Anonymity preserving sequential pattern mining

Anna Monreale; Dino Pedreschi; Ruggero G. Pensa; Fabio Pinelli
Anonymity preserving sequential pattern mining
ARTIFICIAL INTELLIGENCE AND LAW (2014) 22
DOI: 10.1007/s10506-014-9154-6

Anonymity Preserving Sequential Pattern Mining

Anna Monreale · Dino Pedreschi · Ruggero G. Pensa · Fabio Pinelli

Abstract The increasing availability of personal data of a sequential nature, such as time-stamped transaction or location data, enables increasingly sophisticated sequential pattern mining techniques. However, privacy is at risk if it is possible to reconstruct the identity of individuals from sequential data. Therefore, it is important to develop privacy-preserving techniques that support publishing of really anonymous data, without altering the analysis results significantly.

In this paper we propose to apply the *Privacy-by-design* paradigm for designing a technological framework to counter the threats of undesirable, unlawful effects of privacy violation on sequence data, without obstructing the knowledge discovery opportunities of data mining technologies. First, we introduce a k-anonymity framework for sequence data, by defining the sequence linking attack model and its associated countermeasure, a k-anonymity notion for sequence datasets, which provides a formal protection against the attack. Second, we instantiate this framework and provide a specific method for constructing the k-anonymous version of a sequence dataset, which preserves the results of sequential pattern mining, together with several basic statistics and other analytical properties of the original data, including the clustering structure. A comprehensive experimental study on realistic datasets of process-logs, web-logs and GPS tracks is carried out, which empirically shows how, in our proposed method, the protection of privacy meets analytical utility.

A. Monreale, D. Pedreschi
University of Pisa, Italy and KDD Lab.- ISTI-CNR, Pisa, Italy
E-mail: {annam, pedre}@di.unipi.it

R. G. Pensa
Dept. of Computer Science, University of Torino
E-mail: pensa@di.unito.it

F. Pinelli
IBM Research, Dublin, Ireland
E-mail: fabiopin@ie.ibm.com

1 Introduction

The pervasive, automated collection of personal data, combined with increasingly sophisticated data mining techniques, is leading to a new generation of personalized intelligent services. On the other hand, the collection and the disclosure of personal, often sensitive, information increases the risks of privacy violation. For this reason, many recent research works have focused on privacy-preserving data mining and anonymity-preserving data publishing [9,32,12,20]; several techniques have been proposed, that allow to extract knowledge while trying to protect the privacy of users and customers (or respondents) represented in the data[1]. Remarkably, the techniques based on the k-anonymity concept provide anonymous versions of the original datasets, where the personal data of each respondent is indistinguishable from those of other k respondents.

An important field in data mining research concerns the analysis of sequence data. User's actions as well as customer transactions are stored together with their time-stamps, making the temporal sequentiality of the events a powerful source of information. For instance, web-logs describe the full activity of website visitors. Also, the spreading of mobile devices, such as cell phones, GPS devices and RFIDs, has become a great source of spatio-temporal data. Companies and public institutions can now study the sequential/mobile behavior of their customers/citizens to improve offered services. A lot of advanced techniques have been investigated to extract patterns and models in databases of sequences [8,30], as well as in databases of moving objects (trajectories) [18]. For both legal and ethical reasons, the data owners (or custodians) should not compromise the privacy of their customers and users, and therefore should not reveal their personal sensitive information. The point is that a long sequence of events occurred to an individual, or of locations visited by an individual, may reveal a lot about the individual itself, even if it is de-identified in the data. Quoting Robert O'Harrow Jr. in *No Place to Hide* (Free Press, 2005): "Most of privacy violations are not caused by the revelation of big personal secrets, but by the disclosure of many small facts in a row. Like killer bees, one is just a nuisance, but a swarm can be lethal." In the last few years, several techniques have been proposed to develop technological frameworks for countering the threats of undesirable and unlawful effects of privacy violation, without obstructing the knowledge discovery opportunities of data mining technologies. The common result obtained is that no general method exists which is capable of both dealing with *generic personal data* and preserving *generic analytical results*.

In this paper, we propose the application of the *Privacy-by-design* paradigm for designing a framework to counter the threats of privacy violation on sequence data, without obstructing the knowledge discovery opportunities of data mining technologies. First, we discuss the Privacy-by-design principle, introduced in the '90s by Ann Cavoukian, the Information and Privacy Commissioner of Ontario - Canada, by highlighting how it has been embraced by United States and Europe. Then, we describe the idea of Privacy-by-design in data mining domain proposed in [26] and apply it for the design of framework which assures an anonymous publication of sequence data, while preserving a good quality of some data analysis.

The proposed framework is based on the well-known notion of k-anonymity. The concept of k-anonymity (and variants) has been extensively studied for relational data in tabular format, as a formal protection model against privacy breaches; instead, to

[1] In statistics, the problem has been extensively studied in the field of *statistical disclosure control*.

the best of our knowledge, no principled extension of this concept to sequential data has been put forward. A discussion on related works is reported in Sec. 8, which highlights how many extensions to the original idea of k-anonymity have been put forward, to overcome its limited ability to protect privacy in the general case of tabular data. Within this context, the contribution of this paper is twofold.

First, we introduce a k-anonymity framework for sequence data, by defining the sequence linking attack model and its associated countermeasure, namely the k-anonymous version of a sequence dataset. It provides a formal protection framework against the attack. We justify how this framework achieves a strong protection in the case of sequential data, despite the rather weak protection of ordinary k-anonymity for tabular data.

Second, we instantiate this framework by providing a specific method for constructing the k-anonymous version of a sequence dataset, with the aim of preserving sequential pattern mining. We empirically validate the effectiveness of our anonymization technique with realistic web-log and process-log data, as well as with a large-scale, real-life dataset of GPS trajectories of vehicles with on-board GPS receivers, tracked in the city of Milan, Italy. The results of our experiments, where we compare the set of sequential patterns obtained before and after the application of our anonymization technique, show that we substantially preserve such frequent sequential patterns, especially in dense datasets. Clearly, preserving frequent (sequential) patterns is a hard task, so we hoped to obtain, as a collateral benefit of our approach, that other interesting analytical properties of the original data are preserved by the proposed anonymization method. Remarkably, we found in our experiments that, besides sequential patterns, also various basic statistics are preserved after our transformation, such as the distribution of sequence lengths and the frequency of individual elements in the sequences, as well as the clustering structure of the original dataset.

The combined effects of our findings, reported in this paper, is that a simple and effective anonymity protection model exists for personal data of a sequential nature, and that such model admits a practical and efficient method to obtain non-trivial anonymous versions of sequential datasets, where analytical utility is preserved to a broad extent.

The rest of the paper is organized as follows. In Section 2 we discuss the Privacy-by-design paradigm. After recalling the basics of sequential pattern mining in Section 3.1, we introduce in Section 3.2 the k-anonymity framework for sequence data and study the associated attack and protection model. Then, we state in Section 4 the privacy-preserving k-anonymization problem and propose a solution in Section 5. The experimental results for pattern mining and clustering are presented in Section 6. Section 7 discusses the impact of our solution on the society and the legal ground. Finally, Section 8 gives an account of relevant related works, and Section 9 concludes.

2 Privacy-by-design

Privacy-by-design is an approach to protect privacy by inscribing it into the design specifications of information technologies from the very start. It was developed by Ontario's Information and Privacy Commissioner, Dr. Ann Cavoukian, in the 1990s, as a response to the growing threats to online privacy that were beginning to emerge at that time. This paradigm represents a significant innovation with respect to the traditional approaches for protecting privacy, which focus on providing remedies for privacy

breaches after-the-fact. On the contrary, Privacy-by-design requires that organizations think about privacy in a proactive mode; in other words, we have a significant shift from a reactive model to proactive one that requires to prevent privacy issues instead of to remedy to them.

In the last year, many companies are realizing the necessity to adopt and consider privacy at every stage of their business. They have been turning to the concept and principle of Privacy-by-design to integrate privacy requirements into their business model. The main problems related to this promising paradigm, considered the next generation of the privacy protection, are that: (1) in many contexts it is not completely clear which are the approaches to incorporate Privacy-by-design, and (2) it is growing the need for a new legal framework to protect privacy where the major aspect should be Privacy-by-design.

2.1 Privacy-by-design in Law

The Privacy-by-design model for privacy and data protection has been recognized in legislation. Privacy officials in Europe and the United States are embracing this paradigm as never before.

In 2010, at the annual conference of "Data Protection and Privacy Commissioners" the International Privacy Commissioners and Data Protection Authorities approved a resolution recognizing Privacy-by-design as an *essential component of fundamental privacy protection* [2]. This resolution encourages the adoption of the principles of Privacy-by-design as part of an organization's default mode of operation, and invites Data Protection and Privacy Commissioners to promote this paradigm for the incorporation of its foundational principles in privacy policy and legislation in their respective jurisdictions.

A year earlier, the EU Article 29 Data Protection Working Party and the Working Party on Police and Justice issued a joint Opinion, advocating for incorporating the principles of Privacy-by-design into a new EU privacy framework [1]. In March 2010, the European Data Protection Supervisor recommended to "include unequivocally and explicitly the principle of Privacy-by-design into the existing data protection regulatory framework" [34]. This recommendation was taken into consideration in the recent revision of the Data Protection Directive (95/46/EC). The European Union Data Protection Directive has always included provisions requiring data controllers to implement *technical and organizational measures* in the design and operation of ICT, But this has proven insufficient. Therefore, in the comprehensive reform of the data protection rules proposed on January 25, 2012 by the EC, the new data protection legal framework introduces, with respect to the Directive 95/46/EC, the reference to *data protection by design and by default* (Article 23 of the Proposal for a Regulation and Article 19 of the Proposal for a Directive). These articles compel the controller to "implement appropriate technical and organizational measures and procedures in such a way that the processing will meet the requirements of this Regulation and ensure the protection of the rights of the data subject." and to "implement mechanisms for ensuring that, by default, only those personal data are processed which are necessary for each specific purpose of the processing ...".

Privacy-by-design has been embraced with the same enthusiasm in the United States. In the last years the U.S. Federal Trade Commission hosted a series of public roundtable discussions on privacy issues in the digital age and in a recent staff report

[29] it describes a proposed framework with three main recommendations: *privacy by design*, *simplified consumer choice*, and *increased transparency of data practices*.

Moreover, in April 2011, Senators John Kerry (D-MA) and John McCain (R-AZ) proposed their legislation entitled "Commercial Privacy Bill of Rights Act of 2011" that would require companies that collect, use, store or transfer consumer information to implement a version of Privacy-by-design when developing products.

2.2 Privacy-by-design in Data Mining

As stated above, in many contexts it is not clear what means applying the Privacy-by-design principle and which is the best way to applying it for obtaining the desired result. In this section, we present the articulation of the general "by design" principle in the data mining domain proposed in [26].

In the literature, several techniques have been proposed to develop technological frameworks for countering the threats of undesirable and unlawful effects of privacy violation, without obstructing the knowledge discovery opportunities of data mining technologies. The common result obtained is that no general method exists which is capable of both dealing with "generic personal data" and preserving "generic analytical results". Monreale in [26] introduce the idea to inscribe privacy protection into the knowledge discovery technology by design, so that the analysis incorporates the relevant privacy requirements from the very start, evoking the concept of Privacy-by-design discussed above.

The articulation of the general "by design" principle in the data mining domain is that higher protection and quality can be better achieved in a goal-oriented approach. In such an approach, the data mining process is designed with assumptions about:

(a) the sensitive personal data that are the subject of the analysis;
(b) the attack model, i.e., the knowledge and purpose of a malicious party that has an interest in discovering the sensitive data of certain individuals;
(c) the category of analytical queries that are to be answered with the data.

These assumptions are fundamental for the design of a privacy-preserving framework. First of all, the techniques for privacy preservation strongly depend on the nature of the data to be protected. Second, a valid framework has to define the attack model based on a specific adversary's background knowledge and an adequate countermeasure. Different assumptions on the background knowledge entail different defense strategies.

Finally, a privacy-preserving strategy should find an acceptable trade-off between data privacy and data utility. To reach this goal it is fundamental to consider the category of analytical queries to be answered with the transformed data for understanding which data properties is necessary to preserve. As an example, the design of a defense strategy for spatio-temporal data should consider that these data could be used to analyze collective mobility behavior in a city.

Under the above assumptions, we claim that it is conceivable to design a privacy-preserving analytical process able to: i) transform the data into an anonymous version with a quantifiable privacy guarantee - i.e., the probability that the malicious attack fails; and ii) guarantee that a category of analytical queries can be answered correctly, within a quantifiable approximation that specifies the data utility, using the transformed data instead of the original ones

In the following, we show how we apply the Privacy-by-design for the design of a framework for the publication of anonymous sequence data. First, we analyze the privacy issues related to this kind of data, second, we identify the attack model and third, we provide a method for assuring anonymity in sequence data taking into consideration the data analysis that we want to maintain valid.

3 Privacy Model

In this section we discuss the privacy issues in the publication of sequence data and provide a formal definition for the attack model we consider here. Before presenting the core ideas of our framework, we recall some useful preliminaries on frequent sequential pattern mining.

3.1 Preliminaries: frequent sequential pattern mining

We briefly review the basics of frequent sequential pattern mining. Let $\mathcal{I} = \{l_1, l_2, \ldots, l_n\}$ denote a set of items (e.g., events, actions, spatial locations or regions). A sequence $S = s_1 s_2 \ldots s_m$ ($s_i \in \mathcal{I}$) is an ordered list of items; an item can occur multiple times in a sequence. A sequence $T = t_1 t_2 \ldots t_w$ is a subsequence of S ($T \preceq S$) if there exist integers $1 \leq i_1 < \ldots < i_w \leq m$ such that $\forall 1 \leq j \leq w \; t_j = s_{i_j}$. A sequence database $\mathcal{D}$ is a multiset of sequences $\mathcal{D} = \{S_1, S_2, \ldots, S_N\}$. The support of a sequence T in a database $\mathcal{D}$ is the number of sequences in the database containing T, i.e.: $supp_{\mathcal{D}}(T) = |\{S \in \mathcal{D} | T \preceq S\}|$. The relative frequency of a sequence T in a database $\mathcal{D}$ is given by $freq_{\mathcal{D}}(T) = supp_{\mathcal{D}}(T)/|\mathcal{D}|$, where $|\mathcal{D}|$ is the total number of sequences in $\mathcal{D}$. Given a frequency threshold σ, a sequence T is called σ-frequent in a database $\mathcal{D}$ if $freq_{\mathcal{D}}(T) \geq \sigma$ (or $supp_{\mathcal{D}}(T) \geq \sigma \cdot |\mathcal{D}|$). A σ-frequent sequence is also called σ-frequent sequential pattern. The collection of all σ-frequent (sequential) patterns in $\mathcal{D}$ is denoted by $\mathcal{S}(\mathcal{D}, \sigma)$.

The frequent sequential pattern mining problem is formulated as follows: given a sequential database $\mathcal{D}$ and a frequency threshold σ, find all σ-frequent sequential patterns, i.e. the collection $\mathcal{S}(\mathcal{D}, \sigma)$. Since its first definition, many algorithms for sequential patterns have been proposed [8,30].

3.2 Sequence Linking Attack

An intruder who gains access to a published database of personal (micro-)data can conduct attacks on this database in order to make inferences, also on the basis of background knowledge that (s)he possesses. We generically refer to this agent as an attacker; specifically, we refer to the *linking attack model*, i.e., the ability to link the released data to other external information, which enables the re-identification of (some of) the respondents associated with the data. In relational data, linking is made possible by *quasi-identifiers*, i.e., attributes that, in combination, can uniquely identify individuals, such as birth date and gender. The remaining attributes represent the private respondent's information, that may be violated by the linking attack. In privacy-preserving data publishing techniques, such as k-anonymity, the goal is precisely to find countermeasures to this attack, and to release person-specific data in such a way that the ability to link to other information using the quasi-identifier(s) is limited.

In the case of sequential (person-specific) data, where each record is a temporal sequence of events occurred to a specific person, the above dichotomy of attributes into quasi-identifiers (QI) and private information (PI) does not hold any longer: here, a (sub)sequence of events can play both the role of QI and the role of PI. To see this point, consider the case where sequences represent trajectories, i.e., lists of locations visited by an individual in the given order: the attacker may know a sequence of locations visited by some specific person P: e.g., by shadowing P for some time, the attacker may learn that P was in the shopping mall, then in the park, and then at the train station, represented by the sequence $<$*mall, park, station*$>$. The attacker could employ such sequence to retrieve the complete trajectory of the P in the released dataset: this attempt would succeed, provided that the attacker knows that P's sequence is actually present in the dataset, if the known sequence $<$*mall, park, station*$>$ is compatible with (i.e., is a subsequence of) just one sequence in the dataset. In this example of a linking attack in the sequence domain, the subsequence known by the attacker serves as QI, while the entire sequence is the PI that is disclosed after the re-identification of the respondent. Clearly, as the example suggests, is rather difficult to distinguish QI and PI: in principle, any specific location can be the theater of a shadowing actions by a spy, and therefore any possible sequence (of locations, in this example) can be used as a QI, i.e., as a means for re-identification. Put another way, distinguishing between QI and PI among the elements of a sequence, being them locations or events, means putting artificial limits on the attacker's background knowledge; on the contrary, it is required in privacy and security research to have assumptions on the attacker's knowledge that are as liberal as possible, in order to achieve maximal protection.

As a consequence of this discussion, we take in this work the radical assumption that *any sequence that can be linked to a small number of individuals is a potentially dangerous QI and a potentially sensitive PI*; then, we study an anonymity model that tries to achieve the maximal protection possible under this challenging assumption. The crucial point in defining the sequence linking attack lies exactly in the definition of QI and PI, which is formalized by the concept of *harmful sequence*, parametric w.r.t. an anonymity threshold k.

Definition 1 (k-Harmful Sequence) Given a sequence dataset $\mathcal{D}$ and an anonymity threshold k, a sequence T is *k-harmful* (in $\mathcal{D}$) iff $0 < supp_{\mathcal{D}}(T) < k$.

In other words, a sequence is k-harmful if it is a subsequence of a number of sequences in $\mathcal{D}$ smaller than k and greater than 0. Essentially, harmful sequences are potentially dangerous QIs because they occur only a few times in the dataset (but at least once): thus, a harmful sequence can be used to select a few specific complete sequences in the dataset. Moreover, each harmful sequence reveals information pertaining to a small (but not empty) set of persons, hence information that is private in the sense that it reveals a specific, unusual behavior, which potentially violates the right to privacy of a few individuals that follow a path off the crowd (perhaps revealing personal preferences, habits, etc.) Conversely, non-harmful sequences are not considered dangerous, neither as QI nor as PI: a non-harmful sequence either does not occur in the dataset (and therefore does not help the attacker) or occurs so many times that (i) it is not useful as a QI, as it is compatible with too many subjects, and (ii) it is not useful as PI, as it reveals a sequential behavior common to many people. We now formalize the sequence linking attack model and our proposed countermeasures, based on the discussion above.

Definition 2 (Sequence Linking Attack) The attacker, given a published sequence dataset $\mathcal{D}$ where each sequence is uniquely associated with a de-identified respondent, tries to identify the sequence in $\mathcal{D}$ associated with a given respondent X, based on the following additional knowledge: (i) a (QI) sequence T relative to X, and (ii) the fact that respondent X is present in $\mathcal{D}$. We denote by $prob_{\mathcal{D}}(T)$ the probability that the sequence linking attack with a QI sequence T succeeds (over $\mathcal{D}$).

From a data protection perspective, we aim at controlling the probability $prob_{\mathcal{D}}(T)$, for any possible QI sequence T. The linking attack can be performed by using either a harmful or a non-harmful sequence. Clearly, harmful sequences are dangerous because the attacker has a high probability to uniquely identify the entire sequence of a respondent. In general, given an arbitrary sequence dataset $\mathcal{D}$, there's no way to prevent re-identification. To solve this problem, we introduce the *k-anonymous version* of a sequence dataset $\mathcal{D}$, parametric w.r.t. an *anonymity threshold* $k > 1$.

Definition 3 (k-Anonymous version of a Sequence Dataset) Given an anonymity threshold $k > 1$ and two sequence datasets $\mathcal{D}$ and $\mathcal{D}'$, we say that $\mathcal{D}'$ is a *k-anonymous version* of $\mathcal{D}$ iff each k-harmful sequence in $\mathcal{D}$ is not k-harmful in $\mathcal{D}'$.

Here, k plays the same role as in ordinary k-anonymity, i.e., k is the minimal acceptable cardinality of a set of indistinguishable objects, considered a sufficient protection of anonymity in the given situation. Notice that, according to Def. 3, the harmful sequences in the original dataset become non-harmful in the anonymous version, while no constraints apply to non-harmful sequences in the original dataset: those may be either non-harmful or harmful in the anonymous version. We now show that the probability of success of a linking attack in the k-anonymous version of a sequence dataset has an upper bound of $\frac{1}{k}$.

Theorem 1 *Given a k-anonymous version $\mathcal{D}'$ of a sequence dataset $\mathcal{D}$, we have that, for any QI sequence T, $prob_{\mathcal{D}'}(T) \leq \frac{1}{k}$.*

Proof Two cases arise.
Case 1: if T is a k-harmful sequence in $\mathcal{D}$, then, by Def. 3, T is not a k-harmful sequence in $\mathcal{D}'$, i.e., either $supp_{\mathcal{D}'}(T) = 0$, which implies $prob_{\mathcal{D}'}(T) = 0$, or $supp_{\mathcal{D}'}(T) \geq k$, which implies $prob_{\mathcal{D}'}(T) = \frac{1}{supp_{\mathcal{D}'}(T)} \leq \frac{1}{k}$.
Case 2: if T is not a k-harmful sequence in $\mathcal{D}$, then, by Def. 3, T can have an arbitrary support in $\mathcal{D}'$. If T is not k-harmful in $\mathcal{D}'$, then the same reasoning in Case 1 applies; otherwise, $0 < supp_{\mathcal{D}'}(T) < k$. In this case, we have that the probability of success of the linking attack via T to person X is the probability that X is present in $\mathcal{D}'$ times the probability of picking X in $\mathcal{D}'$, i.e.,

$$prob_{\mathcal{D}'}(T) = \frac{supp_{\mathcal{D}'}(T)}{supp_{\mathcal{D}}(T)} \times \frac{1}{supp_{\mathcal{D}'}(T)} = \frac{1}{supp_{\mathcal{D}}(T)} \leq \frac{1}{k}$$

where the final inequality is justified by the fact that X is present in $\mathcal{D}$ by the assumption of the linking attack, and therefore $supp_{\mathcal{D}}(T) \geq k$ due to the hypothesis that T is not k-harmful in $\mathcal{D}$. This concludes the proof.

Thanks to Thm. 1, we have a formal mechanism to control the probability of success of the sequence linking attack: given $\mathcal{D}$, choose a suitable anonymity threshold k and publish a k-anonymous version of $\mathcal{D}$: thus, we are guaranteed that such probability has an upper bound of $\frac{1}{k}$.

It is natural to ask ourselves whether this level of privacy protection is adequate, in reference to the specified attack model: to further discuss this point, let's play the devil's advocate. Consider the following continuation of the above example, where person P has been shadowed by a spy in its itinerary represented by the sequence $S =<mall,$ *park, station*$>$; assume that the attacker retrieves in the k-anonymous version of the sequence dataset a large number ($\geq k$) of sequences containing S as a subsequences, and that all such sequences contain the location *red-light-district*, after S. In this situation, the attacker, albeit cannot identify precisely his victim's itinerary with high probability, can safely conclude that P has visited the *red-light-district* anyway, no matter which precise itinerary has been followed. This resembles the motivation for introducing *l-diversity*, an extension of k-anonymity aimed at avoiding precisely the situation where all the tuples from the same anonymity group, albeit large, share the same value of a PI attribute (see Sec. 8 for references). In our sequential setting, is this situation a bug or a feature? Two cases arise, depending on the status we want to assume for the location/event *red-light-district*: it is certainly a PI, but is it a QI or not?

- *red-light-district* is a PI but not a QI: this option introduces an asymmetry between PI and QI, similar to what happens in the original k-anonymity framework: in our example, this implies that the attacker *cannot shadow* person P in location *red-light-district*, which is an arbitrary assumption. In general, this option weakens considerably the attack model, as it makes rather unsupported assumptions about what the attacker can or cannot use as a QI. This observation lies at the heart of the many critics raised against k-anonymity in the tabular case. Even if distinguishing between PI and QI may allow, in principle, to avoid an inference like the one depicted in the example above, this possibility comes at the price of making the overall protection model weaker and unrealistic. This is the reason why we propose to abolish the distinction from QI and PI in our framework.
- *red-light-district* is both a PI and a QI: this the option we chose in our model, and therefore the conclusion that person P has visited the *red-light-district* can indeed be taken. Then two possible situations apply: either this is not a sensitive information, as it is shared with many other persons (at least $k-1$), or having visited the *red-light-district* is perceived as sensitive for anybody. Clearly, the option depends on the particular situation and perception. In the first case, there's nothing to do, this is precisely what our models accounts for. In the second, it is clear that *red-light-district* is a sensitive location per se, and therefore the data should be suitably sanitized from this information, with some form of hiding or concealment. This is an orthogonal issue w.r.t. the model presented in this paper, which may very well co-exist with some pre-processing aimed at hiding the globally sensitive locations or events in the sequential data.

In the end, the above discussion brings evidence that our sequence anonymity model, albeit simple, has a solid motivation. But clearly, according to our definition, there are many possible k-anonymous versions of a sequence dataset $\mathcal{D}$, corresponding to different ways of dealing with the k-harmful sequences of $\mathcal{D}$. For example, harmful sequences can be either discarded or, on the contrary, replicated or generalized (e.g., by removing some of their items); also, any combination of the above techniques yields a k-anonymous dataset. Also, many trivial and not useful examples of k-anonymous version of a given dataset exist, such as the empty dataset (which is a k-anonymous version of *any* given dataset).

From a statistics/data mining point of view, we are only interested in the k-anonymous versions that preserve some interesting analytical properties of the original dataset $\mathcal{D}$. The goal of the rest of this paper is exactly to illustrate the practical significance of our framework, by providing a first practical instance of our model, where the protection of privacy meets analytical utility.

4 Pattern-preserving k-anonymity

We now tackle the problem of constructing a k-anonymous version of $\mathcal{D}$ that preserves the collection of *frequent sequential patterns* in the original data set $\mathcal{D}$. Our approach is based on a specific way of hiding all the k-harmful sequences, capable of controlling the introduced distortion and producing excellent results in the case of *dense* sequential datasets, according to a notion of density that will be clarified later (see Section 5). As a side effect, we obtain that the k-anonymous version also preserves the clustering structure of the original dataset. The *pattern-preserving k-anonymization problem* can be formulated as follows:

Definition 4 (optimal P2kA problem) Given a sequence dataset $\mathcal{D}$, and an anonymity threshold $k > 1$, find a k-anonymous version $\mathcal{D}'$ of $\mathcal{D}$ such that the collection of all $\frac{k}{|\mathcal{D}|}$-frequent patterns in $\mathcal{D}$ is preserved in $\mathcal{D}'$, i.e., the following two conditions hold:

$$\mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|) = \mathcal{S}(\mathcal{D}, k/|\mathcal{D}|)$$
$$\forall T \in \mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|)\ freq_{\mathcal{D}'}(T) = freq_{\mathcal{D}}(T).$$

Clearly, the above requirement is quite strict, as the pattern collection in the anonymous version is supposed to coincide with that in the original dataset. In this paper, we present a method that approximates the optimal solution, i.e., one which assures that (i) $\mathcal{D}'$ is indeed a k-anonymous version of $\mathcal{D}$, and (ii) $\mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|)$ and $\mathcal{S}(\mathcal{D}, k/|\mathcal{D}|)$ are "similar". In particular the two conditions of Def. 4 are relaxed to:

$$\mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|) \subseteq \mathcal{S}(\mathcal{D}, k/|\mathcal{D}|)$$
$$\forall T \in \mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|)\ freq_{\mathcal{D}'}(T) \simeq freq_{\mathcal{D}}(T).$$

In the experimental section (see Section 6) we express this similarity in terms of two measures which quantify how much pattern support changes, and how many frequent patterns we miss. As a first step towards an "optimal" algorithm, we show that our algorithm provides excellent results on real dense datasets in term of pattern similarity (see Section 6), and guarantees that the disclosed dataset is k-anonymous.

5 The BF-P2kA algorithm

In this section we present our *BF-P2kA* (Brute Force Pattern-Preserving k-Anonymization) algorithm (Algorithm 1), which allows to anonymize a dataset of sequences $\mathcal{D}$. The rationale behind our approach is that infrequent subsequences are potentially dangerous, and should be hidden in the disclosed dataset. To perform this step, a straightforward solution would consist in extracting all infrequent sequential patterns (say, sequential patterns with support less than k), and search for those patterns within each sequence in $\mathcal{D}$. Once a pattern T is found in a sequence S, all occurrences of T in S should be

removed. The first issue concerning this solution is the extraction of infrequent patterns, which can be computationally expensive even for small datasets. Moreover, this action does not ensure that frequent sequential patterns (with support higher than k) are preserved. Indeed, for each deletion of the infrequent pattern T, the support of all its proper subsequences (including those subsequences with support higher than k) is decremented by 1.

In fact, we show that solving the relaxed *P2kA* problem is **NP**-hard. The relaxed P2kA problem can be formulated as a special case of the sequence hiding problem introduced in [4,3]. Given a support threshold k and a dataset of sequences $\mathcal{D}$, let $\mathcal{S}_{<k}$ be the set of infrequent sequential patterns in $\mathcal{D}$, and $\mathcal{S}_{\geq k}$ the set of frequent sequential patterns in $\mathcal{D}$. The sequence hiding formulation of the P2kA problem requires to transform $\mathcal{D}$ in a database $\mathcal{D}'$ such that

1. $\forall S \in \mathcal{S}_{<k}$, $supp_{\mathcal{D}'}(S_i) = 0$[2]
2. $\sum_{S \in \mathcal{S}_{\geq k}} |supp_{\mathcal{D}}(S) - supp_{\mathcal{D}'}(S)|$ is minimized

Solving the sequence hiding problem is demonstrated to be **NP**-hard in [4,3]. Hence, we propose a heuristic that provides an approximated solution in polynomial time. Instead of handling sequences directly, our algorithm operates on a prefix tree which guarantees good performances on dense datasets, both in terms of computational time and percentage of preserved frequent sequential patterns.

Our approach consists of three steps. During the first step, the sequences in the input dataset $\mathcal{D}$ are used to build a prefix tree $\mathcal{T}$. The second step, given a minimum support threshold k, anonymizes the prefix tree. This means that sequences whose support is less than k are pruned from the prefix tree. Then part of these infrequent sequences is recovered by updating the corresponding branches in the pruned prefix tree. The third and last step post-process the anonymized prefix tree, as obtained in the previous step, to generate the anonymized dataset of sequences $\mathcal{D}'$.

Algorithm 1: BF-P2kA($\mathcal{D}$, k)

Input: A sequence database $\mathcal{D}$, an integer k
Output: A k-anonymous sequence database $\mathcal{D}'$
// **Step I: PrefixTree construction**
$\mathcal{T} = PrefixTreeConstruction(\mathcal{D})$;
// **Step II: PrefixTree anonymization**
$\mathcal{L}_{cut} = \emptyset$;
foreach $v \in \mathcal{N}$ *s.t.* $\exists (R, v) \in \mathcal{E}$ **do**
  $\mathcal{L}_{cut} = \mathcal{L}_{cut} \cup TreePruning(v, \mathcal{T}, k)$;
end
$\mathcal{T}' = TreeReconstruction(\mathcal{T}, \mathcal{L}_{cut})$;
// **Step III: Generation of anonymized sequences**
$\mathcal{D}' = SequenceGeneration(\mathcal{T}')$;
return $\mathcal{D}'$

[2] In the original formulation, the requirement is that the support is $\leq$ a given threshold.

5.1 Step I: Prefix Tree Construction

The first step of the *BF-P2kA* algorithm (Algorithm 1) is the construction of a prefix tree $\mathcal{T}$, given a list of sequences $\mathcal{D}$. A prefix tree is more compact than a list of sequences. It is defined as a triplet $\mathcal{T} = (\mathcal{N}, \mathcal{E}, R)$, where $\mathcal{N} = \{v_1, \ldots, v_N\}$ is a finite set of N labeled nodes, $\mathcal{E} \in \mathcal{N} \times \mathcal{N}$ is a set of edges, and $R \in \mathcal{N}$ is a fictitious node and represents the root of the tree. Each node of the tree (except the root) has exactly one parent and it can be reached through a unique path, which is a sequence of edges starting with the root node. An example of path for the node d (denoted $\mathcal{P}(d, \mathcal{T})$) is the following:

$$\mathcal{P}(d, \mathcal{T}) = (R, a), (a, b), (b, c), (c, d).$$

Each node $v \in \mathcal{N}$ has entries in the form $\langle id, item, support \rangle$ where id is the identifier of the node v, $item$ represents an item of a sequence, and $support$ is the support of the sequence represented by the path from R to v.

Algorithm 2: PrefixTreeConstruction($\mathcal{D}$)

Input: A sequence database $\mathcal{D}$
Output: A prefix tree $\mathcal{T} = (\mathcal{N}, \mathcal{E}, R)$
$R = CreateNode(NULL, 0)$;
$\mathcal{N} = \{R\}$; $\mathcal{E} = \{\}$;
foreach *distinct* S *in* $\mathcal{D}$ **do**
  $current = R$;
  for $1 \leq i \leq |S|$ **do**
    if $\exists (current, v) \in \mathcal{E}$ *s.t.* $v.item = s_i$ **then**
      $v.support = v.support + supp_{\mathcal{D}}(S)$;
      $current = v$;
    else
      $v = CreateNode(s_i, supp_{\mathcal{D}}(S))$;
      $\mathcal{N} = \mathcal{N} \cup v$;
      $e = CreateEdge(current, v)$;
      $\mathcal{E} = \mathcal{E} \cup e$;
      $current = v$;
    end
  end
end
return $\mathcal{T} = (\mathcal{N}, \mathcal{E}, R)$

The *PrefixTreeConstruction* function considers each element s_i of every sequence S, starting from the first element, and updates the corresponding node in $\mathcal{T}$ (i.e., the node v s.t. $v.item = s_i$) by adding the support of S to $v.support$. This process is iterated until a path from the root to a node corresponding to s_i already exists in $\mathcal{T}$. Otherwise, it creates the nodes corresponding to the remaining elements of S and updates $\mathcal{T}$ accordingly, setting the support of new nodes to $supp_{\mathcal{D}}(S)$.

Step II: Prefix Tree Anonymization

The main phase of Algorithm 1 is the second step; we introduce some notions to simplify its explanation.

Definition 5 (minimum prefix) Let $S = s_1 s_2 \dots s_n$ and $T = t_1 t_2 \dots t_k$ be two sequences such that T is a subsequences of S and s_p is the first item of S such that $T \preceq s_1 s_2 \dots s_p$. The sequence $S' = s_1 \dots s_p$ is the *minimum prefix* of S containing the sub-sequence T.

Let us consider the sequences $S = ABCDECDF$ and $T = ACD$. The sequence $S' = ABCD$ is the minimum prefix of S containing the sub-sequence T.

We also recall the well-known notions of edit distance and Longest Common Subsequence.

Definition 6 (Edit distance) Let S and T be two sequences. The **edit distance** between S and T is given by the minimum number of operations needed to transform a sequences into the other, where an operation is an insertion, deletion, or substitution of a single item.

Definition 7 (LCS) Let $\mathcal{T}$ be a set of sequences. The **Longest Common Subsequence** (LCS) is the longest subsequence common to all sequences in $\mathcal{T}$.

The first operation performed during step II is the pruning of the prefix tree with respect to the minimum support threshold. This operation is executed by the *TreePruning* function (see Algorithm 3), which modifies the tree by pruning all the infrequent subtrees and updating the support of the path to the last frequent node. *TreePruning* visits the tree and, when the support of a given node v is less than the minimum support threshold k, computes all the sequences represented by the paths which contain the node v and which start from the root and reach the leaves of the sub-tree with root v. Note that for construction each node of this sub-tree has support less than k. All the computed (k-harmful) sequences and their supports are inserted into the list $\mathcal{L}_{cut}$. Next, the subtree with root n is cut from the tree. Therefore, the procedure *TreePruning* returns a pruned prefix tree and the list $\mathcal{L}_{cut}$. After the pruning step, the algorithm tries to reattach the harmful sequences in $\mathcal{L}_{cut}$ onto the pruned tree, using the *TreeReconstruction* function (see Algorithm 4). For each harmful sequence S in $\mathcal{L}_{cut}$, *TreeReconstruction* computes the LCS between S and every sequence represented by the tree. Suppose that T is the sequence such that the computed LCS is subsequence of T. Then, the *TreeReconstruction* function selects the path of the tree that represents the minimum prefix (see Definition 5) of T containing the LCS, and increases the support of the related nodes by adding the support of S in $\mathcal{L}_{cut}$. If there are more LCSs having the same length, the function *ClosestLCS* function returns the LCS and the sequence in $\mathcal{T}$ such that the edit distance between them is minimum. This choice allows to increase the support of a limited set of nodes not belonging to the LCS, thus reducing the noise.

Step III: Generation of anonymized sequences

The second step of Algorithm 1 returns an anonymized prefix tree, i.e., a prefix tree where only k-frequent subsequences are represented. The third step our method allows

to generate the anonymized dataset $\mathcal{D}'$. This phase is performed by the *Sequence-Generation* procedure, which visits the anonymized prefix tree and generates all the represented sequences. In general, the number of sequences represented in $\mathcal{T}'$ is equal or less than the size of the original database, since some pruned sequences can not be appended during the *TreeReconstruction* step.

Algorithm 3: TreePruning(v, $\mathcal{T}$, k)

Input: A node v, a prefix tree $\mathcal{T} = (\mathcal{N}, \mathcal{E}, R)$, an integer k
Output: A list of infrequent sequences $\mathcal{L}_{cut}$
$\mathcal{N}_{temp} = \{\}$; $\mathcal{E}_{temp} = \{\}$;
$\mathcal{L}_{cut} = \{\}$;
if $v.support < k$ **then**
  $newnode = CreateNode(v.item, v.support)$;
  $\mathcal{N}_{temp} = \mathcal{N}_{temp} \cup newnode$ $current = v$;
  repeat
    $oldnode = v_i$ s.t. $(v_i, current) \in \mathcal{E}$;
    $newnode = CreateNode(oldnode.item, v.support)$;
    $oldnode.support = oldnode.support - v.support$
    $\mathcal{N}_{temp} = \mathcal{N}_{temp} \cup newnode$;
    $e = CreateEdge(newnode, current)$;
    $\mathcal{E}_{temp} = \mathcal{E}_{temp} \cup e$;
    $current = oldnode$;
  until $current = R$;
  $R_{temp} = current$;
  $\mathcal{T}_{temp} = (\mathcal{N}_{temp}, \mathcal{E}_{temp}, R_{temp})$;
  $\mathcal{T}_{sub} = (\mathcal{N}_{sub}, \mathcal{E}_{sub}, R_{sub}) = SubTree(\mathcal{T}, v)$;
  $\mathcal{T}_{temp} = \mathcal{T}_{temp} \cup \mathcal{T}_{sub}$;
  $\mathcal{T} = \mathcal{T} \setminus \mathcal{T}_{sub}$;
  $\mathcal{L}_{cut}$ = the set of all sequences in $\mathcal{T}_{temp}$;
else
  foreach $v_i \in \mathcal{N}$ *s.t.* $\exists (v, v_i) \in \mathcal{E}$ **do**
    $\mathcal{L}_{cut} \cup TreePruning(v_i, \mathcal{T}, k)$;
  end
end
return $\mathcal{L}_{cut}$

We now show that (i) our approach guarantees that the disclosed dataset $\mathcal{D}'$ is a k-anonymous version of $\mathcal{D}$, and (ii) the set of sequential patterns in $\mathcal{D}'$ is a subset of those in $\mathcal{D}$.

Theorem 2 *Given a sequence dataset $\mathcal{D}$ and an anonymity threshold $k > 1$, the dataset $\mathcal{D}'$ returned by Algorithm 1 satisfies the following conditions:*

1. *$\mathcal{D}'$ is a k-anonymous version of $\mathcal{D}$,*
2. *$\mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|) \subseteq \mathcal{S}(\mathcal{D}, k/|\mathcal{D}'|)$.*

The proof is based on the fact that the *TreeReconstruction* function does not alter the structure of the pruned tree.

Proof (sketch)

1. By construction, the pruning step in Algorithm 3 prunes all the subtrees with support less than k, then the prefix tree $\mathcal{T}$ only contains k-frequent sequences. Nevertheless, the reconstruction step (see Algorithm 4) does not change the tree structure of $\mathcal{T}$, it only increases the support of existing sequences which are already k-frequent in $\mathcal{D}$. In conclusion, at the end of the second step of Algorithm 1, the sequential patterns which are represented in $\mathcal{T}'$ are at least k-frequent in $\mathcal{D}$.
2. At the end of the pruning step in Algorithm 3, all infrequent branches in $\mathcal{T}$ are cut off. However, this could also imply that some k-frequent sequential patterns are pruned out, if they are only supported by multiple infrequent paths in the prefix tree $\mathcal{T}$. Then, the prefix tree $\mathcal{T}$ contains a subset of the $\mathcal{S}(\mathcal{D}, k)$. Moreover, as already stated, during the reconstruction step the tree structure of $\mathcal{T}$ is unchanged, i.e., patterns represented in $\mathcal{T}'$ were still represented in $\mathcal{T}$ after the pruning step. Finally, the set of sequential patterns supported by $\mathcal{D}'$ is a subset of those supported by $\mathcal{D}$.

Algorithm 4: TreeReconstruction($\mathcal{T}$, $\mathcal{L}_{cut}$)

Input: A prefix tree $\mathcal{T}$, a list of sequences $\mathcal{L}_{cut}$
Output: An anonymized reconstructed prefix tree $\mathcal{T}'$
foreach *distinct* $S \in \mathcal{L}_{cut}$ **do**
  $cand = ClosestLCS(S, \mathcal{T})$;
  $L =$ the set of nodes in $\mathcal{T}$ belonging to the first minimum prefix containing $cand$;
  if L *is not empty* **then**
    foreach $v \in L$ **do**
      $v.support = v.support + supp_{\mathcal{L}_{cut}}(S)$;
    end
  end
end
return $\mathcal{T}$

Even if our approach does not assure that $\mathcal{S}(\mathcal{D}', k/|\mathcal{D}'|) = \mathcal{S}(\mathcal{D}, k/|\mathcal{D}|)$, we show in Section 6 that the difference between the two sets is very small in practice. In particular, we will show that, generally speaking, the higher the density of the dataset, the higher the number of preserved sequential patterns. For this reason, we give a formal definition of density which is based on our prefix tree representation.

Definition 8 (Density of a sequence dataset) Given a sequence dataset $\mathcal{D} = \{S_1, \ldots, S_N\}$, its *density* is the compression ratio introduced by the related prefix tree $\mathcal{T} = (\mathcal{N}, \mathcal{E}, Root(\mathcal{T}))$, and is defined as

$$density(\mathcal{D}) = 1 - \frac{|\mathcal{N}|}{\sum_{i=1}^{N} |S_i|}$$

where $|\mathcal{N}$ is the number of nodes in $\mathcal{T}$, and $|S_i|$ is the length of sequence S_i.

When the compression introduced by the prefix tree is low (i.e., the dataset is sparse), some frequent subsequences tend to appear in different isolated and infrequent

branches which are likely to be pruned during the pruning step of our algorithm. Moreover, during the reconstruction step, a significant portion of these branches can not be recovered. On the other hand, in denser datasets, this situations are more unlikely to happen, since many sequences are represented by a small part of the prefix tree. Indeed, a significant portion of frequent patterns is preserved.

5.2 Complexity Analysis

In this section we discuss the time complexity of the *BF-P2kA* algorithm. We use N to denote the number of sequences in $\mathcal{D}$, and n to denote the sum of lengths of sequences in $\mathcal{D}$, i.e., the size of $\mathcal{D}$.

Theorem 3 The *BF-P2kA* algorithm anonymizes a sequence database $\mathcal{D}$ in $O(n^2)$ time.

First of all, we observe that both the *PrefixTreeConstruction* function and the last step, i.e., the generation of the anonymized sequences, require $O(n)$ time. The time complexity of the anonymization step instead depends on the *TreeReconstruction* Algorithm that has to compute the LCS and the edit distance for each infrequent sequence. These two operations require $O(n^2)$ time using dynamic programming.

Assuming that the average length of the sequences is significantly smaller than the number of sequences in $\mathcal{D}$, we can assert that the overall complexity of our algorithm is $O(N^2)$. Notice that, in most real-world applications, the compression introduced by the prefix tree is significant (see Section 6). In these cases, the complexity of our approach turns out to be subquadratic. Finally, another influencing factor is the k parameter, but it can only be estimated experimentally. In Section 6 we analyze the running time of our algorithm for increasing values of k.

5.3 Running example

We present now an example which shows how our approach works. We consider the dataset of sequences presented in Fig. 1(a) and a minimum support threshold equal to 2. During the first phase of our method the algorithm builds the prefix tree depicted in Fig. 2(a), which represents the sequences in a more compact way.

During the anonymization step, the prefix tree is modified by the *TreePruning* procedure with respect to the minimum support threshold. In particular, this procedure searches the tree for all the highest nodes in the tree hierarchy with support less than 2 and returns the two nodes $\langle 12, S, 1\rangle$ and $\langle 13, D, 1\rangle$. Next, it selects the paths that contain these nodes and which start from the root and reach each leaves belonging to the subtrees of these nodes. Then, it generates all the sequences represented by these paths and inserts them into the list $\mathcal{L}_{cut}$, which now contains the sequences $(BKS, 1)$ and $(DEJF, 1)$.

Finally, the *TreePruning* procedure eliminates from the tree the subtrees induced by the infrequent nodes listed above and updates the support of the remaining nodes. The prefix tree obtained after the pruning step is shown in the Fig. 2(b).

The infrequent sequences within $\mathcal{L}_{cut}$ are then processed in this way: $(BKS, 1)$ increases the support of nodes $\langle 10, B, 2\rangle$ and $\langle 11, K, 2\rangle$, producing $\langle 10, B, 3\rangle$ and $\langle 11, K, 3\rangle$; $(DEJF, 1)$ increases the support of nodes $\langle 1, A, 6\rangle$, $\langle 7, D, 3\rangle$, $\langle 8, E, 3\rangle$ and $\langle 9, F, 3\rangle$.

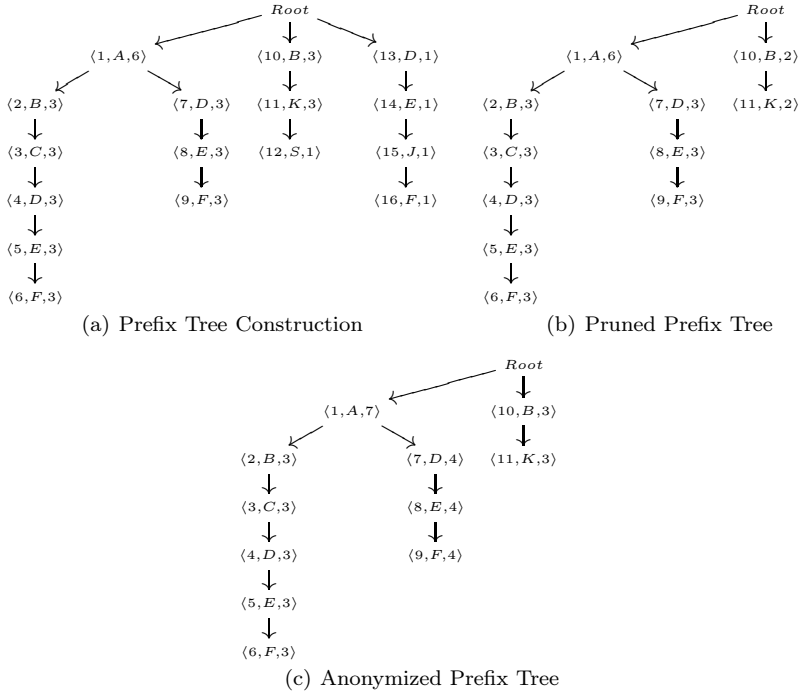
s_1	A B C D E F
s_2	A B C D E F
s_3	A B C D E F
s_4	A D E F
s_5	A D E F
s_6	A D E F
s_7	B K S
s_8	B K
s_9	B K
s_{10}	D E J F

(a) A dataset of sequences

s'_1	A B C D E F
s'_2	A B C D E F
s'_3	A B C D E F
s'_4	A D E F
s'_5	A D E F
s'_6	A D E F
s'_7	A D E F
s'_8	B K
s'_9	B K
s'_{10}	B K

(b) Anonymized sequences

Fig. 1 A toy example



(a) Prefix Tree Construction (b) Pruned Prefix Tree

(c) Anonymized Prefix Tree

Fig. 2 Prefix tree processing

Therefore we obtain $\langle 1, A, 7\rangle$, $\langle 7, D, 4\rangle$, $\langle 8, E, 4\rangle$ and $\langle 9, F, 4\rangle$.
The prefix tree obtained after the anonymization step is shown in Fig. 2(c). Finally, the *SequencesGeneration* procedure provides the anonymized sequence dataset shown in Fig. 1(b).

6 Experiments and results

In this section, we present some applications of our anonymization approach on several real-world datasets. The first bunch of experiments concerns a process-log datasets and a web-log dataset. The second group of experiments is related to several datasets of sequences obtained by preprocessing a huge set of moving objects. Finally, we report on a clustering task applied to a small process-log dataset.

Since our main goal is to preserve sequential patterns as much as possible, we compare the collections of pattern extracted before and after the anonymization process. To measure the similarity between two collection of patterns, we use two metrics. The first one is the popular F-Measure which is computed as $F = 2(precision \cdot recall)/(precision + recall)$, where the precision is computed as the ratio of σ-frequent patterns in $\mathcal{D}'$ which are also σ-frequent in $\mathcal{D}$, and the recall is computed as the portion of σ-frequent patterns in $\mathcal{D}$ that are also σ-frequent in $\mathcal{D}'$. However, even though a pattern belongs to both collections, it may happens that its support in $\mathcal{D}'$ is significantly different than its original support in $\mathcal{D}$. Hence, we introduce the second metric measures the pattern frequency similarity and is defined as:

$$SupSim = \frac{1}{\left|\hat{\mathcal{S}}(\sigma)\right|} \sum_{s \in \hat{\mathcal{S}}(\sigma)} \frac{\min\{freq_{\mathcal{D}'}(s), freq_{\mathcal{D}}(s)\}}{\max\{freq_{\mathcal{D}'}(s), freq_{\mathcal{D}}(s)\}}$$

where $\hat{\mathcal{S}}(\sigma) = \mathcal{S}(\mathcal{D}', \sigma) \cap \mathcal{S}(\mathcal{D}, \sigma)$. It quantifies the similarity in support of each pattern belonging to both collections.Both measures range between 0 and 1. When two collections of subsequences are identical, the two measures are all equal to 1. To extract the collections of sequential patterns, we used PrefixSpan [30] in all our experiments. All the experiments were performed on a 2.0GHz Intel Core 2 Duo processor with 2GBytes RAM, running Windows.

6.1 Experiments on process-logs and web-logs

We worked on a process log and a web-log datasets. The first dataset comes from the usage of a real-world system developed by Think3[3], which is an object repository managing system that allows the users to operate on the same objects from different locations. This dataset contains about 300,000 transactions on 11 tasks, for a total of about 1 million of performed tasks. The logs span along 6 months of executions. The second comes from webserver logs for msnbc.com and news-related portions of msn.com for the entire day of September, 28, 1999 (Pacific Standard Time)[4]. Each sequence in the dataset corresponds to page views of a user during that twenty-four hour period. Each event in the sequence corresponds to a user's request for a page. Requests are recorded at the level of 17 page categories (as determined by a site administrator). The number of total event sequences is about 1 million. In the following we refer to these datasets as think3 and msnbc datasets, respectively.

Due to PrefixSpan limitations, we preprocessed the think3 and msnbc in order to consider only sequences with at most 51 and 25 events respectively. We also removed all small sequences (less than 4 and 5 events respectively). To extract the collection of

[3] http://www.think3.com

[4] http://archive.ics.uci.edu/ml/

Dataset	$\|\mathcal{I}\|$	$\|\mathcal{D}\|$	Length	σ	Compr.
think3	11	61367	7.34	0.025	96.7%
msnbc	17	286260	9.37	0.015	58.0%

Table 1 Dataset statistics

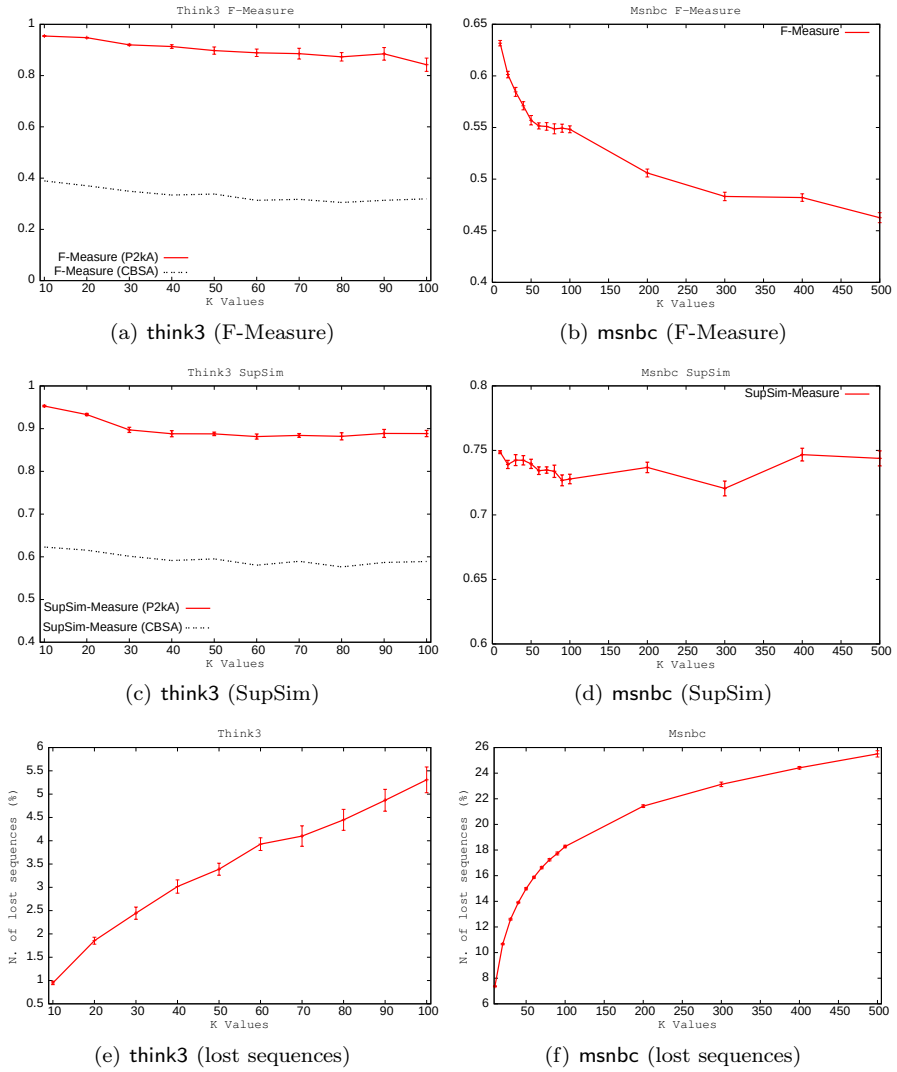
frequent patterns, we used a minimum support threshold of 1.5% for msnbc and 2.5% for think3, in order to make the pattern comparison task feasible. Tab. 1 reports some basic statistics of the three datasets, together with the compression rate introduced by the prefix tree.

Our experiments were conducted as follows: we first anonymized the three datasets using different values of k. Since the output of our algorithm depends on the order in which sequences are processed, we launched *BF-P2kA* 20 times for each value of k. Then, for each value of k we compared the collection of frequent patterns extracted from the original dataset and the collection extracted from all the k-anonymized dataset. We averaged the results on the 20 trials for each value of k. For think3, we also compared our results with those obtained by applying the condensation-based anonymization approach presented in [7]. At the best of our knowledge, this is the only candidate competitor in this area. Notice however that this work is intended as a way to provide anonymized strings, and preserve some statistics for classification purposes, rather than a way to preserve frequent sequential pattern results.

The results are reported in Fig. 3. The support similarity score is satisfactory in general, and even for high anonymization thresholds the values of SupSim are quite high. Concerning the F-Measure measure, for think3 it is quite high for any value of k (see Fig. 3(a)). For msnbc the value of F (see Fig. 3(b)) decrease slowly from a value of 0.63 (for $k = 10$) down to 0.47 (for $k = 500$). As expected, the condensation-based method is not able to preserve local sequential patterns: the values of F are always below 0.4, while *SupSim* is always around 0.60 (see Fig. 3(c)). We also measured the percentage of sequence that were lost after the anonymization process. For think3 the loss is limited to a minimum of 1% for $k = 10$, till a maximum of 5% for $k = 100$ (see Fig. 3(e)). For msnbc the loss is higher but still reasonable: it increases logarithmically from a minimum of 5% for $k = 10$ to a maximum of 25% for $k = 500$ (see Fig. 3(f)). Notice also that all the measures are quite stable: the processing order of sequences does not influence significantly the quality of the results.

To give an idea of the influence of the k parameter on time performances, Fig. 4 shows the running time of our algorithm applied to msnbc dataset. The processing time decreases exponentially with increasing values of k and, in this particular case, it starts to be acceptable for $k = 100$, which is a reasonable value for this large dataset.

We also measured some basic statistics for assessing the distortion introduced by our anonymization algorithm. In particular, we computed the distribution of sequence lengths before and after anonymization, and the frequency of each single item (in terms of item count divided by the total number of items in the datasets). The results are depicted in Fig. 5. For think3, the distortion is always very low, for both $k = 50$ and $k = 100$. By comparing these statistics with those obtained by processing the same dataset using the condensation-based method, we notice that the length distortion introduced by this method is quite similar to the one induced by our algorithm (see Fig. 5(c)). The item frequency distortion however is sensibly higher (see Fig. 5(d)) for some items. In the case of msnbc, due to its sparseness, several long (and infrequent)



(a) think3 (F-Measure) (b) msnbc (F-Measure)

(c) think3 (SupSim) (d) msnbc (SupSim)

(e) think3 (lost sequences) (f) msnbc (lost sequences)

Fig. 3 Anonymization results for a process-log and a web-log datasets

sequences are transformed in shorter ones (see Fig. 5(e)). However the trend of the distribution is maintained. Finally, the item frequency distortion is acceptable in general (see Fig. 5(f)). We do not report error bars here because the results were extremely stable (the standard deviation was always below 10^{-4}).

6.2 Experiments on moving objects

Here, we present an application to a moving object dataset. Object trajectories are first transformed into sequences of crossed locations, and then processed with our anonymization approach.

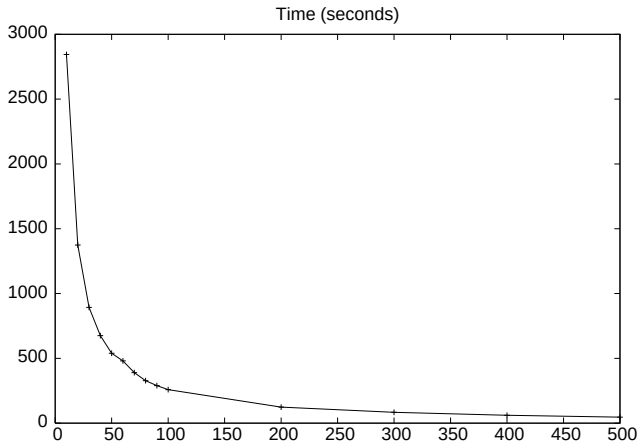


Fig. 4 Running time on msnbc

Density	Regions	$\|\mathcal{D}\|$	Length	Compr.
0.010	113	82341	8.327	60.4%
0.035	31	28663	9.152	86.3%
0.038	16	23744	6.239	93.8%

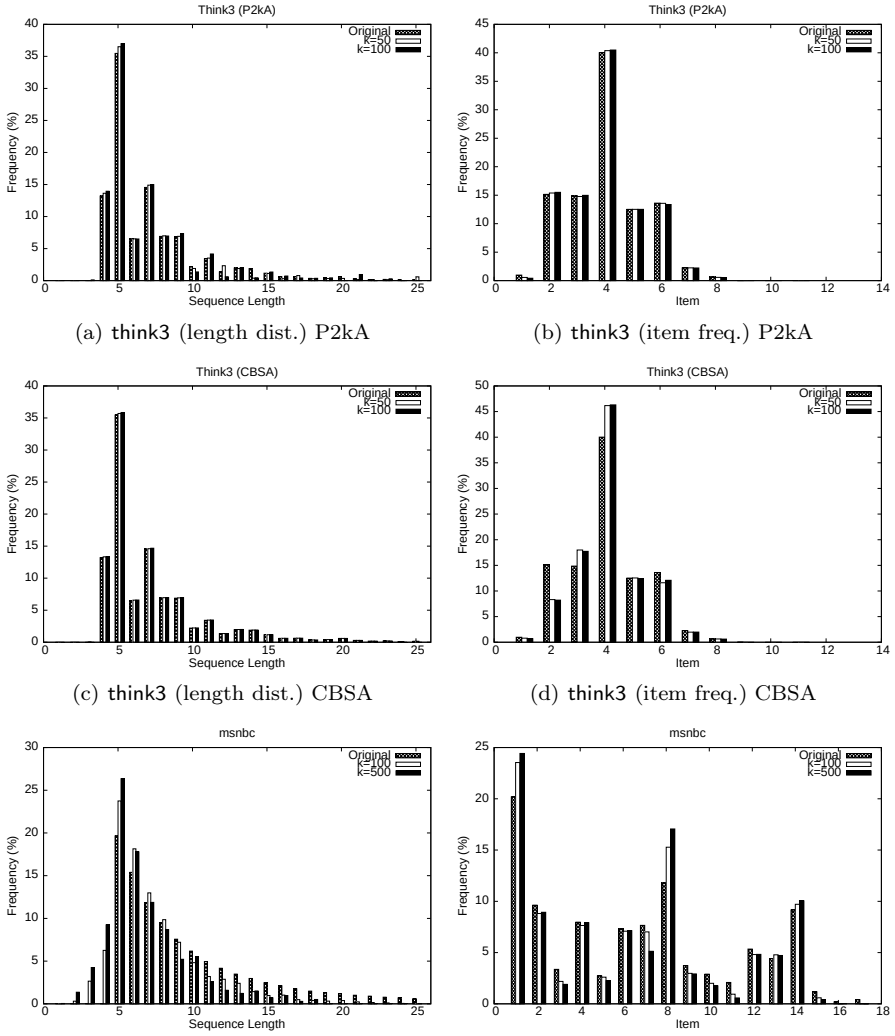
Table 2 Statistics for different density threshold

We explain now the procedure used to obtain the input datasets. We used a set of GPS trajectories made available within the European project *GeoPKDD*[5]; these data are acquired from private cars equipped with a GPS receiver under a special car insurance contract, and cover a whole week in the metropolitan area of Milan, Italy. Each trajectory is a sequence of pairs of coordinates x and y with relative timestamp. To perform our experiments, we preprocessed these data using the definition of Regions of Interest (RoI's) given in [18], where the authors discretize the working space through a regular grid with cells of small size. Then the density of each cell is computed by considering each single trajectory and incrementing the density of all the cells that contain any of its points. Finally a set of RoI's is extracted by means of a simple heuristics using a density threshold.

As a result, a set of Roi's provides a coverage of dense cells through different sized, disjoint, rectangular regions with some form of local maximality. Once the set of RoI's has been extracted, we preprocess all the input trajectories translating each one from a sequence of points to a sequence of RoI's. The order of visit is maintained by means of timestamps. An example of this simple procedure of translation is shown in Fig. 6 — on the left we can see all the trajectories and a set of RoI's extracted; on the right we show a trajectory and we evidence which RoI's it crosses. This new dataset represents the input dataset for the anonymization algorithm.

The datasets used in our experiments are built using all the trajectories in the dataset described above with different density thresholds. These values have been chosen in order to obtain an adequate number of RoI's, since low density values correspond

[5] `http://www.geopkdd.eu`



(a) think3 (length dist.) P2kA (b) think3 (item freq.) P2kA

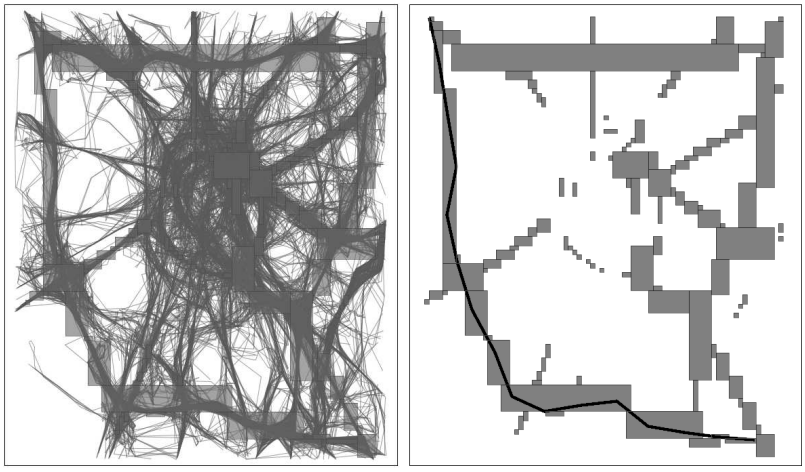
(c) think3 (length dist.) CBSA (d) think3 (item freq.) CBSA

(e) msnbc (length dist.) (f) msnbc (item freq.)

Fig. 5 Dataset statistics before and after anonymization

to few big regions, and higher values produce few small regions. In that way, we obtain different sets of RoI's meaning different sets of items in the input sequences. Tab. 2 summarizes the datasets used in our experiments (together with the compression introduced by the prefix tree). Notice that the number of trajectories is different among the datasets because trajectories that do not cross any region are dropped.

Our experiments were conducted as follows: we first anonymized the three datasets using values of k between 10 and 500. Then, for each value of k we compared the collection of frequent patterns extracted from the original dataset and the collection extracted from the k-anonymized dataset using different support thresholds. Chosen support threshold are such that the number of patterns is significant for our comparison



(a) (b)

Fig. 6 Trajectories and regions.

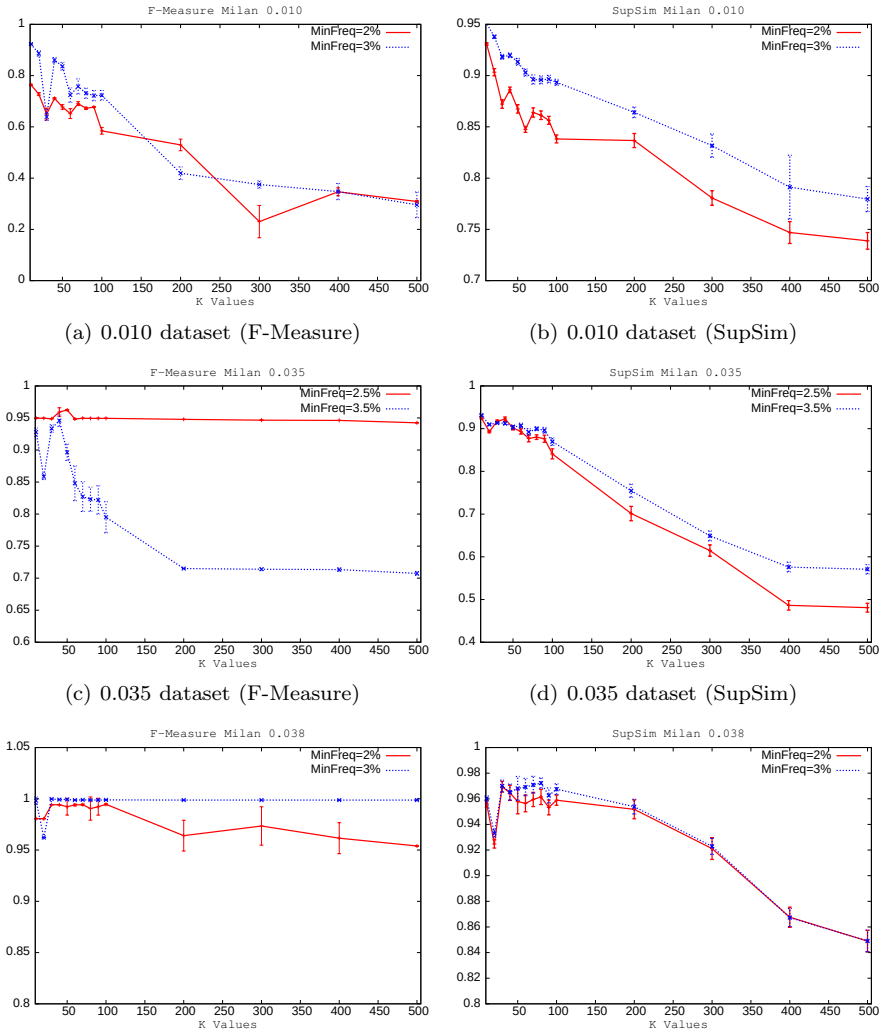
purposes, and the one-to-one comparison is feasible in a reasonable amount of time, given our computational resources.

In Fig. 7 and Fig. 8 we report the results of our experiments. As expected, the denser the datasets, the better the results. However, for high support thresholds and relatively low values of k (say, less than 100), our algorithm performs well with all the datasets, both in terms of F-Measure and support similarity. For higher values of k, we have a low accuracy and a good support approximation in the first dataset (Fig. 7(a) and 7(b)), a good accuracy and a low support approximation in the second dataset (Fig. 7(c) and 7(d)), and a very good accuracy and support approximation in the third one (Fig. 7(e) and 7(f)). Notice that, even for high support thresholds, the number of extracted patterns is still significant for our evaluation. Finally, the number of lost sequences (see Fig. 8) is quite low for the first two datasets (less than 10% and 20% respectively for any value of k), and it is still reasonable for the last dataset when $k < 100$.

6.3 Clustering results

So far, we have discussed the impact of our k-anonymization approach on the collections of local patterns. Although the main goal of the *BF-P2kA* algorithm is to preserve the collection of extracted sequential patterns, we also investigated its impact on clustering results. For this task, we used the *HAC* algorithm [27], which is a hierarchical agglomerative approach using edit distance (see Definition 6). In particular, we use a variant which stops the aggregation process when a desired number of clusters is obtained. Since this algorithm requires high computational time and memory resources, applying it to the datasets described in Sections 6.1 and 6.2 turns out to be unfeasible. Hence, we apply this algorithm to a tractable data sample related to the ProM tutorial[6]

[6] `http://www.processmining.org`



(a) 0.010 dataset (F-Measure)

(b) 0.010 dataset (SupSim)

(c) 0.035 dataset (F-Measure)

(d) 0.035 dataset (SupSim)

(e) 0.038 dataset (F-Measure)

(f) 0.038 dataset (SupSim)

Fig. 7 Values of F-Measure and SupSim for different location datasets

(a process mining toolkit), containing 2104 syntectic logs from a technical assistance chain process of a phone company.

In Fig. 9 we report the (normalized) distance matrices, sorted by cluster labels, for the original dataset and for k=50 and k=100 (for this experiment we processed sequences in their original order). Notice that, after the anonymization process, the order of the data instances is not strictly preserved. For this reasons, some blocks in the matrix are shifted. Nevertheless, the clustering structure does not change significantly. For $k = 50$, the two distance matrices are similar, while, for $k = 100$, the distances within a same cluster are smoothed, but the clustering structure is preserved. Actually, the separation of clusters becomes sharper after anonymization: this is a natural conse-

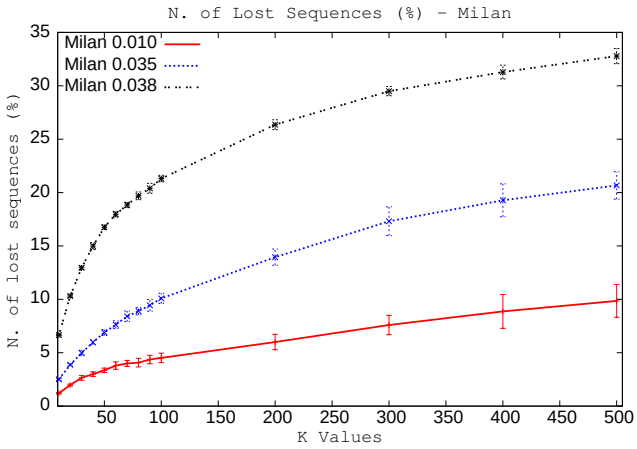


Fig. 8 Lost sequences on Milan traffic data

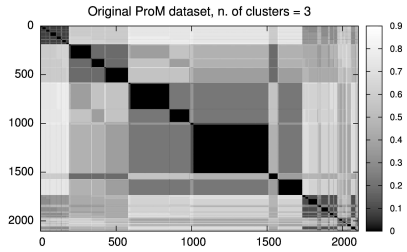
quence of our approach, which hides the infrequent subsequences. From the clustering point of view, this turns out to be a kind of outlier removal or smoothing step. As a future work, it is worth investigating the aspects of clustering in a k-anonymization framework more accurately.

7 Considerations on societal and legal impact

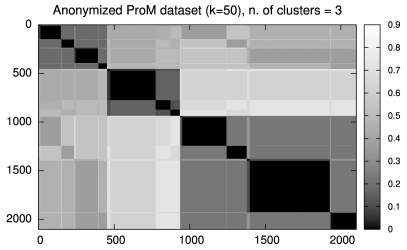
Privacy is highly critical in different scenarios. In a European company it is not allowed to collect private data for any reason even for research and/or business purposes. The European Directive requires that users must be informed and explicitly agree that the specified data can be collected and used for specific goals. Clearly, this is a very big limitation considering the great value of the knowledge that is intrinsically hidden in the human data left every day by people as digital traces. The opportunities and the ability of transforming these data, describing in details our society, into knowledge could help the understanding of many complex phenomena. The problem is that for doing that we need to collect and analyze these *big data* and the privacy regulation tends to limit these opportunities with the aim of protecting one of the most important individual personal right, the *privacy's right*.

The framework proposed in this paper together with our theoretical and empirical results, presented on the above sections, show that by following the *privacy-by-design* principle, adapted in the data mining domain, we can find a data environment that respects the privacy's right at individual level while enabling the free usage of data. As a consequence the application of our technique permits the diffusion of novel opportunities in terms of applications and services that exploit the availability of big data. In other words, our framework is an example of how it is possible to set an environment where we can provide services and applications useful for the collectivity, thanks to the contribution of each user, who provides his/her activity logs, while protecting his/her right to have a private life.

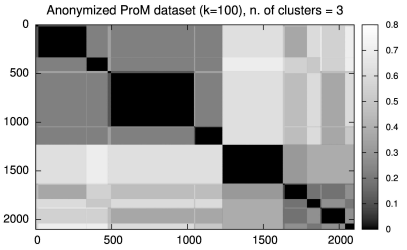
In our specific context, where data have a sequential nature, making data anonymous means to have the possibility of collecting and analyzing sequence data such



(a) Original data



(b) k=50



(c) k=100

Fig. 9 Clustering results on ProM dataset (3 clusters)

as web logs, spatio-temporal data, query logs without putting at risk the privacy of individuals represented in the data. Unfortunately, many examples of re-identification from supposedly anonymous data have been reported in the scientific literature and in the media, from health records to query logs to GPS trajectories. A very famous example is the case of the AOL user re-identified by analyzing the 20 million Web search queries collected by AOL and released on the Internet in 2006 [11]. The application of our privacy transformation would make the re-identification of the user hard and controllable, because after the anonymizatoin process we have a theoretical guarantee on the probability of re-identification of each user.

Our framework has also an impact on the legal ground; its main goal is to make anonymous the user personal data according to Directive 1995/46/EC. To this end, the framework is defined according to the content of this directive and provides technological tools to legal experts for the evaluation of data anonymity. The key concepts of the directive that we took into consideration in the design of our framework are: *anonymous data*, *personal data* and *identifiable person*. The concept of "anonymous data" is not explicitly reported in the directive; but, this notion can be derived from the definition of "personal data" that is defined as any information relating to an identified or identifiable natural person *(data subject)*. Here, an "identifiable person" is defined as a person who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity. The directive also states that the princi-

ple of protection must apply to any information concerning an identified or identifiable individual while shall not apply to data rendered anonymous in such a way that *the data subject is no longer identifiable*.

In our context, we identified "personal data", i.e., the sequences of user actions which can be both sensitive information to be protected and knowledge to be used as a means of re-identification (as discussed in Section 3.2), and designed a data transformation strategy for making any data subject no longer identifiable by a sequence of known user actions.

Our framework also provides a criterium for measuring data anonymity, i.e., the *probability of success of a sequence linking attack*. This represents a possible solution to one of the main legal open problems: "how to measure the degree of anonymity in the data".

Lastly, the application of our framework, based on the principle of the privacy by design, is an example of how this paradigm can help accountable organizations in their responsibility for personal information protection. An accountable organization should identify privacy risks and appropriately taking them into account in developing its business models and related technologies. In few words it should mitigate and minimize the risks and effects of privacy breaches. This result is guaranteed by our framework since we consider the possible privacy risks in the publication of sequence data and provide both a defense strategy against them and a tool for the risk evaluation.

8 Related work

A lot of recent research works have focused on techniques for privacy-preserving data mining [9] and privacy-preserving data publishing. Important techniques include perturbation, condensation, and data hiding with conceptual reconstruction. The first step before data publishing is to remove the personally identifying information. In [32] (and much earlier in statistics by [15]), it has been shown that removing personally identifying information is not enough to protect privacy. In this work, Samarati and Sweeney propose a classification of the attributes in *quasi-identifiers* and sensitive attributes. Moreover, [31] propose k-anonymity to generalize the values of quasi-identifier attributes in each record so that it is indistinguishable with at least $k-1$ other records with respect to the quasi-identifier. Although it has been shown that the k-anonymity framework presents some flaws and limitations [21], and that finding an optimal k-anonymization is NP-hard [24], the k-anonymity model is still practically relevant and in recent years a large research effort has been devoted to develop algorithms for k-anonymity [12, 20].

Anonymity in spatio-temporal data. Recently, privacy-preserving data mining has been studied in conjunction with spatio-temporal data and trajectory mining [19, 14]. k-anonymity is the most popular method for the anonymization of spatio-temporal data. It is often used both in works concerning privacy issues in location-based services (LBSs) [13, 23] and in those on anonymity of trajectories [5, 28, 38, 36]. In the work presented in [5], the authors study the problem of privacy-preserving publishing of moving object database. They propose the notion of (k, δ)-anonymity for moving objects databases, where δ represents the possible location imprecision. In particular, this is a novel concept of k-anonymity based on co-localization that exploits the inherent uncertainty of the moving objects whereabouts. In this work authors also propose an approach, called *Never Walk Alone*, for obtaining a (k, δ)-anonymous moving objects

database. The method is based on trajectory clustering and spatial translation. [28] address privacy issues regarding the identification of individuals in static trajectory datasets. They provide privacy protection by: (1) first enforcing k-anonymity, meaning every released information refers to at least k users/trajectories, (2) then reconstructing randomly a representation of the original dataset from the anonymization. [38] study problem of k-anonymization of moving object databases for the purpose of their publication. They observe the fact that different objects in this context may have different quasi-identifiers and so, anonymization groups associated with different objects may not be disjoint. Therefore, a novel notion of k-anonymity based on spatial generalization is provided. In this work, authors propose two approaches in order to generate anonymity groups that satisfy the novel notion of k-anonymity. These approaches are called *Extreme Union* and *Symmetric Anonymization*. Finally, we are also aware of very recent work [36], where Terrovitis and Mamoulis suggested a suppression-based algorithm that, given the head of the trajectories, reduces the probability of disclosing the tail of the trajectories. This work is based on the assumption that different attackers know different and disjoint portions of the trajectories and the data publisher knows the attacker knowledge. So, the proposed solution is to suppress all the dangerous observations in the database.

Sequence Anonymity. An interesting work is presented by [6] where the authors propose a hiding technique. In particular, they address the problem of hiding sequences considered sensitive from a sequential database. A first work attacking the problem of limiting disclosure of sensitive rules by reducing their significance, while leaving unaltered or minimally affecting the significance of others, non-sensitive rules is [10]. One of the most important contributions of this paper is the proof that finding an optimal sanitization of a dataset is NP-hard. A heuristic using greedy search is thus proposed. In [16] the objective is to hide individual sensitive rules instead of all rules produced by some sensitive itemsets. The work in [33] proposes two distortion-based heuristic techniques for selectively hiding sensitive rules.

In literature, the work that is more related to ours is [7], where authors propose a condensation model for anonymization of string data inspired to k-anonymity. This approach differs from ours since it generates a synthetic dataset based on a probabilistic model. Moreover, authors do not provide any formal upper bound for the privacy guarantees. Finally, in our work we want to preserve the sequential pattern mining results while Aggarwal et al. are interested in preserving the behavior of the anonymous dataset in the context of classification applications.

An interesting work is presented in [22] where Malin introduces a formal model for privacy protection, called k-unlinkability, to prevent trail re-identification in distributed data. The property introduced differs from ours as it based on the k-map [35] notion adapted for distributed environments, while our notion of k-harmful sequence is related to k-anonymity. k-map differs from the k-anonymity as in the first model each record in the released dataset links to at least k entities in the real world, while in the second model each record occurs at least k times in the released data.

Finally, some research works have focused on anonymity on transaction databases but these techniques are not suitable for sequence data [17,37].

9 Conclusion and future work

We introduced a new definition of k-anonymity for personal sequential data which provides an effective privacy protection model, and presented a method that transforms sequential datasets into a k-anonymous form, while preserving the utility of data with reference to a variety of analytical properties. For the design of the proposed framework for anonymous publishing of sequence data we have applied the Privacy-by-design principle after defining its articulation in the data mining field. Through a wide set of experiments with various real-life sequential datasets, we demonstrated that the proposed technique substantially preserves sequential pattern mining results both in terms of number of extracted patterns and their support; results are extremely interesting in the case of dense datasets. Although we developed our method with the main goal of preserving pattern mining, we also found evidence that various analytical properties of the original data are preserved, including the distribution of sequence lengths and the frequency of individual elements, as well as the original clustering structure.

Clearly, the proposed anonymization method is not the only possible instance of the proposed model, and more research is needed to investigate whether better alternatives exist to produce a k-anonymous version of high quality. Certainly, our proposed solution compares favorably with other first-cut possibilities. Among those, we mention the possibility of publishing directly the sequential patterns that are found frequent w.r.t. to the anonymity threshold k; this option is weak for two reasons: first, only patterns (and associated support) are published, not real data, and second, mining at very low support thresholds generally yields an output of unmanageable size, as we verified experimentally on our datasets, even for rather large values of k. Another option would be to try and construct a new dataset that admits the desired collection of frequent patterns as a result, an operation known as *inverse mining*, already shown to be a high complexity task already in the case of itemset mining [25]; this option is therefore even more unpractical than the previous. Our method, instead, has a quadratic worst-case complexity (in the size of the database), which is often over-pessimistic in many practical cases with dense datasets.

Further research will also investigate new approaches to preserve pattern mining results also in other challenging contexts, such as sparse datasets or long sequences. One possible strategy might require the usage of a different and more compact data structure, instead of the prefix tree which is used here. Moreover, another investigation possibility could be oriented to a relaxed privacy constraint. Instead of guaranteeing the full satisfaction of k-anonymity, we could enable better pattern mining results despite of a less aggressive (and slightly more risky) pruning step.

References

1. Article 29 data protection working party and working party on police and justice, the future of privacy: Joint contribution to the consultation of the european commission on the legal framework for the fundamental right to protection of personal data. 02356/09/en, wp 168 (dec. 1, 2009), http://ec.europa.eu/justice/policies/privacy/docs/wpdocs/2009/wp168_en.pdf.
2. Privacy by design resolution. *International Conference of Data Protection and Privacy Commissioners (2010)*, Jerusalem, Israel, October 27-29, 2010.
3. O. Abul, M. Atzori, F. Bonchi, and F. Giannotti. Hiding sensitive trajectory patterns. In *Proc. IEEE ICDM Workshops*, pages 693–698, 2007.

4. O. Abul, M. Atzori, F. Bonchi, and F. Giannotti. Hiding sequences. In *Proc. IEEE ICDE Workshops*, pages 147–156, 2007.
5. O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proc. IEEE ICDE*, pages 376–385, 2008.
6. Osman Abul, Maurizio Atzori, Francesco Bonchi, and Fosca Giannotti. Hiding sequences. In *ICDE Workshops*, pages 147–156, 2007.
7. Charu C. Aggarwal and Philip S. Yu. A framework for condensation-based anonymization of string data. *Data Min. Knowl. Discov.*, 16(3):251–275, 2008.
8. R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. IEEE ICDE*, pages 3–14, 1995.
9. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. ACM SIGMOD*, pages 439–450, 2000.
10. M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. S. Verykios. Disclosure limitation of sensitive rules. In *Proc. KDEX'99*, pages 45–52, 1999.
11. M. Barbaro and T. Zeller Jr. A face is exposed for aol searcher no. 4417749. *The New York Times*, August 9, 2006.
12. R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Proc. IEEE ICDE*, pages 217–228, 2005.
13. Claudio Bettini and Sergio Mascetti. Preserving k-anonymity in spatio-temporal datasets and location-based services, 2006.
14. F. Bonchi, Y. Saygin, V. S. Verykios, M. Atzori, A. Gkoulalas-Divanis, S. V. Kaya, and E. Savas. *Privacy in Spatiotemporal Data Mining. In [19]*, pages 297–329. Springer, 2008.
15. T. Dalenius. The invasion of privacy problem and statistics production — an overview. *Statistik Tidskrift*, 12:213–225, 1974.
16. E. Dasseni, V. S. Verykios, A. K. Elmagarmid, and E. Bertino. Hiding association rules by using confidence and support. In *Proc. IHW 2001*, pages 369–383, 2001.
17. Gabriel Ghinita, Yufei Tao, and Panos Kalnis. On the anonymization of sparse high-dimensional data. In *ICDE*, pages 715–724, 2008.
18. F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *Proc. ACM SIGKDD*, pages 330–339, 2007.
19. F. Giannotti and D. Pedreschi, editors. *Mobility, Data Mining and Privacy*. Springer, 2008.
20. K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proc. IEEE ICDE*, page 25, 2006.
21. A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. *L*-diversity: Privacy beyond *k*-anonymity. *TKDD*, 1(1), 2007.
22. Bradley Malin. k-unlinkability: A privacy protection model for distributed data. *Data Knowl. Eng.*, 64(1):294–311, 2008.
23. Sergio Mascetti, Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. k-anonymity in databases with timestamped data. In *TIME*, pages 177–186, 2006.
24. Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *PODS '04*, pages 223–228, New York, NY, USA, 2004. ACM.
25. Taneli Mielikainen. On inverse frequent set mining. In *2nd Workshop on Privacy Preserving Data Mining (PPDM 2003)*, pages 18–23, 2003.
26. Anna Monreale. Privacy by design in data mining. *Ph.D. Thesis at Department of Computer Science, University of Pisa,*, 2011.
27. M. Nanni. Speeding-up hierarchical agglomerative clustering in presence of expensive metrics. In *Proc. PAKDD*, LNCS 3518, pages 378–387. Springer, 2005.
28. M. E. Nergiz, M. Atzori, and Y. Saygin. Perturbation-driven anonymization of trajectories. Technical Report 2007-TR-017, ISTI-CNR, Pisa, Italy, 2007. 10 pages.
29. Federal Trade Commission (Bureau of Consumer Protection). Preliminary staff report, protecting consumer privacy in an era of rapid change: A proposed framework for business and policy makers, at v, 41 (dec. 2010), http://www.ftc.gov/os /2010/12/101201privacyreport.pdf.
30. J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, and M. Hsu U. Dayal. Prefixspan: Mining sequential patterns by prefix-projected growth. In *Proc. IEEE ICDE*, pages 215–225, 2001.
31. P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *Proc. PODS*, page 188, 1998.
32. P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, March 1998.

33. Y. Saygin, V. S. Verykios, and C. Clifton. Using unknowns to prevent discovery of association rules. *SIGMOD Record*, 30(4):45–54, 2001.
34. European Data Protection Supervisor. Opinion of the european data protection supervisor on promoting trust in the information society by fostering data protection and privacy. (mar. 18, 2010).
35. L. Sweeney. Computational disclosure control: a primer. *Ph.D. thesis, Dept. of Electrical Eng. and Computer Science, MIT*, 2001.
36. Manolis Terrovitis and Nikos Mamoulis. Privacy preservation in the publication of trajectories. In *MDM*, pages 65–72, 2008.
37. Yabo Xu, Benjamin C. M. Fung, Ke Wang, Ada Wai-Chee Fu, and Jian Pei. Publishing sensitive transactions for itemset utility. In *ICDM*, pages 1109–1114, 2008.
38. Roman Yarovoy, Francesco Bonchi, Laks V. S. Lakshmanan, and Wendy Hui Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *EDBT*, pages 72–83, 2009.