Graph k-Anonymity through k-Means and as Modular Decomposition

Klara Stokes

Dept. of Computer and Information Science
Linköping University
581 83 Linköping, Sweden

Abstract. In this paper we discuss k-anonymous graphs in terms of modular decomposition and we present two algorithms for the k-anonymization of graphs with respect to neighborhoods. This is the strictest definition of k-anonymity for graphs. The first algorithm is an adaptation of the k-means algorithm to neighborhood clustering in graphs. The second algorithm is distributed of message passing type, and therefore enables user-privacy: the individuals behind the vertices can jointly protect their own privacy. Although these algorithms are not optimal in terms of information loss, they are a first example of algorithms that provide k-anonymization of graphs with respect to the strictest definition, and they are simple to implement.

Keywords: k-anonymity, graph, modular decomposition, message passing, distributed algorithm, k-means, user-privacy.

1 Introduction

Data privacy is a field that is concerned with privacy issues that appear in the collection and distribution of data. A standard example is when data collected anonymously by a National Statistical Bureau should be made accessible for researchers. It is well-known that side-information may allow records to be linked to the individuals they represent. This is a threat, in the first place to the privacy of the individual, but, in extension, to the correctness of statistical research, since the citizen may refuse to answer the survey if she holds doubts whether the survey is really anonymous. Statistical disclosure control (SDC) is a research area dedicated to the development of data protection techniques that allow for the anonymous release of data to third parties, or that allow third parties to query the data that stays on a secure server, so that all released information respect the anonymity of individuals [11].

Similar privacy issues appear for data collected by companies. Social network providers hold extensive databases over their users, a goldmine for researchers in social science and advertising agencies. In 2006 AOL released search logs that contained 20 million web queries from 658,000 AOL users posted during a period of 3 months. The released data was anonymized by replacing the identity of the users by a random index, but this quickly showed insufficient as several sequences of queries were linked to real persons. AOL removed the query logs from the Internet, but the files were of course already downloaded by many people [1]. Because of this accident, most companies probably would not consider to release data "anonymously". Still, the high market value of data, and a general tendency of people to make mistakes and forget, suggest that this was not the last time such a data leak occurred.

Most methods for SDC have been developed for microdata, data structured in table form. A very popular class of methods are those that achieve k-anonymity [23, 24, 28]. It is clear that k-anonymity suffers from some weaknesses (see for example [17]), however, it is conceptually very simple to understand and it is also achievable, making it attractive when compared to some of its alternatives. For example, differential privacy is a concept that, apart from being more difficult to understand for the uninitiated, in current implementations would not produce an anonymized graph. Instead, differential privacy is typically used to ensure that queries on the data are privacy preserving, while the data itself is not published.

The idea behind k-anonymity is based on the concept of quasi-identifier, coined by Tore Dalenius in 1986 [4]. A quasi-identifer in a data table is a collection of attributes in the so-called public domain, which, when combined, can serve as an identifier of at least some records. It can for example be possible for an adversary holding relevant side-information to reidentify the individual behind a record from its entries for the attributes *address*, *gender*, *age*. A data table is k-anonymous with respect to a quasi-identifier QI if every record appears in at least k copies in the table when restricted to QI. We say that these copies form an anonymity set with respect to QI.

It is well-known that a graph structure in the data can be used as a quasi-identifier. For example, in data coming from social networks, the set of friends of an individual can be used to reidentify her record, even if the identifiers of the table are removed. However, it has not always been clear how k-anonymization should be applied to graphs. Indeed, properties like the degree or the centrality of a vertex in the social graph, can be used for reidentification of individual vertices. Several graph properties have been pointed out as particularly interesting. The literature considers k-anonymity for graphs with respect to a list of distinct quasi-identifiers as for example vertex degree and neighborhood topology.

The main problem in k-anonymity for graphs is still to define efficient algorithms that can transform a given graph into a k-anonymous graph with small information loss. A variant of this problem is to define distributed algorithms for the k-anonymization of graphs in which all calculations are performed locally. Such algorithms would break the ground for systems in which the users of communication systems and social networks would be able to k-anonymize their own digital footprints, without the collaboration of the server or data holder. One of the motivations behind this paper is to explore such user-driven anonymization techniques.

This paper is structured as follows. The second section contains the preliminaries. In the third section we express k-anonymity for graphs in terms of modular decomposition. Section 4 presents algorithms for clustering of graphs with respect to open neighborhoods, which are based on the k-means algorithm. The paper ends with the conclusions.

2 Preliminaries

2.1 k-Anonymity for Graphs

A graph (V, E) is a set of vertices V and a set of edges E consisting of non-ordered pairs of elements from V. The open neighborhood of a vertex $v \in V$ is the set $N(v) = \{u \in V : (v, u) \in E\}$. The closed neighborhood of v is $\overline{N(v)} = N(v) \cup \{v\}$. The degree of v is the cardinality of $N(v)$. An adjacency matrix of the graph is a $|V| \times |V|$ matrix $A = (a_{ij})$ with the rows and columns indexed by V and such that $a_{ij} = 0$ if $(i, j) \notin E$ and $a_{ij} = 1$ if $(i, j) \in E$. Permuting columns and rows of an adjacency matrix of a graph produces another adjacency matrix of the same graph.

An isomorphism f between two graphs $G = (V, E)$ and $G' = (V', E')$ is a bijective function $f : V \to V'$ with an edge-preserving property, i.e. $uv \in E$ if and only if $f(u)f(v) \in E'$. A graph automorphism on a graph G is a graph isomorphism $f : G \to G$. Given a graph $G = (V, E)$, the set of automorphisms on G forms a group, denoted by $Aut(G)$. An element f in $Aut(G)$ acts on a vertex v in V as $fv := f(v)$. The orbit of a vertex v for a subgroup $F \subseteq Aut(G)$ is the subset of vertices $\{fv : f \in F\}$.

A (micro-)data set is k-anonymous with respect to a quasi-identifier QI if every record equals $k - 1$ other records when restricted to the attributes in QI. The achieved level of privacy will depend on whether the quasi-identifier was correctly determined.

When k-anonymity is applied to graphs, typically the goal is to partition the vertex set into anonymity sets, according to the properties attached to each vertex. Different flavours of k-anonymity for graphs exist in the literature, differing in choice of quasi-identifier. For example, Liu and Terzi developed methods for k-anonymity with respect to the degrees of the vertices [14], Zhou and Pei considered k-anonymity with respect to isomorphic neighborhoods [30], and other authors considered more generic structural quasi-identifiers [3, 9, 31]. Methods that these authors use for k-anonymization are, for example, simulated annealing and greedy algorithms which heuristically aim at minimizing some measure of information loss. There are also approaches for edge k-anonymization, in which the edges are partitioned into anonymity sets instead of the vertices. This list of previous work is not exhaustive, and an interested reader is referred to surveys like [15, 19] for further information. Observe that, in general, a graph that is k-anonymous with respect to some quasi-identifier may fail to be so for another quasi-identifier.

Actually, it was observed already in 1971 that the computationally correct quasi-identifier for social networks is the neighborhood of the vertices [16], although the result was not discussed in the context of data privacy, and the concept of quasi-identifier was not yet defined then. The same fact was later rediscovered and explored in [26], which also provided a characterization of k-anonymous graphs with respect to the open neighborhoods. The same article sketched an algorithm which, given a graph, transforms it into a k-anonymous graph (with respect to open neighborhoods). We repeat it here (Algorithm 1), since it is essential for the content of this paper. In the current article, Algorithm 1 is implemented in combination with a distributed k-means algorithm for clustering open neighborhoods in graphs.

Graphs that are k-anonymous with respect to open neighborhoods are in general not k-anonymous with respect to closed neighborhoods, and vice versa. If the quasi-identifier is the open neighborhood, then vertices in the same anonymity set must be disconnected. Indeed, if two vertices u and v are connected, then $v \in N(u)$ and $u \in N(v)$, but $u \notin N(u)$ and $v \notin N(v)$, so u and v cannot have the same open neighborhood. If instead the quasi-identifier is the closed neighborhood, then vertices in the same anonymity set must be connected. If two vertices u and v are not connected then $u \in \overline{N(u)}$ and $v \in \overline{N(v)}$, but $u \notin \overline{N(v)}$ and $v \notin \overline{N(u)}$, so they cannot have the same closed neighborhood. Figure 1 shows examples of graphs with these properties.

The correct choice between these two quasi-identifiers depends on the type of graph that is considered. As was observed in [16], an identity relation in a (social) network is represented by a self-loop at each vertex. In general, a relation with this property is called reflexive. If the data represented by the graph contain such a reflexive relation, then the correct quasi-identifier is the closed neighborhood. If the graph instead represents a relation that is not reflexive, so that the graph is without self-loops, then the correct quasi-identifier is the open neighborhood. The so-called "social graph", representing friendships in an online social network, is typically defined as a graph without self-loops. The correctness of these quasi-identifiers can be understood through the fact that when any of these two corresponding criteria are satisfied (reflexive relation together with k-anonymity with respect to closed neighborhoods, or non-reflexive relation together with k-anonymity with respect to open neighborhood), then an adjacency matrix of the graph will be k-anonymous with respect to all columns simultaneously, when considered as a table with $|V|$ rows and $|V|$ columns.

Two vertices u and v in G are structurally equivalent if u relates to every vertex in G in exactly the same ways as v does. If this occurs, then u and v are absolutely equivalent within the graph, indeed they are substitutable. Consequently a graph that is k-anonymized with respect to open/closed neighborhoods will be k-anonymized to any other graph property attached to the vertices, when considered as vertices of a graph without/with a loop at each vertex. Two vertices with the same neighborhood also share the same degree, centrality, etc.

In some cases, achieving an acceptable privacy level only requires a weaker form of structural equivalence, like the one in which the anonymity sets are orbits of the vertices under the action of some subgroup of the automorphism group of the graph [31]. Observe that k-anonymity with respect to both open and closed neighborhoods implies k-anonymity in terms of automorphisms, but the contrary is not true. However, as has been observed by some authors, k-anonymizing a graph with respect to automorphisms and with gain in information loss is a computationally difficult problem, while k-anonymity with respect to neighborhoods is much more straightforward.

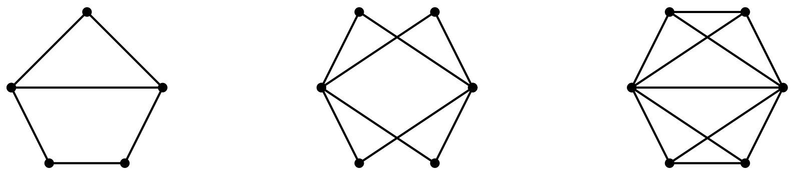
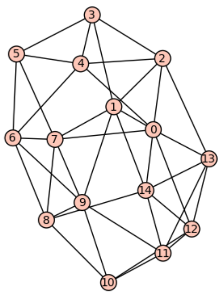


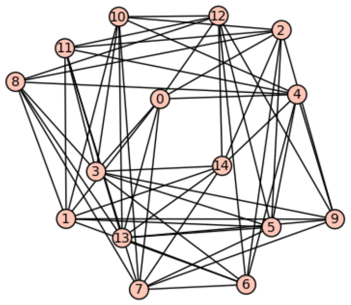
Fig. 1. The graph on the left is not k-anonymous with respect to open neighborhoods, closed neighborhoods, nor automorphisms, for any $k > 1$. The graph in the middle is 2-anonymous with respect to open neighborhoods, but not k-anonymous with respect to closed neighborhoods for $k > 1$. It is also 2-anonymous with respect to automorphisms. The graph on the right is 2-anonymous with respect to closed neighborhoods and automorphisms, but not k-anonymous with respect to open neighborhoods for $k > 1$.

The definition of k-anonymity for graphs with respect to neighborhoods is very strict, implying that k-anonymization may cause large information loss when the original graph is far from being k-anonymous. However, this information loss may be proportionally small when there are substantial data quantities attached to the vertices in the graph. In this case, it is possible to apply hybrid protection methods to the data set, combining k-anonymity for graphs with, for example, synthetic data generators applied to the data attached to each anonymity set, separately [26]. In this process it is of great importance that the anonymity sets are respected over all columns of the relevant quasi-identifiers. Otherwise it is possible to attack the protection of the data by intersecting overlapping anonymity sets.

A graph that is k-anonymous with respect to neighborhoods has a rather particular structure. As we saw previously, vertices in the same anonymity set are either all connected (closed neighborhoods) or they are not connected at all (open neighborhoods). In both cases, two vertices u and v in different anonymity sets A and B are connected if and only if all vertices in A are connected to all vertices in B. Therefore, the graph essentially consists of many copies of a smaller graph: the induced subgraph on a subset of vertices with one vertex from each anonymity set. With an appropiate drawing of the graph this phenomenon can be observed easily. This is not fully developed in Figure 2, which represents a graph with only two anonymity sets, indeed it is the complete bipartite graph on $7 + 8$ vertices.

In some cases it is not necessary to provide a concrete graph as output from the anonymization algorithm, but it may be enough to present the clusters and the relations between the clusters [3]. In any case, the choice of the clustering algorithm is essential, in particular for controlling the information loss. This article focuses on the k-means clustering algorithm for the k-anonymization of graphs.





$$\begin{pmatrix}
0&1&1&0&1&0&0&1&0&0&0&0&1&1&1\\
1&0&1&1&0&0&0&1&0&1&0&0&0&0&1\\
1&1&0&1&1&0&0&0&0&0&0&0&0&1&0\\
0&1&1&0&1&1&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&1&1&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&1&1&0&0&0&0&0\\
1&1&0&0&0&1&1&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&1&0&0&0&1\\
0&1&0&0&0&0&1&1&1&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&1&1&1\\
1&0&0&0&0&0&0&0&0&0&1&1&0&1&1\\
1&0&1&0&0&0&0&0&0&0&0&1&1&0&1\\
1&1&0&0&0&0&0&0&1&0&0&1&1&1&0
\end{pmatrix}
\qquad
\begin{pmatrix}
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
1&0&0&0&0&1&1&0&1&1&1&1&0&0&1\\
0&1&1&1&1&0&0&1&0&0&0&0&1&1&0
\end{pmatrix}$$

Fig. 2. The 7-anonymous graph on the right was obtained through k-anonymization of the Newman-Watts-Strogatz graph on the left, using Algorithm 1 and Algorithm 4. Below the adjacency matrices of the two graphs.

2.2 The k-Means Algorithm

In exploratory data mining, clustering is the task of grouping elements together on the basis of some similarity, dividing data into cells of a Voronoi diagram. Clustering has many applications in machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. In centroid-based clustering, each cluster is represented by a centroid. The centroid can be for example the vector representing the average of all the vectors in the cluster. The centroid does not necessarily belong to the data set.

There is an intrinsic relation between k-anonymity and centroid-based clustering. Indeed, k-anonymity can be attained by clustering the data and replacing the records by the centroid of the corresponding cluster. One of the most important algorithm for centroid-based clustering is the k-means algorithm or Lloyd's algorithm (Algorithm 2), defined by Stuart Lloyd in 1957 [13].

There exist variants of the k-means algorithm that replace the original Euclidean average by the median, the mode, etc., see for example [12]. Other variants explore the possibility to use other similarities than the one induced by the Euclidean distance, as for example those induced by the Malahanobis distance or the Hamming distance. For graphs, the similarity induced by the Manhattan distance and the 2-path similarity were proposed in [27].

There are several ways to initialize cluster centroids for the k-means algorithm, see for example [22]. Two common procedures are to initialize the centroids as

```
Input: A graph $G = (V, E)$ and a natural number $k \leq |V|$
Output: A graph $G' = (V, E')$ that is $k$-anonymous with respect to open
        neighborhoods
Calculate a family of clusters $C$ of $V$ so that vertices in the same cluster have
similar open neighborhoods;
foreach $C_i \in C$ do
    foreach $(u, v) \in C_i$ do
        if $uv \in E$ then
            Delete $uv$;
        end
    end
end
foreach $C_i \neq C_j \in C$ do
    if $\sharp\{(u, v) \in C_i \times C_j : \exists uv \in E\} > |C_i||C_j|/2$ then
        foreach $(u, v) \in C_i \times C_j$ do
            if $uv \notin E$ then
                Add $uv$ to $E$;
            end
        end
    end
    else
        foreach $(u, v) \in C_i \times C_j$ do
            if $uv \in E$ then
                Delete $uv$ from $E$;
            end
        end
    end
end
Return $G$;
```

Algorithm 1. An algorithm for k-anonymization of graphs

random points in the data domain, or as random points in the data set. In the first case, the points are not required to belong to the data set. The k-means algorithm does not have to return a globally optimal result, and the outcome is typically different for different initial input to the algorithm. A good solution is to run the algorithm several times, and choose the best result.

2.3 Message Passing Algorithms

A message passing algorithm is an iterative algorithm in which each step is executed on the vertices of a graph. The result of this execution is then forwarded in messages to the neighbors of the vertices in the graph, before the next iteration. There are several important examples of message passing algorithms. Belief propagation is a message passing algorithm for inference in graphical models as Bayesian networks or Markov random fields. Sometimes it is used as a synonym for the sum-product algorithm, originally described by Judea Pearl in 1982 [21],

```
Input: A data set of records $X = (x_j)$ and an integer $k$
Output: A list of cluster centers $C = (c_i)$
Initialize a set of cluster centers $C' := C := c_1, \ldots, c_k$;
repeat
    $C := C'$;
    foreach $x_j$ do
        Assign $x_j$ to the closest cluster centroid $c_i$, creating the Voronoi
        diagram with corresponding Voronoi cells $A_i$;
    end
    foreach $A_i$ do
        Calculate the new centroid of $A_i$, $c'_i$;
    end
until $C = C'$;
```

Algorithm 2. The k-means algorithm

and with applications for example in the decoding of low-density parity-check (LDPC) codes. However, in its general form it is simply a message passing algorithm that takes advantage of a factorization of the global function into local functions of a subset of the variables. Defined in this way, the Viterbi algorithm [29] is another important example of belief propagation.

A belief propagation algorithm has an associated factor graph, which models the factorization of the problem into local subfunctions. The factor graph is a bipartite graph with the variables in one set and the subfunctions in the other. There is an edge between a variable vertex v and a subfunction vertex f if f is a function of v.

Belief propagation algorithms can typically be proved to be exact on acyclic graphs (trees). Surprisingly, they also tend to give non-exact but approximately good result on graphs with cycles. However, then convergence is not ensured, and oscillation may occur [18].

2.4 Modular Decomposition of Graphs

A module in a graph $G = (V, E)$ is a subset of vertices $M \subseteq V$ that share the same neighbors in $V \setminus M$. A strong module is a module that does not overlap other modules. The modules of a graph form a partitive family defining a decomposition scheme for the graph with an associated decomposition tree composed of the graphs strong modules [7]. This tree is a representation of the structure of the graph and is therefore a first step in many algorithms.

A congruence partition is a partition of the vertices V in which the parts are modules. A factor is the induced graph on the vertices in one part of a congruence partition. The modules that are maximal with respect to inclusion form a congruence partition of V, called a maximal modular partition. Every graph has a unique maximal modular partition. If two vertices v_1 and v_2 in two disjoint modules M_1 and M_2 are connected, then all vertices in M_1 are connected to all vertices in M_2. In particular this is true for the maximal modules. The quotient graph of G is defined as the graph with the maximal modules of G as vertices and an edge between two vertices if the corresponding modules are connected.

For a survey on algorithms for modular decomposition of graphs, see [8].

3 k-Anonymous Graphs in Terms of Modular Decomposition

In this section we characterize k-anonymity for graphs with respect to neighborhoods in terms of modular decomposition. It is easy to see that a graph G that is k-anonymous with respect to open neighborhoods is a graph such that the cardinality of each part in the maximal modular partition P of G is larger than or equal to k, and additionally, the factors of G with respect to P are totally disconnected.

A graph G that is k-anonymous with respect to closed neighborhoods still has a maximal modular partition P such that each part has cardinality larger than or equal to k. However, in this case the factors of G with respect to P are complete graphs. We summarize these results in the following theorem.

Theorem 1. *Let G be a graph. If G is k-anonymous with respect to the open (closed) neighborhoods, then G has a maximal modular partition $P = \{V_1, \ldots, V_m\}$ such that $|Vi| \geq k$ for all $i = 1, \ldots, m$ and such that the factors of G with respect to P are completely disconnected (complete) graphs.*

Observe that Theorem 1 provides an efficient way of testing for k-anonymity in graphs. Just apply an algorithm for modular decomposition to obtain the maximal modular partition and check that the factors are as required. As an extension, the modular decomposition tree could be used for k-anonymization.

In general, the factors of the maximal modular partition of a graph can be any graph. This motivates the use of modular decomposition of graphs for a more flexible definition of k-anonymity, in which only the edges between the different modules are anonymized. This can be useful for example in cases in which some edges are considered to be more sensitive than others. As an example of this, consider a situation in which it can be assumed that some edges are not in the public domain of knowledge, so that they are not in the quasi-identifier. Then, according to the model of k-anonymity, these edges cannot be used for reidentification, and there is no need to k-anonymize them.

4 Algorithms for Clustering of Graphs with Respect to Open Neighborhoods

This section describes algorithms for centroid-based clustering of graphs with respect to open neighborhoods. These algorithms can be used in combination with Algorithm 1 to produce a k-anonymous graph.

4.1 A k-Means Algorithm for Graphs

Only small modifications to the template of the k-means algorithm is needed in order to obtain a straight-forward iterative centroid-based clustering algorithm for graphs, Algorithm 3.

For a vertex v in a graph $G = (V, E)$, represent $N(v)$ as the row vector in an adjacency matrix of G indexed by v. Then $N(v)$ is a vector in $\{0,1\}^{|V|}$. For two vectors x and y in $\{0,1\}^{|V|}$, let $d(x,y) := |\{i : x_i \neq y_i\}|$ be the Hamming distance of x and y. Observe that for two vertices $u, v \in V$, the Hamming distance betweeen $N(v)$ and $N(u)$ equals the symmetric difference between the neighborhoods of u and v as sets.

For a vertex $v \in V$, denote by $c(v)$ the centroid vector c from a set of centroids C that minimizes $d(N(v), c)$. Denote the clusters by $A(c) := \{v \in V : c(v) = c\}$.

As we saw in Section 2.2, cluster centroids can be initialized in several ways. Although there exist more advanced initialization algorithms for the k-means algorithm, it is commonly accepted that random initialization is a good alternative, combined with several runs of the algorithm, see for example [22]. We suggest that randomly generated centroids should be chosen from a distribution that adjusts well to the relevant family of graphs. This can be achieved easily by choosing the centroids randomly from the data set, which is the method that will be used in this article. Other initialization methods can of course be used.

The k-means algorithm involves a step of calculation of the mean between several vectors. For open neighborhoods, the mean does not adjust well, since it is not obvious how to define the mean of a set of neighborhoods. Instead other aggregation operators can be more suitable. The mode of a set of elements is the element that appears most often in the set. In contrast to the mean or the median, the mode can be used also for nominal data. We use the mode for vectors applied to each coefficients independently. Given a set of vectors it returns the vector with coefficients equal to the mode of the coefficients of all vectors in the set. Observe that defined in this way, the mode of a data set of vectors does not necessarily belong to the data set. The mode of a set of neighborhoods with threshold t is then a neighborhood that contains a vertex v if a proportion larger than a threshold $t \in [0,1]$ of the original neighborhoods contain v.

4.2 A Distributed k-Means Algorithm

In this section we present a distributed version of the k-means open neighborhood clustering algorithm for graphs. The idea is to approximate Algorithm 3 by decomposing the calculation of one iteration of the algorithm into factors that can be calculated by each vertex independently, using only local knowledge. As was pointed out in [27], two vertices in a graph have similar neighborhoods if their 2-path similarity is high. The 2-path similarity between two vertices u and v is defined as the number of paths of length two between u and v. It is therefore clear that all vertices in an anonymity set of a vertex v in a k-anonymous graph of non-zero degree will all be on distance at most two from v. This suggests that a clustering in graphs with respect to open neighborhoods can

```
Input: A graph $(V, E)$ and an integer $m$
Output: A set of $m$ cluster centroids $C = \{c_1, \dots, c_m\}$
Initialize a set of cluster centroids $C' := C := \{c_1, \dots, c_m\}$;
repeat
    foreach $v \in V$ do
        $C := C'$;
        Calculate $c(v)$ as $c \in C$ minimizing $d(c, N(v))$;
        Assign $v$ to $A(c(v))$.
    end
    foreach $c \in C$ do
        Calculate the new centroid $c'$ of $A(c)$ as the mode with threshold $t$ of
        the vectors $N(u)$ for $u \in A(c)$;
    end
until $C = C'$;
```

Algorithm 3. A k-means algorithm for open neighborhood clustering of graphs

be approximated with local computations on each vertex, requiring a knowledge of the graph topology that covers vertices at distance at most three, or in other words, the neighborhoods of the neighbors of neighbors. However, in practice, the algorithm can be defined so that it requires only message passing between vertices at distance one. Also, much of the information, the neighborhoods, can be passed once in the beginning of the algorithm and stored for future use.

The distributed k-means algorithm is described in Algorithm 4. As in Section 4.1, for a vertex $v \in V$, denote by $c(v)$ the centroid vector c from a set of centroids C that minimizes $d(N(v), c)$ and the clusters by $A(c) := \{v \in V : c(v) = c\}$.

The algorithm starts by initializing a set of cluster centroids. Also, each vertex v posts a message to all its neighbors containing its neighborhood $N(v)$, which is stored by these vertices. The rest of the algorithm is an iteration of the two steps in Algorithm 3 and an extra third approximation step, adapting the algorithm to distributed execution. Since the execution of the algorithm moves between neighbors, it is useful to follow the execution as it moves from one vertex v and its neighbor u.

In Step 1 each vertex v calculates the cluster centroid that is closest to its neighborhood $N(v)$ and posts the result to all its neighbors $u \in N(v)$. In Step 2 each vertex u updates the cluster centroids $C = \{c_1, \dots, c_m\}$. For each centroid c_i, the new centroid c_i' is calculated as the mode of the coefficients of the set of neighborhood vectors $\{N(v) : v \in N(u) \text{ and } c(v) = c_i\}$. Then u posts the set of new centroids C' to the neighbors $v \in N(u)$. In Step 3 each vertex v calculates an approximation C'' of the new centroids as a combination of the different C''s received from its neighbors. Then the algorithm returns to step 1 with $C := C''$.

Instead of having the vertices post the set of new centroids C' to their neighbors, one can let them post C' on a public message board. In any case, where the original k-means algorithm produced one set of new centroids, the distributed

```
**Input**: A graph $(V, E)$ and an integer $m$
**Output**: A set of cluster centroids $C = \{c_1, \dots, c_m\}$
Globally initialize a set of cluster centroids $C'' := C := \{c_1, \dots, c_m\}$ and post
them to the public board;
**foreach** $v \in V$ **do**
    Post $N(v)$ to $u \in N(v)$;
    Receive and store $N(u)$ from $u \in N(v)$;
    **repeat**
        *Step 1*
        Set $C := C''$;
        Calculate $c(v)$, as $c \in C'$ that minimizes $d(c, N(v))$;
        Post $c(v)$ to $u \in N(v)$;
        *Step 2*
        Receive $c(u)$ from $u \in N(v)$;
        Calculate the new centroids $C'_v := \{c'_1, \dots, c'_m\}$ as the mode of the
        neighborhoods in the set $\{N(u) : u \in N(v) \cap A(c_i)\}$, with $c_i \in C$;
        Post $C'_v$ to $N(v)$;
        *Step 3*
        Receive $C'_u$ from $u \in N(v)$ and combine them all into new centroids $C''$;
    **until** $C = C''$;
**end**
```

Algorithm 4. A distributed k-means algorithm for clustering graphs

version of the algorithm produces many sets of new centroids that must be combined in some effective way. Our experiments show that it works to combine the new centroids $C' = \{c'_1, \dots, c'_m\}$ into the vectors $C'' = \{c''_1, \dots, c''_m\}$ that have ones at the indices where there is at least one c'_i with a one at these indices. This strategy can be used for the calculation of C'' both locally and globally at the public message board.

5 Experiments

Algorithm 4 was implemented in Sage [25] and executed on random graphs generated using the Barabási-Albert model [2] and the Newman-Watts-Strogatz model [20]. These random graph models were chosen since they share properties with real social graphs: the former gives graphs that are scale-free and the graphs generated using the latter model have the small world property.

5.1 The *k*-Means Algorithm on Graphs

In a first experiment, for each of these two graph models and for each number of vertices in $\{30, 60, 120\}$, Algorithm 4 with public board was tested on 20 different graphs. For each graph, 10 different initializations of the cluster centroids were tried, and the result with the smallest information loss was chosen. In this experiment, information loss was measured as the sum of the symmetric difference error (SSDE), the symmetric difference between the cluster centers (considered as sets of vertices) and the neighborhoods of the vertices in each cluster, summed over all vertices of the graph. The number of iterations on each initialization was set to 15, since it was observed that in general convergence was achieved well before 10 iterations. Table 1 shows the average information loss in SSDE among 20 tested graphs on $|V|$ vertices, divided by the number of vertices $|V|$.

Table 1.

$\|V\|$	e	m	$SSDE/\|V\|$
30	3	3	3.9
60	3	3	4.8
120	3	3	5.3

Barabási-Albert graphs with parameters $(|V|, e)$

$\|V\|$	e	p	k	$SSDE/\|V\|$
30	4	0.2	3	3.9
60	4	0.2	3	4.0
120	4	0.2	3	4.2

Newman-Watts-Strogatz graphs with parameters $(|V|, e, p)$

5.2 Constructing k-Anonymous Graphs

In a second experiment, once the clusters were obtained, Algorithm 1 was applied to the original graph, and a k-anonymized graphs was constructed. For each of the two random graph models, and for each number of vertices $|V|$ in $\{30, 60, 120\}$, Algorithm 4 was first applied to 20 different graphs, with the same settings as the experiment in Section 5.1. The only change was the use of an x-factor when updating the centroids, allowing to adjust the approximate number of vertices in the centroids. Then Algorithm 1 was applied, using the resulting clusters to construct a k-anonymous graph.

Several problems were observed during both experiments described in this article. For example, although m centroids were demanded, the algorithm typically returned fewer centroids. Also, the k-means algorithm is not designed to ensure that clusters contain at least k vertices. Since our goal is to construct a k-anonymous graph, this was an issue. In particular, outliers were a problem, creating clusters with single vertices. All these problems are typical for the k-means algorithm and there are methods to deal with them available in the literature. Note though that for all the tested graphs, 10 different initializations were enough in order to automatically find a clustering such that each cluster contained at least k vertices. The results presented in Table 2 all correspond to graphs that are k-anonymous.

Currently, there is no measure of information loss for graphs that can be considered to be clearly better than other measures. In general, information loss should be quantified according to the utility of the anonymized graph in the context in which it should be used. This means that the best measure of information loss may vary with the context. Sometimes it may be interesting to preserve some particular property of the graph, i.e. diameter, girth, degree sequence. Then, please note then that a graph that is k-anonymous with respect to neighborhoods, with $k \geq 2$ and minimum degree 2, must have girth 3 if neighborhoods are closed and can not have girth larger than 4 if neighborhoods are open. To see this, fix a vertex v and observe that there must be a vertex

Table 2.

$\lvert V\rvert$	e	k	$SSDE/\lvert V\rvert$	$ED/\lvert V\rvert$
30	3	3	4.13	4.16
60	3	3	5.56	6.68
120	3	3	6.64	10.75

Barabási-Albert graphs with parameters $(|V|, e)$

$\lvert V\rvert$	e	p	k	$SSDE/\lvert V\rvert$	$ED/\lvert V\rvert$
30	4	0.2	3	3.93	3.68
60	4	0.2	3	4.51	4.95
120	4	0.2	3	4.61	6.6

Newman-Watts-Strogatz graphs with parameters $(|V|, e, p)$

$u \neq v$ with the same neighbors as v. Since v has at least two neighbors, if neighborhoods are closed, there will be a cycle of length 3, if neighborhoods are open, there will be a cycle of length 4. Therefore the girth is not a very interesting measure of information loss in this context.

In this experiment, information loss was measured both as in Section 5.1 and as the number of edges that had to be removed or added in order to construct the k-anonymous graph from the original graph. In Table 2, the average of the information loss over the 20 graphs is presented as the sum of symmetric difference error (SSDE), the symmetric difference between centroids and neighborhoods as sets as in Table 1, and as the edge difference (ED), the number of edges added or removed by the anonymization algorithm. In both cases, information loss was divided by the number of vertices in the graphs. The SSDE was slightly higher than in Table 1, due to the use of the x-factor.

6 Conclusions

The contribution of this article was two-fold. On the one hand we have characterized k-anonymous graphs in terms of modular decomposition, motivating a more flexible definition of k-anonymity for graphs. On the other hand we have described two algorithms for the k-anonymization of graphs with respect to open neighborhoods. Algorithm 3 is an adaptation of the k-means algorithm to the open neighborhood clustering problem in graphs. Algorithm 4 is a distributed version of Algorithm 3. The algorithms have been implemented using Sage [25]. Applications for distributed algorithms for k-anonymization of graphs can be found for example in user-driven privacy enhancing technologies for social networks.

Acknowledgements. The calculations in this paper was made using Sage [25]. The author is financed by the Swedish government through the National Graduate School in Computer Science (CUGS). Partial support by the Spanish MEC projects ARES (CONSOLIDER INGENIO 2010 CSD2007-00004) and ICWT (TIN2012-32757), is acknowledged.

References

1. AOL query logs, `www.aolstalker.com`
2. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
3. Campan, A., Truta, T.M.: Data and structural anonymity in social networks. In: ACM SIGKDD International Workshop on Privacy, Security, and Trust in KDD (2008)
4. Dalenius, T.: Finding a needle in a haystack. Journal of Official Statistics 2(3), 329–336 (1986)
5. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
6. Domingo-Ferrer, J., Torra, V.: Ordinal, continuous and heterogenerous k-anonymity through microaggregation. Data Mining and Knowledge Discovery 11(2), 195–212 (2005)
7. Gallai, T.: Transitiv orientierbare graphen. Acta Mathematica Acad. Sci. Hungar. 18, 25–66 (1967)
8. Habib, M., Paul, C.: A survey of the algorithmic aspects of modular decomposition. Journal of Computer Science Review 4(1), 41–59 (2010)
9. Hay, M., Miklau, G., Jensen, D., Towsley, D., Weis, P.: Resisting structural reidentification in anonymized social networks. In: International Conference on Very Large Data Bases (2008)
10. Hedetniemi, S.T., Laskar, R.C.: Bibliography on domination in graphs and some basic definitions of domination parameters. Discrete Mathematics 86(1-3), 257–277 (1990)
11. Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Schulte Nordholt, E., Spicer, K., de Wolf, P.-P.: Statistical Disclosure Control. Wiley (2012)
12. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall (1981)
13. Lloyd, S.P.: Least square quantization in PCM. Bell Telephone Laboratories Paper (1957); Later published as: Least squares quantization in PCM. IEEE Transactions on Information Theory 28(2), 129–137 (1982)
14. Liu, K., Terzi, E.: Towards identity anonymization on graphs. In: ACM SIGMOD International Conference on Management of Data, pp. 93–106 (2008)
15. Liu, K., Das, K., Grandison, T., Kargupta, H.: Privacy-preserving data analysis on graphs and social networks. In: Kargupta, H., Han, J., Yu, P., Motwani, R., Kumar, V. (eds.) Next Generation of Data Mining. CRC Press (2008)
16. Lorrain, F., White, H.C.: Structural equivalence of individuals in social networks. Journal of Mathematical Sociology 1, 49–80 (1971)
17. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: L-diversity: Privacy beyond k-anonymity. J. ACM Trans. Knowl. Discov. Data 1(1), 3 (2007)
18. Mooij, J.M., Kappen, H.J.: Sufficient Conditions for Convergence of the SumProduct Algorithm. IEEE Transactions on Information Theory 53(12), 4422–4437 (2007)
19. Nettleton, D.F., Torra, V.: Data Privacy for Simply Anonymized Social Network Logs Represented as Graphs - Considerations for Graph alteration Operations. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 19(suppl. 1), 107–125 (2011)
20. Newman, M.E.J., Watts, D.J., Strogatz, S.H.: Random graph models of social networks. Proc. Nat. Acad. Sci. USA 99, 2566–2572 (2002)

21. Pearl, J.: Reverend Bayes on inference engines: A distributed hierarchical approach. In: Proceedings of the Second National Conference on Artificial Intelligence, AAAI 1982, Pittsburgh, PA, pp. 133–136. AAAI Press, Menlo Park (1982)
22. Peña, J.M., Lozano, J.A., Larrañaga, P.: An Empirical Comparison of Four Initialization Methods for the K-Means Algorithm. Pattern Recognition Letters 20(10), 1027–1040 (1999)
23. Samarati, P.: Protecting Respondents' Identities in Microdata Release. IEEE Trans. on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
24. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k–anonymity and its enforcement through generalization and suppression. SRI Intl. Tech. Rep. (1998)
25. Stein, W.A., et al.: Sage Mathematics Software (Version 5.0), The Sage Development Team (2012), `http://www.sagemath.org`
26. Stokes, K., Torra, V.: Reidentification and k-anonymity: a model for disclosure risk in graphs. Soft Computing 16(10), 1657–1670 (2012)
27. Stokes, K., Torra, V.: On some clustering approaches for graphs. In: FUZZ-IEEE 2011, pp. 409–415 (2011)
28. Sweeney, L.: k-anonymity: a model for protecting privacy. Int. J. of Unc., Fuzz. and Knowledge Based Systems 10(5), 557–570 (2002)
29. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory 13(2), 260–269 (1967)
30. Zhou, B., Pei, J.: Preserving privacy in social networks against neighborhood attacks. In: IEEE International Conference on Data Engineering (2008)
31. Zou, L., Chen, L., Tamer Özsu, M.: k-Automorphism: a general framework for privacy preserving network publication. VLDB Endowment 2(1), 946–957 (2009)