Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-019-01515-8

ORIGINAL RESEARCH

Cover-up: a probabilistic privacy-preserving graph database model

Klara Stokes[1]

Received: 20 February 2019 / Accepted: 23 September 2019

Abstract
A new graph database model is introduced that allows for an efficient and straightforward privacy-preserving mechanism. A probabilistic graph database model is also proposed, perhaps less suitable for lossless storage, but adapted for the use of statistical analysis that preserves the privacy of the individuals behind the data. Parallels are drawn to concepts in combinatorics such as clique complexes and incidence geometries.

Keywords Privacy · Graph database · Clique complex · Incidence geometry

1 Introduction

In this article, a data set is a collection of properties of objects. To the data set, there is attached meta data, or to be more precise, the types of the properties (attributes). For example, if the objects are the set of books in a library, then the data set can be the collection of the titles, the authors, the number of pages, the shelf the book can be found on, and so on. The types (or the attributes) of the data are then "title", "author", "number of pages", "shelf". The properties are the actual titles, authors and so on. We will assume that each object can have only one property of each type.

Definition 1 (*Data graph*) A data graph (V, E) representing a data set D with s types is an s-partite graph, such that the vertices in the partition V_i represent the properties of type i, and the data in D is represented by a set of labeled edges E: there is an edge between two vertices if there is some object with both properties, and the edge is labeled with the proportion of objects having the pair of properties in question.

If the proportion is always taken with respect to an absolute number n of total objects in the data set, then the data graph can equivalently be represented as the number n together with an s-partite graph with multiple edges (between vertices) so that each edge between two vertices corresponds to an object with both properties.

This data graph is an example of a graph database model. Graph databases form part of the NoSQL paradigm, and are often suitable for large amounts of data, featuring good scalability and flexibility in front of different use cases.

The demand for data privacy protection is increasing, following the explosive development of data collection in society. Nothing suggests that data collection will cease, but rather will it evolve and transform. Privacy protection can be motivated by law or by commercial reasons. Indeed, customers with protected privacy are happy customers. Although many methods exist for data privacy protection, there is currently a great need for reliable and scalable methods that allow for the general publication of privacy protected databases.

The purpose with this article is to suggest modifications of the graph database model in Definition 1 that allow to represent large quantities of data about individuals in a naturally scalable and useful way, while also preserving the privacy of the individuals behind the data. First we present a deterministic model that preserves the overall features of the data, while erasing or modifying individual traces of data with trivial (hence fast) operations. The data structure allows the privacy preserving operations to be precise enough to put the information loss where it should be: at the private information. The method preserves the big picture and returns a useful database reflecting the general features of the population. Additionally, it is possible to obtain nice theoretical control over the information loss (in the sense of understanding what is lost).

✉ Klara Stokes
klara.stokes@mu.ie

[1] National University of Ireland Maynooth, Maynooth, Ireland

Published online: 28 October 2019

Definition 2 A *cover-up privacy-preserving data graph with parameter k* of the data set D is the graph obtained from the data graph of D by modifying the label indicating the proportion of objects in the database with a given pair of properties from $\frac{m}{n}$ to $\frac{ik}{n}$ where ik is the result of rounding m to the closest integer multiple of k.

The label $\frac{ik}{m}$ can be seen as an approximation of the probability that there is an object with the two properties.

The second model is a probabilistic generalization of the first. Instead of labeling edges with simple probabilities, the edges are here labeled with discrete random variables and probability distributions.

Definition 3 A *probabilistic privacy-preserving data graph* of the data set D is an n-partite complete graph $K_{V_1,\ldots,V_n} = (V, \tilde{E})$ with the same vertex set as the data graph $G(D) = (V, E)$ and a discrete random variable with a discrete probability distribution on each edge.

2 Related work

2.1 Database models

2.1.1 Graph databases

Hierarchical database models were popular from their introduction in the 1960's until the relational database model became popular (Knuth 1968; Long et al. 2000). They are still the preferred model in certain applications. A hierarchical database structures data in form of a tree, an acyclic graph. Every record is a node, and relations between records are modeled as edges. Records can have types, and the possible entries of a record depends of the type. For a more recent reference for the use of trees as data structures, see (Lima 2014).

The Network Model is more general than the hierarchical model, in allowing cycles in the graph. It was first described by Bachman in the end of the 1960s (Bachman 1973), and was included in CODASYL. The model was developed further until the beginning of the 1980s, but did not have substantial influence on commercial products. Instead, the commercial focus was almost completely on the relational database model.

Graph database models became important in the 1990s again, when they were used to index web pages. Ten years later, several commercial graph databases were available, featuring the ACID (Atomicity, Consistency, Isolation and Durability) properties. As social networks analysis became increasingly popular, graph databases grew increasingly important. Today there is a long list of commercial graph databases available at the market.

Graph databases belong to the NoSQL paradigm (Robinson et al. 2013; Deka 2017). This simply means that a graph database model is different from a relational database model. The two most commonly used graph models are the labeled-property graph and the resource description framework (RDF). Both these models feature *directed* graphs. The first model represent data as nodes using edges to model relations between data. Both the nodes and the edges can store information in form of properties and labels. The labeled-property graph model is currently used in many popular graph databases, as for example Neo4j. The RDF model treats every new data as a new node, allowing this node to have a type, if so desired. This model is for example used in the Open Graph protocol (2017).

The data graph model that is used in this article is similar to other graph database models, however not equal. The literature contains a vast collection of graph database models, indeed it is rather natural to model data in terms of graphs. One main difference from the two models just mentioned is that our data graph model uses an *undirected* graph. There are of course other models using undirected graphs, in particular such models are common in social network analysis. Also, in our data graph the data attached to an object is represented by a clique (a complete subgraph), while most graph databases use a tree for this purpose—the attributes of an object would not typically be connected, at least not only for the sake of being attributes of the same object.

Note that the particular features of the data graph model in this article, being rather simplistic and mathematical in nature, should not be considered restrictive, and many of the results can surely be generalized and modified in many ways to fit into different commercial design requirements.

2.1.2 Data in table form

Because most commercial databases are still based on the relational model, and because the graph data model used in this article probably differ from other graph data models known to the reader, it is of interest to compare the data graph model of this article, not only to graph database models, but also to relational database models.

When a data set is represented in a relational database model as a table, typically the rows represent the objects, the columns represent the types and the (i, j)th entry is the property of type j of the object i. For example, a table storing personal data would then typically have one row per person, and one column per each attribute (or type), such as "name", "age", and so on.

A table representation can be constructed from our graph data representation, by making one row for each *maximal clique* of the graph. If there is no unique identifier, then it could happen that a row was constructed that does not correspond to any object with

these properties. Imagine for example the data set with 3 types "name", "age" and "disease", 3 individuals $A = [John, 45, Heartattack]$, $B = [John, 34, Diabetes]$ and $C = [Sarah, 45, Diabetes]$. The graph has 2 nodes of each of the 3 types, hence a total of 6 nodes. The edges are (45, *Heartattack*), (34, *Diabetes*), (45, *Diabetes*) (*John*, 45), (*John*, 34), (*Sarah*, 45), (*John*, *Heartattack*), (*John*, *Diabetes*), (*Sarah*, *Diabetes*). There is therefore a maximal clique consisting of the nodes *John*, 45, and *Diabetes*, without there being any individual with these properties. However, if the data contains a unique identifier for each individual, then this situation will not occur. For example, if the identifiers A, B, C are given nodes in the data graph above, then they will be nodes of a forth type. The nodes A, *John*, 45, *Heartattack* form a maximal clique, as do the nodes B, *John*, 34, *Diabetes*, but there is no maximal clique that contains all the nodes *John*, 45, *Diabetes*, simultaneously.

Indeed, it is not difficult to see that if the data graph contains a unique identifier for each individual, then every maximal clique corresponds to a unique individual, and storage is lossless. However, since we want to build a privacy preserving database, we are typically not interested in ensuring that every individual is readily identifiable in the database. Instead, we want to achieve the contrary: we want to make it as hard as possible to identify an individual in the database.

2.1.3 Data in tensor form

A tensor is in computer science often regarded as a generalization of matrices to higher dimensions. When a data set is represented as a tensor, the $(i_1, \dots, i_n)$th entry represents a value for the i_1th to the i_nth attribute. For example, for $n = 3$, i_1 can be indexed by usernames, i_2 by device types, and i_3 by internet consumption per month. For a very small data set containing the internet consumption during 12 months of 3 users A, B and C, who all have a laptop and a smartphone using the internet, the data could be modeled as 12 matrices on top of each other, one for each month, with 3 rows labelled A, B and C and 2 columns labelled "Laptop" and "Smartphone", the entries being the internet consumption of the corresponding device belonging to the corresponding user, during the corresponding month.

Data in the form of a data graph as in Definition 1 can be transformed into tensor form by assigning each type one dimension. Then if there are is a clique labeled with m in the data graph, indicating that there are m objects with these properties, it would be represented by an entry of m in the tensor. The absence of a clique with certain indices, would be represented by a 0. Machine learning with tensor data has recently become very popular, and is implemented for example in TensorFlow (2019).

2.2 Data privacy

Data privacy is concerned with hiding sensitive data about individuals when publishing useful data about the same individuals. There are two main strategies available in the literature.

- Strategy 1. A protected database is released on which third parties can execute any query of their choise.
- Strategy 2. The database is kept secret, third parties can query the database and the answers that are returned are protected.

It has been readily proved that removing identifying data as name and social security number is not enough protection before releasing a database to third parties (this is sometimes referred to as naïve anonymization). The literature comprises a long list of more advanced protection methods providing privacy protection according to another shorter list of privacy definitions. Below a short summary of the concepts that are most relevant to the content of this article is provided. The reader is referred to (Torra 2017) for a more complete overview.

2.2.1 Noise addition

A simple protection method, with a long history, is noise addition. Following Strategy 1, some appropiate noise would be added the database for its subsequent release. The first extensive testing of noise addition is from 1983 (Spruill 1983). For an overview of different noise addition methods of this type, see (Brand 2002). If instead Strategy 2 is followed, then the noise is added to the answer of the query. The first methods implementing differential privacy followed this procedure, as do some methods preceeding the definition of differential privacy (Blum et al. 2005).

2.2.2 *k*-Anonymity

A database (in form of a table) satisfies k-anonymity when for each record in the database there are $k - 1$ other records that are indistinguishable (Samarati and Sweeney 1998; Samarati 2001; Sweeney 2002). Methods to produce a k-anonymous database X_p from an original database X include generalization and suppression (Sweeney 2002) and microaggregation (Domingo-Ferrer et al. 2001; Domingo-Ferrer and Mateo-Sanz 2002). For example, if we have a database that includes information on the age and the postal code of some patients in a hospital, a k-anonymized version of this database may use ages in ranges and counties instead of towns so that the change in

granularity allow to have at least k patients for any combination of (age range, county).

The concept of k-anonymity is simple and relatively easy to implement, but it also suffers from a series of problems. The first problem is *identification of the quasi-identifiers*. It is simply not reasonable to assume that it can be foreseen which combination of attributes will be used by an attacker for reidentification. One way to solve this problem is to assume that any combination of attributes is a potential quasi-identifier. However, applying k-anonymity using this assumption implies ensuring that every record is *repeated* at least k times, causing a lot of redundancy and information loss. A more flexible solution is to assume that any combination of at least t attributes is a potential quasi-identifier (Stokes 2012).

Another problem is *the risk of attribute disclosure*. This means that an attacker could be able to say that a given individual has a certain property of sensitive attribute, without necessarily being able to reidentify this individual. For example, this can happen if the individual belongs to an anonymity set in which every member has the same property for this sensitive attribute. The concept of l-diversity was invented to solve this problem, ensuring that at least l values of the sensitive attributes appear in each anonymity set (Machanavajjhala et al. 2006).

2.2.3 Differential privacy

In differential privacy (Dwork 2006, 2008), we submit queries about the data set, typically modelled as a table, and the answers are randomized to ensure privacy. A query is modelled as a random function Q from the data set D to some codomain S. For a given $\epsilon > 0$, differential privacy is then ensured if, for all pairs of subsets D_1 and D_2 of the data set, differing in one record (row), and for all subsets $S' \subseteq S$, the ratio $\frac{P(Q(D_1)\in S')}{P(Q(D_2)\in S')}$ is smaller than e^{ϵ}. By taking logarithms at both sides, we can rewrite this as

$$\log\left(P(Q(D_1) \in S')\right) - \log\left(P(Q(D_2) \in S')\right) < \epsilon.$$

The randomization of the query depends of the method that is used. Noise can be added to the data before the function Q is evaluated (in the domain), or it can be added to the answer (in the codomain). One could imagine that by adding the noise to the data, before the query is asked, the data could be released as a protected data set. Then, one could think, any possible query could be asked on the data set, while still preserving the privacy. Therefore, although differential privacy originally was defined for Strategy 2, it can in theory also be applied to Strategy 1. One of the more promising areas for practical use of differential privacy following Strategy 1 is the local model (Dwork and Roth 2014).

2.2.4 Integral privacy

Integral privacy is an alternative to differential privacy, designed to work with machine learning rather than with statistical methods (Torra and Navarro-Arribas 2016). A machine learning model satisfies integral privacy if the space of databases that could have generated that model is, for some given appropiate measure, large enough.

3 The cover-up privacy-preserving data graph model

3.1 Constructing the data graph

Given a data set D, consider the graph $G_T(D)$ defined so that there is a vertex for each property held by each object. Assuming that every object has n properties, each object is represented by a clique with one vertex of each of the n types, and this clique is disconnected from the rest of the graph. Since the graph consists of disconnected cliques, it is not a very interesting graph. However, note that this is very similar to representing the data in terms of a table: every clique is a record.

A data graph $G(D)$, following Definition 1, can be constructed from $G_T(D)$ by

- *Step 1*. Identify all vertices that have the same type and property.

The modification is in general not lossless. Although all edges are preserved, the information that is lost is the partition of the edge set that was present in the disjoint cliques of $G_T(D)$. In other words, it is in general no longer clear to which clique each edge belongs.

A database is often required to point out clearly which are the properties of a given object. For example, the doctor expects the database to return the medical history, given the social security number of the patient. Therefore many graph database models would use directed edges from the social security number to the nodes representing the medical history. Our data graph $G(D)$ can easily achieve the same by choosing the type *social security number* to be a unique identifier. Then the edges incident with a node of type *social security number* can be considered to be directed, in the same way as the edges of an undirected tree becomes directed when a root is chosen.

However, the topic of this article is data privacy, so we do not want to link information to individuals. Rather the goal is to detach the information from the individual.

3.2 Naïve anonymization of the data graph

Removing unique identifiers would correspond to naïve anonymization. But naïve anonymization has been proven to be insufficient as data protection method. More advanced methods are needed to protect the data.

Note the effect of naïve anonymization on the data graph $G(D)$. By Lemma 2, there may be cliques that are not contained in a maximal clique. This can imply that naïve anonymization results in a database in which there are maximal cliques of data for which there was no object in the data set. That would not happen in a table. Note that naïve anonymization is the first step in most anonymization methods, for example in the methods of this paper.

3.3 Protecting the data graph

Consider the data graph $G(D)$ again, representing multiple edges between a pair of nodes (a, b) as one labeled edge $(a, b, \frac{m}{n})$, where m is the number of edges between a and b, and n is the number of objects in the population represented in the database. If the label was removed, information loss would of course increase. The lost information is the number of objects in the data set that have a certain set of properties.

Now, data privacy is all about controlling information loss. We want to lose the unnecessary and information that is sensitive at individual level, while keeping the useful information about the large masses. It therefore seems reasonable to keep an approximation about how many objects in the data set have a certain set of properties, while we indeed would like to lose the information saying whether there is actually any individual with a certain set of properties or not.

We want to keep the big picture, while avoiding too much detail. We modify $G(D)$ in the following way:

- *Step 2.* Pick a k (typically at least 3) and for each edge of $G(D)$, round off m to the closest integer multiple ik of k, and label the edge with $\frac{ik}{n}$. Remove all edges labelled with zero.

We call the resulting labelled data graph $G_P(D)$, where P means protected. Note that the construction that we followed until this point, somehow implements a notion of anonymity, while still representing the data set.

The information loss occurring between $G(D)$ and $G_P(D)$ consists mainly of an uncertainty of the exact number of objects with a given pair of properties. The privacy protection can be regarded as a substantially more complex version of k-anonymity.

3.4 Reducing information loss: labeling cliques

Some of the information loss occurs already in the construction of $G(D)$. More precisely, the number of objects with property set P may be partially lost when $|P| > 2$. It is likely that the extent of this information loss highly depends on the structure of the data. In case it is desired that the database carries more information than $G(D)$ can do, it is possible to label not only edges with the frequency of occurrence, but also each clique of size, say, at most t. A protected graph can be constructed from this labeled clique graph in the same way as $G_P(D)$ was constructed from $G(D)$. Every frequency of occurrence $\frac{m}{n}$ is replaced by $\frac{ik}{n}$, where ik is the integer multiple of k closest to m.

3.5 The data graph in its general form

We summarize the discussion of this section with the definition of the protected data graph:

Definition 4 A *cover-up privacy-preserving data graph* of the data set D is the graph constructed from D in the following steps:

1. Represent each object in the data set with a complete graph on the node set of pairs (*type,property*). The result is a set of disjoint cliques.
2. (a) Identify all nodes with the same pair (*type,property*).
   (b) Merge all multiple edges into single edges.
   (c) Label every clique of size at most t by $\frac{m}{n}$ where m is the positive integer number of objects in the data set with all the properties in the clique, and n is the total possible number of objects with the properties in the same clique.
3. (a) Pick an integer k (typically at least 3).
   (b) Modify the labels at the cliques from $\frac{m}{n}$ to $\frac{ik}{n}$, where ik is the closest integer multiple of m.
   (c) Remove all edges labelled with zero.

The number n can be the total number of objects in the data set, but the model is not limited to use this definition of n. For example, if the data set is the union of several smaller data sets, perhaps the types in these smaller data sets are different. Then it would be natural to let n be the number of objects in the smaller data sets that contain the relevant types.

The protected graph of Sect. 3.3 and Definition 2 is obtained from Definition 4 by taking $t = 2$. Indeed, a clique of size two is simply an edge.

3.6 Data graphs and combinatorics

3.6.1 The data graph as a flag complex

An abstract simplicial complex is a family of finite sets (called simplices) that is closed under the operation of taking subsets. An empty simplex in an abstract simplicial complex C is a set Δ such that every subset of Δ consisting of a pair of elements are in some simplex of C, but Δ is not. A flag complex is an abstract simplicial complex without empty simplices. Any flag complex can be constructed from its underlying 1-skeleton, which is a graph, as its clique complex. The clique complex of a graph is the abstract simplicial complex consisting of the set of cliques in the graph.

If we forget the labels, the data graph $G(D)$ is a simple graph. The clique complex of this graph is a flag complex. Because of the way the graph is constructed, there may be some clique that does not come from a set of properties of any object. A lossless representation of the data set D would have preserved that information, allowing the structure to have empty simplices. We see that a lossless representation of the data similar to our data graph would mean modeling the data rather as an abstract simplicial complex with empty simplices, and not as the clique complex of the data graph.

We get the following result.

Lemma 1 *The data set D is represented with some information loss by the flag complex defined as the clique complex of data graph $G(D)$. The abstract simplicial complex defined by taking the set of properties of each object as maximal simplices on the same 1-skeleton $G(D)$ is a lossless representation of D.*

Using the graph $G(D)$ to represent D is clearly more efficient from a storage point of view compared to using the abstract simplicial complex defined by the data. If we also store information about the number of objects in the data set corresponding to each clique of the graph, then the representations are similar in storage efficiency. An empty simplex would have the label 0.

3.6.2 The data graph as a pregeometry

In incidence geometry, a *pregeometry* is a set X, a type function $\tau : X \to I$ from the set X to a set I of types, and an incidence relation $\sim$, such that if two elements from X are incident, then they have different type. The incidence graph of a pregeometry is the $|I|$-partite graph with X as node set and an edge between $a, b \in X$ if and only if a and b are incident.

A clique in the incidence graph of a pregeometry is called a flag. Since elements of the same type cannot be incident, all elements in a flag have different types. A flag with one element of each possible type is called a chamber. A pregeometry is a *geometry* if every flag is contained in a chamber.

Lemma 2 *A data graph for a data set in which each object has only one property of each type is a pregeometry, however not necessarily a geometry. The data corresponding to one object is represented by a maximal flag (a chamber).*

Proof It is clear that the underlying graph of the data graph is a pregeometry. Indeed, the set X is the set of nodes representing properties, the type function is the function that assigns to each property a type, and the incidence relation is defined by the edges of the graph—two properties are related if there is some object that have both.

To prove that it is a geometry, we would have to show that every flag is contained in a chamber. But this is not true in general. For example, consider the flag John, 45 and Diabetes from the example in Sect. 2.1.2. Add a unique identifier to the data set. This corresponds to adding 3 nodes of a forth type, each connected to the nodes in the flag with 3 properties of the corresponding individual. There will therefore be no node of type 4 connected to all nodes in the flag John, 45 and Diabetes, so it will not be contained in a maximal flag. □

For someone that likes incidence geometry, this can of course be somewhat disappointing. From a computer science point of view the observation of Lemma 2 is of similar importance as Lemma 1. The data graph model may contain information (in form of a flag) which was not present in the original data set.

3.7 Information loss

The major factors giving the information loss of representing D by the protected data graph $G_P(D)$ have already been described. To summarize, the information can be decomposed into two components:

1. *Information loss coming from using the data graph model*. This information loss is due to the fact that the data graph (and its clique complex) forgets the original partition of the data sets into disjoint cliques. Lemma 1 and Lemma 2 give further useful descriptions of the consequences. Note that if t is the number of types in Definition 4, then there is no information loss coming from using the data graph model!
2. *Information loss coming from rounding off the labels*. This means that the model forgets the exact number of objects with a given set of properties. Note that this is different from the information loss of most methods giving k-anonymous tables. Methods that provide k-anonymous tables often use generalization and suppression.

Data is first partitioned into similar clusters of size at least k. Then attributes are generalized in order to fit all members of the cluster. Records with attributes that are difficult to cluster can be suppressed, either by removing the entire record, or just suppressing the particular attribute (Sweeney 2002). Our protected data graph method has a different information loss, because attributes are not generalized. Attributes can be suppressed when rounding to zero, but the rounding can as well give a protected database that indicates a higher number of objects with a given set of properties than the actual number.

3.8 Disclosure risk

As was described in Sect. 2.2.2, two weaknesses of k-anonymity are (1) the risk of missing out a quasi-identifier and (2) the risk of attribute discloure because of low diversity within the anonymity sets.

The cover-up data graph model does not use quasi-identifiers, but treats all attributes equal. Therefore the first of the risks is not a problem.

Attribute disclosure can occur in a k-anonymous table when all records with property set P have the same sensitive property p_S. In the protected data graph this corresponds to a clique C_1 with node set $P \cup \{p_S\}$ such that there is no clique C_2 with node set $P \cup \{q_S\}$, with $p_S \neq q_S$ of the same type and non-zero proportions on both C_1 and C_2. This can clearly occur, however the rounding implies that some individuals probably have been removed from the database, while some fake individuals might have been added. Applying an argument of uncertainty, attribute disclosure occurs only with a given probability of success.

The risk of reidentification in a data graph seems to correspond to the risk that an adversary successfully identifies a clique, or a maximal clique, as corresponding to a given individual. This is clearly possible. If an individual has some publicly known property set P and there is a clique in the datagraph with this node set, one could argue that reidentification has occurred. But all edges in the clique are labeled with a positive integer multiple of k, carrying the information that there is at least k objects with the given properties. Therefore, what the adversary found is not the data of the individual, but an anonymity set of the individual. Consequences are therefore limited to possible attribute disclosure, which we have seen above can only have probabilistic success.

4 The probabilistic privacy-preserving data graph model

The privacy-preserving data graph model has a frequency or proportion label on each edge. By interpreting this frequency as a probability, the model becomes *random*. The model can be generalized further, by allowing edges to be labeled not only by probabilities, but by *probability distributions*. We put a probability distribution on each edge of the graph, and we do the same for the non-edges, but only between nodes of different type.

Definition 5 A *probabilistic privacy-preserving data graph* is an n-partite complete graph $K_{V_1,\ldots,V_n} = (V, \tilde{E})$, with the same vertex set as the data graph $G(D) = (V, E)$ and a discrete random variable with a discrete probability distribution on each edge.

For example, consider an edge (a, b) labeled with the random variable representing the number of objects in the data set with both properties a and b, together with the binomial distribution with parameters n and p. Then this can be interpreted so that the probability that k out of the n objects that could possibly have both the properties a and b follows the binomial distribution with parameters n and p, and therefore equals $\binom{n}{k} p^k (1-p)^{n-k}$.

For a different example, consider a graph in which an edge (a, b) is labeled with the random variable returning the adjacency matrix of a bipartite graph with s vertices of type a and t vertices of type b and the probability distribution of this random variable is defined so that it reflects the properties of the underlying data set. In such a situation, the random variables on the different edges are not likely to be independent.

The model allows to represent complicated scenarios, using a variety of discrete random variables and probability distributions. By using stochastic processes instead of probability distributions, also changes over time can be taken into account.

This privacy-preserving graph database model differs in nature from the model from Definition 4. It is not merely a graph database with probabilities on the edges, it is a graph database with a collection of random variables on the edges. The great flexibility and the probabilistic language should make this representation of a data set highly suitable as a privacy-preserving data release method for statistical analysis.

5 Summary and future work

A model for representing data in terms of a graph in a privacy preserving manner was presented. Information loss and disclosure risk was discussed, as were relations to some mathematical concepts within the area of combinatorics. In particular, abstract simplicial complexes and incidence geometries were used to express the nature of the information loss theoretically. Also a model that represents the data in terms of a graph of random variables was suggested.

Future work involves, for example,

- the study of the models in practice, in particular with respect to information loss, something that would imply defining statistical methods adapted to the model,
- obtaining a better understanding of the disclosure risk of Definition 5,
- applying the protection method, not to the clique complex of the data graph, but to the abstract simplicial complex that represents the data in a lossless manner,
- to study the compatibility of probability distributions on the edges of the graph of Definition 5.

Acknowledgements The author acknowledges partial support from the Spanish MEC project ICWT (TIN2016-80250-R).

References

Bachman CW (1973) The programmer as navigator. Commun ACM 16(11):653–658

Blum A, Dwork C, McSherry F, Nissim K (2005) Practical privacy: the SuLQ framework. In: Proceedings of PODS 2005, pp 128–138

Brand R (2002) Microdata protection through noise addition. In: Domingo-Ferrer J (ed) Proceedings of inference control in statistical databases, LNCS, vol. 2316, pp 97–116

Deka GC (2017) NoSQL: database for storage and retrieval of data in cloud. Chapman and Hall, London

Domingo-Ferrer J, Mateo-Sanz JM (2002) Practical data-oriented microaggregation for statistical disclosure control. IEEE Trans Knowl Data Eng 14(1):189–201

Domingo-Ferrer J, Mateo-Sanz JM, Torra V (2001) Comparing SDC methods for microdata on the basis of information loss and disclosure risk. In: Pre-proceedings of ETK-NTTS, 2001, vol 2, pp 807–826

Dwork C (2006) Differential privacy. In: ICALP 2006, LNCS 4052, pp 1–12

Dwork C (2008) Differential privacy: a survey of results. In: TAMC 2008, LNCS 4978, pp 1–19

Dwork C, Roth A (2014) The algorithmic foundations of differential privacy. Found Trends Theor Comput Sci 9(3–4):211–407

Ito S, Kikuchi H, Nakagawa H, Attacker models with a variety of background knowledge to de-identified data. J Ambient Intell Human Comput **(in press)**

Knuth DE (1968) The art of computer programming, Volume I: fundamental algorithms, 3rd edn. Addison-Wesley, Reading

Lima M (2014) The book of trees: visualizing branches of knowledge. Princeton Architectural Press, New York

Li X, Zhang C, Jung T, Qian J, Chen L (2016) Graph-based privacy-preserving data publication. IEEE INFOCOM 2016—the 35th annual IEEE international conference on computer communications, San Francisco, CA, pp 1–9

Long R, Harrington M, Hain R, Nicholls G (2000) IMS primer, IBM International Technical Support Organization, SG24-5352-00

Machanavajjhala A, Gehrke J, Kifer D, Venkitasubramaniam M (2006) l-diversity: privacy beyond k-anonymity. In: 22nd international conference on data engineering (ICDE'06), Atlanta, GA, USA, pp 24–24

Open graph protocol (2017). http://ogp.me/. Accessed Feb 2019

Robinson I, Webber J, Eifrem E (2013) Graph databases. O'Reilly Media Inc., Newton

Salás J (2019) Sanitizing and measuring privacy of large sparse datasets for recommender systems. J Ambient Intell Human Comput **(in press)**

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

Samarati P, Sweeney L (1998) Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression. SRI Intl. Tech, Rep

Spruill NL (1983) The confidentiality and analytic usefulness of masked business microdata. In: Proceedings of the section on survery research methods, vol 1983, American Statistical Association, pp 602–610

Stokes K (2012) On computational anonymity. In: Privacy in statistical databases (PSD 2012), pp 336–347

Sweeney L (2002) Achieving *k*-anonymity privacy protection using generalization and suppression. IJUFKS 10(5):571–588

TensorFlow. https://www.tensorflow.org/. Accessed Feb 2019

Torra V (2017) Data privacy. Springer, Berlin

Torra V, Navarro-Arribas G (2016) Integral privacy. In: Proceedings of CANS 2016, LNCS 10052, pp 661–669