different simi-

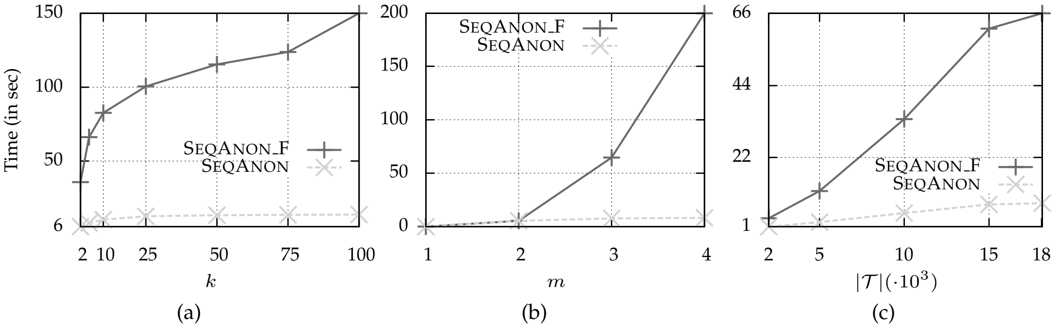


Figure 20: Runtime of SEQANON and SEQANON_F for the Oldenburg dataset and *(a)* varying k, *(b)* varying m, and *(c)* varying dataset size.

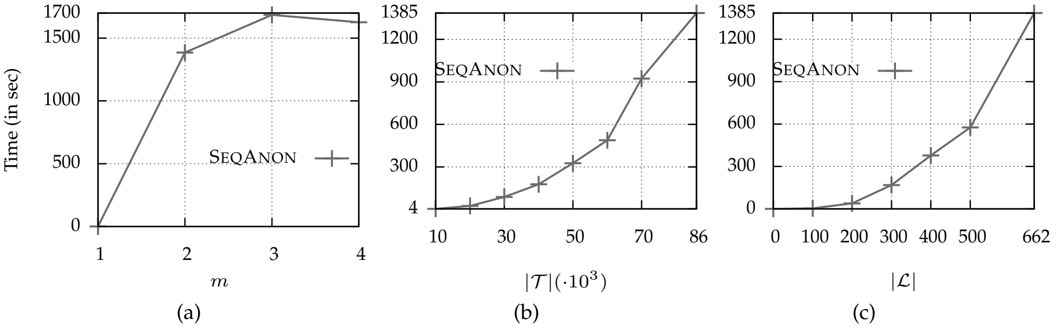


Figure 21: Runtime of SEQANON for the Gowalla dataset and *(a)* varying m, *(b)* varying dataset size, and *(c)* varying number of locations.

larity measures. Last, we report the runtime of the U-SEQANON algorithm in Figure 22b. As expected, the use of utility constraints incurs some overhead, but it does not affect the scalability of the algorithm.

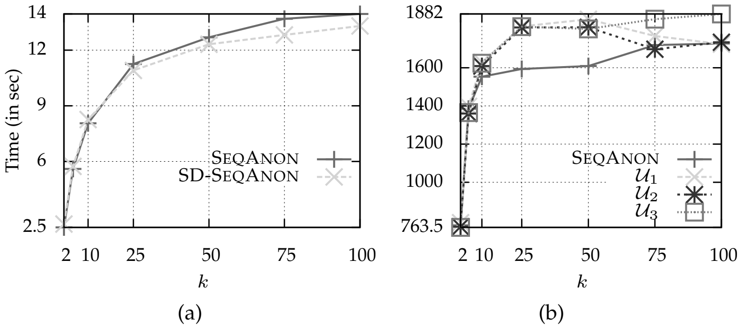


Figure 22: Runtime for varying k and for *(a)* SEQANON and SD-SEQANON (Oldenburg dataset), *(b)* SEQANON and U-SEQANON (Gowalla dataset).

6 Conclusions

In this paper, we proposed a new approach to publishing trajectory data in a way that prevents identity disclosure. Our approach makes realistic privacy assumptions, as it adapts k^m-anonymity to trajectory data, and allows the production of truthful data that preserve important data utility characteristics. To realize our approach, we developed three anonymization algorithms that employ the apriori principle. These algorithms aim at preserving different data characteristics, including location distance and semantic similarity, as well as user-specified utility requirements. The efficiency and effectiveness of these algorithms was demonstrated through extensive experiments.

Acknowledgements

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

References

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376–385, 2008.

[2] O. Abul, F. Bonchi, and M. Nanni. Anonymization of moving objects databases by clustering and perturbation. *Information Systems*, 35(8):884–910, 2010.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 3–14, 1995.

[4] F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Trajectory anonymity in publishing personal mobility data. *Special Interest Group on Knowledge Discovery and Data Mining Explorations*, 13(1):30–42, 2011.

[5] L. Bonomi and L. Xiong. A two-phase algorithm for mining sequential patterns with differential privacy. In *CIKM*, pages 269–278, 2013.

[6] T. Brinkhoff. A framework for generating network-based moving objects. *GeoInformatica*, 6(2):153–180, 2002.

[7] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 638–649, New York, NY, USA, 2012. ACM.

[8] R. Chen, B. Fung, N. Mohammed, B. Desai, and K. Wang. Privacy-preserving trajectory data publishing by local suppression. *Inf. Sci.*, 231:83–97, 2013.

[9] R. Chen, B. C. M. Fung, and B. C. Desai. Differentially private trajectory data publication. *Computing Research Repository*, abs/1112.2020, 2011.

[10] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *ACM SIGKDD*, KDD '11, pages 1082–1090. ACM, 2011.

[11] J. Domingo-Ferrer and R. Trujillo-Rasua. Microaggregation- and permutation-based anonymization of movement data. *Journal of Information Sciences*, 208:55–80, Nov. 2012.

[12] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

[13] A. Freitas. A critical review of multi-objective optimization in data mining: a position paper. *ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations*, 6(2):77–86, 2004.

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

[15] B. C. M. Fung, K. Wang, and P. Yu. Top-down specialization for information and privacy preservation. In *ICDE*, pages 205–216, 2005.

[16] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis. Fast data anonymization with low information loss. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 758–769, 2007.

[17] A. Gionis, H. Mannila, T. Mielikäinen, and P. Tsaparas. Assessing data mining results via swap randomization. *ACM Transactions on Knowledge Discovery Data*, 1(3), Dec. 2007.

[18] A. Gkoulalas-Divanis and G. Loukides. PCTA: privacy-constrained clustering-based transaction data anonymization. In *PAIS*, pages 1–10, 2011.

[19] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, pages 217–228, 2006.

[20] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD*, pages 49–60, 2005.

[21] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE*, page 25, 2006.

[22] D. Lin, S. Gurung, W. Jiang, and A. Hurson. Privacy-preserving location publishing under road-network constraints. In *Proceedings of the 15th International Conference on Database Systems for Advanced Applications*, DASFAA '10, pages 17–31, 2010.

[23] G. Loukides and A. Gkoulalas-Divanis. Utility-preserving transaction data anonymization with low information loss. *Expert System with Applications*, 39(10):9764–9777, 2012.

[24] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. *Knowledge Information Systems*, 28(2):251–282, 2011.

[25] B. Malin. k-unlinkability: A privacy protection model for distributed data. *Data Knowl. Eng.*, 64(1):294–311, 2008.

[26] N. Mohammed, B. C. M. Fung, and M. Debbabi. Walking in the crowd: anonymizing trajectory data for pattern analysis. In *CIKM*, pages 1441–1444, 2009.

[27] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel. Movement data anonymity through generalization. *Transactions on Data Privacy*, 3(2):91–121, 2010.

[28] A. Monreale, R. Trasarti, D. Pedreschi, C. Renso, and V. Bogorny. C-safety: a framework for the anonymization of semantic trajectories. *Transactions on Data Privacy*, 4(2):73–101, 2011.

[29] M. E. Nergiz, M. Atzori, and Y. Saygin. Towards trajectory anonymization: a generalization-based approach. In *SPRINGL*, pages 52–61, 2008.

[30] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Mining sequential patterns by pattern-growth: the prefixspan approach. *Knowledge and Data Engineering, IEEE Transactions on*, 16(11):1424–1440, Nov 2004.

[31] G. Poulis, G. Loukides, A. Gkoulalas-Divanis, and S. Skiadopoulos. Anonymizing data with relational and transaction attributes. In *ECML/PKDD (3)*, pages 353–369, 2013.

[32] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[33] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *PODS*, pages 188–, 1998.

[34] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[35] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*,

pages 65–72, 2008.

[36] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *The International Journal on Very Large Data Bases*, 20(1):83–106, 2011.

[37] R. Trujillo-Rasua and J. Domingo-Ferrer. On the privacy offered by (k, δ)-anonymity. *Information Systems*, 38(4):491–494, June 2013.

[38] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu. Utility-based anonymization using local recoding. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 785–790, 2006.

[39] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.

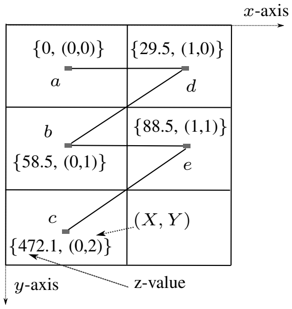
Select-Organize-Anonymize: A framework for trajectory data anonymization

Giorgos Poulis*, Spiros Skiadopoulos*, Grigorios Loukides†, Aris Gkoulalas-Divanis‡
* University of Peloponnese, Email: {poulis,spiros}@uop.gr
† Cardiff University, Email: g.loukides@cs.cf.ac.uk
‡ IBM Research - Ireland, Email: arisdiva@ie.ibm.com

***Abstract*—Advances in positioning technologies together with the wide adoption of GPS-enabled smartphones enable accurate and low-cost tracking of user location. This allows the collection of large amounts of person-specific mobility data that offer remarkable opportunities for data analysis. Yet, the sharing of such data poses significant privacy risks. This enunciates the need for privacy-preserving, trajectory data publishing methods. Existing approaches are either limited in their privacy specification component or they incur significant, and often unnecessary, data distortion. In response, we propose a novel framework for anonymizing trajectory data that prevents the disclosure of both identity and sensitive location information, while retaining data utility. Our framework involves: (i) selecting similar trajectories, by employing Z-ordering or data projections on frequent sub-trajectories, (ii) organizing the selected trajectories into carefully constructed clusters, and (ii) anonymizing each cluster separately. We develop algorithms to realize our framework, which are effective and efficient, as verified by extensive experiments.**

id	trajectory
t_1	(d, a, c, e)
t_2	(b, a, e, c)
t_3	(a, d, e, f)
t_4	(b, d, e, c)
t_5	(d, g, c)
t_6	(d, e)

(a)



(b)

Fig. 1: (a) The original database $\mathcal{T}$ (b) Z-ordering

I. Introduction

The enormous advances in positioning technologies, such as GPS, GSM, UMTS and RFID, along with the rapid developments in the wireless communications industry, have made possible the accurate tracking of user location, at a low cost [7]. This, together with the wide adoption of GPS-enabled smartphones, gave rise to novel applications, which are based on user location. At the same time, the mobility data that are collected from these applications provide a valuable resource for understanding human behavior, as well as for supporting various processes. The sharing, however, of person-specific movement data, for research or other purposes, poses significant challenges, as it may threaten individuals' privacy.

Specifically, the publishing of person-specific trajectories (i.e., sequences of locations visited by individuals) can lead to *identity disclosure*, even when they are devoid of identifying information [3], [21]. Identity disclosure attacks are possible, when an individual can be associated with a sequence of locations in the published data. Consider, for example, the dataset in Fig. 1a, where each trajectory t_i corresponds to an individual and contains an ordered list of locations visited by them. Observe that identity disclosure is possible, based on the sequence of locations d and a, because this sequence appears only in t_1. A sequence of locations that may lead to identity disclosure, is called *quasi-identifier* (QID) [21]. Existing approaches prevent identity disclosure by either anonymizing: (i) each trajectory as a whole (e.g., by producing cylindrical tubes that contain many trajectories [1]), or (ii) parts of trajectories, based on specific QIDs [21], [23]. The first category of approaches, termed *QID-blind*, employ *clustering and perturbation* [1], [16], while approaches that fall into the second category, termed *QID-aware*, use *generalization* and *suppression* [15], [21], [23].

QID-blind approaches are limited into their privacy specification component and may harm the utility of the published data, unnecessarily. This is because: (i) they assume that an attacker may know all locations visited by an individual, which is extremely difficult, due to the high-dimensionality and sparsity of trajectory data [21], (ii) clustering-based methods may lose information about the direction of movement of co-clustered trajectories, and (iii) perturbation-based approaches may generate false associations, harming data *truthfulness*. On the other hand, QID-aware approaches assume that data owners possess (i) detailed knowledge of QIDs (e.g., an attacker knows a certain sequence of locations about an individual [21]), which is unlikely in the context of trajectory data publishing [3], and (ii) taxonomies that organize all locations in terms of semantic similarity, which is restrictive for real-world trajectory data publishing applications [18].

Adapting k^m-anonymity [22] to trajectory data and employing a *distance-based* generalization model has been proposed recently [18] to address some of these limitations. However, the approach of [18] may risk the disclosure of *sensitive* location information (e.g., the fact that an individual visited a psychiatric clinic) and incur excessive distortion, as it applies distance-based generalization to the entire dataset.

In this paper we propose a novel anonymization framework, which allows preventing the disclosure of both identity and

978-0-7695-5109-8/13 $31.00 © 2013 IEEE
DOI 10.1109/ICDMW.2013.136

sensitive location information, while publishing minimally distorted data. This is achieved by performing a series of operations: (i) *Select*, which identifies similar trajectories based on summary information about them, (ii) *Organize*, which sorts the selected trajectories with respect to similarity and groups them into clusters, and (iii) *Anonymize*, which constructs clusters that prevent both types of disclosure.

Our work makes the following specific contributions:

- We develop two algorithms, called ZGA and SGA, to realize our anonymization framework. ZGA performs *Select* by capturing location similarity, based on *Z-ordering* [20], whereas SGA by measuring trajectory similarity using projections of trajectories on *frequent subtrajectories* [2]. These algorithms subsequently organize similar trajectories, using *Gray order* [19], and form clusters which are anonymized independently.
- We design ℓ^m-ANON, an algorithm for anonymizing the clusters created by ZGA or SGA. This algorithm employs distance-based generalization to preserve privacy and works in an apriori-like fashion.
- We experimentally demonstrate that our approach is effective at preserving data utility and very efficient.

The rest of the paper is organized as follows. Section II formulates the problem and Section III presents our framework. Related work and experiments are presented in Sections IV and V, respectively. Section VI concludes the paper.

II. Problem formulation

Let $\mathcal{L}$ be a set of locations visited by individuals. Each location in $\mathcal{L}$ corresponds to a different region (cell) of a two-dimensional, uniform grid, and we may use the (X, Y)-coordinates of the cell to refer to its associated location. Some locations in $\mathcal{L}$ are *sensitive*, as they represent places that individuals are not willing to be associated with, such as an oncology clinic. Following [14], we assume that sensitive locations are specified by data owners.

A *trajectory* t is an ordered list of locations $(l_1, \ldots, l_n)$, where $l_i \in \mathcal{L}$, $1 \leq i \leq n$. The *size* of the trajectory $t = (l_1, \ldots, l_n)$, denoted by $|t|$, is the number of its locations, i.e., $|t| = n$. A trajectory s is a *subtrajectory of*, or is *contained in*, a trajectory $t = (l_1, \ldots, l_n)$, when: (i) $|s| \leq |t|$, and (ii) there is a mapping f such that $\lambda_1 = l_{f(1)}, \ldots, \lambda_\nu = l_{f(\nu)}$ and $f(1) < \cdots < f(\nu)$. Thus, s is formed by removing some locations from t, while maintaining the order of all other locations.

Given a set of trajectories $\mathcal{T}$, the *support* of a subtrajectory s, denoted by $\sup(s, \mathcal{T})$, is defined as the number of distinct trajectories in $\mathcal{T}$ that contain s. These trajectories are referred to as the *supporting* trajectories of s, and their set is denoted with $\mathcal{T}_s$. Given a set of trajectories $\mathcal{T}$ and a minimum support threshold $minSup$, specified by data owners, the set of *frequent* subtrajectories contains a subtrajectory s, if and only if $\sup(s, \mathcal{T}) \geq minSup$ [2].

Example 1: A trajectory dataset is shown in Fig. 1a and is comprised of trajectories t_1 to t_6 (id is shown for reference purposes only). Each trajectory contains some of the locations in the set $\mathcal{L} = \{a, b, ..., g\}$, and the locations f and g are sensitive. A subtrajectory of the trajectory $t_1 = (d, a, c, e)$ is $s = (a, e)$, which has a support of 3. Assuming that $minSup = 3$, the subtrajectories (d, c) and (d, e) are both *frequent*, unlike (d, c, e) and (d, e, c).

In the following, we define the concept of *trajectory key*, which is essentially a projection of the trajectory on the feature space of an ordered set of subtrajectories.

Definition 1 (Trajectory key): Given a trajectory t in $\mathcal{T}$ and an ordered set of subtrajectories $S = (s_1, \ldots, s_l)$, where $s_i \in \mathcal{T}$, $1 \leq i \leq l$, the key of t, denoted as $\mathcal{K}_t^S$ is a set of size $|S|$, whose i-th element is 1, if s_i is contained in t, and 0 if not.

Example 2: Consider the trajectory t_1 in Fig. 1a and assume the set $\big((d, e), e, d\big)$, which contains the three most frequent subtrajectories in $\mathcal{T}$ ordered in decreasing support; the key for t_1 is (111), as all three of them are subtrajectories of t_1.

As mentioned before, a location in $\mathcal{L}$ is associated with a distinct pair of (X, Y)-coordinates. To measure similarity between locations, we use the *Z-order* (or *Morton order*), a function that maps locations to numbers, called *z-values*. The z-value for a location is calculated by interleaving the binary representations of the location's (X, Y) coordinates [20], and the mapping preserves the locality of locations. That is, locations with "similar" coordinates are mapped to numbers whose difference is "small". Based on this function, we obtain the Z-ordering of locations, as explained below.

Definition 2 (Z-ordering of locations): Given a set of locations in $\mathcal{L}$ and their corresponding z-values, Z-ordering is defined as the ordering obtained by sorting the locations in ascending order of their z-values.

The Z-ordering of all nonsensitive locations in $\mathcal{T}$, is shown in Fig. 1b. To publish $\mathcal{T}$ in a way that prevents both the disclosure of individuals' identity and sensitive location information, we use the k^m-anonymity and l^m-diversity privacy principles, which were originally developed for transaction data [22]. The following definition explains how these principles can be adapted to trajectory data and combined together, in order to obtain the principle of $(k, \ell)^m$-anonymity.

Definition 3 ($(k, \ell)^m$-anonymity): Given parameters k and ℓ, which are specified by data owners, a trajectory dataset $\mathcal{T}$ satisfies $(k, \ell)^m$-anonymity, if and only if: (i) $\sup(s, \mathcal{T}) \geq k$, for any subtrajectory s, comprised of m nonsensitive locations in $\mathcal{L}$, and (ii) $\sup(s', \mathcal{T}_s) \leq \ell$, where s' is any subtrajectory of sensitive locations that are contained in $\mathcal{T}_s$.

Example 3: The dataset in Fig. 1a is $(2, 2)^1$-anonymous, because each nonsensitive location a to e has support of at least 2, and none of the sensitive locations f and g co-occurs with all the supporting trajectories of a or e. However, this dataset is not $(2, 2)^2$-anonymous, as the subtrajectory (a, d) of t_3 co-occurs with f, in no other trajectory.

$(k, \ell)^m$-anonymity guarantees that an attacker, who knows any subtrajectory s of m nonsensitive locations about an individual, cannot link the individual to fewer than k trajectories in the published dataset, nor can associate the individual with a sensitive location with a probability that exceeds $1/\ell$.

To enforce $(k,\ell)^m$-anonymity, we employ the distance-based generalization model [18], which replaces nonsensitive locations with *generalized locations*. A *generalized location* $\{l_1,\ldots,l_v\}$, is a set of at least two nonsensitive locations $l_1,\ldots,l_v$ in $\mathcal{L}$, which is interpreted as any of the locations $l_1,...,l_v$. Sensitive locations are not generalized, because they typically need to be retained intact in applications. Thus, given a trajectory t in $\mathcal{T}$, we may replace a nonsensitive location l_i in t, where $1 \leq i \leq u$, with a generalized location $\{l_1,...,l_v\}$.

Clearly, generalization must avoid unnecessarily distorting the original dataset $\mathcal{T}$, thus we need to quantify distortion. To achieve this, we adopt the *location*, *trajectory*, and *trajectory dataset* distance measures [18], which are applicable to distance-based generalization and are defined as follows.

Definition 4 (Distance): Given a nonsensitive location l that is generalized to $\{l_1,\ldots,l_v\}$, the *location distance* between l and $\{l_1,\ldots,l_v\}$, is defined as:

$$\mathcal{D}_{loc}(l,\{l_1,\ldots,l_v\}) = \text{avg}\{\mathit{EuclDist}(l,l_i) \mid 1 \leq i \leq v\}$$

where *EuclDist* is the Euclidean distance. The *trajectory distance* between $t=(l_1,\ldots,l_n)$ and its generalized counterpart $t'=(l'_1,\ldots,l'_n)$ is defined as:

$$\mathcal{D}_{traj}(t,t') = \text{avg}\{\mathcal{D}_{loc}(l_i,l'_i) \mid 1 \leq i \leq n\}$$

The *trajectory dataset distance* between $\mathcal{T}=\{t_1,\ldots,t_u\}$ and its generalized counterpart $\mathcal{T}'=\{t'_1,\ldots,t'_u\}$, where the trajectory t_i is generalized to trajectory t'_i, $1 \leq i \leq u$, is defined as:

$$\mathcal{D}(\mathcal{T},\mathcal{T}') = \text{avg}\{\mathcal{D}_{traj}(t_i,t'_i) \mid 1 \leq i \leq u\}$$

These measures quantify distortion, based on the distance between the original and generalized data and apply average to combine specific distances, as illustrated below. Normalizing these measures is possible by dividing each of them with the maximum distance between locations in $\mathcal{T}$.

Example 4: Consider trajectory t_1 of Fig. 1a and let $\mathit{EuclDist}(a,b)=1$, $\mathit{EuclDist}(a,c)=1$ and $\mathit{EuclDist}(b,c)=2$. If locations a and c of t_1 are generalized to location $\{a,b,c\}$, then the location distances are $\mathcal{D}_{loc}(a,\{a,b,c\})=(0+1+1)/3=2/3$ and $\mathcal{D}_{loc}(c,\{a,b,c\})=1$. Also, if trajectory $t_1=(d,a,c,e)$ is generalized to $(d,\{a,b,c\},\{a,b,c\},e)$ the trajectory distance $\mathcal{D}_{t_1}=(0+2/3+1+0)/4 \approx 0.42$.

The problem we consider is formulated as follows.

Problem: Given a trajectory dataset $\mathcal{T}$, and parameters k, ℓ, and m, construct a $(k,\ell)^m$-anonymous version $\mathcal{T}'$ of $\mathcal{T}$, such that $\mathcal{D}(\mathcal{T},\mathcal{T}')$ is minimized.

III. SELECT-ORGANIZE-ANONYMIZE FRAMEWORK

This section presents our framework for enforcing $(k,\ell)^m$-anonymity to a trajectory dataset $\mathcal{T}$. Our framework performs the following operations:

1) *Select*: This operation selects similar trajectories, based on different properties of their keys.
2) *Organize*: In this operation, the selected trajectories are sorted, based on their *Gray order* [19], and grouped into carefully constructed clusters.
3) *Anonymize*: This operation constructs a minimally distorted $(k,\ell)^m$-anonymous dataset, by anonymizing the trajectories of each cluster separately.

To realize this framework, we develop two algorithms, henceforth called ZGA (for Z-ordering, Gray-code ordering, and Anonymize) and SGA (for Subtrajectory selection, Gray-code ordering, and Anonymize). As is evident from their names, these algorithms perform *Select*, based on different notions of trajectory key similarity. In addition, we introduce ℓ^m-ANON, an algorithm that enforces $(k,\ell)^m$-anonymity, using distance-based generalization. In what follows, we present the details of the ZGA, SGA, and ℓ^m-ANON algorithms.

ZGA algorithm. This algorithm performs the *Select* operation, based on the similarity of nonsensitive locations. To achieve this, it constructs the Z-ordering of the locations, which preserves the locality of locations, as discussed in Section II. Then, ZGA performs *Organization* by: (i) creating a key, for each trajectory in the dataset, based on Z-ordering, (ii) sorting trajectories, based on the Gray order of their keys, and (iii) formulating clusters, based on the latter sorting order. Thus, trajectories that contain many similar, nonsensitive locations are organized together.

Consider, for instance, three trajectories with keys $t=(001)$, $t'=(011)$, and $t''=(101)$, which are sorted in Gray order. Note that t is more similar to t' than to t'', because: (i) t and t' differ by only one bit (the 2nd bit from left to right), and (ii) the bit in which they differ is adjacent to the bit they share (the 3rd bit from left to right). Thus, ZGA organizes trajectories that share many neighboring locations, which can be subsequently generalized with low information loss. Due to its efficient computation and effectiveness, Gray order has been employed as an alternative to more computationally demanding clustering algorithms (e.g., in [10], [22]). However, the way that Gray order is combined with Z-ordering is new, to the best of our knowledge. Subsequently, the SGA algorithm performs the *Anonymize* operation, by applying ℓ^m-ANON to each cluster separately.

Algorithm ZGA takes as input a trajectory dataset $\mathcal{T}$, as well as parameters k, ℓ, m, and $\mathcal{C}$, and it constructs the $(k,\ell)^m$-anonymous counterpart $\mathcal{T}'$ of $\mathcal{T}$. The parameter $\mathcal{C}$ specifies the number of clusters, which will be created in the *Organization* operation, and is set by data owners. Automated specification of $\mathcal{C}$ is possible [13] but left as future work. After initialization, ZGA constructs the Z-ordering of all nonsensitive locations in $\mathcal{L}$ and stores it in a set S (Steps 1-3). Next, the algorithm stores each trajectory $t \in \mathcal{T}$ and its key $\mathcal{K}_t^S$ in a 2D array D, which is then sorted according to the Gray order of the stored trajectory keys (Steps 4-7). After that, ZGA assigns the ordered trajectories of D into clusters (Step 8). Each cluster is then anonymized using ℓ^m-ANON (to be discussed later) and added into $\mathcal{T}'$ (Steps 9-9). Last, $\mathcal{T}'$ is returned (Step 12).

Example 5: Consider applying ZGA to the dataset in Fig. 1a,

Algorithm: ZGA

```
Input: A dataset $\mathcal{T}$, parameters $k$, $\ell$, $m$, and $\mathcal{C}$
Output: A $(k,\ell)^m$-anonymous dataset $\mathcal{T}'$
1  $\mathcal{T}' := \varnothing$ // Initialize output
2  $D := S := \varnothing$ // Initialize $D$ and $S$
3  $S :=$ Z-ordering of nonsensitive locations in $\mathcal{L}$
4  for every trajectory $t \in \mathcal{T}$ do
5      $D(t).traj := t$
6      $D(t).key := \mathcal{K}_t^S$
7  sort $D$ according to $D.key$ in Gray order
8  $G :=$ set of clusters, each containing $|D|/\mathcal{C}$ consecutive
   trajectories from $D$
9  for every cluster $C \in G$ do
10     $\tilde{T} := \ell^m$-ANON$(C, k, \ell, m)$
11     $\mathcal{T}' := \mathcal{T}' \cup \tilde{T}$
12 return $\mathcal{T}'$
```

Algorithm: SGA

```
Input: A dataset $\mathcal{T}$, parameters $k$, $\ell$, $m$, $\mathcal{C}$, and $K$
Output: A $(k,\ell)^m$-anonymous dataset $\mathcal{T}'$
1  $\mathcal{T}' := \varnothing$ // Initialize output
2  $D := \varnothing$ // Initialize $D$
3  $S$:= the set of top $K$ frequent subtrajectories, comprised of
   nonsensitive locations, in descending order of support
4  for every trajectory $t \in \mathcal{T}$ do
5      $D(t).traj := t$
6      $D(t).key := \mathcal{K}_t^S$
7  sort $D$ according to $D.key$ in Gray order
8  $G$:= set of $\mathcal{C}$ clusters, each containing $|D|/\mathcal{C}$ consecutive
   trajectories from $D$
9  for every cluster $C \in G$ do
10     $\tilde{T} := \ell^m$-ANON$(c, k, \ell, m)$
11     $\mathcal{T}' := \mathcal{T}' \cup \tilde{T}$
12 return $\mathcal{T}'$
```

when all parameters are set to 2. The algorithm constructs the Z-ordering $\{a, b, d, e, c\}$ by sorting these locations in ascending order of their z-values, and populates the 2D-array D with all trajectories and their keys. Then, ZGA sorts D based on the Gray order of trajectory keys (Step 7). The unsorted and sorted keys of these trajectories are shown in Figs. 2a and 2b, respectively. As $\mathcal{C} = 2$, two clusters containing $\{t_6, t_3, t_1\}$ and $\{t_4, t_2, t_5\}$ are created and anonymized separately by ℓ^m-ANON (Steps 9-9). Last, the dataset $\mathcal{T}'$ in Fig. 2c is returned.

SGA algorithm. This algorithm performs the *Select* operation of our framework, based on the notion of frequent subtrajectories, and the *Organization* operation, by creating trajectory keys from the selected subtrajectories. That is, trajectories are projected on distinct sets of nonsensitive locations, which are visited by most individuals in a specific order, and trajectories with "similar" projections are brought together. In this way, ZGA organizes trajectories that share many frequent subtrajectories, which can be subsequently anonymized with low information loss. To see this, observe that a subtrajectory of m nonsensitive locations and a support of at least k, is k^m-anonymous. Then, the sorting, clustering, and anonymization of trajectories are performed, as in the ZGA algorithm.

Algorithm: ℓ^m-ANON

```
Input: A cluster $C$, parameters $k$, $\ell$, and $m$
Output: A $(k,\ell)^m$-anonymous set $C'$
1  $C' := C$ // Initialize output
2  for $i := 1$ to $m$ do
3      $\mathcal{S} := \varnothing$ // Initialize $\mathcal{S}$
4      for every subtrajectory $s \in C'$ with size $i$ do
5          compute sup$(s', C'_s)$
6          if sup$(s, C') < k$ or sup$(s', C'_s) > \ell$ then
7              $\mathcal{S} := \mathcal{S} \cup s$ // Insert $s$ to $\mathcal{S}$
8      sort $\mathcal{S}$ in increasing order of sup$(s, C')$
9      for every subtrajectory $s \in \mathcal{S}$ do
10         while sup$(s, C') < k$ or sup$(s', C'_s) > \ell$ do
11             find the location $l_1$ in $s$ with the minimum support
               in $C'$
12             find the location $l_2 \in C'$ such that $l_2 \neq l_1$ and
               $\mathcal{D}_{loc}(l_1, l_2)$ is minimum
13             replace each occurrenc of $l_1$ and $l_2$ in $C'$ with the
               generalized location $\{l_1, l_2\}$
14 return $C'$
```

The SGA algorithm takes as input a trajectory dataset $\mathcal{T}$, as well as parameters k, ℓ, m, $\mathcal{C}$, and K, and it returns $\mathcal{T}'$, the $(k,\ell)^m$-anonymous counterpart of $\mathcal{T}$. The parameter K controls the number of subtrajectories, contained in trajectory keys. Specifically, SGA creates a key with the top K frequent subtrajectories in $\mathcal{T}$ (i.e., those with the largest support). These subtrajectories are comprised of nonsensitive locations only. K is set by data-owners and its impact will be assessed in Section V. After initialization, SGA finds the top K frequent subtrajectories, using the method in [17], which is selected due to its efficiency (Steps 1-3). Then, in Steps 7-9, SGA performs sorting, clustering, and anonymization, and it returns the $(k,\ell)^m$-anonymous dataset $\mathcal{T}'$, in Step 12.

Example 6: Consider applying SGA to the dataset in Fig. 1a, when all parameters are set to 2. The algorithm finds the top 2 frequent subtrajectories, d and e, and stores them in S, in descending order of support (Step 3). Then, it constructs the 2D-array D, using all trajectories in $\mathcal{T}$ and their keys, which is sorted, as in Fig. 2b (Steps 4-7). Next, SGA creates the clusters $\{t_5, t_1, t_3\}$ and $\{t_4, t_6, t_2\}$, which are anonymized, and produces the dataset in Fig. 3d (Steps 8-12). This dataset differs from the output of ZGA (Fig. 2c), due to the different notion of trajectory similarity used by SGA.

ℓ^m-ANON algorithm. This algorithm is used by ZGA and SGA, to enforce $(k,\ell)^m$-anonymity to a cluster produced by these algorithms, with minimal distortion. Given a cluster C, and parameters k, ℓ and m, ℓ^m-ANON constructs the $(k,\ell)^m$-anonymous counterpart C' of C. To achieve this effectively and efficiently, it employs distance-based generalization [18] and the apriori principle [2]. That is, it first considers generalizing subtrajectories, containing one location, and then applies the same procedure to increasingly larger subtrajectories, as long as $(k,\ell)^m$-anonymity is not satisfied.

In more detail, ℓ^m-ANON initializes C' to the input cluster C and iterates over all subtrajectories of size i (Steps 1-4).

	location				
id	c	e	b	d	a
t_1	1	1	0	1	1
t_2	1	1	1	0	1
t_3	0	1	0	1	1
t_4	1	1	1	1	0
t_5	1	0	0	1	0
t_6	0	1	0	1	0

(a)

	location				
id	c	e	b	d	a
t_6	0	1	0	1	0
t_3	0	1	0	1	1
t_1	1	1	0	1	1
t_4	1	1	1	1	0
t_2	1	1	1	0	1
t_5	1	0	0	1	0

(b)

id	trajectory
t'_6	$(\{d,a,c,e\},\{d,a,c,e\})$
t'_3	$(\{d,a,c,e\},\{d,a,c,e\},\{d,a,c,e\},f)$
t'_1	$(\{d,a,c,e\},\{d,a,c,e\},\{d,a,c,e\},\{d,a,c,e\})$
t'_4	$(\{a,b,d\},\{a,b,d\},e,c)$
t'_2	$(\{a,b,d\},\{a,b,d\},e,c)$
t'_5	$(\{a,b,d\},g,c)$

(c)

Fig. 2: (a) key of $\mathcal{T}$ using Z-ordered locations (b) Gray ordered key, and (c) output of ZGA

$F.sb$	$F.\sup$
d	5
e	5
(d,e)	4
c	4
(d,c)	3
a	3

(a)

	freq. subtraj.	
id	e	d
t_1	1	1
t_2	1	0
t_3	1	1
t_4	1	1
t_5	0	1
t_6	1	1

(b)

	freq. subtraj.	
id	e	d
t_5	0	1
t_1	1	1
t_3	1	1
t_4	1	1
t_6	1	1
t_2	1	0

(c)

id	trajectory
t'_5	$(\{d,a,e,c\},g,\{d,a,e,c\})$
t'_1	$(\{d,a,e,c\},\{d,a,e,c\},\{d,a,e,c\},\{d,a,e,c\})$
t'_3	$(\{d,a,e,c\},\{d,a,e,c\},\{d,a,e,c\},f)$
t'_4	$(\{a,b,d\},\{a,b,d\},e,c)$
t'_6	$(\{a,b,d\},e)$
t'_2	$(\{a,b,d\},\{a,b,d\},e,c)$

(d)

Fig. 3: (a) ordered array F containing subtrajectories and its respective support (presenting the 5 most frequent) (b) key of $\mathcal{T}$ using 2 most frequent subtrajectories (c) gray ordered key, and (d) output of Algorithm SGA

Note that i increases from 1 to m in each iteration. For each subtrajectory of size i, it calculates the support $\sup(s', C'_s)$, where C'_s is the set of supporting trajectories of s in C', and s' is the sensitive location with the largest support in C'_s (Step 5). Then, ℓ^m-ANON populates a set $\mathcal{S}$ with all subtrajectories that require protection, either because they have a lower support than k in C', or because they co-occur with s' in more than ℓ subtrajectories in C'_s (Steps 6-7). After considering all subtrajectories in C', the algorithm sorts $\mathcal{S}$ with respect to the support of its members, in increasing order (Step 8). Next, it considers each subtrajectory in $\mathcal{S}$ and applies distance-based generalization to it, so that: (i) its support is at least k, and (ii) it does not co-occur with s' in more than ℓ subtrajectories in C'_s (Steps 9-13). Note that the generalization aims at minimizing the *trajectory distance* measure (see Definition 4), and it is applied to each subtrajectory in $\mathcal{S}$. After that, the algorithm proceeds to the next iteration, if i does not exceed m. Otherwise, ℓ^m-ANON returns C', which satisfies $(k,\ell)^m$-anonymity (Step 14).

Example 7: Consider applying ℓ^m-ANON to a cluster containing t_4, t_2, and t_5 in Fig. 1a, when all parameters are set to 2. The algorithm starts by considering the subtrajectories of size 1 in Fig. 4a and calculates $\sup(s', C'_s)$, for each of them (Steps 2-5). Then, it adds a into $\mathcal{S}$, since its support is $1<k$ (Steps 6-7). No other subtrajectory satisfies the check in Step 6, so ℓ^m-ANON sorts $\mathcal{S}$ and applies generalization. Thus, the generalized location $\{a,b\}$, which has minimum $\mathcal{D}_{loc}(a,b)$, is constructed and replaces a and/or b, in all trajectories of C' (Steps 8-13). This produces the cluster in Fig. 4b. After that, ℓ^m-ANON considers the subtrajectories of size 2 in Fig. 4c and applies the same procedure. This creates the cluster, shown in Fig. 4d, which contains $\{a,b,d\}$ and satisfies $(2,2)^2$-anonymity. As all subtrajectories of size 2 have been considered, ℓ^m-ANON returns the cluster in Fig. 4d.

IV. Related work

Trajectory data anonymization has attracted significant attention, due to the pervasive use of location-aware devices that led to tremendous increase in the volume of collected spatiotemporal data about individuals. Bonchi et al. [3] surveyed works on trajectory data anonymization and classified them into *motion-pattern based* and *location based*, according to the adversarial model they adopt. Motion-pattern based methods investigate how anonymized data may allow an attacker to predict individuals' locations, based on their mobility patterns [9], [11], whereas the goal of location based methods is to prevent the inference of individuals' identity and/or sensitive location information from anonymized data (e.g., [1], [14], [16], [18], [21], [23]). Our method falls into the latter category, and, more specifically, to the subcategory of *QID-aware* methods, which guard against attackers with specific background knowledge. In what follows, we discuss QID-aware, location based methods that are relevant to the one we propose. We also note that there are *QID-blind* methods [1], [16], which do not consider specific locations that may risk privacy.

Terrovitis et al. [21] considered multiple attackers, each knowing a different set of places of interest (POIs), visited by individuals. To preserve privacy in this setting, the authors proposed limiting the probability of associating these POIs to an individual's record in the published trajectory dataset. This is achieved through a suppression-based method, which aims at removing the least number of POIs from trajectories, so that the remaining trajectories are protected with respect to the knowledge of each adversary. Unlike [21], our method additionally prevents the inference of sensitive location information, and employs generalization, which generally preserves data utility better than suppression.

Yarovoy et al. [23] considered trajectories containing time information, in addition to POIs, and assumed that each individual has a different set of POIs and times that need to be protected. To offer protection, the authors followed a k-anonymity based approach, which protects trajectories based

s	$\sup(s, C')$
a	1
b, d, e	2
c	3

(a)

id	trajectory
t'_2	$(\{a,b\},\{a,b\},e,c)$
t'_4	$(\{a,b\},d,e,c)$
t'_5	(d,g,c)

(b)

s	$\sup(s, C')$
$(\{a,b\},\{a,b\})$	1
$(\{a,b\},d)$	1
(d,e)	1

(c)

id	trajectory
t'_2	$(\{a,b,d\},\{a,b,d\},e,c)$
t'_4	$(\{a,b,d\},\{a,b,d\},e,c)$
t'_5	$(\{a,b,d\},g,c)$

(d)

Fig. 4: (a) Support for subtrajectories of size $i = 1$ (b) transformed set C' after the processing of subtrajectory a, (c) support for subtrajectories of size $i = 2$, and (d) the final $(2, 2)^2$-anonymous result C'

on individuals' privacy requirements. In practice, eliciting privacy requirements from individuals may be challenging [3], so the use of privacy principles that provide more uniform protection, such as $(k, \ell)^m$-anonymity that we adopt, is more feasible. The authors of [23] developed two generalization-based algorithms, which employ Hilbert curves. The algorithms proposed in [23] follow a very different process of generalizing data than that of our algorithms and may create overlapping groups of records. This is mainly because the algorithms in [23] adopt a different privacy model and consider time information.

Recently, a k^m-anonymity-based algorithm for trajectory data has been proposed in [18]. This algorithm, called SeqAnon, works in an apriori-like fashion (i.e., it aims at protecting increasingly larger combinations of individuals' locations from identity disclosure) and applies generalization to the entire dataset. However, SeqAnon does not provide protection against the inference of individuals' sensitive location information, and may heavily generalize data. This is because it creates a large number of generalized locations compared to our algorithms, as our experiments demonstrate.

Methods that employ differential privacy [8] have also been proposed [5], [6]. These methods focus on specific data analytic tasks, such as query answering or frequent pattern mining [2], and they employ noise addition. Thus, unlike our approach, they harm data truthfulness, which is necessary to preserve in many applications [12].

V. Experimental evaluation

This section presents an experimental evaluation of our algorithms, in terms of data utility and efficiency.

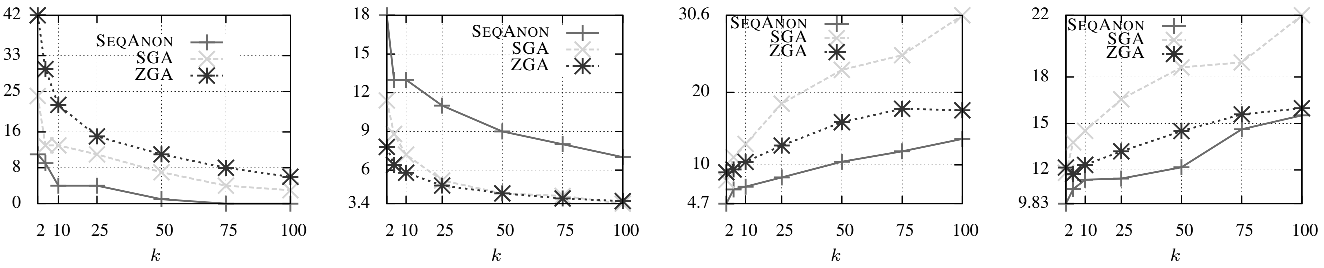
Experimental setup. All algorithms were evaluated using a dataset of moving objects in the Oldenburg city. The dataset was constructed using Brinkhoff's data generator [4], which is employed by many related works [1], [16], [21], [23]. The dataset consists of 18,143 trajectories, whose average length is 4.72 and are created as in [18]. We compare our algorithms, referred to as SGA and ZGA, respectively, with the SeqAnon algorithm [18], which is the most closely related to them. All algorithms were implemented in C++ and tested on an Intel Core i7 at 2.2 GHz with 6 GB of RAM.

Data utility. In this section, we evaluate the impact of parameters k, m, ℓ, and $\mathcal{C}$ on data utility, whose default values are $k = 5$, $m = 2$, $\ell = 2$, and $\mathcal{C} = 5$. The parameter K in SGA is fixed to 10, because this offered a good trade off between utility and efficiency, as we found empirically (results omitted due to space limitations). To quantify data utility, we measure the number of original and generalized locations in the anonymized datasets. For the generalized locations, we also measure their average size and distance, similarly to [18].

We first considered the effect of k, which varied in [2, 100]. Observe, in Figs. 5a and 5b, that the number of original locations, as well as that of generalized locations, decreases with k. This is because all algorithms create fewer and larger generalized locations (i.e., they generalize more original locations together), as k increases. Both SGA and ZGA retain many more original locations than SeqAnon, which helps data utility. Specifically, ZGA and SGA retained 81% and 28% more original locations than SeqAnon, on average. This is because they are applied to each cluster separately. This result is particularly encouraging, as our algorithms prevent both types of disclosure, unlike SeqAnon. Furthermore, ZGA and SGA created fewer generalized locations than SeqAnon, by 88% and 87.8% (on average), respectively.

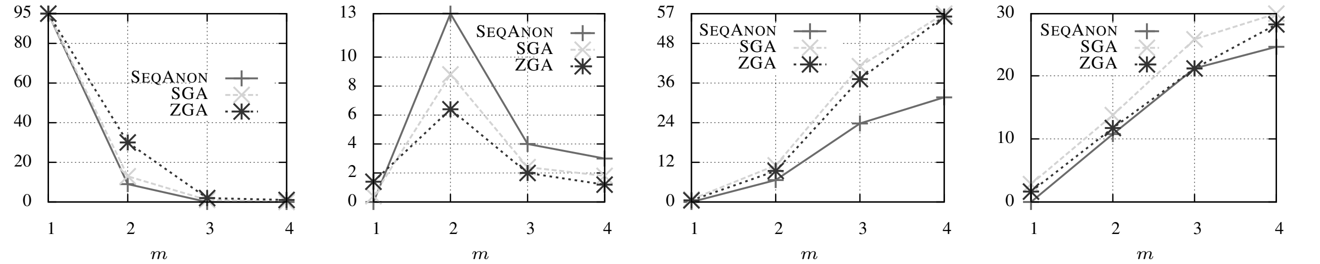
Fig. 5c reports the average number of locations in a generalized location, and Fig. 5d the average distance of all locations contained in each generalized location, which is computed as a percentage of the distance between the two furthest locations in a generalized location. Both of these statistics increase with k, as the distortion required to preserve privacy increases. However, the algorithms behave differently. That is, SeqAnon creates generalized locations that contain a small number of locations that are "close" to one another, whereas the generalized locations constructed by SGA are more, and consist of more distant original locations. This is because, the distribution of clusters created by SGA is rather skewed (i.e., a small number of clusters with dissimilar trajectories influence the average statistics in Figs. 5c and 5d). We also observed that the issue becomes evident for $k > 10$ and can be ameliorated by creating signatures comprised of more subtrajectories. On the other hand, the generalized locations constructed by ZGA are similar in terms of distance to those created by SeqAnon.

To study the impact of m on data utility, we varied this parameter in [1, 4]. Since the dataset we use has an average of 4.72 locations per trajectory, using $m = 3$ (respectively $m = 4$) means that we assume that an attacker knows approximately 65% (respectively 85%) of a user's locations. So, for $m = 3$ and $m = 4$, we expect that few locations will be published intact. On the contrary, for $m = 1$, all locations have support at least k, and a negligible amount of generalization is required. This is true for all algorithms, as can be seen in Fig. 6a. For $m = 2$, all algorithms create many generalized locations containing locations that are close to one another, as shown in Figs. 6b and 6c. Moreover, more generalization needs to be applied to satisfy privacy, as m increases. This leads



(a) (b) (c) (d)

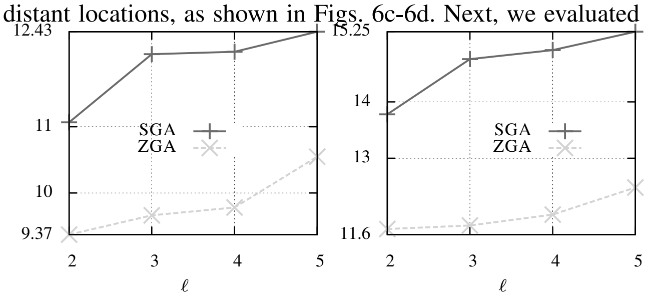
Fig. 5: (a) number of original locations published, (b) number of generalized locations published, (c) avg. number of generalized locations' size, and (d) avg. % of distance in generalized locations for parameter k.



(a) (b) (c) (d)

Fig. 6: (a) number of original locations published, (b) number of generalized locations published, (c) avg. number of generalized locations' size, and (d) avg. % of distance in generalized locations for parameter m.

to generalized locations with larger sizes that contain more distant locations, as shown in Figs. 6c-6d. Next, we evaluated the impact of parameter ℓ. As SEQANON does not satisfy l^m-diversity, we only report results for ZGA and SGA in Fig. 7. Notice that ZGA created generalized locations that consist of fewer and less distant locations, thereby helping data utility. On average, the generalized locations, created by ZGA consist of 82.5% fewer locations than those constructed by SGA, and their average distance is lower by 81.9%. The superiority of ZGA is attributed to the fact that it captures the distance of original locations more effectively, and it is evident for all tested values of ℓ.



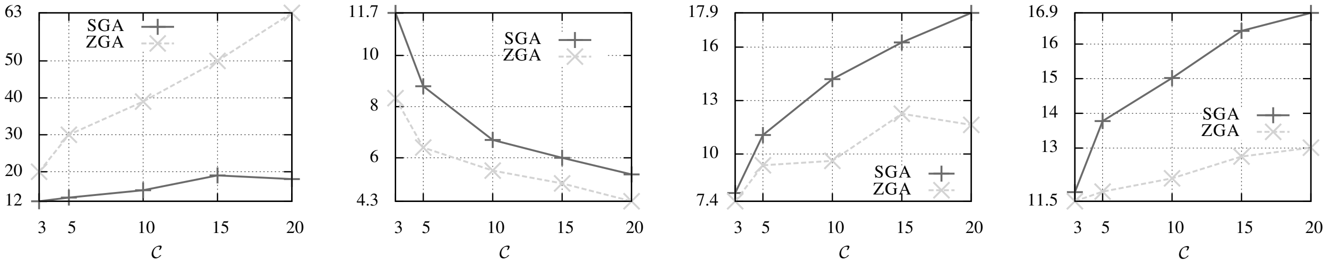
(a) (b)

Fig. 7: (a) avg. number of generalized locations' size and (b) avg. % of distance in generalized locations for ℓ.

Last, the effect of parameter $\mathcal{C}$, which controls the size of clusters, on data utility was investigated. The results in Fig. 8 demonstrate that ZGA outperforms SGA by a large margin, particularly for larger values of $\mathcal{C}$. Specifically, ZGA permits the publishing of 2.54 times more original locations than SGA, and it creates 78% fewer generalized locations, as shown in Figs. 8a and 8b, respectively. Furthermore, the generalized locations that are constructed by ZGA contain 22.7% fewer original locations, on average, as can be seen in Fig. 8c. The locations in these generalized locations are also "close" to one another, as can be observed from Fig. 8d. In fact, the average percent of distance for the locations of ZGA is smaller than the corresponding percent for SGA, by at least 14.7%. This confirms that taking into account the distance of original locations allows preserving data utility better than doing so based on their frequency, as SGA does through the use of frequent subtrajectories.

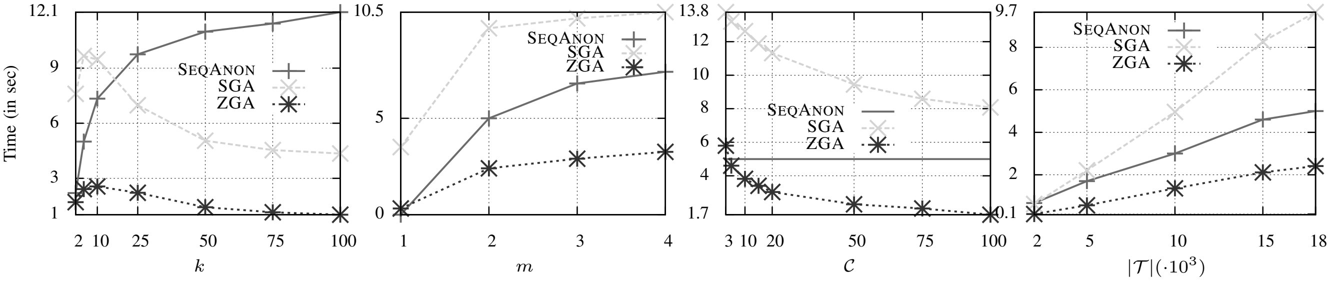
Efficiency. We report results for parameters k, m, $\mathcal{C}$, and dataset size, which affect runtime the most. The results for k in Fig. 9a show that ZGA and SGA are more efficient than SEQANON by 48.3% and 13.6%, respectively, when $k > 10$, but need more time, for smaller k values. This is because, they enforce l^m-diversity, which requires applying more generalization, when k is smaller. That is, our algorithms need to progressively increase the support of generalized locations to larger values than k, until l^m-diversity is satisfied, when $k < 25$. Furthermore, ZGA and SGA need less time as k increases, unlike SEQANON. This is because, SEQANON needs to consider a much larger number of potential generalizations to deal with subtrajectories with a lower support than k. Also, ZGA is more efficient than SGA by 72.8%, on average, as it does not require frequent subtrajectory mining.

The impact of m is shown in Fig. 9b. As m increases, all algorithms need more time, but ZGA outperforms SEQANON and SGA by 55% and 72%, on average. However, all algorithms scale well with m, as they employ the apriori principle. Fig. 9c shows the effect of $\mathcal{C}$. As expected, our algorithms are significantly faster as $\mathcal{C}$ increases, because they run on smaller clusters. On the other hand, SEQANON is not affected by this parameter, as it is applied to the entire dataset. In addition,



(a) (b) (c) (d)

Fig. 8: (a) number of original locations published, (b) number of generalized locations published, (c) avg. number of generalized locations' size, and (d) avg. % of distance in generalized locations for parameter $\mathcal{C}$.



(a) (b) (c) (d)

Fig. 9: Runtime with respect to (a) k, (b) m, (c) $\mathcal{C}$, and (e) dataset size

ZGA outperforms SeqAnon and SGA by 33% and 71% on average, respectively.

Last, we examined the impact of dataset size and report the results in Fig. 9d. In this experiment, we used increasingly larger subsets of records, which are contained in all larger sets. Observe that all algorithms scale equally well with the dataset size and that ZGA is more efficient than SeqAnon and SGA by 61% and 75%, on average.

VI. Conclusions

In this paper, we proposed a novel framework for anonymizing trajectory data. Our framework enforces $(k, \ell)^m$-anonymity on trajectory data, using two generalization-based algorithms that follow a Select-Organize-Anonymize paradigm. The benefit of our framework is that it enables the generation of truthful data with low distortion, in an efficient and scalable manner.

Acknowledgements

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

References

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376–385, 2008.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, 1995.

[3] F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Trajectory anonymity in publishing personal mobility data. *SIGKDD Expl*, 13(1):30–42, 2011.

[4] T. Brinkhoff. A framework for generating network-based moving objects. *GeoInf*, 6(2):153–180, 2002.

[5] R. Chen, G. Acs, and C. Castelluccia. Differentially private sequential data publication via variable-length n-grams. In *CCS*, 2012.

[6] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou. Differentially private transit data publication: a case study on the montreal transportation system. In *KDDM*, KDD '12, pages 213–221. ACM, 2012.

[7] R. Clarke. Person location and person tracking - technologies, risks and policy implications. *ITP*, 14(2):206–231, June 2001.

[8] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

[9] G. Ghinita, M. L. Damiani, C. Silvestri, and E. Bertino. Preventing velocity-based linkage attacks in location-aware applications. In *ICAGIS*, pages 246–255, 2009.

[10] G. Ghinita, P. Kalnis, and Y. Tao. Anonymous publication of sensitive transactional data. *KDE*, 23(2):161–174, 2011.

[11] W. Jin, K. LeFevre, and J. M. Patel. An online framework for publishing privacy-sensitive location traces. In *DEWMA*, pages 1–8, 2010.

[12] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. *Knowl. Inf. Systems*, 28(2), 2011.

[13] G. Loukides and J. Shao. An efficient clustering algorithm for k-anonymisation. *CSTJ*, 23(2):188–202, 2008.

[14] A. Monreale, G. L. Andrienko, N. V. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, and S. Wrobel. Movement data anonymity through generalization. *TDP*, 3(2):91–121, 2010.

[15] A. Monreale, R. Trasarti, D. Pedreschi, C. Renso, and V. Bogorny. C-safety: a framework for the anonymization of semantic trajectories. *TDP*, 4(2):73–101, 2011.

[16] M. E. Nergiz, M. Atzori, and Y. Saygin. Towards trajectory anonymization: a generalization-based approach. In *SPRINGL*, pages 52–61, 2008.

[17] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. Mining sequential patterns by pattern-growth: the prefixspan approach. *KDE*, 16(11):1424–1440, 2004.

[18] G. Poulis, S. Skiadopoulos, G. Loukides, and A. Gkoulalas-Divanis. Distance-based k^m-anonymization of trajectory data. In *MDM*, 2013.

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press, 1992.

[20] R. Raman and D. Wise. Converting to and from dilated integers. *ToC*, 57(4):567–573, 2008.

[21] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*, pages 65–72, 2008.

[22] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *VLDB J.*, 20(1):83–106, 2011.

[23] R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.

SECRETA: A System for Evaluating and Comparing RElational and Transaction Anonymization algorithms[*]

Giorgos Poulis
University of Peloponnese
poulis@uop.gr

Aris Gkoulalas-Divanis
IBM Research-Ireland
arisdiva@ie.ibm.com

Grigorios Loukides
Cardiff University
g.loukides@cs.cf.ac.uk

Spiros Skiadopoulos
University of Peloponnese
spiros@uop.gr

Christos Tryfonopoulos
University of Peloponnese
trifon@uop.gr

ABSTRACT

Publishing data about individuals, in a privacy-preserving way, has led to a large body of research. Meanwhile, algorithms for anonymizing datasets, with relational or transaction attributes, that preserve *data truthfulness*, have attracted significant interest from organizations. However, selecting the most appropriate algorithm is still far from trivial, and tools that assist data publishers in this task are needed. In response, we develop SECRETA, a system for analyzing the effectiveness and efficiency of anonymization algorithms. Our system allows data publishers to evaluate a specific algorithm, compare multiple algorithms, and combine algorithms for anonymizing datasets with both relational and transaction attributes. The analysis of the algorithm(s) is performed, in an interactive and progressive way, and results, including attribute statistics and various data utility indicators, are summarized and presented graphically.

1. INTRODUCTION

Publishing data about individuals is essential for applications, ranging from marketing to healthcare. Several marketing studies, for example, seek to find product combinations that appeal to customers with specific demographic profiles, while a large class of medical studies aims to discover associations between patient demographics and diseases. To enable these applications, data must be published in a way that preserves privacy and utility.

Towards this goal, numerous algorithms that prevent the disclosure of individuals' private and sensitive information, while maintaining *data truthfulness* (i.e., generate data that can be analyzed at a record level), have been proposed [4,6,7,10]. These algorithms work by transforming attribute values in a dataset (e.g., replacing them with more general values), and are applicable to either relational or transaction (set-valued) attributes. For example, an individual's year of birth is modeled as a relational attribute, while his/her purchased items are modeled as a transaction attribute. Furthermore, these algorithms can be combined, using a recent approach [9], to anonymize datasets with both relational and transaction attributes, referred to as RT-datasets.

While there is a growing interest for publishing protected and truthful data from governmental [8] and industrial organizations [1], selecting the most appropriate algorithm, for a given dataset and publishing scenario, remains a challenging and error-prone task. This is because both the effectiveness and efficiency of algorithms depend on: (a) *data characteristics* (e.g., the distribution of values in an attribute), (b) *various input parameters which affect the level of privacy and utility* (e.g., hierarchies that govern data transformation), and (c) *data utility requirements* (e.g., the need to accurately answer a certain query workload, or to adhere to constraints on the way values are transformed).

To assist data publishers in this task, we propose SECRETA, the first system for evaluating and comparing anonymization algorithms for relational, transaction, and RT datasets. Our system integrates 9 popular algorithms under a common, benchmark-oriented framework, and it allows data publishers to apply and analyze the performance of one or more of these algorithms. SECRETA operates in two modes, namely *Evaluation* and *Comparison*.

The Evaluation mode can be used to configure and evaluate the effectiveness of a given algorithm, with respect to data utility and privacy, as well as its efficiency. For capturing data utility, we employ several information loss measures [7, 12] and support data utility requirements. These requirements can be expressed using queries and/or *utility constraints* [7], which are specified by data publishers or generated automatically. Furthermore, SECRETA enables the use of 20 different combinations of algorithms to anonymize RT-datasets. The selection and management of these combinations is performed in an intuitive way that allows preserving different aspects of data utility.

The Comparison mode offers data publishers the ability to design and execute benchmarks for comparing multiple anonymization algorithms. These benchmarks facilitate an interactive and progressive comparison of sets of algorithms, with respect to their utility and efficiency. The results of the comparative analysis are summarized and presented graphically, allowing for fast and intuitive understanding of the effectiveness and efficiency of different algorithms.

To our knowledge, SECRETA is the only system that permits a comprehensive evaluation and comparison of recent anonymization techniques. The Cornell Anonymization Toolkit [11] demonstrates a single algorithm for relational data, also supported by SECRETA, while TIAMAT [3] does

[*] More details about the demo, together with additional screen shots, are available at: `http://secreta.uop.gr/`.

not support algorithms for transaction data, nor methods for anonymizing RT-datasets. Moreover, none of these systems employs utility requirements. We believe that the distinctive features of SECRETA can greatly assist data publishers in making informed decisions on publishing anonymized data.

2. OVERVIEW OF SECRETA

This section describes the components of our system, which we broadly divide into *frontend* and *backend* components. The frontend offers a Graphical User Interface (GUI), which enables users to: (a) issue anonymization requests, and (b) visualize and store experimental results. The backend consists of components for servicing anonymization requests and for conducting experimental evaluations. The architecture of SECRETA is presented in Figure 1.

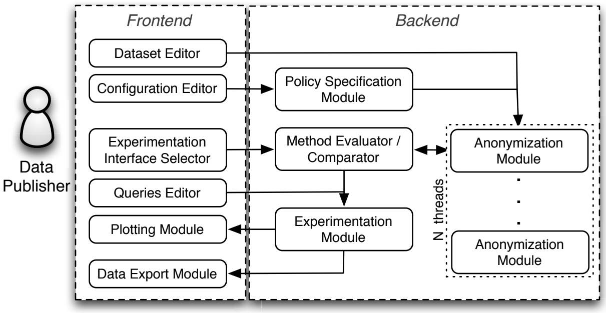


Figure 1: Architecture of SECRETA

2.1 Frontend of SECRETA

The frontend is implemented using the QT framework (`https://qt-project.org`). Using the provided GUI, users can: (a) select datasets for anonymization, (b) specify hierarchies and query workloads, (c) select and configure anonymization algorithms, (d) execute experiments and visualize the experimental results, and (e) export anonymized datasets and experimental results, in a variety of formats. In what follows, we detail the components of the frontend.

Dataset Editor: It enables users to select datasets for anonymization. The datasets can have relational and/or transaction attributes, and they need to be provided in a Comma-Separated Values (CSV) format. Once a dataset is loaded to the Dataset Editor, the user can modify it (edit attribute names and values, add/delete rows and attributes, etc.) and store the changes. The user can also generate data visualizations, such as histograms of attributes. Figure 2 shows a loaded dataset and some visualizations.

Configuration Editor: It allows users to select hierarchies and to specify utility and privacy policies. Hierarchies are used by all anonymization algorithms, except COAT [7] and PCTA [5], whereas utility and privacy policies are only used by these two algorithms to model such requirements. Hierarchies and policies can be uploaded from a file, or automatically derived from the data, using the algorithms in [7].

Queries Editor: This component allows specifying query workloads, which will be used to evaluate the utility of anonymized data in query answering. The system supports the same type of queries as [12], and uses Average Relative Error (ARE) [12] as a defacto utility indicator. The query workloads can be loaded from a file and edited by the user, or be inserted directly using the GUI (see Figure 2).

Experimentation Interface Selector: This component selects the operation mode of SECRETA. Figure 3 shows an interface of the Evaluation mode, in which users can evaluate a given algorithm, while Figure 4 shows an interface of the Comparison mode, which allows users to compare multiple algorithms. Through these interfaces, users can select and configure the algorithm(s) to obtain the anonymized data, store the anonymized dataset(s), and generate visualizations that present the performance of the algorithm(s).

Plotting Module: This module is based on the QWT library (`http://qwt.sourceforge.net/`) and supports a series of data visualizations that help users analyze their data and understand the performance of anonymization algorithms, when they are applied with different configuration settings. Specifically, users can visualize information about: (a) *the original/anonymized dataset* (e.g., histograms of attributes, relative difference of the frequency between an original and a generalized value), and (b) *anonymization results*, for *single* and *varying* parameter execution. In single parameter execution, the results are derived with fixed, user-specified parameters and include frequencies of generalized values in relational or set-valued attributes, runtime, etc. In varying parameter execution, the user selects the start/end values and step of a parameter that varies, as well as fixed values for other parameters. The plotted results include data utility indicators and runtime vs. the varying parameter.

Data Export Module: This module allows exporting datasets, hierarchies, policies, and query workloads, in CSV format, and graphs, in PDF, JPG, BMP or PNG format.

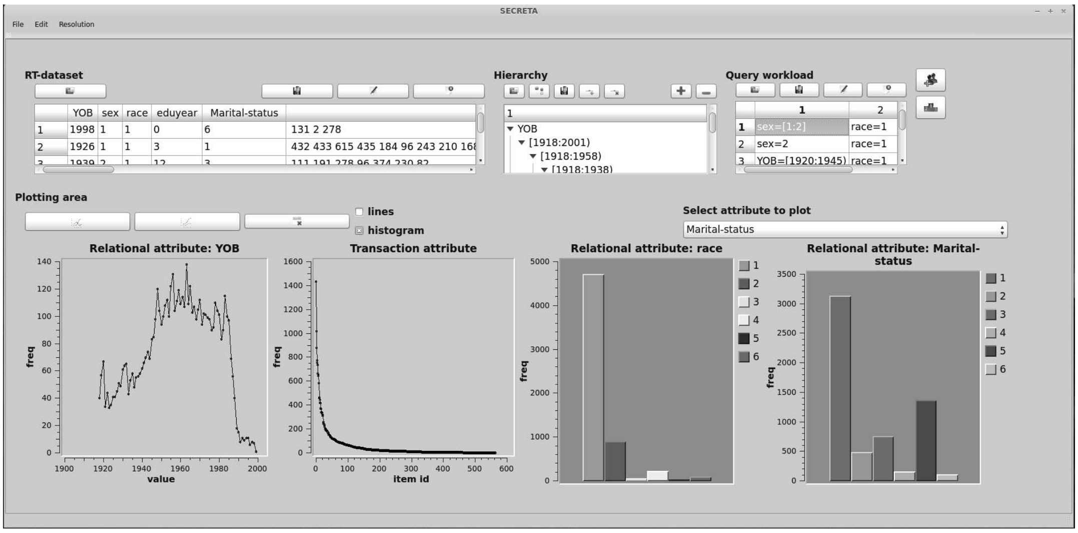
2.2 Backend of SECRETA

The backend of our system is implemented in C++. For each mode of operation, SECRETA invokes one or more instances of the Anonymization Module with the specified algorithm and parameters. The anonymization results are collected by the Method Evaluator/Comparator component and forwarded to the Experimentation Module. From there, results are forwarded to the Plotting Module, for visualization, and/or to the Data Export Module, for data export.

Policy Specification Module: This module invokes algorithms that automatically generate hierarchies [10], as well as the strategies in [7], which generate privacy and utility policies. The hierarchies and/or policies are used by the Anonymization Module (to be described later).

Method Evaluator/Comparator: This component implements the functionality that is necessary for supporting the interfaces of the Evaluation and of the Comparison mode. Based on the selected interface, anonymization algorithm(s) and parameters, this component invokes one or more instances (threads) of the Anonymization Module. After all instances finish, the Method Evaluator/Comparator component collects the anonymization results and forwards them to the Experimentation Module.

Anonymization Module: This component is responsible for executing an anonymization algorithm with the specified configuration. SECRETA supports 9 algorithms; 4 of them are applicable to datasets with relational atrtributes (Incog-



	YOB	sex	race	eduyear	Marital-status	
1	1998	1	1	0	6	131 2 278
2	1926	1	1	3	1	432 433 615 435 184 96 243 210 16
3	1939	2	1	12	3	111 191 278 96 374 230 82

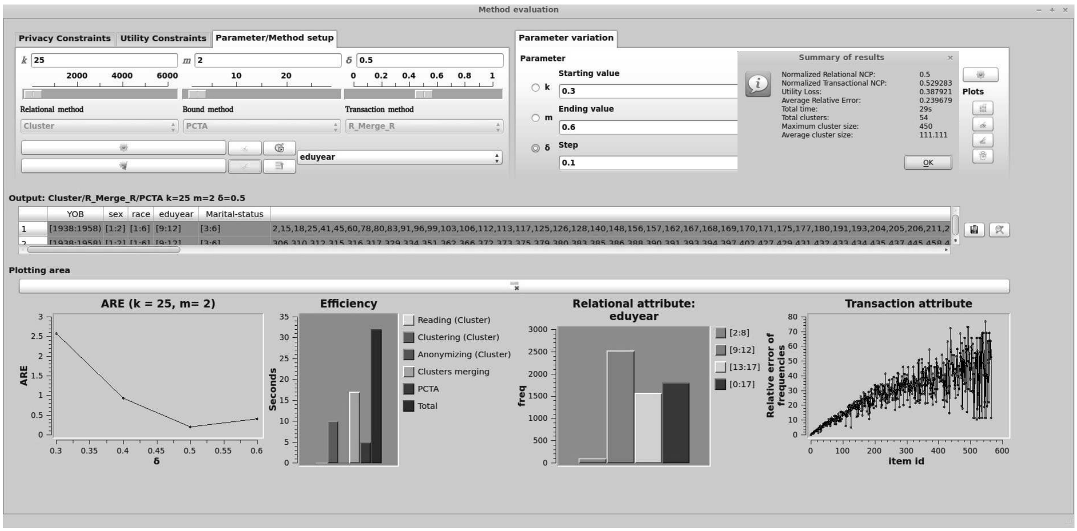
1
▼ YOB
▼ [1918:2001)
▼ [1918:1958)
▼ [1918:1938)

	1	2
1	sex=[1:2]	race=1
2	sex=2	race=1
3	YOB=[1920:1945)	race=1

☐ lines
☒ histogram

Select attribute to plot: Marital-status

Figure 2: Main screen of SECRETA



k 25 *m* 2 *δ* 0.5

Relational method: Cluster
Bound method: PCTA
Transaction method: R_Merge_R
eduyear

Parameter
○ k ○ m ◉ δ
Starting value: 0.3
Ending value: 0.6
Step: 0.1

Summary of results	
Normalized Relational NCP:	0.5
Normalized Transactional NCP:	0.529283
Utility Loss:	0.387921
Average Relative Error:	0.239679
Total time:	29s
Total clusters:	54
Maximum cluster size:	450
Average cluster size:	111.111

OK

Output: Cluster/R_Merge_R/PCTA k=25 m=2 δ=0.5

	YOB	sex	race	eduyear	Marital-status	
1	[1938:1958)	[1:2]	[1:6]	[9:12]	[3:6]	2,15,18,25,41,45,60,78,80,83,91,96,99,103,106,112,113,117,125,126,128,140,148,156,157,162,167,168,169,170,171,175,177,180,191,193,204,205,206,211,2
2	[1938:1958)	[1:2]	[1:6]	[9:12]	[3:6]	306 310 312 315 316 317 329 334 351 362 366 372 373 375 379 380 383 385 386 388 390 391 393 394 397 402 427 429 431 432 433 434 435 437 445 458 4

Figure 3: Evaluation mode: Method evaluation screen of SECRETA

nito [6], Cluster [9], Top-down [4], and Full subtree bottom-up), and 5 to datasets with transaction attributes (COAT [7], PCTA [5], Apriori, LRA and VPA [10]). Additionally, it supports 3 bounding methods (**R**MERGE$_r$, **T**MERGE$_r$, **RT**MERGE$_r$) [9], which enable the anonymization of *RT*-datasets by combining two algorithms, each designed for a different attribute type (e.g., Incognito and COAT).

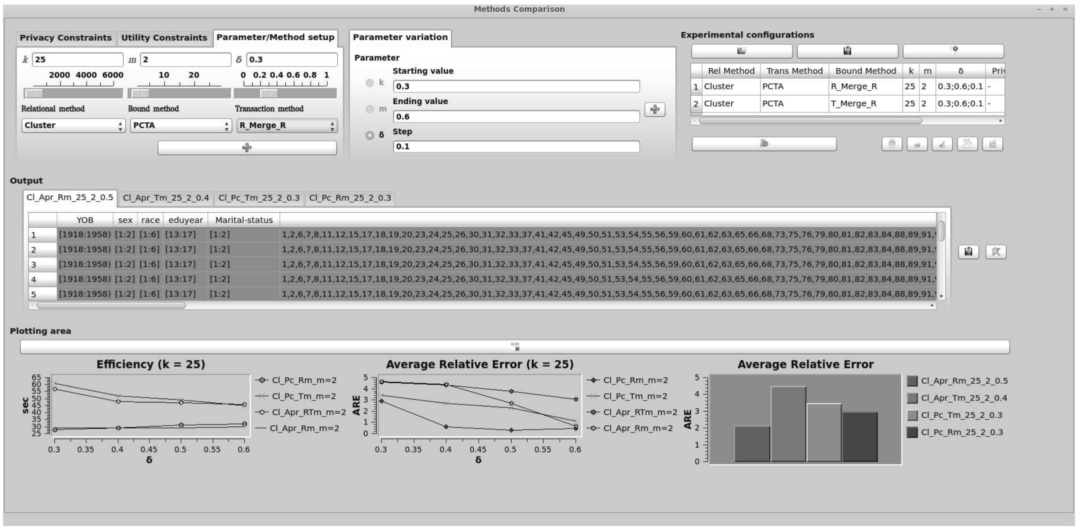
Experimentation Module: This module is responsible for producing visualizations of the anonymization results and of the performance of the anonymization algorithm(s), in the case of *single* and *varying* parameter execution. For visualizations involving the computation of ARE, input is used from the Queries Editor module. The produced visualizations are presented to the user, through the Plotting Module, and can be stored to disk, using the Data Export module.

3. DEMONSTRATION PLAN

During the demonstration, attendees will be able to use SECRETA to: (a) create, edit and analyze a dataset, and (b) execute two different scenarios that demonstrate the modes, functionality range, and potential of the system.

Using the Dataset Editor: The demonstration will start by allowing the user to load a ready-to-use *RT*-dataset. After that, the user will be able to edit the attribute names of the dataset, as well as the values in some records. These operations can be performed directly from the input area (top-left pane in Figure 2), and the user may overwrite the existing dataset with a modified one, or export it to a file. Subsequently, the user will analyze the dataset by plotting histograms of the frequency of values in any attribute (bottom pane in Figure 2).

Using the Configuration and Queries Editor: The user will load a predefined hierarchy from a file. This hierarchy is fully browsable and editable, through the hierarchy area (top-mid pane in Figure 2). Then, the user will



Methods Comparison

Privacy Constraints | Utility Constraints | Parameter/Method setup

k 25 m 2 δ 0.3

Relational method: Cluster

Bound method: PCTA

Transaction method: R_Merge_R

Parameter variation

Parameter

○ k ○ m ● δ

Starting value: 0.3

Ending value: 0.6

Step: 0.1

Experimental configurations

	Rel Method	Trans Method	Bound Method	k	m	δ	Pri
1	Cluster	PCTA	R_Merge_R	25	2	0.3;0.6;0.1	-
2	Cluster	PCTA	T_Merge_R	25	2	0.3;0.6;0.1	-

Output

Cl_Apr_Rm_25_2_0.5 | Cl_Apr_Tm_25_2_0.4 | Cl_Pc_Tm_25_2_0.3 | Cl_Pc_Rm_25_2_0.3

	YOB	sex	race	eduyear	Marital-status	
1	[1918:1958)	[1:2]	[1:6]	[13:17]	[1:2]	1,2,6,7,8,11,12,15,17,18,19,20,23,24,25,26,30,31,32,33,37,41,42,45,49,50,51,53,54,55,56,59,60,61,62,63,65,66,68,73,75,76,79,80,81,82,83,84,88,89,91,
2	[1918:1958)	[1:2]	[1:6]	[13:17]	[1:2]	1,2,6,7,8,11,12,15,17,18,19,20,23,24,25,26,30,31,32,33,37,41,42,45,49,50,51,53,54,55,56,59,60,61,62,63,65,66,68,73,75,76,79,80,81,82,83,84,88,89,91,
3	[1918:1958)	[1:2]	[1:6]	[13:17]	[1:2]	1,2,6,7,8,11,12,15,17,18,19,20,23,24,25,26,30,31,32,33,37,41,42,45,49,50,51,53,54,55,56,59,60,61,62,63,65,66,68,73,75,76,79,80,81,82,83,84,88,89,91,
4	[1918:1958)	[1:2]	[1:6]	[13:17]	[1:2]	1,2,6,7,8,11,12,15,17,18,19,20,23,24,25,26,30,31,32,33,37,41,42,45,49,50,51,53,54,55,56,59,60,61,62,63,65,66,68,73,75,76,79,80,81,82,83,84,88,89,91,
5	[1918:1958)	[1:2]	[1:6]	[13:17]	[1:2]	1,2,6,7,8,11,12,15,17,18,19,20,23,24,25,26,30,31,32,33,37,41,42,45,49,50,51,53,54,55,56,59,60,61,62,63,65,66,68,73,75,76,79,80,81,82,83,84,88,89,91,

Plotting area

Figure 4: Comparison mode: Methods comparison screen of SECRETA

load a preconstructed query workload from a file, edit the query values using the query workload area (top-right pane in Figure 2), and follow either of the two following scenarios.

Evaluating a method for *RT*-datasets: In this scenario, the users will configure, apply, and evaluate a method, in a series of steps. First, they will use the "Method evaluation" interface (Figure 3) and set the values for parameters k, m, δ, by inputting them directly in the form, or by using the corresponding slider (top-left pane in Figure 3). Then, they may select two algorithms, one for anonymizing the relational attributes, and one for the transaction attribute, and a bounding method for combining the selected algorithms.

Next, the users will initiate the anonymization process. When this process ends, a message box with a summary of results will be presented and the anonymized dataset will be displayed in the output area (middle pane in Figure 3). Last, the users will select a number of data visualizations. These visualizations will be presented in the plotting area (bottom pane in Figure 3) and may illustrate any combination of the following: (a) ARE scores for various parameters (e.g., for varying δ and fixed k and m), (b) the time needed to execute the algorithm and its different phases, (c) the frequency of all generalized values, in a selected relational attribute, and (d) the relative error between the frequency of the transaction attribute values, in the original and the anonymized dataset.

Comparing methods for *RT*-datasets: In this scenario, the users will compare multiple anonymization methods. Using the "Methods comparison" interface (shown in Figure 4), they will: (a) select algorithms for anonymizing each type of attributes, as well as a bounding method, (b) set the values for parameters that will be fixed, as described above (top-left pane in Figure 4), and (c) choose a varying parameter (top-mid pane in Figure 4), along with its start/end value and step. The choices for (a) to (c) comprise a configuration, which will be added into the experimenter area (top-right pane in Figure 4). Similar configurations will be created by the users for at least another method. After the methods are applied, the users will select various graphs, which will be displayed in the plotting area (bottom pane in Figure 4).

4. CONCLUSION

In this paper, we presented SECRETA, a system that helps data publishers analyze the performance of anonymization algorithms and make informed decisions on publishing anonymized data. Our system allows evaluating and comparing a range of different algorithms, in an interactive and progressing way. In the future, we will extend our system, by incorporating additional algorithms, such as those in [2].

Acknowledgements

G. Poulis is supported by Heraclitus II, S. Skiadopoulos by EICOS/Thalis, and G. Loukides by a Research Fellowship from the Royal Academy of Engineering.

5. REFERENCES

[1] http://www-03.ibm.com/software/products/us/en/infosphere-optim-data-privacy/.

[2] J. Cao, P. Karras, C. Raïssi, and K. Tan. *rho*-uncertainty: Inference-proof transaction anonymization. *PVLDB*, 3(1):1033–1044, 2010.

[3] C. Dai, G. Ghinita, E. Bertino, J.-W. Byun, and N. Li. Tiamat: a tool for interactive analysis of microdata anonymization techniques. *PVLDB*, 2(2), 2009.

[4] B. Fung, K. Wang, and P. Yu. Top-down specialization for information and privacy preservation. In *ICDE*, 2005.

[5] A. Gkoulalas-Divanis and G. Loukides. Utility-guided clustering-based transaction data anonymization. *TDP*, 5(1):223–251, 2012.

[6] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *SIGMOD*, 2005.

[7] G. Loukides, A. Gkoulalas-Divanis, and B. Malin. COAT: Constraint-based anonymization of transactions. *KAIS*, 28(2):251–282, 2011.

[8] National Institutes of Health, 2013. Data repositories. http://www.nlm.nih.gov/NIHbmic/nih_data_sharing_repositories.html.

[9] G. Poulis, G. Loukides, A. Gkoulalas-Divanis, and S. Skiadopoulos. Anonymizing data with relational and transaction attributes. In *ECML/PKDD*, 2013.

[10] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. *VLDB J.*, 20(1):83–106, 2011.

[11] X. Xiao, G. Wang, and J. Gehrke. Interactive anonymization of sensitive data. In *SIGMOD*, 2009.

[12] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A.-C. Fu. Utility-based anonymization using local recoding. In *KDD*, pages 785–790, 2006.

Fuzzy Times on Space-time Volumes

Manos PAPADAKIS[2], Martin DOERR[1] and Dimitris PLEXOUSAKIS[1]
[1]Institute of Computer Science, Foundation for Research and Technology - Hellas,
N. Plastira 100, Heraklion, 700 13, Greece, Email: {martin, dp}@ics.forth.gr
[2]Computer Science Department, University of Crete,
Voutes Campus, 700 13, Heraklion Greece, Email: mpapad@csd.uoc.gr

Abstract: This paper is a study on the formal representation of the temporal extent of periods, as defined in CIDOC CRM. Our proposal exploits positive and negative evidence, gained by observations, to approximate the definite and indefinite bounds of a period, stating whether it is considered as active or possibly active. We contribute to the modelling of imprecise temporal information by proposing a generalized definition of time intervals which allows the representation of fuzziness that characterizes archaeological findings and historical data. Moreover, we provide an alternative to Allen's operators, based on fuzzy time intervals. Finally, we extend our fuzzy interval model to represent four-dimensional volumes, which are associated with the spatiotemporal nature of periods; we also provide a spatiotemporal version of the basic temporal relations to approximate the temporal topology of periods. Our study has a crucial impact to fields associated with reality modelling, especially observation-based sciences.

1. Introduction

The study of the past through the definition, description and association of past periods [7] is an important part of historical, archaeological and other research processes. According to CIDOC CRM [5] [10], a period is regarded as a "set of coherent phenomena or cultural manifestations bounded in time and space". Analysing periods temporally and spatially allows researchers to assume a spatiotemporal perspective of reality [6], where periods are treated as 4-dimensional volumes and their possible associations are considered as topological relations, such as inclusion, separation and overlap.

Since the past is not directly observable [1], evidence about past periods or events is derived through the observation process among traces that were left by their related phenomena. For instance, wood combustion by-products within the soil of a woodland provides clues that refer to an event of a wildfire over that area, at a time that is specified by the dating process of the individual findings. However a basic problem of inferring the extent of larger spatiotemporal phenomena from empirical evidence is that observations are necessarily point wise and scarce which leads to the need of hypotheses of interpolating and complementing the space and time between.

Data obtained about periods is imprecise due to limitations of observation, definition and information loss from the past, which leads to uncertainty with regard to their spatiotemporal modelling. For instance, a meeting in time cannot be expressed using Allen's 'meets' operator [3], because the time interval endpoints of periods are fuzzy, rendering time point equality inapplicable. In addition, the forms of evidence for the spatiotemporal extent of a period in the past are too limited for defining its true bounds [6],[1]. Moreover, relations based only on total time projections of periods do not provide a complete view of a temporal topology among periods due to variations in space. The examples mentioned above introduce a second problem in which the limited precision of the observations lead to the assumption of fuzzy layers in order to approximate the real boundaries or to describe the total association of two periods under consideration.

In the rest of this document, we first provide a concise analysis of the aforementioned issues, motivated by the example of the conquest of a city. Afterwards, we propose solutions to address all issues, followed by a brief overview of potential applications.

2. Objectives

The main objective of this paper is to contribute to the theoretical foundations of spatiotemporal modelling from observation data, focusing on the impact of time imprecision on space-time volumes. Furthermore we propose an approach that devises a general theory which connects the knowledge derived from empirical observations with the reasoning on approaching the extent and the topological relationships over space-time volumes.

There are several approaches on modelling imprecise time however, they mainly focus on pure temporal reasoning. The prevalent idea is the usage of disjunctions of possible temporal relationships [17]. An alternative method introduces the attachment of a possibility value among Allen's operators [4]. Particularly, each temporal relation is loosened or strengthen in order to express for instance, the concept of possible before and definitely before. A related approach to the previous work focuses on the expression of indeterminacy and incompleteness of temporal information using three valued logic [18]. Finally, an approach approximates the boundaries of the Ancient Millan periods ("Pre-Roman", "Roman" and so on) by applying fuzzy set theory to model imprecise time intervals.

There are algebras that focus on the description of temporal [3] or spatial topology [19]; however there are quite few approaches, to the best of our knowledge, that concern the description of relationships between regions located in space and time. Particularly, in [11] there is an approach on associating temporal spaces introducing spatiotemporal relations. More specific, temporal and spatial algebras are combined to conclude to a pair of relations like "before overlap", rather than describing exclusively the relations that are formed between the space-time points.

In order to illustrate the need of spatiotemporal modelling based on imprecise observation data as well as the necessity of temporal topology over space-time volumes we assume the scenario of the conquest of a city by force where it has been encountered many times in history. An important result of such events is the change of leadership of the city under attack. In many cases, key evidence that documents such changes is the considerable cultural difference within the city borders between the population defending the city, the invading troops and the resulting population. Inferring from empirical evidence about the conquest events can provide information related to the succession of different cultures in cities under consideration. There are several historical sources that demonstrate the city conquest scenario, for instance, the siege of the ancient Troy which was excavated by Torfmann [16] and described in Homer's Iliad. However for the sake of simplicity we describe the objectives of the current paper using a simplified imaginary city conquest event.

2.1 – Time point equality

It is obvious that a conquest event can be considered as a "meeting in time" [1] using temporal terms, in which the intervals that represent the activities of the defending and invading armies "meet" without a discontinuity ("gap") in between. Trying to apply Allen's meet relation to the event of leadership change mentioned above, leads to the problem of seeking the exact time point for which the endpoint equality of the meet relation should be applied; this is not an easy decision, as the conquest event which represents the meeting is not instantaneous, but has a duration which, in some cases, is considerable and differs, depending on the location within the city. Furthermore, even with the assumption that the conquest event is regarded as instantaneous by associating it with the time that the defending army surrenders, it is difficult to pinpoint the exact time point in which the last soldier gave in.

2.2 – Certainty and Impossibility of existence

A habitation event leaves traces, products of human activities, that associate a population and, by correlation, its culture with a certain place. These primary observations can be associated with time intervals through dating processes [1]; by combining the spatiotemporal information gained from semantically related observations, a period with fuzzy space-time bounds can be defined [6], representing the lifetime of the individual culture at that place. However, primary observations can only indicate the possible existence of that population over time; precise bounds cannot be derived. Even historical records are sparse and of limited precision. This hinders any effort to turn a possibility of existence to a precisely limited certainty or impossibility.

2.3 – Temporal relations on space-time volumes

Periods can be modelled by four-dimensional volumes that serve as the union of the spatiotemporal expanses of the set of the constituent coherent phenomena. Going back to the conquest event, there are two periods of habitation in relation to the city, one referring to the pre-conquest population and another involving the post-conquest one. In order to define temporal relations between these periods, time projections must be used, since general spatiotemporal relations do not allow for complete order (e.g., before, meets, starts and so on). Taking into account the corresponding time projections of the individual habitation periods, the resulting relation is formally "overlaps in time"; on the other hand, considering that the city was captured without coexistence between the populations, its natural semantics are a "meets in time" relation. This ambiguity is caused by the fact that the expansion of the invading population happened over time and not instantly.

3. Methodology – Proposed Solutions

3.1 – Negative and Positive observations

The true temporal extent of a period cannot be observed but is approximated through observations [6]. Based on the differentiation between positive and negative evidence, gained by observations, we propose the notion of contradictory observations as an amplified unity criterion. The contextual coherence of positive observations forms a "part of" relation with the period to be defined, as analyzed in CIDOC CRM [10]. On the other hand, negative observations provide evidence of semantic inconsistency, precluding the possibility of co-existence or inclusion relations with the period to be defined. For instance, in our motivating example, findings of everyday objects belonging to the defending people are regarded as positive observations for the period that illustrates the reign of the former civilization; on the contrary, massive findings of weapons that are associated with the conquering army are considered as negative observations.

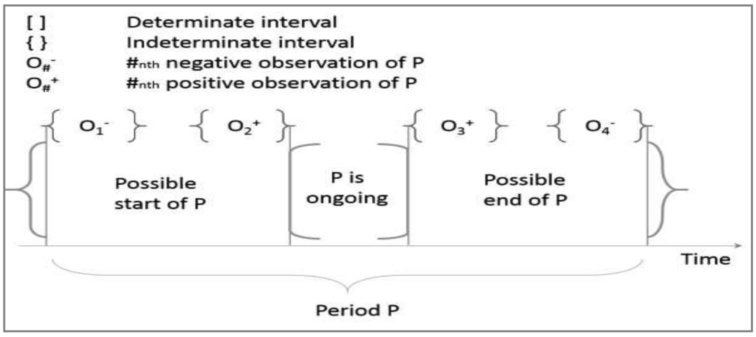


Figure 1: Determinate intervals from indeterminate time

The temporal extent of a set of positive observations forms an inner, determinate interval in which the period is "ongoing" [1] [2], as depicted in Figure 1, whereas negative observations define the outer bounds where the period ceases to exist. It is worth mentioning that a period is a unified entity, according to CIDOC CRM [5]. It would be counter-intuitive to consider that a period is "on-going" only during the time extent derived by the dating process of the positive observations (and inactive otherwise), because the unity property is violated. As a result, a period is "on-going" till there is no positive evidence to prove that claim and a period is "possibly active" till a negative observation proves that the period is inactive. The intermediate intervals between positive and negative observations result in a possibility of existence for the period in question, rather than a precise state (e.g., ongoing or finished).

3.2 – Point-wise Time and Fuzzy intervals

Time [12] is perceived as a set of duration-less points known as instants or moments that is isomorphic to the set of real numbers R. The real timeline is defined as a totally ordered set "Time" under a total order <. In addition, let tempD: Time x Time → R be a real-valued function that refers to the temporal distance between two time points and satisfies the properties of positive definiteness, symmetry and triangle inequality.

A time interval I ⊂ Time is regarded as a set of instants which illustrates the temporal extent of observable phenomena under consideration. There are no instantaneous observable phenomena (i.e. with zero duration). In addition, the concept of an empty time interval cannot be applied in real events, since phenomena cannot be separated from the notion of time.

In the sequel, we provide a definition of intervals, allowing to represent fuzzy and non-fuzzy intervals as specializations of a more general definition. We define the boundaries of an interval or space-time region as sets which can either have a certain thickness, in the case of fuzziness, or are restricted to points or hyper-planes, in the case of precise sets. The boundary becomes the area in which the fuzzy function evaluates to values in (0, 1). In this way, the effect of a fuzzy function is "encapsulated" in the thickness of the boundary, hence our theory becomes independent of the choice and evaluation of a particular fuzzy function.

Let I be a set of time points that represent a valid time interval and $\mathrm{fe}\colon \mathrm{Time} \rightarrow [0,1] \subset R$ be the fuzzy time point membership function of I. We define the following:

1. A neighbourhood of a time point t is a set $\mathrm{N}_\mathrm{t}^\mathrm{r}$ that contains $\{\mathrm{t_i} \in \mathrm{Time} : \mathrm{tempD}(\mathrm{t}, \mathrm{t_i}) \leq \mathrm{r\ where\ r} > 0\} \setminus \{\mathrm{t}\}$.
2. A time point t is a boundary point of I if for all neighbourhoods $\mathrm{N}_\mathrm{t}^\mathrm{r}$ of point t, it holds that $\exists \mathrm{t_1}, \mathrm{t_2} \in \mathrm{N}_\mathrm{t}^\mathrm{r} \colon \mathrm{t_1} \in \mathrm{I}$ and $\mathrm{t_2} \notin \mathrm{I}$. In case of fuzzy boundaries, the fuzzy function evaluates to $\mathrm{fe}(\mathrm{t_1}, \mathrm{I}) > 0$ and $\mathrm{fe}(\mathrm{t_2}, I) < 1$.
3. The boundary $\mathrm{B_I}$ of time interval I is the set of all boundary points of I.

4. A time point t is an interior point of I if there exists a neighbourhood $\mathrm{N}_\mathrm{t}^\mathrm{r} \subseteq \mathrm{I}$ of point t where $\forall t_i \in N_t^r: \mathrm{fe}(\mathrm{t_i}, \mathrm{I}) = 1$.
5. The interior $\mathrm{I_I}$ of time interval I is the set of all interior points of I.
6. The closure $\mathrm{C_I}$ of time interval I, is the set of all interior and boundary points of I: $\mathrm{C_I} = \mathrm{I_I} \cup \mathrm{B_I}$.
7. The exterior $\mathrm{E_I}$ of time interval I is the complement of its closure $\mathrm{C_I}$: $\mathrm{E_I} = \mathrm{Time} \setminus \mathrm{C_I}$.

The following properties hold for a valid interval I:

(a) $\mathrm{I_I} \subseteq \mathrm{I}$
(b) non-empty boundary or interior sets : $\mathrm{B_I} \neq \emptyset$ and $\mathrm{I_I} \neq \emptyset$
(c) bounded closure: $\exists \mathrm{N}_\mathrm{t}^\mathrm{r}: \mathrm{C_I} \subseteq \mathrm{N}_\mathrm{t}^\mathrm{r}$ where $\mathrm{r} < \infty, \mathrm{t} \in \mathrm{Time}$, hence the boundary and interior sets are finite
(d) the boundary set is divided into two subsets that wrap the interior set: $\exists \mathrm{A}, \mathrm{B} \subset \mathrm{B_I} : \mathrm{A} \cap \mathrm{B} = \emptyset \wedge \mathrm{A} \cup \mathrm{B} \neq \emptyset$
(e) convex interior set: $\forall x, y \in I_I$, it holds that $\forall \mathrm{t} \in [0,1]$ $[(1-\mathrm{t})\mathrm{x} + \mathrm{ty}] \in \mathrm{I_I}$ In other words, all points on the line segment that connects two interior points are also elements of the interior set.

According to our definition, a time interval is composed of the interior and boundary sets that refer to the determinate and indeterminate interval of the defining period, respectively, as mentioned in Section 3.1. As our knowledge about the period approximates its real bounds, the boundary set of the corresponding temporal interval shrinks. The highest precision is reached when the boundary set contains only the true endpoints of the period.

3.3 – Fuzzy temporal relations

Information about the relevant topology of precise time intervals can be stated using Allen's operators [3]. However, in cases of imprecise information, the temporal association of fuzzy intervals can be approximated by a set of Allen's operators that hold between the possible endpoints of the imprecise intervals. In this section, we propose an alternative approach of Allen's operators that describes the temporal association of fuzzy intervals. Particularly, endpoint relations proposed by Allen are replaced with set-oriented statements, properly modified in order to adhere to the representation of fuzzy intervals, defined in Section 3.2.

Let $\mathrm{B_A}$, $\mathrm{B_B}$, $\mathrm{I_A}$, $\mathrm{I_B}$, $\mathrm{C_A}$ and $\mathrm{C_B}$ be the boundary, interior and closure sets of two valid time intervals A and B.

1. A 'before' B describes the scenario of disjoint closure sets; particularly interval A happened earlier than B. It is formalized as follows: $\forall \mathrm{a} \in \mathrm{C_A}, \forall \mathrm{b} \in \mathrm{C_B}: \mathrm{a} < \mathrm{b}$
   A 'after' B is the inverse relation of 'before', with the following formalization: $\forall \mathrm{a} \in \mathrm{C_A}, \forall \mathrm{b} \in \mathrm{C_B}: \mathrm{a} > \mathrm{b}$. It is noteworthy that 'before' and 'after' relations can be applied to the interior, closure and boundary sets, as well as any other time-point set. These relations will be used as building blocks for the definition of the rest.
2. A 'meets' B describes meetings in time, where interval B starts at the end of the temporal extent of A, formalized as: $\mathrm{B_A} \cap \mathrm{B_B} \neq \emptyset$ and $\mathrm{I_A}$ 'before' $\mathrm{I_B}$.
3. A 'overlaps' B describes the relation where time interval A pre-exists of B, both share interior points and B keeps to exist after A. The following formalization holds: $\mathrm{I_A} \cap \mathrm{I_B} \neq \emptyset, \mathrm{I_A} \setminus \mathrm{C_B} \neq \emptyset, \mathrm{I_B} \setminus \mathrm{C_A} \neq \emptyset$ and $\mathrm{I_A} \setminus \mathrm{C_B}$ 'before' $\mathrm{C_A} \leftrightarrow$ $\mathrm{I_B} \setminus \mathrm{C_A}$ 'after' $\mathrm{C_B}$.
4. A 'starts' B describes the relation where interval A signifies the start of B. Both intervals start at the same time point, however, they share interior points and B keeps existing after A ends. It is formalized as follows:

$I_A \cap I_B \neq \emptyset, I_A \setminus C_B = \emptyset, I_B \setminus C_A \neq \emptyset$ and $I_B \setminus C_A$ 'after' C_A.

5. A 'during' B describes an inclusion relation like "falls within". Particularly, the following properties hold: $C_A \subseteq I_B$.
6. A 'finishes' B refers to the opposite scenario of 'starts', where interval A signifies the termination of B. Particularly, B predates A, both share interior points and terminate at the same time point. It is formally expressed as follows: $I_A \cap I_B \neq \emptyset, I_A \setminus C_B = \emptyset, I_B \setminus C_A \neq \emptyset$ and $I_B \setminus C_A$ 'before' C_A.
7. A 'equals' B is a special scenario of temporal topology where intervals A and B are related with mutual inclusion. Particularly, A shares interior points with B, while the interior of B falls within A and the interior of A falls within B. The formalization is: $I_A \cap I_B \neq \emptyset, I_A \subset C_B$ and $I_B \subset C_A$.

The inverse relations are formalized similarly, by replacing the boundary, interior and closure sets of interval A to the corresponding sets of B and vice versa.

Fuzzy relations approximate the temporal topology of periods, regardless of their level of fuzziness. Properly loosened constraints are applied in order to avoid any limitation on the boundary's thickness and hence the size of the individual periods. Finally, our approach is compatible with Allen's approach in cases of complete awareness.

3.4 – Meetings in time and transitive events

As already mentioned in Section 3.1, temporal approximation of periods is in many cases imprecise [2] [4], making it difficult or impossible to define a clear-cut relation between pairs of successive intervals. To address this, we adopt the concept of a transitive event, which considers the non-instantaneous nature of a meeting in time. In essence, we consider meetings in time as convex intervals with indefinite bounds (Figure 2), indicating that the phenomenal meeting happened sometime during the transitive event. The temporal extent of the event depends on the intensity of change, with regard to the instance that acts as a transition factor between the meeting intervals. For modelling reasons, we assume a "true" meeting in time happened somewhere within the fuzzy bounds, which cannot be observed, but can be constrained.

Our fuzzy intervals approach, presented in Section 3.2, can describe a fuzzy meeting in time as the set of shared time points between the disjoint interiors of the related periods. The intersecting boundary points form the indeterminate interval that refers to the temporal extent of the transitive event. As far as the relations of the associated periods are concerned, meetings in time can be approximated using the fuzzy relation 'meets'.

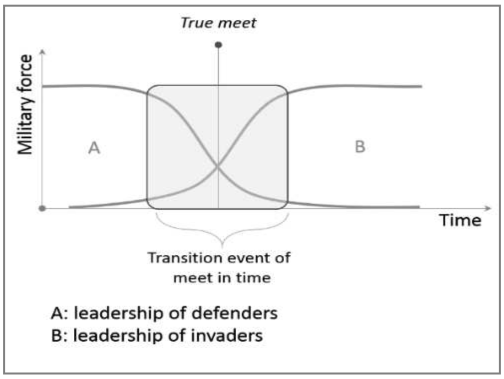

True meet
Military force
A
B
Time
Transition event of meet in time
A: leadership of defenders
B: leadership of invaders

Figure 2: True meet within the transitive event

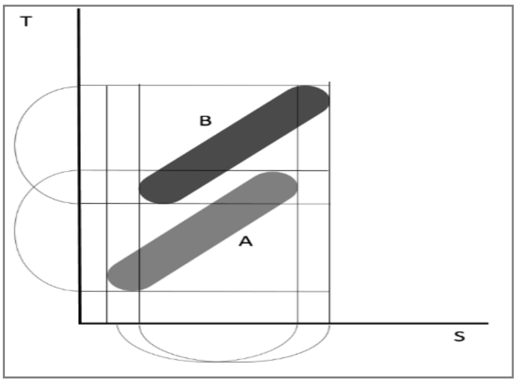


Figure 3: Period A is before period B

3.5 – Spatiotemporal approach of time

As already mentioned in section 2.3, periods are considered as spatiotemporal entities, represented as four-dimensional volumes in space-time. Approaching their temporal topology using only time projections (i.e. by completely ignoring their space extent), may sometimes lead to ambiguity. For instance, as illustrated in Figure 3, periods A and B are associated with a 'total overlap' relation, however their volumes do not touch at any point in their common space, resulting in a 'local before' relation. Such conflicting temporal approaches, derived by local and total relations, is caused by the fact that, in many cases, the space occupied by a period is changing over time. Such cases are frequent in archaeology, but have been widely disregarded in the literature [11].

In the rest of this section, we propose a spatiotemporal approach of temporal relations. First, space-time is defined, proportionally to point-wise time definition, followed by the formalization of fuzzy volumes, which represent the space-time extent of a period. In the end, we propose a set of temporal relations applied on fuzzy volumes based on the fuzzy relations.

3.5.1 – Point-wise Space-Time and fuzzy volumes

Space-time is usually interpreted from a Euclidean space perspective, in which space and time are regarded as three- and one-dimensional systems respectively. For the sake of simplicity, we adopt a non-relativistic model, in which time is treated as universal and constant, being independent of the state of motion of the observer. Furthermore, we only regard observers that are at rest with respect to our spatial reference system. Additional study on reference systems in relative motion could be considered as future work. It is worth mentioning that the theory presented here is independent of geometries on curved surfaces, such as Earth, and relativistic space distortions.

Space-time is defined as a set ST that is isomorphic to the 4-dimensional space R^4 and includes all spatiotemporal points. Each point in ST is considered as a quadruple (x, y, z, t) where $\mathrm{x, y, z} \in \mathrm{Space}$ and $\mathrm{t} \in \mathrm{Time}$, where Space is the known space set defined in physics while Time refers to the timeline. Variables x, y and z refer to Cartesian space coordinates in some arbitrary spatial reference system, while t stands for time values.

A space-time volume $\mathrm{V} \subset \mathrm{ST}$ is regarded as a set of spatiotemporal points that illustrates the spatiotemporal extent of a periods. We offer a generalized definition of space-time volumes that allows space-time representation of periods, regardless of whether fuzzy information is included or not. The fuzzy volumes model is defined by extending the generalized interval definition, proposed in Section 3.2, in 4-dimensional space-time. Although the definitions of neighbourhood, interior, boundary, closure and exterior sets remain the same, defining a valid volume requires the following modifications. Let V be a space-time volume and $\mathrm{B_V}, \mathrm{I_V}$ and $\mathrm{C_V}$ the boundary, interior and closure set respectively. Instead of a divided boundary, as in interval property (d), we define a connected and continued boundary, formalized as:

- $\forall \mathrm{x} \in \mathrm{B_V}$ is implied that $\forall \mathrm{N_x^r} \exists \mathrm{y} \in \mathrm{B_V}: \mathrm{x} \neq \mathrm{y} \wedge \mathrm{y} \in \mathrm{N_x^r}$ and
- $\nexists \mathrm{A}, \mathrm{B} \subset \mathrm{B_V}$ such that $\mathrm{A} \cap \mathrm{B_V} \neq \emptyset, \mathrm{B} \cap \mathrm{B_V} \neq \emptyset, \mathrm{A} \cap \mathrm{B_V} \cap \mathrm{B} \neq \emptyset$ and
  $\mathrm{B_V} = (\mathrm{A} \cap \mathrm{B_V}) \cup (\mathrm{B} \cap \mathrm{B_V})$

Moreover, the convexity property (e) is only required to hold for the time dimension. The interior, boundary and closure sets, associated with 4-dimensional volumes, represent the indeterminate, determinate and whole extent of the period, respectively.

3.5.2 – Spatiotemporal version of fuzzy temporal relations

Information about temporal topology over space-time volumes is gained by the relevant association of their time projections. The projected image of a volume upon the time axis forms a fuzzy interval, composed by the individual projections of the boundary and interior sets of the projected volume. Fuzzy relations that hold on the derived fuzzy intervals approach the temporal topology of the corresponding projected volumes.

We differentiate time-projections based on the amount of space points that are being projected. Time projections that provide an overall image of the occupied space, are regarded as 'total', whereas time projections derived by discrete space slices are considered as 'local'. Consequently, fuzzy relations that describe the temporal association of the individual volumes are distinguished into 'local' or 'total', depending on whether they are related to 'local' or 'total' time-projections, respectively.

'Local' or 'total' temporal topology is used, according to the existence of shared space points between the associated volumes. Particularly, we propose that if there is no space overlap among the volumes, then their temporal topology is described by their 'total' relation. However, in cases of total or partial space overlap, volumes are associated by a set of 'local' relations that relate the time projections of the shared space slices. Instead of using a set of 'local' relations to describe the temporal topology of the individual space-overlapped volumes, we propose the definition of a single prevailing relation that represents every possible temporal association of the shared space slices. Consequently, we introduce a relation hierarchy, defined intuitively and semantically related with data of historical significance. Particularly, a level of prevalence is attached to each relation, forming a classification order from strongest to weakest, as follows: overlaps, equals, {starts, finishes}, meets and {before, during}. Stronger relations cause the weaker ones to be excluded as possible volume associations. There are two pairs of relations with the same level of prevalence, namely 'starts'/'finishes' and 'before'/'during'. These cases do not raise decidability issues, since the former pair is evaluated semantically as a special case of 'equals', while the latter is considered as a counter-intuitive scenario.

Let A and B be two space-time volumes and $\mathrm{T_V}: \{\mathrm{Space}\} \to \{\mathrm{Time}\}$, $\mathrm{S_V}: \{\mathrm{Time}\} \to \{\mathrm{Space}\}$ the time and space projection functions of a set of space and time points over a volume V, respectively. For the sake of simplicity, the variants of functions $\mathrm{T_V}$ and $\mathrm{S_V}$ with arity 0 return the total time or space projection of the volume V; hence, $S_{A\cap B}$ refers to the overlapped space of volumes A and B.

In the rest of this section, we define the spatiotemporal version of the basic temporal relations, proposed in Section 3.3, that hold on space-overlapping volumes, based on the relation subsumption hierarchy introduced above. These relations are illustrated in Figure 4.

1. A 'sp_before' B: $\forall \mathrm{s} \in S_{A\cap B}: \ \mathrm{T_A(s)}$ \`before\` $T_B(\mathrm{s})$
2. A 'sp_during' B: $\forall \mathrm{s} \in S_{A\cap B}: \ \mathrm{T_A(s)}$ \`during\` $T_B(\mathrm{s})$
3. A 'sp_meets' B: $\exists \mathrm{s} \in S_{A\cap B}: \ \langle \mathrm{T_A(s)}$ \`meets\` $\mathrm{T_B(s)}\rangle$ and $\langle \mathrm{I_A} \cap \mathrm{I_B} = \emptyset \rangle$
4. A 'sp_starts' B: there are two interpretations of the 'starts' relation, causal and incidental. An incidental start describes the case where there is an area in the period B that is reached by the starting period A after its beginning, while in the other case a casual start does not allow areas of period B to be reached after the beginning of period A. For instance an incidental start can be considered as the transmission of a message that triggers the start of a new period, which is affected by speed limits, in contrary to a volcano eruption that can be regarded as instant which demonstrates a causal start. They are formalized as follows:
   Causal: $\forall \mathrm{s} \in \mathrm{S_{A\cap B}}: \ \mathrm{T_A(s)}$ \`starts\` $\mathrm{T_B(s)}$
   Incidental: $\exists \mathrm{s} \in \mathrm{S_{A\cap B}}: \langle \mathrm{T_A(s)}$ \`starts\` $\mathrm{T_B(s)}\rangle$ and neither $\langle$A \`sp_finishes\` B$\rangle$ nor $\langle$A \`sp_equals\` B$\rangle$ hold.

5. A ‘sp_finishes’ B: similarly, there are two interpretations of the ‘finishes’ relation, causal and incidental.
   Causal: $\forall s \in S_{A \cap B}$: $T_A(s)$ `finishes` $T_B(s)$
   Incidental: $\exists s \in S_{A \cap B}$ $\langle T_A(s)$ `finishes` $T_B(s)\rangle$ and neither $\langle A$ `sp_starts` $B\rangle$ nor $\langle A$ `sp_equals` $B\rangle$ hold.
6. A ‘sp_equals’ B: $\exists s \in S_{A \cap B}$: $\langle T_A(s)$ `equals` $T_B(s)\rangle$ and $\langle A$ `sp_overlaps` $B\rangle$ does not hold.
   In addition, there is a special case of volume equality, in which a pair of ‘starts’ and ‘finishes’ relations is evaluated into ‘equals’; it is formalized as follows:
   $\exists s_1, s_2 \in S_{A \cap B} : s_1 \neq s_2$ and $\langle T_A(s_1)$ `finishes` $T_B(s_1)\rangle$ and $\langle T_A(s_2)$ `starts` $T_B(s_2)\rangle$ and $\langle A$ `sp_overlaps` $B\rangle$ does not hold.
7. A ‘sp_overlaps’ B: $\exists s \in S_{A \cap B}$: $P_A^T(s)$ `overlaps` $P_B^T(s)$

It should be noted that fuzzy temporal relations for both time intervals and spatiotemporal volumes are considered complete, since there is no combination of intervals (or volumes) that cannot be described by exactly one of the proposed relations. This claim is backed by the fact that our analysis can model every temporal relation that is expressed in CIDOC CRM [5]. As far as the efficiency of calculating such relations is concerned, it depends on the candidate spatiotemporal information systems in which the proposed theory may be embedded and the 3D/4D indexing methods they support; these include spatiotemporal GIS and databases.

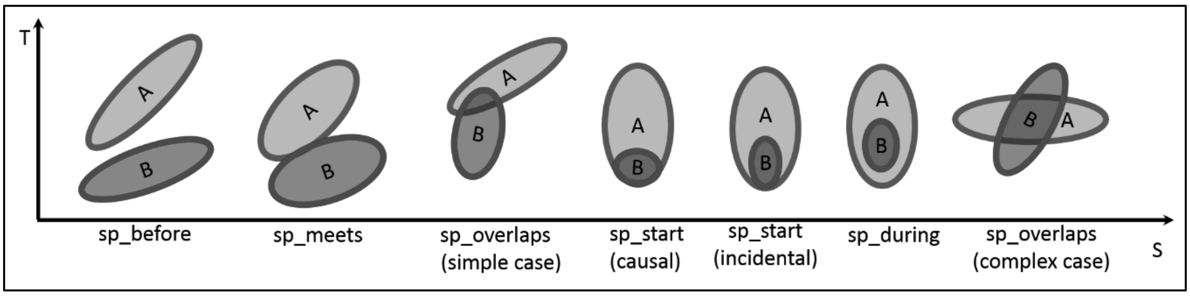


Figure 4: temporal relation on spatiotemporal volumes A and B

4. Applications

Our proposal provides solutions correlated with the temporal modelling of periods and the approximation of their temporal topology, when imprecise information is included. Key applications of such a theory are: the approximation of the spatiotemporal extent of a period, including determinate and indeterminate bounds, based on observations and historical sources; determining the influence that one period has on another based on the analysis of their temporal association; derivation of spatiotemporal relations based on semantic associations; representation of successive periods and meeting in time to realize “continuity” of periods [8]; providing answers to temporal queries applied on spatiotemporal entities like periods.

These applications have a crucial impact on fields associated with reality modelling, especially observation-based sciences, such as archaeology, geology, palaeontology, ecology and anthropology. More specifically, archaeology, geology and palaeontology are related to the revelation of data about various aspects of prehistory, such as human history and evolution, aging of the Earth, evolution of organisms and their association with the environment, periods of extinction and so on. Relating observations and findings with rock layers and layering through stratigraphy [9], the aforementioned sciences can approximate the temporal extent of periods, related to specific layers, and hence reveal information about their temporal association. Additionally studying the temporal association of findings with known periods can contribute to dating methods, by approaching the age or date of existence of the individual findings. Furthermore, our model can provide efficient temporal modelling of the identified geologic periods, through index fossils [13]. Finally, temporal topology extraction methods like the Harris matrix [14] can be evaluated through the association of the extracted results with the temporal topology derived by our model.

Ecology studies the interactions among organisms and their environment, while anthropology examines humans on a social and biological point of view. Scientific observations over long time periods can be associated with our model, defining the determinate and indeterminate boundaries of the corresponding periods. Also, approximating the relevant temporal topology over past periods and understanding past events enhances the reconstruction of possible pasts, by excluding inconsistent instances of our past, and resulting in the most prevailing scenario.

5. Conclusions and Future Work

This paper provides solutions to important issues in spatiotemporal modelling. Particularly, we proposed a model to reconstruct the extent of periods in space-time using empirical evidence and individual observations. It is worth noting that our model can deal with imprecise information by modelling fuzziness. Additionally we extended the basic temporal relations [3] to be applied on 4-dimensional entities like periods. More specific our theory allows independent spatial and dating information in order to approach the spatiotemporal relation that describes the coherent phenomena. Applying our theory conversely our model can provide evaluation of the concluding global spatiotemporal relations stated by historical sources among the phenomena of a space-time volume, by stating precise relations to point in them. The assumptions that were made about the extent of the spatiotemporal volumes can be considered as shortcomings of our spatiotemporal modelling approach. Essentially, the constraints that were stated about the spatiotemporal points, do not allow the illustration of periods in which an entity expands and retreats to the same place (Chinese invasion in Vietnam). A probable solution would be the declaration of phases that compose the super-period. Our approach on period modelling is directly dependent on the occurrence of observations, as a result it can by adopted by any science field that focuses on modelling reality using empirical evidence. In addition our theory can be embedded in GIS systems providing solutions in representing fuzzy boundaries. Future research involves period modelling in absence of direct observations, where additional factors must be considered, such as statistical frequency of observation events and efficiency of detection [15].

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales. Investing in knowledge society through the European Social Fund. In addition our work was partially supported by the EU FP7 project DIACHRON (ICT-2011.4.3, #601043).

6. References

[1] Doerr, M., Plexousakis, D., Kopaka, K., & Bekiari, Ch. (2004). Supporting Chronological Reasoning in Archaeology. In Computer Applications and Quantitative Methods in Archaeology Conference, CAA2004, (pp. 13-17).

[2] Kauppinen, T., Mantegari, G., Paakkarinen, P., Kuittinen, H., Hyvönen, E. & Bandini, S. (2010). Determining relevance of imprecise temporal intervals for cultural heritage information retrieval. Int. J. Hum.-Comput. Stud., 68, (pp. 549-560).

[3] James F. Allen. (1983). Maintaining knowledge about temporal intervals. Communications of the ACM 26, 11, (pp. 832-843).

[4] Cowley W., Plexousakis D., (2000b). An Interval Algebra for Indeterminate Time, Proc. Of the AAAI, Austin, Texas, August 2000.

[5] CRM. The CIDOC Conceptual Reference Model (ISO/CD 21127), http://cidoc.ics.forth.gr/

[6] Doerr, M., & Hiebel, G.H (2013). CRMgeo: Linking the CIDOC CRM to GeoSPARQL through a Spatiotemporal Refinement.

[7] Doerr, M., Kritsotaki, A., & Stead, S. (2004). Which Period is it? A Methodology to Create Thesauri of Historical Periods. Computer Applications and Quantitative Methods in Archaeology Conference, CAA2004, Prato, Italy, (pp. 13-17) April.

[8] Darvill, Timothy. & Oxford University Press. (2008). The concise Oxford dictionary of archaeology.

[9] Gradstein, F. M., J. G. Ogg, and A. G. Smith. "Chronostratigraphy: linking time and rock", A Geologic Time Scale 2004. 1st ed. Cambridge: Cambridge University Press, 2005. 20-46. Cambridge Books Online. Web. 07 April 2014.

[10] Crofts, N., Doerr, M., Gill, T., Stead, S., & Stiff, M., (2005) Definition of the CIDOC Conceptual Reference Model, June 2005 (version 4.2), http://www.cidoc-crm.org/docs/cidoc_crm_version_4.2.pdf

[11] Claramunt, Christophe, and Bin Jiang. (2001) "An integrated representation of spatial and temporal relationships between evolving regions." Journal of geographical systems 3.4: 411-428.

[12] J. Benthem, The Logic of time. D. Reidel Publishing Company, Dordrecht, 1983.

[13] Ghosh, v D. (2006). "Index fossils - Evidences from plant sources". Resonance: 69–77.

[14] E. Harris, M. Brown, and G. Brown, “Practices of archaeological stratigraphy. Academic press, 1993.

[15] Reid, P.C., et al. (2003). "The Continuous Plankton Recorder: concepts and history, from plankton indicator to undulating recorders". Progress in Oceanography 58: 117-175.

[16] Korfmann, M. (2006). Troia: Archäologie eines Siedlungshügels und seiner Landschaft. Mainz am Rhein: P. von Zabern.

[17] Doerr, M. Yiortsou, A. (1998) Implementing a Temporal Datatype.

[18] Gadia, S. K.; Nair, S. S.; and Poon, Y.-C. 1992. Incomplete information in relational temporal databases. In Proceedings of the 18th International Conference on Very Large Data Bases, 395–406.

[19] Bruns HT, Egenhofer M (1996) Similarity of spatial scenes. In: Kraak MJ, Molenaar M (eds) Advances in GISresearch II. Taylor and Francis, London, pp 173–184

Unifying Heterogeneous and Distributed Information about Marine Species through the Top Level Ontology *MarineTLO*

Yannis Tzitzikas(1)(2), Carlo Allocca(1), Chryssoula Bekiari(1), Yannis Marketakis(1), Pavlos Fafalios(1)(2), Martin Doerr(1), Nikos Minadakis(1), Theodore Patkos(1) and Leonardo Candella(3)

(1) Institute of Computer Science, FORTH-ICS, Greece
(2) Computer Science Department, University of Crete, Greece
(3) Consiglio Nazionale delle Ricerche, CNR-ISTI, Pisa, Italy
{tzitzik, carlo, bekiari, marketak, fafalios, martin, minadakn, patkos}@ics.forth.gr, leonardo.candela@isti.cnr.it

Marine species data are scattered across a series of heterogeneous repositories and information systems. There is no repository that can claim to have all Marine Species data. Moreover, information on marine species is made available through different formats and protocols. Our research aims at providing models and methods that allow integrating such information either for publishing it, browsing it, or querying it. Aiming at providing a valid and reliable knowledge ground for enabling semantic interoperability of marine species data, in this paper we motivate a top level ontology, called **MarineTLO** and discuss its use for creating **MarineTLO**-based warehouses. This approach has been implemented in the context of the iMarine operational European research infrastructure.

1 Introduction

Marine species data are widely distributed with few well-established repositories or standard protocols for their archiving and retrieval. Currently, the various laboratories have in place databases for keeping their raw data, while ontologies are primarily used for metadata that describe these raw data. One of the challenges in the iMarine project[1] is to enable users to experience a coherent source of facts about marine entities, rather than a bag of contributed contents. Considering the current setting where each iMarine source has its own model, queries like *"Given the scientific name of a species, find its predators with the related taxon-rank classification and with the different codes that the organizations use to refer to them"*, cannot be formulated (and consequently nor answered) by any individual source. To formulate such queries, we need an expressive conceptual model, while for answering them we also have to assemble pieces of information stored in different sources. For example, Figure 1 illustrates information about the species Thunnus albacares which is stored in different sources (here FLOD, ECOSCOPE, WoRMS, FishBase and DBpedia, more about these sources in the next section). These pieces of information are complementary, and if assembled properly, advanced browsing, querying and reasoning can be provided.

[1] iMarine, FP7 Research Infrastructures, 2011-2014

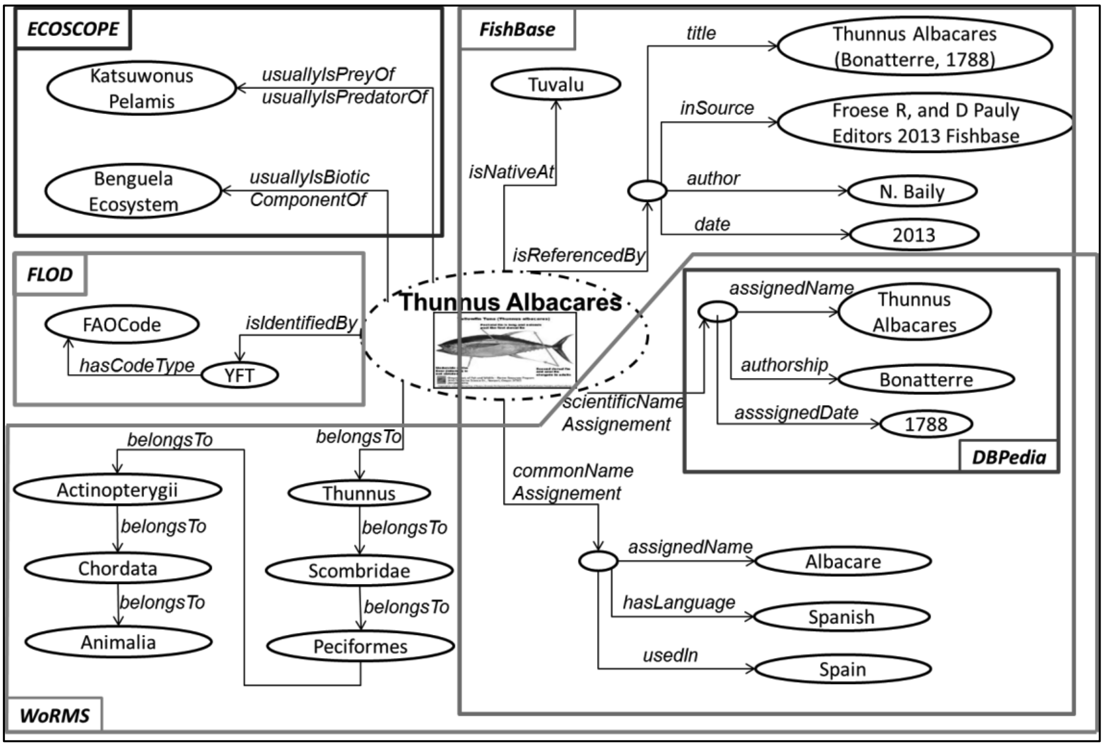


Figure 1: Integrated Information about Thunnus Albacares from five sources

We believe, therefore, that a unified and coherent model for better accessing/reasoning upon and across different marine data sources is a critical and, at the same time, challenging objective, in order to provide a valid and reliable knowledge ground for enabling semantic interoperability of marine data, services, applications and systems. In a nutshell, the key contributions of our work are the following: (a) we identify use cases motivating the need for having harmonized integrated information, (b) we introduce a generic core model, called **MarineTLO**, for schema integration, (c) we describe the mappings between this model and main sources of marine information for building integrated warehouses, (d) we comparatively evaluate two different triplestores for the problem at hand, and (e) we report results regarding the ability of the **MarineTLO**-based warehouse to answer queries which cannot be answered by the underlying sources. To the best of our knowledge, no such warehouse exists.

The rest of this paper is organized as follows. Section 2 discusses the underlying sources and motivating application scenarios, Section 3 describes the proposed approach, Section 4 describes the process for constructing MarineTLO-based warehouses, Section 5 discusses the process for evaluating the ontology, comparatively evaluates two triplestores, and describes the current uses of the MarineTLO-based warehouse. Finally, Section 6 concludes and identifies directions for future work and research.

2 Sources and Motivating Scenarios

In this section, we first describe the main underlying sources (§2.1) and then discuss four motivating scenarios as came up by the organizations participating in iMarine (§2.2).

2.1 Main Underlying Sources

Fisheries Linked Open Data (FLOD) RDF dataset. FLOD, created and maintained by Food and Agriculture Organization (FAO), is dedicated to create a dense network of relationships among the entities of the Fishery domains, and to programmatically serve them to semantic and traditional application environments. The FLOD content is exposed either via a public SPARQL endpoint[2] (suitable for semantic applications) or via a JAVA API to be embedded in consumers' application code. Currently, the FLOD network includes entities and relationships from the domains of Marine Species, Water Areas, Land Areas, Exclusive Economic Zones, and serves software applications in the domain of statistics and GIS.

ECOSCOPE Knowledge Base. IRD[3] offers a public SPARQL endpoint[4] for its knowledge base containing geographical data, pictures and information about marine ecosystems (specifically data about fishes, sharks, related persons, countries and organizations, harbors, vessels, etc.).

WoRMS. TheWorld Register of Marine Species[5] currently contains more than 200 thousand species, around 380 thousand species names including synonyms, and 470 thousands taxa (infraspecies to kingdoms).

FishBase. FishBase[6] is a global database of fish species. It is a relational database containing information about the taxonomy, geographical distribution, biometrics, population, genetic data and many more. Currently, it contains more the 32 thousand species and more than 300 thousand common names in various languages.

DBpedia. DBpedia[7] is a project focusing on the task of converting content from Wikipedia to structured knowledge so that Semantic Web techniques can be employed against it. At the time of writing this article, the English version of the knowledge base of DBpedia describes more than 4.5 million things, containing persons, places, works, species, etc. In our case, we are using a subset of DBpedia's knowledge base containing only fishes (i.e., instances classified under the class http://dbpedia.org/ontology/Fish).

2.2 Motivating Scenarios

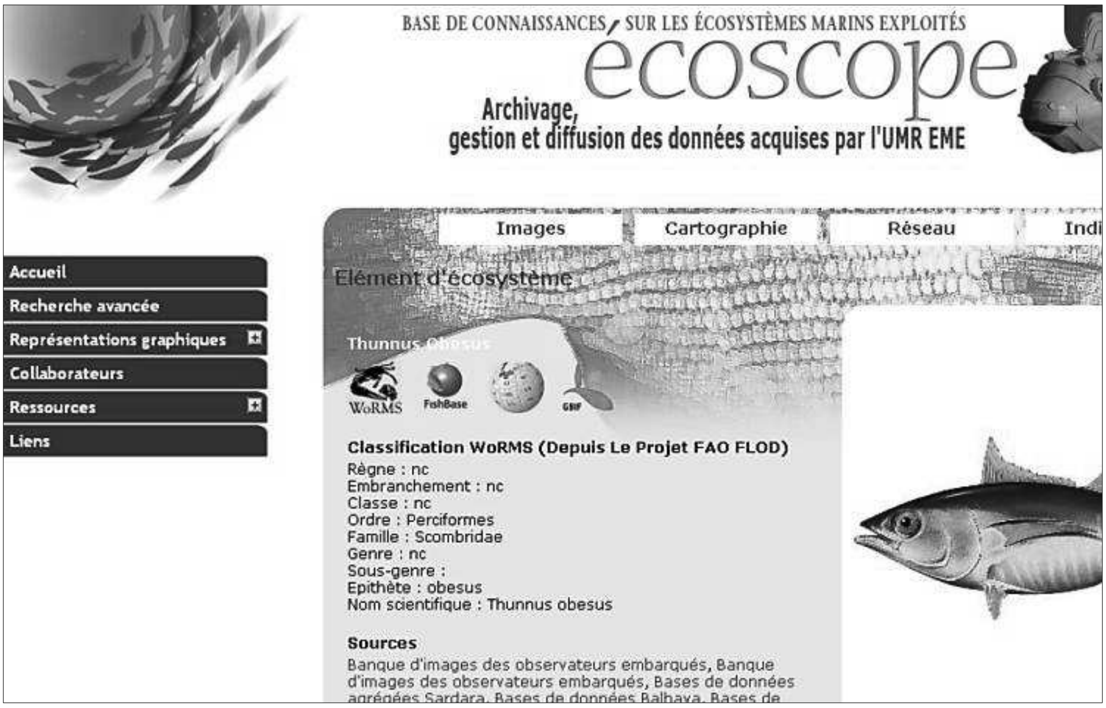
The availability of a top-level ontology for the marine domain would be useful in various scenarios.

For Publishing Linked Data. There is a trend towards publishing Linked Data; consequently a rising issue concerns the structure that is beneficial to use during such publishing. The semantic structure that will be presented can be used by the involved organizations for anticipating future needs for information integration, and thus alleviating the required effort for (post) integration.

Fact Sheets. FactSheetGenerator[8] is an application provided by IRD aiming at providing factual knowledge about the marine domain by mashing-up relevant knowledge distributed across several data sources. Figure 2 shows the results of the current FactSheetGenerator when searching for the species `Thunnus albacares`. Currently, the results are based only on ECOSCOPE and related knowledge stored in other sources (e.g., about commercial codes or taxonomic information) cannot be exploited. The approach that we will present in this

[2] http://www.fao.org/figis/flod/endpoint
[3] Institut de recherche pour le developpement (IRD), France (http://www.ird.fr/)
[4] http://ecoscopebc.mpl.ird.fr/joseki/ecoscope
[5] http://www.marinespecies.org
[6] http://www.fishbase.org
[7] http://dbpedia.org
[8] http://www.ecoscopebc.ird.fr

paper can be exploited for advancing this application, i.e., for providing more complete semantic descriptions.



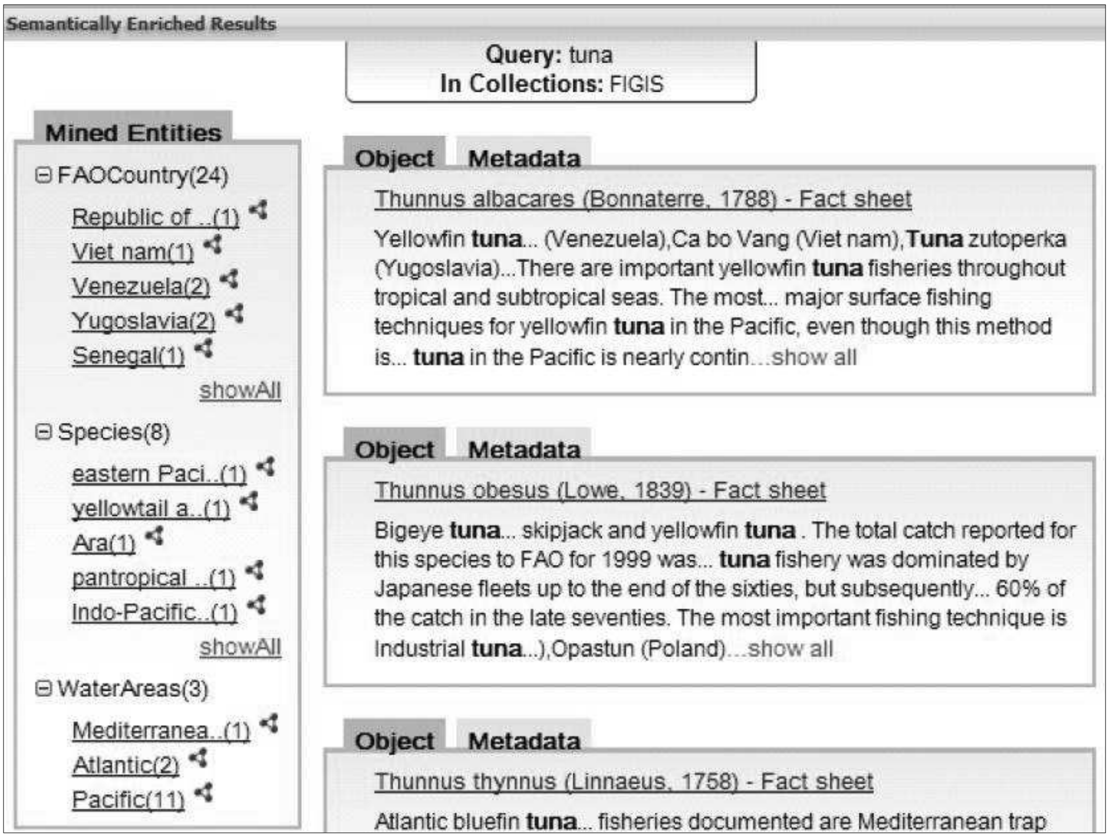
Classification WoRMS (Depuis Le Projet FAO FLOD)
Règne : nc
Embranchement : nc
Classe : nc
Ordre : Perciformes
Famille : Scombridae
Genre : nc
Sous-genre :
Epithète : obesus
Nom scientifique : Thunnus obesus

Sources
Banque d'images des observateurs embarqués, Banque d'images des observateurs embarqués, Bases de données agrégées Sardara, Bases de données Balbaya, Bases de

Figure 2: `Thunnus Albacares` in FactSheetGenerator

For Semantic Post-Processing of the Results of Keyword Search Queries. Another big challenge nowadays is how to integrate structured data with unstructured data (documents and text). The availability of harmonized structured knowledge about the marine domain can be exploited for a *semantic post-processing* of the search results (over dedicated or general purpose search systems). Specifically, the work done in the context of iMarine so far, described in [Fafalios et al., 2012][Fafalios and Tzitzikas, 2013], has proposed a method to enrich the classical (mainly keyword based) searching with *entity mining* that is performed at *query time*. The results of entity mining (entities grouped in categories) complement the query answers with information which can be further exploited by the user in a faceted and session-based interaction scheme [Sacco and Tzitzikas, 2009]. This means that instead of annotating and building indexes for the documents (or web pages), the annotation can be done at query time and using the desired entities of interest. These works show that the application of entity mining over the *snippets* of the top hits of the answers can be performed at real-time, and indicate how semantic repositories can be exploited for specifying the entities of interest and for providing further information about the identified entities.

The initial application within iMarine of this "semantic post-processing" service used FLOD. Figure 3 shows a screen dump of the results for the query `tuna` over a deployment (as a portlet) in an infrastructure where the underlying system is *gcube search* [Simeoni et al., 2007] and the knowledge base is FLOD. The approach presented in this paper has improved this service from various perspectives: more entities can be identified in the results; the system is able to provide more complete information about the identified entities, etc.



Semantically Enriched Results

Query: tuna
In Collections: FIGIS

Mined Entities

⊟ FAOCountry(24)
Republic of ..(1)
Viet nam(1)
Venezuela(2)
Yugoslavia(2)
Senegal(1)
showAll

⊟ Species(8)
eastern Paci..(1)
yellowtail a..(1)
Ara(1)
pantropical ..(1)
Indo-Pacific..(1)
showAll

⊟ WaterAreas(3)
Mediterranea..(1)
Atlantic(2)
Pacific(11)

Object **Metadata**

Thunnus albacares (Bonnaterre, 1788) - Fact sheet

Yellowfin **tuna**... (Venezuela),Ca bo Vang (Viet nam),**Tuna** zutoperka (Yugoslavia)...There are important yellowfin **tuna** fisheries throughout tropical and subtropical seas. The most... major surface fishing techniques for yellowfin **tuna** in the Pacific, even though this method is... **tuna** in the Pacific is nearly contin...show all

Object **Metadata**

Thunnus obesus (Lowe, 1839) - Fact sheet

Bigeye **tuna**... skipjack and yellowfin **tuna** . The total catch reported for this species to FAO for 1999 was... **tuna** fishery was dominated by Japanese fleets up to the end of the sixties, but subsequently... 60% of the catch in the late seventies. The most important fishing technique is Industrial **tuna**...),Opastun (Poland)...show all

Object **Metadata**

Thunnus thynnus (Linnaeus, 1758) - Fact sheet

Atlantic bluefin **tuna**... fisheries documented are Mediterranean trap

Figure 3: Examples of semantic post-processing of search results within gcube

For Enabling Complex Query Services over Integrated Data. `MarineTLO` can be used as the schema for setting up integrated repositories that offer more complex query services, which cannot be supported by the individual underlying sources. In general, there are two main approaches for building and querying such repositories: the materialized integration approach (or *warehouse* approach), and the virtual integration (or *mediator*) approach (both are described in Section 4). The key point is that in both cases a schema is needed; `MarineTLO` can serve this requirement.

3 `MarineTLO`-based Integration

3.1 Design Principles

`MarineTLO` is not supposed to be the single ontology covering the entirety of what exists. It aims at being a *global core model* that i) covers with suitable abstractions the domains under consideration to enable the most fundamental queries, ii) can be extended to any level of detail on demand, and iii) can adequately map and integrate data originating from distinct sources, in a style similar to other related domains [Doerr et al., 2003][Cangemi et al., 2002]. Figure 4 drafts the intended architecture of knowledge models.

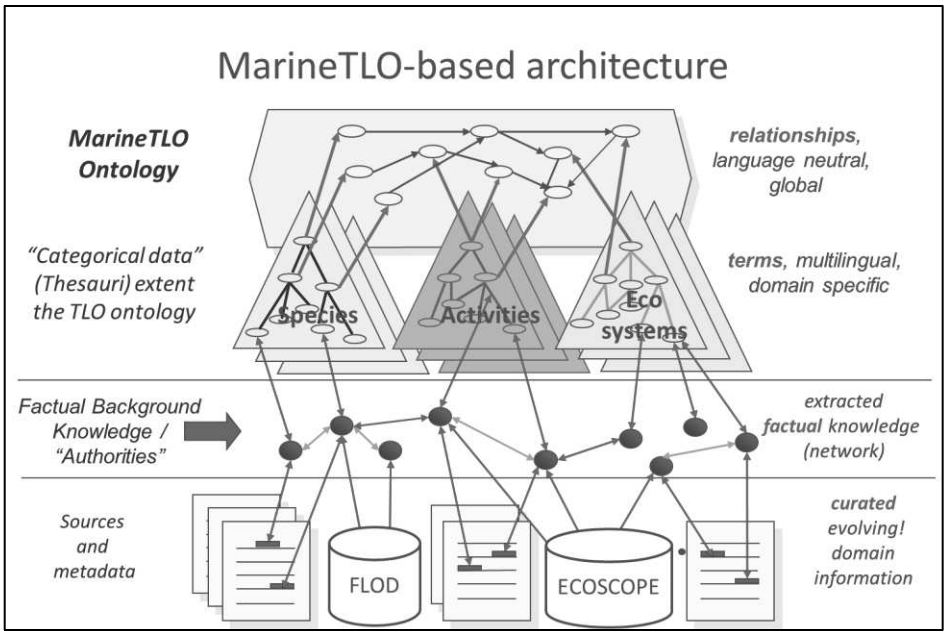

MarineTLO-based architecture
MarineTLO Ontology
relationships, language neutral, global
"Categorical data" (Thesauri) extent the TLO ontology
Species
Activities
Eco systems
terms, multilingual, domain specific
Factual Background Knowledge / "Authorities"
extracted factual knowledge (network)
Sources and metadata
FLOD
ECOSCOPE
curated evolving! domain information

Figure 4: MarineTLO-based architecture

Note that the adoption of a single and coherent core conceptual model has two main benefits: *(a)* reduced effort for improving and evolving it, since the focus is given on one model rather than many [Ibrahim and Pyster, 2004], and *(b)* reduced effort for constructing mappings, since this approach avoids the inevitable combinatorial explosion and complexities that result from pair-wise mappings between individual metadata formats and/or ontologies [Doerr et al., 2003].

Since the marine domain is complex and multiple views or projections should be supported for inference, the **MarineTLO** makes use of (i) categorical and cross-categorical relations as logical derivation of classes and properties of the selected sources, (ii) categories of classes (meta-classes) which support certain type of inference about classes in a way analogous to how classes support certain types of inference about instances and enable the assignment of attribute values to a class. Attention has been given also to the design of **MarineTLO** for preserving *monotonicity*. Since the primary role of **MarineTLO** is the meaningful integration of information in an OpenWorld, it aims to be monotonic in the sense of Domain Theory. That is, the existing constructs and the deductions made from them should remain valid and well-formed, even as new constructs are added to the **MarineTLO**. A particular consequence of this principle is that no class is declared as complement of a sibling concept under a common direct superclass.

Competitive Models. Although many organizations keep marine data, these data are organized based on the needs and activities of the particular organizations. Darwin Core offers a glossary of terms intended to facilitate the sharing of biodiversity information. The philosophy for the development of Darwin Core [Madin et al., 2007][TDWG, 2004][Wilson, 2009][Wieczorek et al., 2012], which intends to keep the standard as simple and open as possible and to develop terms only when there is demand for sharing, is not sufficient. Specifically, the terms are organized into nine categories, often referred to as classes, six of which cover broad aspects of the biodiversity domain (event, location, geological context, occurrence, taxon, and identification). The remaining categories cover relationships to other resources, measurements, and generic information about records. Especially for the record level, Darwin Core recommends the use of a number of terms from Dublin Core (type, modified, language, rights, rights holder, access rights, bibliographic citation, references). Darwin Core was designed to be minimal (only terms shared in common by natural history collections) and flat (no relational structure). A Darwin Core data record leaves the interpretation of the relationships between the whole record and one of its fields to the intuition of the human reader; in other words, it cannot be used to draw logical conclusions (e.g., consistency, equivalence) without human intervention. For instance, if a record level term `dc:ctype` equals to the term "physical object", then it is understood that the observed record documents a taxon, e.g., a mammal specimen; if on the other hand, the `dc:ctype` is missing, the record is understood to represent the taxon itself. Fields like "prey of" or "predator of" are missing. Also, causally related complex events (or composite events) cannot be described. Darwin Core can serve as a data entry questionnaire. One of the major drawbacks of Darwin Core in the Semantic Web context is the lack of a well-defined ontology, i.e., a formal definition of relationships between the kinds of entities ("core schema") of the biodiversity domain including its scientific processes. Such an ontology would define the relationships between concepts, such as biological entities, the events that document where and when they occurred, and the processes through which they are identified as being representative of a taxonomic concept. Without rigorous relationships between concepts and the properties that define them, connections between biodiversity data and related semantically rich information, such as literature and genomes, are difficult to traverse and no reasoning can be applied. This creates obstacles to cross-disciplinary semantic inquiry, such as in the Linked Data distributed data community.

Similar approached have been described also in different domains, i.e., in the medical domain. The Neuroweb Reference ontology [Colombo et al., 2009] is an upper level schema and enables a specific infrastructure to operate over different clinical repositories and to retrieve patients based on a set of specific criteria. This ontology acts as a vocabulary by encoding in a common way the phenotypes (the pathological condition of a patient) of different patients coming from different repositories. A similar approach is followed in the manufacturing domain where a design of top level ontologies is used to provide a ground term for enhancing the collaboration between different labors and partners. In [Mosca et al., 2009] a framework for the development of decision support systems for the engineering domain has been presented. The framework is based on a set of ontologies that describes all the properties of a product so that small and medium enterprises (SME) will be able to easily define the roles of the different labors in the lifecycle of the product (i.e., design, production, testing, etc.).

3.2 The MarineTLO Ontology

For the development and evolution of **MarineTLO** we adopted an iterative and incremental methodology comprising the following steps: (i) ontological analysis of underlying sources, (ii) design, (iii) implementation, and (iv) evaluation. For the implementation we used the OWL Web Ontology Language 2 [Hitzler et al., 2009], while for the needs of evaluation we used the notion of competence queries (described later in this paper). The full version of **MarineTLO**, as well as more information, is available at http://www.ics.forth.gr/isl/MarineTLO.

For the first version of **MarineTLO** we used the descriptions and the data of ECOSCOPE, FLOD and WoRMS sources. As new sources (i.e., DBpedia, FishBase) or new concepts emerged, we updated the **MarineTLO** ontology appropriately. The following list describes its evolution history.

- Version 1 contained 17 classes and 8 unique properties and was designed to capture the scientific names of species and information about predator-prey relationship, coming from ECOSCOPE, FLOD and WoRMS.
- Version 2 contained 57 classes and 22 unique properties. This version captured the same concepts as the previous version, as well as information about WoRMS classification, competitors, images, and species codes coming from FAO and IRD organizations. Furthermore, this version captured specific information about fishes from DBpedia (i.e., scientific and common name of species, images, general description and others).
- Version 3 contained 57 classes and 25 unique properties. This version captured the same concepts as version 2 and furthermore information about water areas, countries, ecosystems, exclusive economic zones, fishing gears, fishing vessels and common names of species. In addition, this version integrated information about the common names of marine species in different language from FishBase.
- Version 4 contained 127 classes and 81 properties. This version captured the concepts of the previous version, as well as information about catch and byCatch[9], biological parameters, statistical indicators (provided by IRD), and publications.

For the needs of the intended applications and the main underlying sources, an extension of the full version is being used. The current version of the extended ontology contains 151 classes and 116 properties. With the name "**MarineTLO**", we hereafter refer to this extension. It is organized in two abstraction levels: schema and metaschema. The metaschema aims at providing a method for classifying the schema level in meaningful abstractions, which can be exploited not only for expressing cross-categorical knowledge, but also for aiding the formulation of generic queries. Figure 5 shows the metaclasses (and how they are organized in a subClassOf hierarchy), and Figure 6 shows a part of the classes in the class level. Between the classes and the metaclasses there are instanceOf relationships (implemented as RDF typeOf relationships) which are omitted from the diagram. We use a set of prefixes to declare classes, metaclasses and properties between them. Particularly, we use the prefix BT for declaring the metaclasses (e.g., BT27_Species) and BC for declaring the classes (e.g., BC8_Actor). For the properties we are using the prefix LT for properties between metaclasses (e.g., LT8_usually_belongs_to), LC for properties between classes (e.g., LC13_is_carried_out_by), and LT for cross-categorical links (e.g., LX6_type_was_attributed_by).

[9] http://en.wikipedia.org/wiki/Bycatch

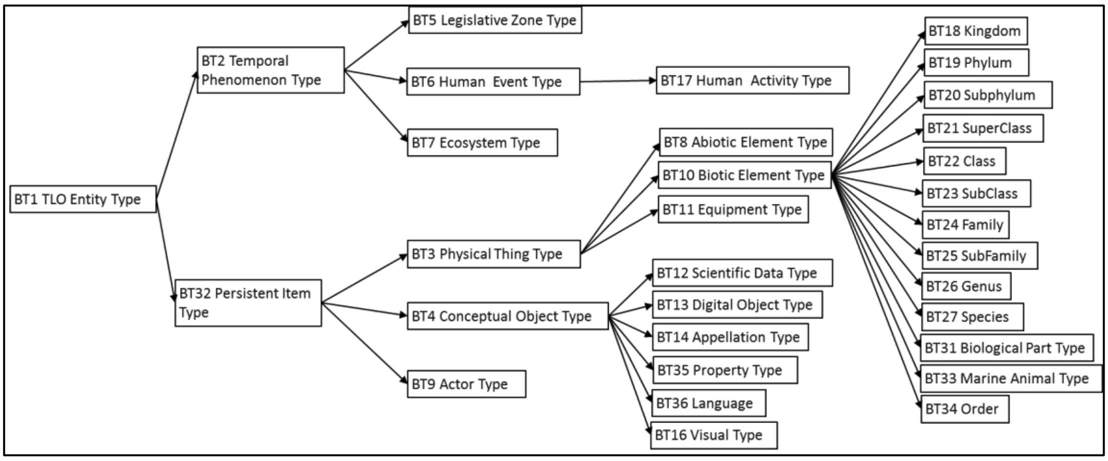


Figure 5: The meta-classes of `MarineTLO`

The example shown in Figure 1 illustrates how pieces of information that come from different sources and concern one particular species, namely `Thunnus Albacares`, are assembled. The labels of the frames indicate the used sources. A more detailed example can be seen in Figure 7. The upper part of Figure 7 depicts the scientific name assignment and the lower part shows the taxonomic classification of `Thunnus Albacares`. Rectangles are used to denote the class name and its corresponding instance (for example, *ns:thunnus_albacares* is an instance of the class **BT27 Species**). In some cases, instead of creating new (or even arbitrary) URIs we are using blank nodes (e.g. the instance of **BT46_Scientific_Name_Assignment**). In those cases, we are using the notation *_:bn* to declare that this particular node is a blank node. Edges are used to denote the properties. Figure 8 shows the same information expressed in RDF. It is evident from this figure that we overcome the issues that arise with new resources; instead of adopting a particular policy for new resources and defining specific namespaces for publishing them, we model them as blank nodes. For example, it is not required to publish a specific URI for the scientific name assignment of `Thunnus Albacares`, however the information connected to it (i.e., the actual name, the year, the authoritative information) are more than useful.

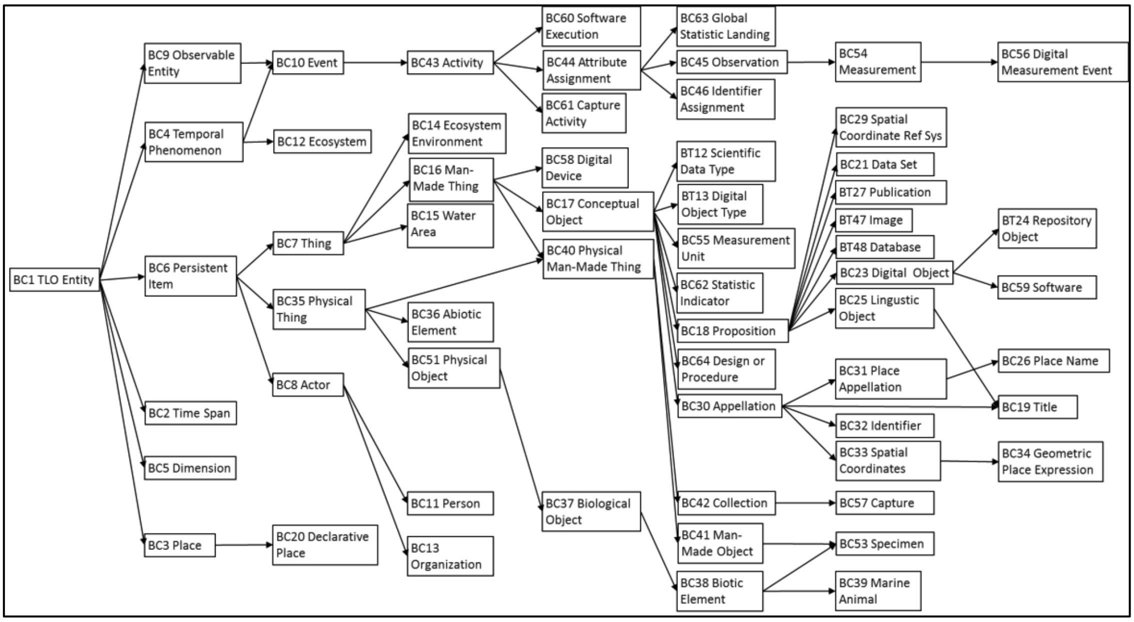


Figure 6: Part of the classes of `MarineTLO`

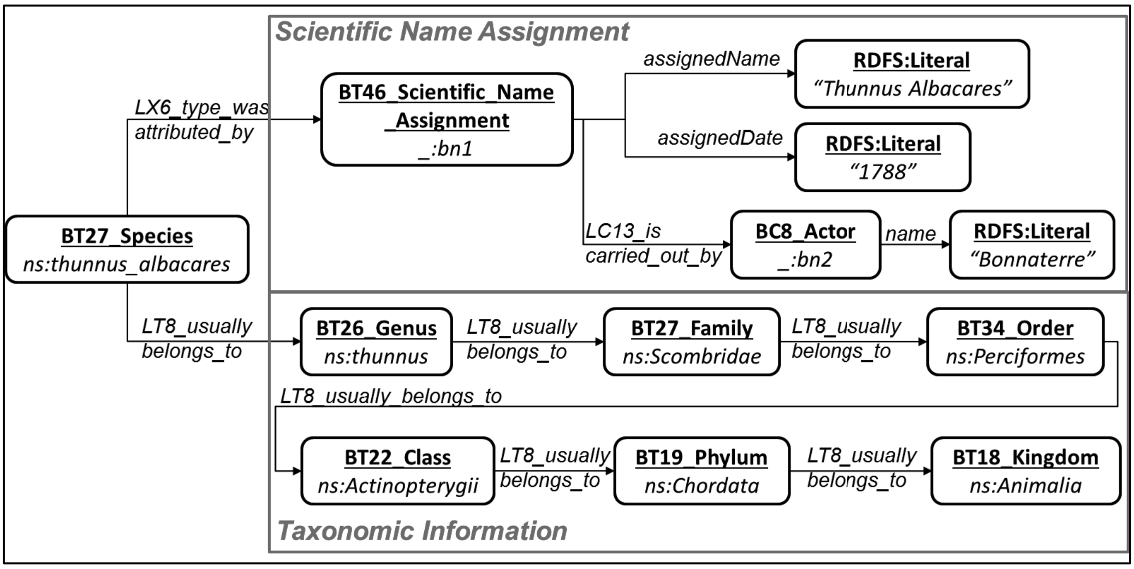


Figure 7: The scientific name assignment and taxonomic information of `Thunnus Albacares`

```
<rdf:RDF xmlns:mtlo="http://www.ics.forth.gr/isl/MarineTLO/v4/marinetlo.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <mtlo:BT27_Species rdf:about="http://url/thunnus_albacares">
        <mtlo:LX6_assigned_attribute_to_type>
            <mtlo:BC46_1_Scientific_Name_Assignment>
                <mtlo:assignedName> Thunnus Albacares </mtlo:assignedName>
                <mtlo:LC13_is_carried_out_by>
                    <mtlo:BC8_Actor>
                        <mtlo:name> Bonnatere </mtlo:name>
                    </mtlo:BC8_Actor>
                </mtlo:LC13_is_carried_out_by>
                <mtlo:assignedDate> 1788 </mtlo:assignedDate>
            </mtlo:BC46_1_Scientific_Name_Assignment>
        </mtlo:LX6_assigned_attribute_to_type>
        <mtlo:LT8_usually_belongs_to>
            <mtlo:BT26_Genus rdf:about="http://www.ics.forth.gr/isl/thunnus">
                <mtlo:LT8_usually_belongs_to>
                    <mtlo:BT27_Familiy rdf:about="http://www.ics.forth.gr/isl/scombridae">
                        <mtlo:LT8_usually_belongs_to>
                            <mtlo:BT34_Order rdf:about="http://www.ics.forth.gr/isl/perciformes">
                                <mtlo:LT8_usually_belongs_to>
                                    <mtlo:BT22_Class rdf:about="http://www.example/actinopterygii">
                                        <mtlo:LT8_usually_belongs_to>
                                            <mtlo:BT19_Phylum rdf:about="http://url/chordata">
                                                <mtlo:LT8_usually_belongs_to>
                                                    <mtlo:BT18_Kingdom rdf:about="http://url/animalia"/>
                                                </mtlo:LT8_usually_belongs_to>
                                            </mtlo:BT19_Phylum>
                                        </mtlo:LT8_usually_belongs_to>
                                    </mtlo:BT22_Class>
                                </mtlo:LT8_usually_belongs_to>
                            </mtlo:BT34_Order>
                        </mtlo:LT8_usually_belongs_to>
                    </mtlo:BT27_Familiy>
                </mtlo:LT8_usually_belongs_to>
            </mtlo:BT26_Genus>
        </mtlo:LT8_usually_belongs_to>
    </mtlo:BT27_Species>
</rdf:RDF>
```

Figure 8. The scientific name assignment and taxonomy of `Thunnus Albacares` in RDF

4 On Constructing MarineTLO-based Warehouses

4.1 Integration Approaches

In general, there are two main integration approaches for such repositories: the materialized integration approach (or warehouse approach), and the virtual integration (or mediator) approach.

Materialized Approach The materialized approach relies on a central repository (RDF triplestore in our case) where all data are to be stored. *Mappings* (in the broad sense) are exploited to *extract* information from data sources, to *transform* it to the target model and then to *store* it at the central repository. Over such a repository more complex queries can be answered.

It is good practice not to modify extracted information after each transformation except for the use of common identifiers. Rather, any need for updating individual information is covered by requesting source providers to make updated sources available. There are some important issues that should be taken into account for designing and maintaining a data warehouse. Firstly (design phase), the information from each source that is going to be used should be selected. Specific views over the sources should be chosen in order to be materialized. Next (maintenance phase), issues should be tackled concerning the warehouse initial population by the source data and the update of the data when sources are refreshed. The notion of *graph spaces* of RDF triplestores can alleviate this problem. The great advantage of materialized integration is its flexibility in transformation logic, the decoupling of the release management of the integrated resource from the management cycles of the sources, and the decoupling of access load from the source servers. The method that we will present can be used for setting up such repositories.

Moreover, the availability of a materialized repository is beneficial for applying entity matching techniques (e.g., see [Noessner et al., 2010]) since more information about the domain entities is available, while the application of these techniques is significantly faster than applying them without having a repository (i.e., by fetching information from the network).

Virtual Approach. On the other hand, the virtual integration approach does not rely on a central repository but leaves the data in the original sources. Mappings (in the broad sense) are exploited to enable *query translation* from one model to another. Then, data from disparate sources is *combined* and returned to the user. The mediator (a.k.a. integrator) performs the following actions. First, it receives a query formulated in terms of the unified model/schema and decomposes the query into *sub-queries*. These queries are addressed to specific data sources. This decomposition is based on the mappings generated between the unified model and the source models, which play an important role in sub-queries' execution plan optimization. Finally, the sub-queries are sent to the wrappers of the individual sources, which transform them into queries over the sources. The results of these sub-queries are sent back to the mediator. At this point, the answers are merged and returned to the user. Besides the possibility of asking queries, the mediator has no control over the individual sources. The great advantage (but in some cases disadvantage) of virtual integration is the real-time reflection of source updates in integrated access. As regards system's complexity (complexity of query rewriting and of execution planning), this depends on the structural complexity of the global view and the differences between this view and that of the underlying models. The higher complexity of the system (and the quality of service demands on the sources) is only justified if immediate access to updates is indeed required.

4.2 The Marine-TLO based Warehouse

We have been investigating the materialized (warehouse) approach. Specifically, we coded the **`MarineTLO`** ontology using OWL 2 and set up a repository using two different triplestores which are described in the sequel. Apart from **`MarineTLO`**, the repository contains the entire FLOD fetched from its SPARQL endpoint, the entire ECOSCOPE downloaded by its web site, and parts of WoRMS extracted using SDDS and TLO Wrapper, part of FishBase extracted using FishBaseReaper and part of DBpedia fetched from its public endpoint.[10] Figure 9 displays the current **`MarineTLO`**-based warehouse.

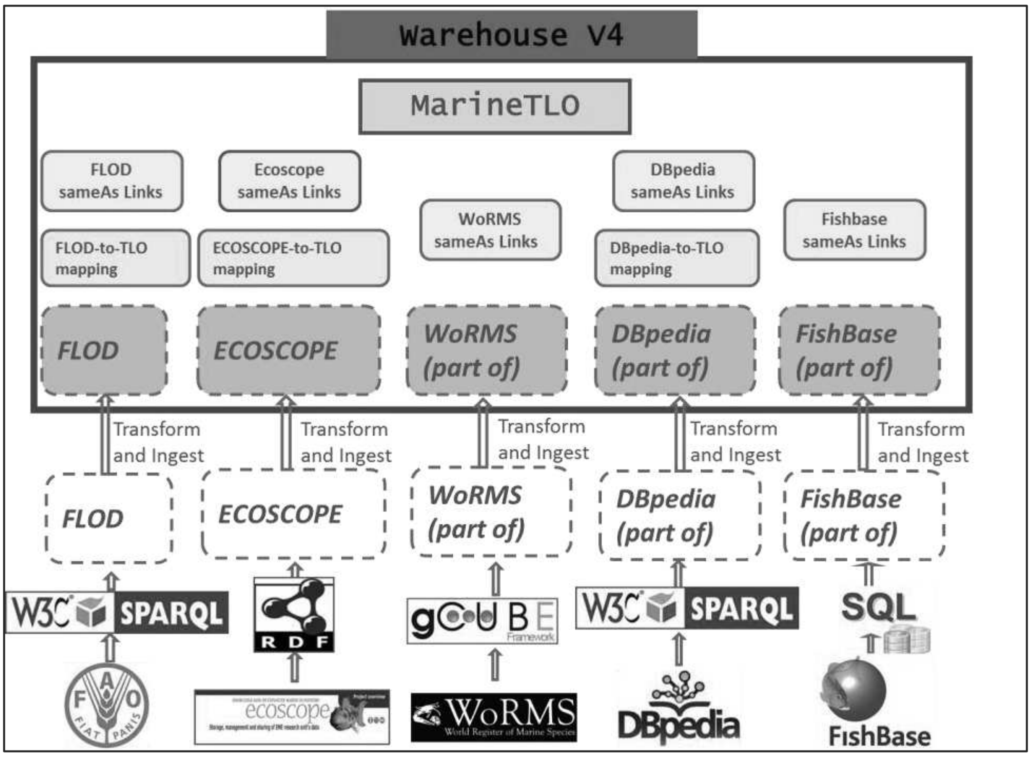


Figure 9: The current version of `MarineTLO`-based warehouse

Used Triplestores. We have comparatively evaluated two different triplestores: OWLIM-Lite[11] and OpenLink Virtuoso[12]. The first has been designed for medium data volumes (less than 100 million statements). It contains a persistence layer, however reasoning and query

[10] SDDS, TLOWrapper and FishBaseReaper will be described in the subsequent sections.
[11] http://owlim.ontotext.com/
[12] http://virtuoso.openlinksw.com/

evaluation are being performed entirely in main memory. On the other hand, Virtuoso supports backward chaining reasoning, meaning that it does not materialize all inferred facts, but computes them at query level. Practically, this means that transitive relations (i.e., rdfs:subClassof, rdfs:subPropertyOf, owl:equivalentClass etc.) are not physically stored in the knowledge base, but they are added to the result set during query answering. In §5.2 we comparatively evaluate these triplestores.

4.3 MarineTLO-based Mappings

For extracting information from the underlying sources and associating them with **MarineTLO**-based descriptions, we use *mappings*. In general, what we call mapping comprises: extensions to the metaschema, extensions to the schema, rdfs:subClassOf, rdf:subPropertyOf, owl:equivalentClass relationships between the elements of MarineTLO and the schema at hand, plus some inference rules. Below, we sketch the defined mappings. For instance, the ECOSCOPE-2-**MarineTLO** mapping consists of rdfs:subClassOf and rdfs:subPropertyOf like those shown in Figure 10. The DBpedia-2-**MarineTLO** mapping contains analogous relationships. Note that we do not use any mappings for FishBase and WoRMS, since the software tools we are using for transforming data from these sources (see §4.4), directly produce instances which are expressed with respect to **MarineTLO** descriptions.

```
(tlo:EcoscopeSpecies, rdfs:subClassOf, tlo:TLOSpecies)
(eco:fish, rdfs:subClassOf, tlo:EcoscopeSpecies)
(eco:is_predator_of, rdfs:subPropertyOf, tlo:usuallyIsPredatorOf)
(eco:is_prey_of, rdfs:subPropertyOf, tlo:usuallyIsPreyOf)
(eco:biotic_component_of, rdfs:subPropertyOf, tlo:usuallyIsComponentOf)
(eco:used_data_source, rdfs:subPropertyOf, tlo:isReferencedBy)
```

Figure 10: Mappings between Ecoscope and MarineTLO

However, in FLOD any resource is an instance of CodedEntity, and for distinguishing a vessel (e.g. vessel_289) from a species (e.g. thunnus albacares) we need to go one step further and look at its code. For instance, we can distinguish FAOSpecies as follows:

FAOSpecies={x | CodedEntity(x) and (∃y isClassifiedByCode(x,y) and SpeciesCode(y))}

The required mapping can be defined using owl:Restriction. This is supported by OWLIM, but it is not supported by Virtuoso. For the latter, we can express this mapping through the following SPARQL INSERT query:

```
INSERT {
 ?x rdf:type < http://www.ics.forth.gr/isl/MarineTLO/v4/marinetlo.owl#FLOD_Species> }
WHERE {
 ?x rdf:type <http://www.fao.org/figis/flod/onto/codedentity.owl#CodedEntity> .
 ?x <http://www.fao.org/figis/flod/onto/codedentityclassification.owl#isClassfiedByCode> ?y .
 ?y rdf:type <http://www.fao.org/figis/flod/onto/linneanspecies.owl#SpeciesCode> }
```

Figure 11: Expressing OWL Restriction as a SPARQL Insert query

4.4 Software for Transforming Instances from Heterogeneous Sources to MarineTLO

In some cases the data of the underlying sources (i.e., FishBase, WoRMS) is described in different formats. In these cases we have to transform the data to RDF. For this reason, we have implemented particular tools for carrying out these transformations, which are described below.

FishBaseReaper. FishBase contains information in relational databases. We have implemented a tool, called FishBaseReaper, that extracts data from these databases and transforms them to RDF instances according to MarineTLO. The tool takes as input a list of concepts of interest (scientific name, ecosystems, bibliographic information), connects to the relational databases of FishBase, and produces as output files (in N-Triples format) that contain the extracted information with respect to **`MarineTLO`** classes and properties. There is also the option of using a specific URI prefix for all the extracted entities, or different URIs prefixed according to the type of each entity (i.e., use of the namespace http://www.ics.forth.gr/isl/MarineTLO/Species for marine species and http://www.ics.fort.gr/isl/MarineTLO/PlaceName for countries).

The Species Data Discovery Service (SDDS). The Species Data Discovery Service [Candela et al., 2014-1], SDDS for short, is a gCube service [Candela et al., 2008] specifically conceived to provide its users with a single access point to species data, both occurrence data and nomenclature data, hosted by a number of distributed databases. It is a plugin-based mediator service for key species data databases including GBIF and OBIS for occurrence data, Catalogue of Life, OBIS, Interim Register of Marine and non-marine Genera (IRMNG), ITIS, NCBI, and WoRMS for nomenclature data.

Given that the set of databases that the service interfaces with is open, it is sufficient to implement a dedicated plugin to (i) transforming the query for species data expressed in a domain specific query language into an equivalent query supported by the specific data provider, (ii) submit the transformed query to the data provider, and (iii) transform the results into the common data format envisaged by the SDDS. Details on the domain specific query language and the unifying data format characterizing SDDS are discussed in [Candela et al., 2014-2]. Here it is worth to highlight that (i) the query language is essentially based on species names (scientific and common names) and supports directives for automatic query expansion based on known species names, (ii) the resulting records are annotated with details on data provenance produced accordingly to the policies of each data source, and (iii) the resulting records can be produced according to known formats and standards including DarwinCore [Wieszorec et al., 2012]. SDDS is offered both via a web-based user interface and a web-based API for programmatic access.

MarineTLO Wrapper. We implemented a tool that uses `SDDS` API and transforms the fetched information into descriptions structured according to the **`MarineTLO`**. Its functionality is performed in two phases: during the first phase, it takes as input a list of scientific names to be retrieved and the data sources to be searched and submits the query to `SDDS`. The output is a Darwin Core Archive (DwC-A) file, containing the classifications of the given input. During the second phase the tool parses the DwC-A archives and produces the descriptions according to **`MarineTLO.`** Finally, the data are exported in RDF or NTRIPLES format.

4.5 Transformation and Entity Matching Rules

Transformation Rules. Some data can be stored into the warehouse as they are fetched, while others may need to be transformed. For example, a literal may need to be transformed into a URI, or to be split for using its constituents, or an intermediate node may need to be created (e.g. instead of (x,hasName,y) to have (x,hasNameAssignement,z),(z,name,y),(z,date,d). In order to handle such cases, we created a number of transformation rules that are applied to the fetched data, before its ingestion to the warehouse. The following table shows some of the transformation rules we applied for the warehouse; for each transformation rule, the upper row shows the initial expression and the lower one shows the transformed expression.

Table 1: Transformation Rules

1	The string “Bonnattere, 1788” for the entry Thunnus Albacares in Worms
	<ns:thunnus_albacares> <mtlo:LX6_type_was_attributed_by> _:bn1 _:bn1 <mtlo:is_carried_out_by> _:bn2 _:bn2 <mtlo:name> “Bonnaterre” _:bn1 <mtlo:assignedDate> “1788”
2	The value (?val) of the property skos:prefLabel for every instance (?inst) of the class http://www.ecoscope.org/ontologies/ecosystems_def/fish
	?inst <mtlo:type_was attributed_by> _:bn1 _:bn1 <mtlo:assignedName> ?val
3	?x <mtlo:usually_is_predator_of> ?z ?y <mtlo:usually_is_predator_of> ?z ?x != ?y
	?x <mtlo:usually_is_competitor_of> ?z

SILK Rules. We created same-as relationships between the entities using an entity matching tool called SILK link[13]. Specifically, i) we inspected the connectivity between the sources, ii) we formulated a number of silk same-as rules, iii) we applied these rules to the sources and iv) we imported the produced same-as relationships into the warehouse. The reason for applying these rules is that they increase the connectivity of the resulting warehouse (this aspect will be discussed in detail later). Table 2 shows some indicative SILK rules[14].

Table 2: Rules for creating owl:sameAs links using SILK

#	Value from Source A	Value from Source B	Example

1	`wormsId` attribute from ECOSCOPE	Integer part of `hasTaxonID` attribute from WoRMS	`wormsId: 127027`
			`hasTaxonId: WoRMS:127027`

2	`prefLabel` attribute (in lower case) from ECOSCOPE	`label` attribute (in latin) from FLOD	`prefLabel: Thunnus albacares`
			`label: “thunnus albacares”@la`

3	`altLabel` attribute from ECOSCOPE	`label` attribute from FLOD	`altLabel: “yellowfin tuna”@en`
			`label: “yellowfin tuna”@en`

4	`prefLabel` attribute from ECOSCOPE	`binomial` attribute from DBpedia	`prefLabel: Thunnus albacares`
			`binomial: Thunnus albacares`

5	`label` attribute tokenized using “ from FLOD	`binomial` attribute (to lower case) from DBpedia	`label: “thunnus albacares”@la`
			`binomial: Thunnus albacares`

6	`label` attribute tokenized using “ from FLOD	WoRMS species URI after removing the namespace, the taxon id, replacing underscores ‘_’ with spaces and converting it to lower case	`label: “thunnus albacares”@la`
			`URI: http://www.marinespecies.org/`
			`entities/ WoRMS:127027/`
			`Thunnus_Albacares`

7	`binomial` attribute from DBpedia	WoRMS species URI after removing the namespace, the taxon id, replacing underscores	`binomial: Thunnus albacares`
			`URI: http://www.marinespecies.org/`
			`entities/ WoRMS:127027/`

[13] http://wifo5-03.informatik.uni-mannheim.de/bizer/silk/

[14] The full list of the SILK rules that are being used for constructing the **`MarineTLO`**-based warehouse can be found at **`MarineTLO`** website http://www.ics.forth.gr/isl/MarineTLO/.

		'_' with spaces and converting it to lower case	`Thunnus_Albacares`

8	`binomial` attribute from DBpedia	FishBase species URI after removing the namespace and replacing underscores '_' with spaces	`Binomial: Thunnus albacares`
			`URI: http://www.fishbase.org/entity#thunnus_albacares`

9	WoRMS species URI after removing the namespace, the taxon id, replacing underscores '_' with spaces and converting it to lower case	FishBase species URI after removing the namespace and replacing underscores '_' with spaces	`URI: http://www.marinespecies.org/entities/ WoRMS:127027/Thunnus_Albacares`
			`URI: http://www.fishbase.org/entity#thunnus_albacares`

10	`label` attribute tokenized using " from FLOD	FishBase species URI after removing the namespace and replacing underscores '_' with spaces	`label: ‟thunnus albacares”@la`
			`URI: http://www.fishbase.org/entity#thunnus_albacares`

11	`prefLabel` attribute from ECOSCOPE	FishBase species URI after removing the namespace and replacing underscores '_' with spaces	`prefLabel: Thunnus albacares`
			`URI: http://www.fishbase.org/entity#thunnus_albacares`

4.6 Assessing the Connectivity of Information

From this activity, we observed that the data fetched from the sources are in many cases problematic (consistency problems, duplicates, wrong values). We noticed that placing them together in a warehouse makes easier the identification of such errors. Furthermore, the availability of the warehouse enables defining `sameAs` connections by exploiting transitively induced equivalences, and can be produced by exploiting SILK matching rules, like the ones described in Section 4.5. In any case, the inspection of the repository for detecting the missing connections that are required for satisfying the needs of the competence queries is an important requirement. To this end, we have devised some metrics for quantifying the value of the warehouse and the value (contribution) of each source to the warehouse. These metrics are described in detail in [Tzitzikas et al., 2014-1], while a vocabulary that allows the representation, exchange and querying of such measurements is described in [Mountantonakis et al., 2014].

4.7 Handling the Provenance

After ingesting data coming from several sources in the warehouse we can still identify their provenance. We support four levels of provenance: (a) at *conceptual modeling level*, (b) at *URIs and values level*, (c) at *triple level*, and (d) at *query level*.

As regards (a), **`MarineTLO`** models the provenance of species names, codes etc. (who and when assigned them). Therefore, there is no need for adopting any other model for capturing provenance (e.g. OPM [Moreau et al., 2011]). As regards (b), we adopt the namespace mechanism for reflecting the source of origin of an individual. For example, the URI `http://www.fishbase.org/entity#thunnus_albacares` denotes that this URI has been derived from FishBase. Furthermore, during the construction of the warehouse there is the option of applying a uniform notation @source to literals (where source can be FLOD, ECOSCOPE, WoRMS, FishBase, DBpedia). As regards (c), we store the triples from each source in a separate graph space. This is useful not only for provenance reasons, but also for refreshing parts of the warehouse, as well as for computing the connectivity metrics that have been described previously. Finally, as regards (d), we have implemented a framework, called

MatWare that is described below, which offers a query rewriting functionality that exploits the graph spaces, and returns the sources that contributed to the query results. The provenance at *URIs and values level* is just an alternative way of modelling provenance. We used this approach for modeling the scientific names of the species in the first versions of the warehouse. In subsequent versions we used the triple level provenance which allows storing data coming from different sources using different graph spaces. In this case the provenance of all the contents of a source is being derived from the graph space. An extensive discussion about provenance in **`MarineTLO`**-based warehouses can be found at [Tzitzikas et al., 2014-2].

4.8 Connecting the Pieces

We developed a framework, called *MatWare* [Tzitzikas et al., 2014-2], that automates the construction, maintenance and quantitative evaluation of warehouses based on **`MarineTLO`**. Figure 12 illustrates the warehouse construction and evolution process, as supported by *MatWare*.

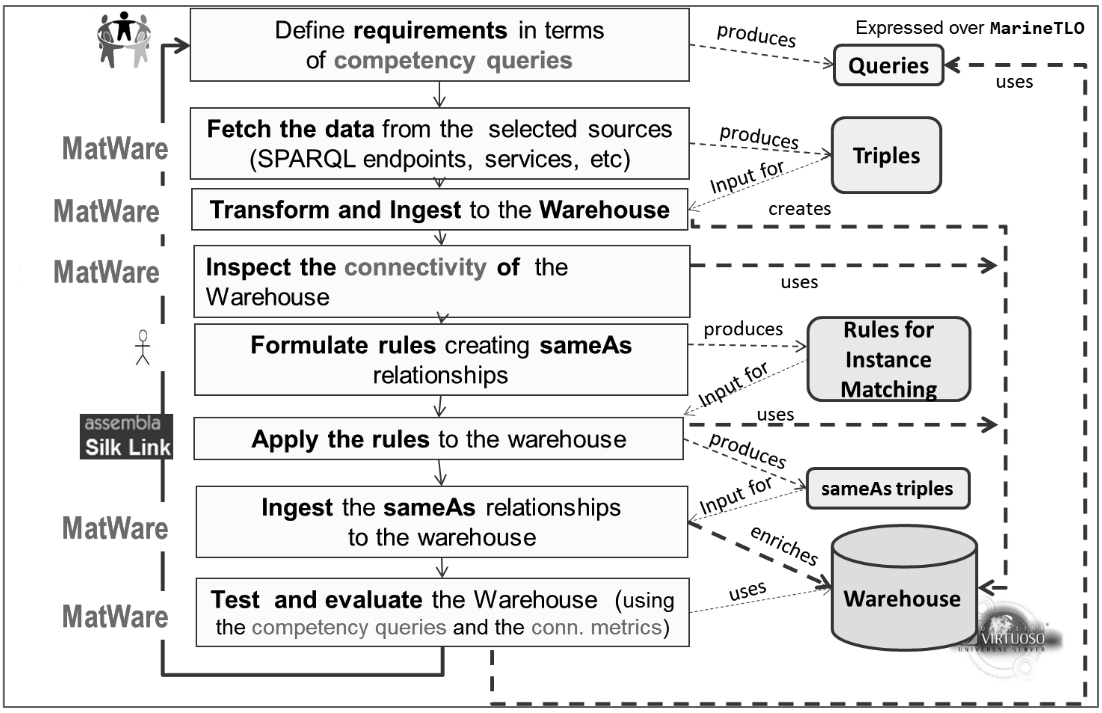

Define requirements in terms of competency queries
produces
Expressed over MarineTLO
Queries
uses
MatWare
Fetch the data from the selected sources (SPARQL endpoints, services, etc)
produces
Triples
Input for
MatWare
Transform and Ingest to the Warehouse
creates
MatWare
Inspect the connectivity of the Warehouse
uses
Formulate rules creating sameAs relationships
produces
Rules for Instance Matching
Input for
assembla Silk Link
Apply the rules to the warehouse
uses
produces
sameAs triples
MatWare
Ingest the sameAs relationships to the warehouse
Input for
enriches
Warehouse
MatWare
Test and evaluate the Warehouse (using the competency queries and the conn. metrics)
uses
VIRTUOSO UNIVERSAL SERVER

Figure 12: The warehouse construction and evolution process supported by *MatWare*.

5 Evaluation and Current Uses of the `MarineTLO`-based Warehouse

5.1 Evaluating the `MarineTLO`-based Warehouse through Competence Queries

For evaluating the structuring of **`MarineTLO`** and the process used for creating the **`MarineTLO`**-based repository, we had to investigate whether they offer the required abstractions for (a) adequately modeling the domain, (b) hosting information coming from different sources, and (c) allowing answering useful queries which cannot be answered by the individual underlying sources. For the latter, we formed a collection of competence queries in collaboration with the involved partners and their priorities. Table 3 shows some fundamental concepts that exist in the competence queries. The columns at the right show which of them are answerable by the underlying sources. We should note that the real *competence queries* include queries that combine more than one of the listed concepts, like the complex query that was described in the introduction of this paper, e.g., "*I want the taxonomic information of the predators of a particular species with the different codes that the organizations use to refer to them*". This particular query requires sources that contain information about: (a) scientific names, (b) species taxonomy, (c) predators, and (d) codes (usually provided by FLOD/FAO). Such queries cannot be answered by any particular source (which is also evident for the particular example from the contents of Table 3), but can now be answered by the **`MarineTLO`**-based Warehouse that contains the required sources. This is the concrete evidence of the benefits offered by the integrated model. A table showing the competence queries we used and their corresponding SPARQL expression can be found at **`MarineTLO`** website.

Table 3: Basic Queries

Concepts	ECOSCOPE	FLOD	WoRMS	DBpedia	FishBase
Species Taxonomy			✓	✓	✓
Scientific/Common Names	✓	✓	✓	✓	✓
Authorships			✓	✓	✓
Predators	✓				
Ecosystems	✓				
Countries					✓
Water Areas		✓			✓
Vessels	✓	✓			
Gears	✓	✓			
EEZ[15]		✓			
Bilbiography	✓		✓		✓
Statistical Indicators					✓

5.2 Comparison of Different Triplestores

Table 4 shows the sizes in triples of the contents of the OWLIM and Virtuoso repositories for the first version of the **`MarineTLO`** and its corresponding sources. The first contains in total 10.8 million triples. This number includes the inferred triples, since this repository materialized them. The creation of the repository from scratch (by loading the corresponding files) takes around 30 minutes. The time is short because the used edition of OWLIM loads everything in main memory. In Virtuoso the number of triples is significantly lower, because the inferred triples are not stored. The creation in this case takes 4h and 20 minutes[16]. The execution of the INSERT query (needed for FLOD) created about 32,000 triples, i.e., the FLOD-originated triples from 2,148,128 increased to 2,180,678.

Table 4: MarineTLO-based warehouses using OWLIM and Virtuoso

KB Part	# triples in OWLIM	# triples in Virtuoso
`MarineTLO`	277	58
FLOD	9,092,087	2,148,128
ECOSCOPE	170,980	84,184
WoRMS	70,174	9,552
FLOD-2-TLO mappings	180	15
ECOSCOPE-2-TLO mappings	205	11

[15] Exclusive Economic Zone

[16] Experiments done using a QuadCore Linux machine with 4 GB RAM with OWLIM version 4.2 and Virtuoso opensource version 6.1

WoRMS-2-TLO mappings	180	8
Total	10,822,758	2,241,956

To test query performance, we used queries provided by the iMarine partners. The average time in OWLIM ranged from 62ms to 8.8 seconds, while in Virtuoso from 31ms to 3.4 seconds. We observe that Virtuoso is faster despite the fact that OWLIM keeps everything in main memory, while Virtuoso does not necessarily do so. In general, performance depends on the capabilities of the adopted triplestore used (for a comparative analysis see [Haslhofer,2011]).

5.2.1 The contents of the MarineTLO-based warehouse(-s)

Based on the above results, we decided to use Virtuoso for the subsequent versions of the warehouse. Similarly to the different versions of the **MarineTLO**, we released 4 different version of the warehouse. Each version contained the corresponding **MarineTLO** version and the required schema mappings, in addition to the following:

- Version 1: contents from FLOD, ECOSCOPE and WoRMS, about the scientific name and predators of species.
- Version 2: contents from FLOD, ECOSCOPE, WoRMS and DBpedia, about the same concepts of Version 1 (i.e., scientific names and predators) plus authorship information of species.
- Version 3: contents from FLOD, ECOSCOPE, WoRMS, FishBase and DBpedia about the same concepts of Version 2 plus common names of species, information about ecosystems, countries, water areas, vessels, gears and EEZ. After the Version 3 release we released another version (named Version 3+) having the same contents with Version 3, however, we used multiple graphspaces for storing data coming from different sources. This allowed us to track easily the provenance of the information in the warehouse (e.g., the fact that "*yellowfin tuna*" is an English common name of the species `thunnus albacares` is derived from WoRMS and FishBase).
- Version 4: contents from FLOD, ECOSCOPE, WoRMS, FishBase and DBpedia about the same concepts of Version 3, containing also information about bibliographic citations and statistical indicators.

Figure 13 shows the differences between the 4 versions of the **MarineTLO**-based warehouse, in terms of the number of triples, species, main concepts and used sources. The first plot (A) shows how the number of species has been increased from 10 thousand (in the first version of the warehouse) to 53 thousand (in the fourth version). The second plot (B) depicts the increment in the size of the triplestore. Data are described in the warehouse as triples (in the form of <*subject, predicate, object*>), so the plot depicts the number of triples for the different versions. Plot (C) shows the different concepts (i.e. scientific names, predators, vessels, etc.) which are included in the different version of the **MarineTLO**-based warehouse and the last one (D) illustrates the number of the underlying sources which are exploited in each version.

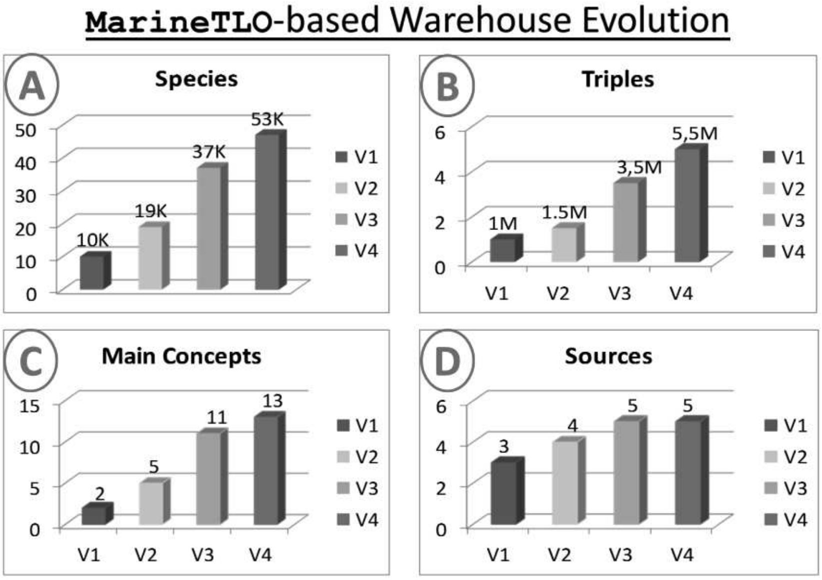


Figure 13: The evolution history of MarineTLO-based warehouse

5.3 Current uses of MarineTLO-based Warehouse

The **MarineTLO**-based warehouse is under constant evolution. At the time of writing, it contained information about 54 thousand species (i.e., scientific and common names, predators, bibliographic resources, ecosystems, water areas etc.). A SPARQL endpoint is available online[17]. Figure 14 shows the contents of the latest version of the **MarineTLO**-based warehouse.

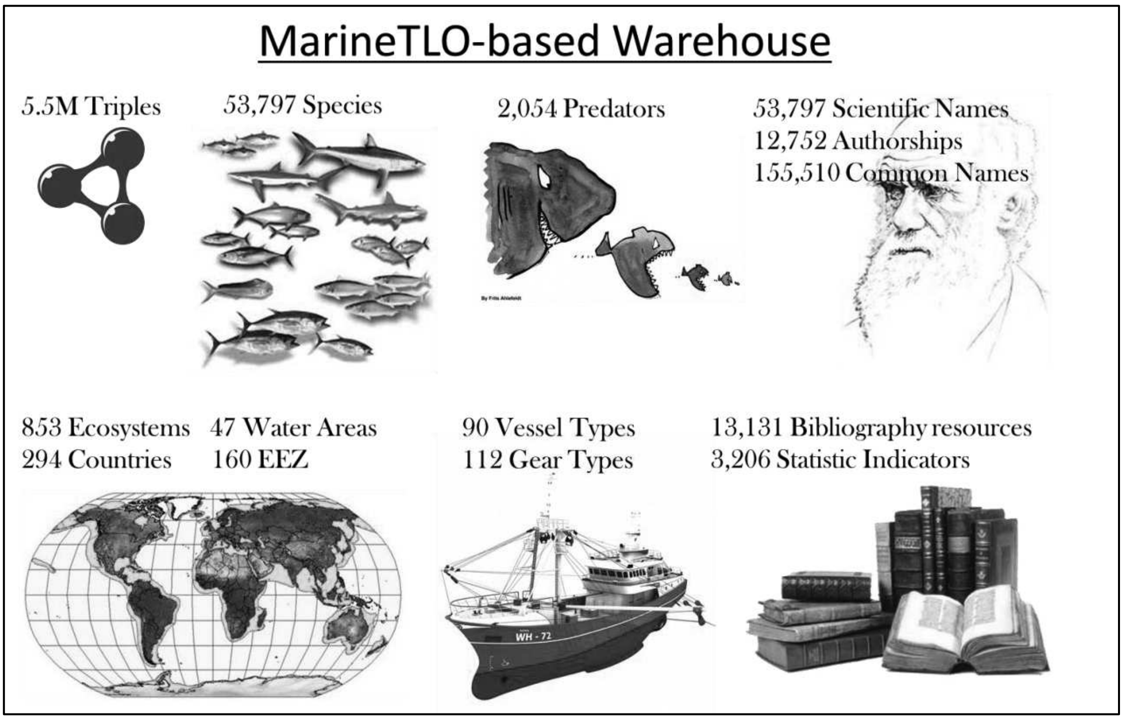


Figure 14: The contents of the MarineTLO-based warehouse (on July 2014)

[17] The warehouse can be accessed from https://i-marine.d4science.org/. Instructions for connecting and using is can be found at http://www.ics.forth.gr/isl/MarineTLO/files/AccessingMarineTLOBasedWarehouse.pdf

This warehouse is currently in use by the X-Search[18] system. Before building the **`MarineTLO`**-based warehouse, X-Search was exploiting FLOD as the underlying knowledge base and was able to detect no more than 11,000 species. Note also that for each species, the **`MarineTLO`**-based warehouse has in average about 30 properties, while in FLOD each species has in average only 6 properties. In addition, the **`MarineTLO`**-based warehouse contains about 200 distinct predicates that connect two URIs (contrary to the about 40 predicates of FLOD), allowing richer experience while browsing on the properties of an entity. The left part of Figure 15(i) depicts an example of (a part of) an entity card. An entity card is a popup window describing a resource (e.g., a species) which is displayed to the user on-demand (by clicking the small icon next to an entity name in Figure 3), offering entity exploration and browsing. In that figure, we divided the card into four groups, each one presenting information derived from different sources. Specifically, group A comes from DBpedia, B from FLOD, C from ECOSCOPE and D from WoRMS. Note that this information is derived at real-time (in less than one second).

Furthermore, the FactSheetGenerator (described in § 2.2) for using this warehouse is under development and will offer more elaborate information. Its current version focuses on tuna species and is called TunaAtlas[19]. An indicative screen of a prototype is given in Figure 15(iii).

Finally, we have developed (and currently improve) an Android application, called `Ichthys` that exploits the contents of the warehouse aiming to offer to end users information about marine species in a user friendly manner. Screen samples are shown in Figure 15(ii).

6 Concluding Remarks

In this paper, we described the design of a top level ontology for the marine domain, intended to satisfy the need for maintaining integrated sets of facts about marine species, and thus assisting ongoing research on biodiversity. The ontology offers a unified and coherent core model for schema mapping, which enables the formulation and answering of complex queries that cannot be answered by any individual source alone. We identified and described use cases and applications that exploit this ontology, and elaborated on the mappings that are required to build integrated warehouses. Finally, we discussed the realization of the mappings given the reasoning capabilities of the selected triplestore and evaluated the warehouse with respect to its completeness and its ability to answer the complex queries.

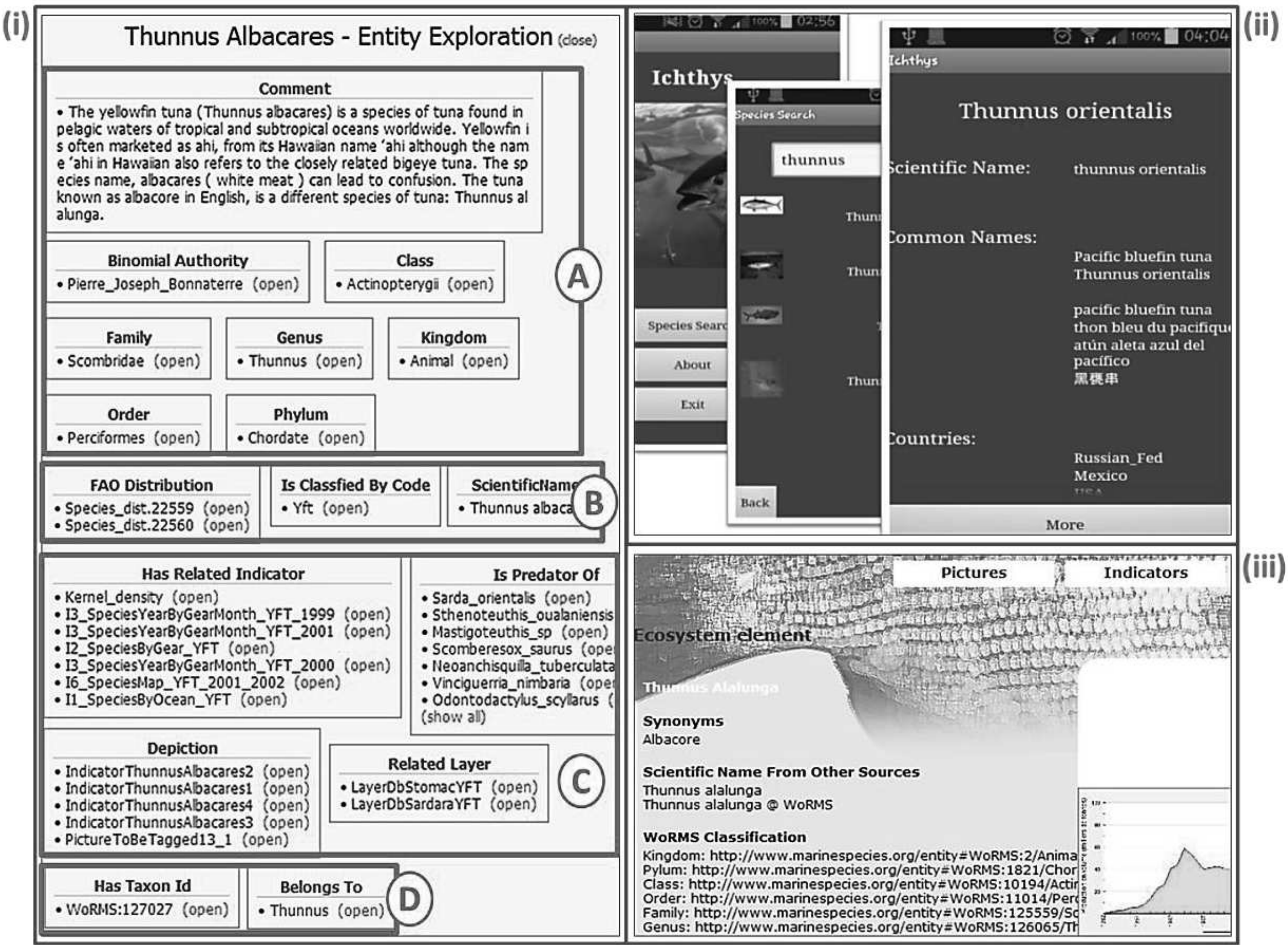
In the future, we plan to continue along the same lines and evolve **`MarineTLO`** by considering more sources and more competence queries, and to enhance the configurability of the workflow used for producing **`MarineTLO`**-based wareshouses.

To conclude, MarineTLO will also be exploited in the context of the LifeWatch Greece project[20], as the core underlying schema of the Lifewatch Greece infrastructures. Towards this end, it will be extended to cover also terrestrial and fresh water domains, microCT scanning processes, genetics, morphometric characteristics, and more.

[18] http://www.ics.forth.gr/isl/X-Search

[19] http://www.i-marine.eu/Content/About.aspx?id=f0fd33e9-b4bf-41b4-a746-46c0981913cc

[20] http://www.lifewatchgreece.eu/



Thunnus Albacares - Entity Exploration (close)

Comment
- The yellowfin tuna (Thunnus albacares) is a species of tuna found in pelagic waters of tropical and subtropical oceans worldwide. Yellowfin i s often marketed as ahi, from its Hawaiian name ʻahi although the nam e ʻahi in Hawaiian also refers to the closely related bigeye tuna. The sp ecies name, albacares (white meat) can lead to confusion. The tuna known as albacore in English, is a different species of tuna: Thunnus al alunga.

Binomial Authority
- Pierre_Joseph_Bonnaterre (open)

Class
- Actinopterygii (open)

Family
- Scombridae (open)

Genus
- Thunnus (open)

Kingdom
- Animal (open)

Order
- Perciformes (open)

Phylum
- Chordate (open)

FAO Distribution
- Species_dist.22559 (open)
- Species_dist.22560 (open)

Is Classfied By Code
- Yft (open)

ScientificName
- Thunnus albaca

Has Related Indicator
- Kernel_density (open)
- I3_SpeciesYearByGearMonth_YFT_1999 (open)
- I3_SpeciesYearByGearMonth_YFT_2001 (open)
- I2_SpeciesByGear_YFT (open)
- I3_SpeciesYearByGearMonth_YFT_2000 (open)
- I6_SpeciesMap_YFT_2001_2002 (open)
- I1_SpeciesByOcean_YFT (open)

Is Predator Of
- Sarda_orientalis (open)
- Sthenoteuthis_oualaniensis
- Mastigoteuthis_sp (open)
- Scomberesox_saurus (ope
- Neoanchisquilla_tuberculata
- Vinciguerria_nimbaria (ope
- Odontodactylus_scyllarus (

(show all)

Depiction
- IndicatorThunnusAlbacares2 (open)
- IndicatorThunnusAlbacares1 (open)
- IndicatorThunnusAlbacares4 (open)
- IndicatorThunnusAlbacares3 (open)
- PictureToBeTagged13_1 (open)

Related Layer
- LayerDbStomacYFT (open)
- LayerDbSardaraYFT (open)

Has Taxon Id
- WoRMS:127027 (open)

Belongs To
- Thunnus (open)

Ichthys

Species Search

thunnus

Species Search

About

Exit

Back

Ichthys

Thunnus orientalis

Scientific Name: thunnus orientalis

Common Names:
Pacific bluefin tuna
Thunnus orientalis

pacific bluefin tuna
thon bleu du pacifiqu
atún aleta azul del pacífico
黑鮪串

Countries:
Russian_Fed
Mexico

More

Pictures

Indicators

Ecosystem element

Thunnus Alalunga

Synonyms
Albacore

Scientific Name From Other Sources
Thunnus alalunga
Thunnus alalunga @ WoRMS

WoRMS Classification
Kingdom: http://www.marinespecies.org/entity#WoRMS:2/Anima
Pylum: http://www.marinespecies.org/entity#WoRMS:1821/Chor
Class: http://www.marinespecies.org/entity#WoRMS:10194/Actin
Order: http://www.marinespecies.org/entity#WoRMS:11014/Perc
Family: http://www.marinespecies.org/entity#WoRMS:125559/Sc
Genus: http://www.marinespecies.org/entity#WoRMS:126065/Th

Figure 15: Usages of MarineTLO-based warehouse. (i) an entity exploration card displayed by XSearch for the species Thunnus Albacares, (ii) Screenshots from the Icthys Android application, (iii) the Tuna Atlas application

Acknowledgement: This work was partially supported by the project iMarine (FP7 Research Infrastructures, 2011-2014) and the LifeWatch Greece project (National Strategic Reference Framework, 2012-2015), and the European Union and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program (Thales, Investing in knowledge society through the European Social Fund).

References

[Candela et al., 2008]: L. Candela, D. Castelli, P. Pagano, gCube: a service-oriented application framework on the grid. ERCIM News, 72 (2008), pp. 48–49 (October).

[Candela et al., 2014-1]: L. Candela, D. Castelli, G. Coro, L. Lelii, F. Mangiacrapa, V. Marioli, P. Pagano, An Infrastructure-oriented Approach for supporting Biodiversity Research. Ecological Informatics, Elsevier, 2014, doi: 10.1016/j.ecoinf.2014.07.006.

[Candela et al., 2014-2]: L. Candela, F. De Faveri, L. Lelli, F. Mangiacrapa, V. Marioli, P. Pagano. Accessing biodiversity databases: a domain specific query language and a unifying data model. Technical Report 2014-TR-26, Istituto di Scienza e Tecnologie dell'Informazione A. Faedo, CNR (2014).

[Colombo et al., 2009]: G. Colombo, D. Merico, Z. Nagy, F. De Paoli, M. Antoniotti, and G. Mauri. Ontological Modeling at a Domain Interface: Bridging Clinical and Biomolecular Knowledge. The Knowledge Engineering Review, Vol. 24:3, 205–224. 2009, Cambridge University Press, doi:10.1017/S0269888909990026.

[Doerr et al., 2003] :M. Doerr, J. Hunter, and C. Lagoze. Towards a core ontology for information integration. Journal of Digital Information, 4:2003, 2003.

[Fafalios et al., 2012]: P. Fafalios, I. Kitsos, Y. Marketakis, C. Baldassarre, M. Salampasis, and Y. Tzitzikas. Web searching with entity mining at query time. In Procs of the 5th Information Retrieval Facility Conference, 2012.

[Fafalios and Tzitzikas,2013]: P. Fafalios and Y. Tzitzikas. X-ENS: Semantic Enrichment of Web Search Results at Real-Time. In Procs of SIGIR'13, 2013.

[Gangemi et al., 2002] A. Gangemi, F. Fisseha, I. Pettman, and J. Keizer. Building an integrated formal ontology for semantic interoperability in the fishery domain. In Procs of ISWC'2002, 2002.

[Haslhofer et al., 2011]: B. Haslhofer, E. Momeni Roochi, B. Schandl, and S. Zander. Europeana RDF store report. 2011.

[Hitzler et al., 2009]: P. Hitzler, M. Krtzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. OWL 2 Web Ontology Language Primer. W3C Recommendation, World Wide Web Consortium, October 2009.

[Ibrahim and Pyster,2004]: L. Ibrahim and A. Pyster. A single model for process improvement. IT Professional, 6(3), May 2004.

[Madin et al., 2007]: J. Madin, S. Bowers, M. Schildhauer, S. Krivov, D. Pennington, and F. Villa. An ontology for describing and synthesizing ecological observation data. Ecological informatics, 2(3):279{296, 2007.

[Moreau et al., 2011]: L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S.Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, J.V. den Bussche, The open provenance model core specification (v1.1), Future Generation Computer Systems 27 (6) (2011) 743–756. doi:10.1016/j.future.2010.07. 005. URL: http://www.sciencedirect.com/science/article/B6V06-50J9GPP-3/2/09d841ac888ed813ccc3cce84383ce27.

[Mosca et al., 2009]: A. Mosca, M. Palmonari, and F. Sartori. An upper-level Functional Design Ontology to Support Knowledge Management in SME-based E-Manufacturing of Mechanical Products. The Knowledge Engineering Review, Vol. 24:3, 265–285. & 2009, Cambridge University Press. doi:10.1017/S0269888909990063.

[Mountantonakis et al., 2014]: M. Mountantonakis, C. Allocca, P. Fafalios, N. Minadakis, Y. Marketakis, C. Lantzaki, and Y. Tzitzikas. Extending VoID for expressing the connectivity metrics of a semantic warehouse. In 1st International Workshop on Dataset Profiling & Federated Search for Linked Data (PROFILES'14), Anissaras, Crete, Greece, May 2014.

[Noessner et al., 2010]: J. Noessner, M. Niepert, C. Meilicke, and H. Stuckenschmidt. Leveraging terminological structure for object reconciliation. Procs of ESWC'10, 2010.

[Sacco and Tziztikas,2009]: G. M. Sacco and Y. Tzitzikas. Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience, volume 25. Springer, 2009.

[Simeoni et al., 2007]: F. Simeoni, L. Candela, G. Kakaletris, M. Sibeko, P. Pagano, G. Papanikos, P. Polydoras, Y. Ioannidis, D. Aarvaag, F. Crestani. A Grid-based Infrastructure for Distributed Retrieval. In Research and Advanced Technology for Digital Libraries, 11th European Conference, ECDL 2007, Budapest, Hungary, September 16-21, 2007, Proceedings, Springer-Verlag, 2007, 4675, 161-173

[TDWG,2004]: TDWG. Darwin core schema (version 1.3), a draft standard of the taxonomic database working group (tdwg), 2004.

[Tzitzikas et al., 2014-1]: Y. Tzitzikas, N. Minadakis, Y. Marketakis, P. Fafalios, C. Allocca, and M. Mountantonakis. Quantifying the connectivity of a semantic warehouse. In 4th International Workshop on Linked Web Data Management (LWDM'14), Athens, Greece, March'2014.

[Tzitzikas et al., 2014-2]Q:Y. Tzitzikas, N. Minadakis, Y. Marketakis, P. Fafalios, C. Allocca, M. Mountantonakis, and I. Zidianaki. MatWare: Constructing and exploiting domain specific warehouses by aggregating semantic data.In 11th Extended Semantic Web Conference (ESWC'14), Anissaras, Crete, Greece, May 2014.

[Wieczorek et al., 2012] J. Wieczorek, D. Bloom, R. Guralnick, S. Blum, M. Döring, R.D.T. Robertson, D. Vieglais. Darwin Core: an evolving community-developed biodiversity data standard. PLoS One, 7 (1) (2012).

[Wilson, 2009]: E. Wilson. Metadata for plant seeds: Taxonomy, standards, issues, and impact. Library Philosophy and Practice (e-journal), page 306, 2009.