An Efficient Clustering Method for k-Anonymization

Jun-Lin Lin*
Department of Information Management
Yuan Ze University
Chung-Li, Taiwan
jun@saturn.yzu.edu.tw

Meng-Cheng Wei
Department of Information Management
Yuan Ze University
Chung-Li, Taiwan
mongcheng@gmail.com

ABSTRACT

The k-anonymity model is a privacy-preserving approach that has been extensively studied for the past few years. To minimize the information loss due to anonymization, it is crucial to group similar data together and then anonymize each group individually. This work proposes a clustering-based k-anonymization method that runs in $O(\frac{n^2}{k})$ time. We experimentally compare our method with another clustering-based k-anonymization method recently proposed by Byun *et al.* Even though their method has a time complexity of $O(n^2)$, the experiments show that our method outperforms their method with respect to information loss and resilience to outliers.

1. INTRODUCTION

Owing to the rapid progress in information technologies, companies can nowadays collect and store huge amounts of data. With this large volume of data, manual analysis is no longer feasible. Instead, automatic or semiautomatic tools that employ data mining techniques have been widely used to support data analysis. In a nutshell, data mining is the process of discovering patterns in data, and the quality of data is crucial for the success of a data mining process. However, as the public becomes more concerned with privacy, more and more people are unwilling to provide real personal data when asked to do so. Besides, companies that wish to use their customers' data for data mining cannot easily do so without compromising their customers' privacy. This is a dilemma between data quality and data privacy.

The k-anonymity model [9, 10] is an approach to protect data from individual identification. It works by ensuring that each record of a table is identical to at least $k-1$ other records with respect to a set of privacy-related attributes, called *quasi-identifiers*, that could be potentially used to identify individuals by linking these attributes to external data sets. For example, consider the hospital data in Table 1, where the attributes *ZipCode*, *Gender* and *Age* are regarded as quasi-identifiers. Table 2 gives a 3-anonymization version of the table in Table 1, where anonymization is achieved via generalization at the *attribute* level [3], i.e., if two records contain the same value at a quasi-identifier, they will be generalized to the same value at the quasi-identifier as well. Table 3 gives another 3-anonymization version of the table in Table 1, where anonymization is achieved via generalization at the *cell* level [3], i.e., two cells with same value could be generalized to different values (e.g., value 75275 in the *ZipCode* column and value *Male* in the *Gender* column).

Table 1: Patient records of a hospital

ZipCode	Gender	Age	Disease	Expense
75275	Male	22	Flu	100
75277	Male	23	Cancer	3000
75278	Male	24	HIV+	5000
75275	Male	33	Diabetes	2500
75275	Female	38	Diabetes	2800
75275	Female	36	Diabetes	2600

Table 2: Anonymization at attribute level

ZipCode	Gender	Age	Disease	Expense
7527*	Person	[21-30]	Flu	100
7527*	Person	[21-30]	Cancer	3000
7527*	Person	[21-30]	HIV+	5000
7527*	Person	[31-40]	Diabetes	2500
7527*	Person	[31-40]	Diabetes	2800
7527*	Person	[31-40]	Diabetes	2600

Table 3: Anonymization at cell level

ZipCode	Gender	Age	Disease	Expense
7527*	Male	[21-25]	Flu	100
7527*	Male	[21-25]	Cancer	3000
7527*	Male	[21-25]	HIV+	5000
75275	Person	[31-40]	Diabetes	2500
75275	Person	[31-40]	Diabetes	2800
75275	Person	[31-40]	Diabetes	2600

Because anonymization via generalization at the cell level generates data that contains different generalization levels

*Corresponding author.

within a column, utilizing such data becomes more complicated than utilizing the data generated via generalization at the attribute level. However, generalization at the cell level causes less information loss than generalization at the attribute level. Hence, as far as data quality is concerned, generalization at the cell level seems to generate better data than generalization at the attribute level.

Anonymization via generalization at the cell level can proceed in two steps. First, all records are partitioned into several groups such that each group contains at least k records. Then, the records in each group are generalized such that their values at each quasi-identifier are identical. To minimize the information loss incurred by the second step, the first step should place similar records (with respect to the quasi-identifiers) in the same group. In the context of data mining, clustering is a useful technique that partitions records into clusters such that records within a cluster are similar to each other, while records in different clusters are most distinct from one another. Hence, clustering could be used for k-anonymization.

This work proposes a new clustering-based method for k-anonymization. This method has a time complexity of $O(\frac{n^2}{k})$, where n is the number of records. It first partitions all records into $\lfloor \frac{n}{k} \rfloor$ groups, and then adjusts the records in each group such that each group contains at least k records. This method differs from previously proposed clustering-based k-anonymization methods in two ways. First, it attempts to build all clusters simultaneously. In contrast, the methods proposed by Byun *et al* [1] and Loukides and Shao [7] build one cluster at a time, which might limit the assignment of records to clusters. Second, it is more resilient to outliers than the method proposed by Byun *et al* [1]. Performance study compares the proposed method against the method proposed by Byun *et al* [1]. The results show that the proposed method outperforms their method with respect to information loss.

The rest of this paper is organized as follows. Section 2 reviews basic concepts on k-anonymization with a focus on clustering-based methods. Section 3 proposes our algorithm, and Section 4 gives a performance study of the proposed algorithm. Finally, Section 5 concludes this paper.

2. BASIC CONCEPTS

The k-anonymity model has attracted much attention for the past few years. Many approaches have been proposed for k-anonymization and its variations. Please refer to Ciriani *et al* [3] for a survey of various k-anonymization approaches. This section first describes the concept of information loss, which will be used throughout this paper to evaluate the effectiveness of k-anonymization approaches. This section then reviews several recently proposed clustering-based k-anonymization approaches.

2.1 Information Loss

The notion of information loss is used to quantify the amount of information that is lost due to k-anonymization. The description in this subsection is based on Byun *et al* [1]. Please refer also to Byun *et al* [1] for details.

Let $\mathcal{T}$ denote a set of records, which is described by m numeric quasi-identifiers $N_1, \ldots, N_m$ and q categorical quasi-identifiers $C_1, \ldots, C_q$. Let $\mathcal{P} = \{P_1, \ldots, P_p\}$ be a partitioning of $\mathcal{T}$, namely, $\bigcup_{i \in [1,p]} P_i = \mathcal{T}$, and $P_{\hat{i}} \cap P_{\check{i}} = \emptyset$ for any $\hat{i} \neq \check{i}$. Each categorical attribute C_i is associated with a taxonomy tree T_{C_i}, that is used to generalize the values of this attribute.

Consider a set $P \subset \mathcal{T}$ of records. Let $\hat{N}_i(P)$, $\check{N}_i(P)$ and $\overline{N}_i(P)$ respectively denote the max, min and average values of the records in P with respect to the numeric attribute N_i. Also, let $\breve{C}_i(P)$ denote the set of values of the records in P with respect to the categorical attribute C_i, and let $T_{C_i}(P)$ denote the maximal subtree of T_{C_i} rooted at the lowest common ancestor of the values of $\breve{C}_i(P)$. Then, the *diversity* of P, denoted by $D(P)$, is defined as:

$$D(P) = \sum_{i \in [1,m]} \frac{\hat{N}_i(P) - \check{N}_i(P)}{\hat{N}_i(\mathcal{T}) - \check{N}_i(\mathcal{T})} + \sum_{i \in [1,q]} \frac{H(T_{C_i}(P))}{H(T_{C_i})} \quad (1)$$

where $H(T)$ represents the height of tree T.

Let r' and r^* be two records, then the distance between r' and r^* is defined as the diversity of the set $\{r', r^*\}$, i.e., $D(\{r', r^*\})$.

The *centroid* of P is a record whose value of attribute N_i equals $\overline{N}_i(P)$, and value of attribute C_i equals the lowest common ancestor of the values of $\breve{C}_i(P)$ in the taxonomy tree T_{C_i}. The distance between a record r and a cluster P is defined as the number of records in P times the distance between r and the centroid of P.

To anonymize the records in P means to generalize these records to the same values with respect to each quasi-identifier. The amount of information loss occurred by such a process, denoted as $L(P)$, is defined as:

$$L(P) = |P| \times D(P). \quad (2)$$

where $|P|$ represents the number of records in P.

Consider a partitioning $\mathcal{P} = \{P_1, \ldots, P_p\}$ of $\mathcal{T}$. The total information loss of $\mathcal{P}$ is the sum of the information loss of each $P_{i \in [1,p]}$. To ensure the data quality of $\mathcal{T}$ after anonymization, an anonymization method should try to construct a partitioning $\mathcal{P}$ such that total information loss of $\mathcal{P}$ is minimized.

2.2 Clustering-Based Approaches

Byun *et al* [1] proposed the greedy k-member clustering algorithm (k-member algorithm for short) for k-anonymization. This algorithm works by first randomly selecting a record r as the seed to start building a cluster, and subsequently selecting and adding more records to the cluster such that the added records incur the least information loss within the cluster. Once the number of records in this cluster reaches k, this algorithm selects a new record that is the furthest from r, and repeats the same process to build the next cluster.

Eventually, when there are fewer than k records not assigned to any clusters yet, this algorithm then individually assigns these records to their closest clusters. This algorithm has two drawbacks. First, it is slow. The time complexity of this algorithm is $O(n^2)$. Second, it is sensitive to outliers. To build a new cluster, this algorithm chooses a new record that is the furthest from the first record selected for previous cluster. If the data contains outliers, it is likely that outliers have a great chance of being selected. If a cluster contains outliers, the information loss of this cluster increases.

Loukides and Shao [7] proposed another greedy algorithm for k-anonymization. Similar to the k-member algorithm, this algorithm builds one cluster at a time. But, unlike the k-member algorithm, this algorithm chooses the seed (i.e., the first selected record) of each cluster randomly. Also, when building a cluster, this algorithm keeps selecting and adding records to the cluster until the diversity (similar to information loss) of the cluster exceeds a user-defined threshold. Subsequently, if the number of records in this cluster is less than k, the entire cluster is deleted. With the help of the user-defined threshold, this algorithm is less sensitive to outliers. However, this algorithm also has two drawbacks. First, it is difficult to decide a proper value for the user-defined threshold. Second, this algorithm might delete many records, which in turn cause a significant information loss. The time complexity of this algorithm is $O(\frac{n^2 \log(n)}{c})$, where c is the average number of records in each cluster.

Chiu and Tsai [2] proposed a weighted feature C-means clustering algorithm for k-anonymization. Unlike the previous two algorithms, this algorithm attempts to build all clusters simultaneously by first randomly selecting $\lfloor \frac{n}{k} \rfloor$ records as seeds. This algorithm then assigns all records to their respective closest clusters, and subsequently updates feature weights to minimize information loss. This algorithm iterates this step until the assignment of records to clusters stops changing. As some clusters might contain less than k records, a final step is needed to merge those small clusters with large clusters to meet the constraint of k-anonymity. The time complexity of this algorithm is $O(\frac{cn^2}{k})$, where c is the number of iterations needed for the assignment of records to clusters to converge.

3. ONE-PASS K-MEANS ALGORITHM

This section proposes the One-pass K-means Algorithm (OKA for short) for k-anonymization. This algorithm derives from the K-Means algorithm, but it only runs for one iteration. The OKA algorithm proceeds in two stages: the *clustering* stage and the *adjustment* stage.

3.1 Clustering Stage

Let $\mathcal{T}$ denote the set of records to be anonymized, and $K = \lfloor \frac{n}{k} \rfloor$, where n is the number of records and k is the threshold value for k-anonymity. During the clustering stage, the OKA algorithm first sorts all records in $\mathcal{T}$ by their quasi-identifiers. The algorithm then randomly picks K records as the seeds to build K clusters. Then, for each record $r \in \mathcal{T}$, the algorithm finds the cluster that is closest to r, assigns r to this cluster, and subsequently updates the centroid of this cluster. The distance between a cluster and a record r is defined as the number of records in the clusters times the distance between r and the centroid of the cluster. The clustering stage of the OKA algorithm is shown in Figure 1. It has a complexity of $O(\frac{n^2}{k})$.

Input: a set $\mathcal{T}$ of n records; the value k for k-anonymity
Output: a partitioning $\mathcal{P} = \{P_1, \ldots, P_K\}$ of $\mathcal{T}$

1. Sort all records in $\mathcal{T}$ by their quasi-identifiers;
2. Let $K := \lfloor \frac{n}{k} \rfloor$;
3. Randomly select K distinct records $r_1, \ldots, r_K \in \mathcal{T}$;
4. Let $P_i := \{r_i\}$ for $i = 1$ to K;
5. Let $\mathcal{T} := \mathcal{T} \setminus \{r_1, \ldots, r_K\}$;
6. **While** $(\mathcal{T} \neq \emptyset)$ **do**
7. Let r be the first record in $\mathcal{T}$;
8. Calculate the distance between r to each P_i;
9. Add r to its closest P_i; update centroid of P_i;
10. Let $\mathcal{T} := \mathcal{T} \setminus \{r\}$;
11. **End of While**

Figure 1: OKA algorithm: the clustering stage

The clustering stage of the OKA algorithm is similar to the traditional K-Mean, with three distinctions. First, the centroid of a cluster is updated whenever a record is added to the cluster. Hence, each centroid accurately reflects the current center of a cluster, and consequently, the quality of subsequent assignment of records to clusters improves. Second, because all records are pre-sorted by their quasi-identifiers, identical records (with respect to quasi-identifiers) will always be assigned to the same cluster. Finally, the distance between a record and a cluster depends not only on the distance between the record and the centroid of the cluster, but also on the size of the cluster. This reduces the chance of generating clusters that are too large or too small.

3.2 Adjustment Stage

After the clustering stage, a set of clusters are constructed, but some of these clusters might contain less than k records. Therefore, further adjustment is needed to satisfy the k-anonymity constraint. The goal of the adjustment stage is to adjust the set of clusters constructed in the clustering stage such that every cluster contains at least k records. It is important that such an adjustment should also try to minimize the total information loss.

Figure 2 shows the adjustment stage of the OKA algorithm. First, some records are removed from those clusters with more than k records, and these records are subsequently added to those clusters with less than k records. The records that are removed from a cluster are those most distant from the centroid of the cluster. Those removed records are added to their respective closest clusters with less than k records until no such a cluster exists. If all clusters contain no less than k records and there are still some records not yet assigned to any cluster, then these records are simply assigned

to their respective closest clusters. The time complexity of the adjustment stage is also $O(\frac{n^2}{k})$.

Input: a partitioning $\mathcal{P} = \{P_1, \ldots, P_K\}$ of $\mathcal{T}$
Output: an adjusted partitioning $\mathcal{P} = \{P_1, \ldots, P_K\}$ of $\mathcal{T}$

1. Let $R := \emptyset$;
2. **For** each cluster $P \in \mathcal{P}$ with $|P| > k$ **do**
3. Sort records in P by distance to centroid of P;
4. **While** ($|P| > k$) **do**
5. $r \in P$ is the record farthest from centroid of P;
6. Let $P := P \setminus \{r\}$; $R := R \cup \{r\}$;
7. **End of While**
8. **End of For**
9. **While** ($R \neq \emptyset$) **do**
10. Randomly select a record r from R;
11. Let $R := R \setminus \{r\}$;
12. **If** $\mathcal{P}$ contains cluster P_i such that $|P_i| < k$ **then**
13. Add r to its closest cluster P_i satisfying $|P_i| < k$;
14. **Else**
15. Add r to its closest cluster;
16. **End If**
17. **End of While**

Figure 2: OKA algorithm: the adjustment stage

4. EXPERIMENTAL RESULTS

This section compares the performance of the OKA algorithm against that of the k-member algorithm [1]. Both algorithms are implemented in Java and run on a Desktop PC with Intel Core2Duo 2.2 GHz CPU and 2GB of RAM under MS Window Vista operating system.

This experiment uses the Adult dataset from the UC Irvine Machine Learning Repository [5], which is considered as a standard benchmark for k-anonymization. Eight attributes of the Adult dataset are used as the quasi-identifiers, including *age, work class, education, marital status, occupation, race, gender*, and *native country*. Among these eight attributes, *age* and *education* are treated as numeric attributes, while the other six attributes as categorical attributes. The taxonomy trees for these six categorical attributes are based on those defined in [4].

Figure 3 shows the information loss of both algorithms. The OKA algorithm consistently causes less information loss than the k-member algorithm. Figure 4 shows that the OKA algorithm takes much less time than the k-member algorithm.

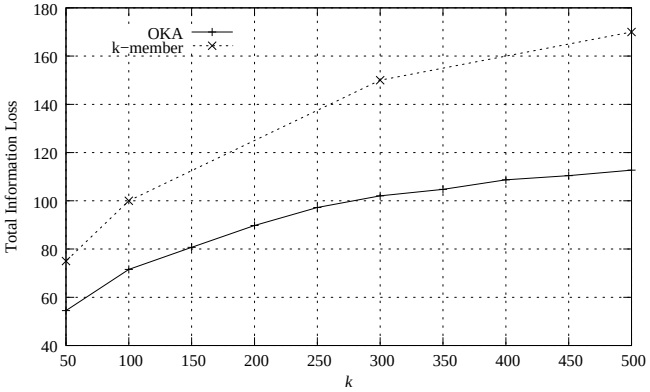

Figure 3: Information loss vs. k

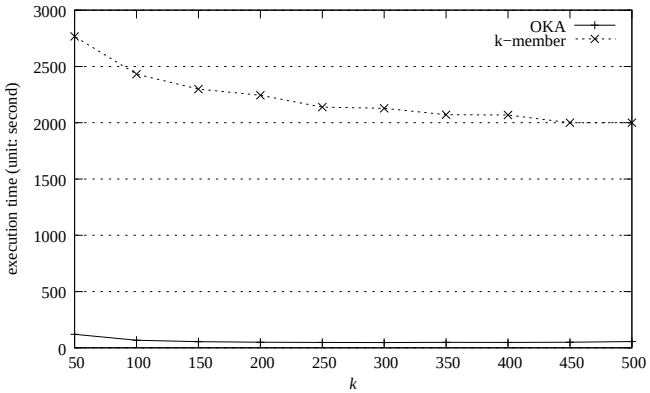

Figure 4: Execution time vs. k

5. CONCLUSIONS

This work proposes an algorithm for k-anonymization. Although the proposed OKA algorithm outperforms the k-member algorithm in terms of both time and information loss, there is still room for possible improvement. First, increasing the number of iterations during the clustering stage might help reducing the information loss and improving the resilience to outliers. Second, during the adjustment stage of the OKA algorithm, it may be better to suppress some of the small clusters rather than adding more records to them. Again, such a decision should depend on which method incurs more information loss. Third, many variations of the k-anonymity model have been proposed to further protect the data from identification, e.g., l-diversity [8], t-closeness [6], (α, k)-anonymity [11]. It should be possible to extend the OKA algorithm to these models.

6. REFERENCES

[1] J.-W. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymization using clustering techniques. In *Internal Conference on Database Systems for Advanced Applications (DASFAA)*, 2007.

[2] C.-C. Chiu and C.-Y. Tsai. A k-anonymity clustering method for effective data privacy preservation. In *Third International Conference on Advanced Data Mining and Applications (ADMA)*, 2007.

[3] V. Ciriani, S. D. C. di Vimercati, S. Foresti, and P. Samarati. K-anonymity. *Security in Decentralized Data Management (to appear)*.

[4] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, pages 205–216, Washington, DC, USA, 2005. IEEE Computer Society.

[5] S. Hettich and C. Merz. Uci repository of machine learning datasets. 1998.

[6] N. Li and T. Li. t-closeness: Privacy beyond k-anonymity and l-diversity. In *International Conference on Data Engineering (ICDE)*, 2007.

[7] G. Loukides and J. Shao. Capturing data usefulness and privacy protection in k-anonymisation. In *Proceedings of the 2007 ACM symposium on Applied Computing*, 2007.

[8] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE 2006)*, 2006.

[9] P. Samarati. Protecting respondent's privacy in microdata release. *TKDE*, 13(6), 2001.

[10] L. Sweeney. K-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[11] R. C.-W. Wong, J. Li, A. W.-C. Fu, and K. Wang. (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.