Privacy Problems with Anonymized Transaction Databases

Taneli Mielikäinen

HIIT Basic Research Unit
Department of Computer Science
University of Helsinki, Finland
`Taneli.Mielikainen@cs.Helsinki.FI`

Abstract. In this paper we consider privacy problems with anonymized transaction databases, i.e., transaction databases where the items are renamed in order to hide sensitive information. In particular, we show how an anonymized transaction database can be deanonymized using non-anonymized frequent itemsets. We describe how the problem can be formulated as an integer programming task, study the computational complexity of the problem, discuss how the computations could be done more efficiently in practice and experimentally examine the feasibility of the proposed approach.

1 Introduction

A *transaction database* is a bag (i.e., a multi-set) of *itemsets* (called *transactions* in the transaction database) that are finite subsets of the set of items. The prominent example of transaction data is market basket data. In that case each transaction corresponds to one purchase event and each item is a salable product. Also the popular representation of text documents as a set of words contained in the document can be seen as transaction data: items are possible words in documents and each transaction represents one document. A collection of web pages is yet another example of transaction databases. For each web page in the collection there is one transaction containing its out-going (or alternatively incoming) links. Thus, the items in this case are the links. Transaction databases have important role in data mining. For example, association rules were first defined for transaction databases [1].

Many times the owner of the transaction database is not willing to reveal the database to others due to confidentiality or potential valuability of the database but might still be interested to share the data if the privacy problems can be avoided. For example, the data owner might be interested to compute and possibly also to distribute some summaries computed from the data. Hiding sensitive information about the data has been recognized to be an important aspect of data mining and it has recently been studied actively, see e.g. [2,3].

One particularly simple approach to hide sensitive information is to rename the set $\mathcal{I}$ of items e.g. by a random bijection $f : \mathcal{I} \to \mathcal{J}$ where $\mathcal{J}$ is a set with same cardinality as $\mathcal{I}$.

A great advantage of this approach to anonymize transaction databases is that the existing methods to manipulate transaction data can be directly applied to the anonymized version of the data if the methods depend only on the syntactical structure of data. Examples of such methods are e.g. decision tree induction [4], learning Bayesian networks [5] and association rule mining [6].

A downside of the method is that the data mining results computed from the anonymized transaction database are not always very informative. Sometimes it is not necessary to remove the anonymization from the data mining results. For example, in the case of classification, the data owner can provide the anonymized data to a data miner that constructs the classifier for the anonymized data and gives that to the data owner. As the data owner knows the mapping to anonymize the data, she can classify the transactions of other parties on demand by anonymizing the transactions, classifying them by the classifier and deanonymizing the classification result, i.e., the class of the transaction. The knowledge about the data represented by the classifier is not shared by sharing the anonymized version of the classifier.

Even more difficult problems have to be faced when the data mining result is inherently representing some knowledge about the original data and the purpose of the knowledge is to be shared with other parties. This is the case e.g. with association rules. They are a summary of the data as conditional empirical probabilities over sets of items. The data owner might be willing to share (or more probably sell) this kind of summary information about her database although she does not want to reveal the actual database.

It has recently been shown that finding a database compatible with given collection of association rules or frequent itemsets is NP-hard [7,8]. Thus, at least some privacy is preserved when releasing the frequent itemsets of the transaction database although it is known that frequent itemsets tell much also about other itemsets [9]. That is, releasing frequent itemsets might be an acceptable privacy risk for the data owner.

If the data owner can compute the association rules by herself then there is no need for anonymizing the data. Sometimes, however, the data owner is not interested or capable of conducting the data mining by herself but is willing to use the services of some data miner. In that case it is quite reasonable to assume that the data owner is not interested or willing to do very advanced obscuration of the database.

There are methods for hiding sensitive association rules if the sensitive rules are known [10,11,12,13]. (The problem of hiding all sensitive rules optimally has been shown to be NP-hard [14].) This requires, however, that the association rules that should not be revealed are known in advance and modifying the transactions changes also the frequencies of the itemsets. When the data miner mines association rules from the anonymized database, the data owner can interactively control which itemsets she is willing to reveal and which must be hidden. (There are methods for hiding such rules that tolerate certain attacks [15].)

In this paper we show that it is very dangerous to share the anonymized data and non-anonymized data mining results. In particular, we show that it is possible to estimate the original names of frequent items from anonymized data and non-anonymized frequent itemsets in non-deterministic polynomial time even in the case when the frequencies are independently perturbed by noise, some frequent itemsets are omitted and some itemsets are falsely claimed to be frequent. Furthermore, we show that even the simplest variants of the problem are at least as hard as graph isomorphism problem, but the exact maximum likelihood solution can be feasible in practice and that more efficiently solvable relaxed versions of the problems give reasonable solutions for the exact problem.

The rest of the paper is organized as follows. In Section 2 we describe how the problem of finding the original item names can be formulated as an integer programming task of reasonable size. In Section 3 we study the computational issues of the problem: we show that it is at least as difficult as graph isomorphism and discuss how the integer programming formulations could be relaxed and simplified without losing too much of the accuracy of the solution. The feasibility of the integer programming approach and its relaxations are evaluated experimentally in Section 4. Section 5 is a short conclusion.

2 Deanonymizing Transactions

To be able to consider anonymized transaction database, it is useful to define transaction databases and anonymizations of the transaction databases:

Definition 1 (transaction databases). *Let the collection of all items be denoted by $\mathcal{I}$. A set of items is called an* itemset*. A* transaction database *D is a bag of* transactions*, i.e., a multiset of subsets $D_1, \ldots, D_n$ of $\mathcal{I}$.*

Definition 2 (anonymization). *An anonymization of a transaction database D over $\mathcal{I}$ is a random mapping $f : \mathcal{I} \to \{1, \ldots, |\mathcal{I}|\}$. We use the shorthands $f(X) = \{f(A) : A \in X\}$ for itemsets $X \subseteq \mathcal{I}$, $f(\mathcal{F}) = \{f(X) : X \in \mathcal{F}\}$ for itemset collections $\mathcal{F} \subseteq 2^{\mathcal{I}}$, and $f(D) = \{f(D_i) : 1 \leq i \leq n\}$ for transaction databases.*

The most prominent data mining task for transaction databases is finding all σ-frequent itemsets in the database.

Problem 1 (discovery of σ-frequent itemsets). Given a transaction database D and minimum frequency threshold $\sigma \in [0, 1]$, find all σ-frequent itemsets, i.e., find the collection

$$\mathcal{F}(\sigma, D) = \{X \subseteq \mathcal{I} : fr(X, D) \geq \sigma\} .$$

where the *frequency* $fr(X, D)$ of an itemset X in D is the fraction of transactions containing X, i.e.,

$$fr(X, D) = \frac{|\{i : X \subseteq D_i \in D\}|}{|D|}.$$

There exist highly optimized methods for finding the σ-frequent itemsets but still the resources needed can be too high for an ordinary user [16]. Thus, the data owner might buy this service from a specialized data mining company if the privacy issues are not of the highest priority. It would be desirable that the privacy of the database would not be sacrificed completely. A very simple and computationally cheap solution is to first anonymize the database and give that to the data miner and deanonymize the data mining result. More specifically we consider the following procedure:

Anonymization of the database. The data owner chooses a random anonymization function f and sends $f(D)$ and a minimum frequency threshold $\sigma \in [0,1]$ to the data miner.

Mining the anonymized database. The data miner computes the collection $\mathcal{F}(\sigma, f(D))$ of frequent itemsets and sends $\mathcal{F}(\sigma, f(D))$ with their frequencies to the data owner.

Deanonymization of the σ-frequent itemsets. The data owner computes $\mathcal{F}(\sigma, D) = f^{-1}(\mathcal{F}(\sigma, f(D)))$ and publishes $\mathcal{F}(\sigma, D)$.

If the data miner is honest then this procedure does not cause any privacy problems and in fact the anonymization would be unnecessary. In this case the only information revealed to possibly malicious parties is the collection of σ-frequent itemsets in D and their frequencies. We assume that the data miner computes the frequent itemsets correctly. However, there is no reason (excluding possible ethical and legal reasons) why the data miner should not try to find out more about the data. The frequent itemsets can leak some information about the relationship between the anonymized and the original databases. Thus, the anonymization approach is not secure in a strict cryptographic sense. The data owner, however, can hope that not too much is leaked. As an extreme case, the leakage could reveal the whole original database, i.e., it might be possible to estimate the correct names of the items from the non-anonymized frequent itemsets and their frequencies.

Problem 2 (deanonymization of transaction databases from σ-frequent itemsets). Given an anonymized transaction database $f(D)$, the collections $\mathcal{F}(\sigma, f(D))$ and $\mathcal{F}(\sigma, D)$ of σ-frequent itemsets, find a bijection $g : f(\mathcal{I}) \to \mathcal{I}$ such that $X \in \mathcal{F}(\sigma, f(D)) \iff g(X) \in \mathcal{F}(\sigma, D)$.

One possible solution mapping g for the above problem is the inverse of the original anonymization function f. Clearly, only the frequent items can be deanonymized based on frequent itemsets since infrequent items do not occur in the collection of frequent itemsets. Fortunately, the frequent items are often considered the most important ones among all items.

Problem 2 is essentially about finding a matching between the items in $\mathcal{I}$ and the integers in $\{1, \ldots, |\mathcal{I}|\}$ that fulfills some additional constraints. Thus, the problem can be formulated as solving integer inequalities as follows. For each pair of itemsets $X \in \mathcal{F}(\sigma, D)$ and $Y \in \mathcal{F}(\sigma, f(D))$ with $|X| = |Y|$ and $fr(X, D) = fr(Y, f(D))$ there is an indicator variable $u_{X,Y} \in \{0,1\}$ indicating whether all items in X are matched with the integers in Y.

The variables $u_{X,Y}$ are constrained as follows. It is required that each itemset X in $\mathcal{F}(\sigma, D)$ is matched exactly to one itemset of same cardinality in $\mathcal{F}(\sigma, f(D))$ and the vice versa. That is,

$$\sum_{X \in \mathcal{F}(\sigma,D)} u_{X,Y} = 1 \quad \text{and}$$
$$\sum_{Y \in \mathcal{F}(\sigma,f(D))} u_{X,Y} = 1.$$

Furthermore, it is required that if $X \in \mathcal{F}(\sigma, D)$ and $Y \in \mathcal{F}(\sigma, f(D))$ are matched then all their items are matched with each other. This constraint can be expressed by requiring that the sum of indicator variables between the items of X and Y sum up at least to value $|X| \, u_{X,Y} = |Y| \, u_{X,Y}$, i.e.,

$$\sum_{A \in X, B \in Y} u_{A,B} - |X| \, u_{X,Y} \geq 0.$$

A bit vector u fulfilling these constraints is a solution to Problem 2: each itemset in $\mathcal{F}(\sigma, D)$ (and thus also each item) is matched with exactly one itemset in $\mathcal{F}(\sigma, f(D))$ with $|X| = |Y|$ and $fr(X, D) = fr(Y, f(D))$, and if itemsets $X \in \mathcal{F}(\sigma, D)$ and $Y \in \mathcal{F}(\sigma, f(D))$ are matched then the items in X are Y are matched with each other.

If the frequent itemsets are restrictive enough, it is possible to find out the correct names for the items, i.e., the data miner could construct the original data from the anonymized data and frequent itemsets.

The data owner is probably willing to use only quite simple techniques to obscure the frequent itemsets if she is not interested to mine the data by herself.

First, the data owner could perturb the frequencies of the frequent itemsets a small amount independently, e.g., by adding to each frequency a Gaussian zero-mean noise with small variance. With high noise levels this can reduce the chance to find a naming of the items that is close to the correct one but it also reduces the usefulness of the itemsets. Let $c_{X,Y}$ be the cost of matching $X \in \mathcal{F}(\sigma, D)$ with $Y \in \mathcal{F}(\sigma, f(D))$ with $|X| = |Y|$. Then the problem of solving the integer inequalities becomes an integer programming problem:

$$\begin{aligned} \min_u & \sum_{X \in \mathcal{F}(\sigma,D), Y \in \mathcal{F}(\sigma,f(D))} c_{X,Y} u_{X,Y} \\ \text{subject to} & \\ & \sum_{X \in \mathcal{F}(\sigma,D)} u_{X,Y} = 1 \\ & \sum_{Y \in \mathcal{F}(\sigma,f(D))} u_{X,Y} = 1 \\ & \sum_{A \in X, B \in Y} u_{A,B} - |X| \, u_{X,Y} \geq 0 \\ & u_{X,Y} \in \{0, 1\} \end{aligned}$$

If the additive noise $n_{X,Y}$ in the frequencies is Gaussian and loss of matching $X \in \mathcal{F}(\sigma, D)$ with $Y \in \mathcal{F}(\sigma, f(D))$ with $|X| = |Y|$ is defined to be $c_{X,Y} = |fr(X,D) - fr(Y, f(D)) + n_{X,Y}|^2$ then the optimal solution of the integer program gives the maximum likelihood estimate of the correct item naming.

Second, the data owner could omit some frequent itemsets and add some false frequent itemsets. That situation can be modeled by adding one new indicator variable u_Z with cost c_Z for each $Z \in \mathcal{F}(\sigma, D) \cup \mathcal{F}(\sigma, f(D))$ indicating that itemset Z is not matched with any itemset and minimizing the cost. That is,

$$\min_u \sum_{Z \in \mathcal{F}(\sigma,D) \cup \mathcal{F}(\sigma,f(D))} c_Z u_Z$$

subject to

$$u_Y + \sum_{X \in \mathcal{F}(\sigma,D)} u_{X,Y} = 1$$

$$u_X + \sum_{Y \in \mathcal{F}(\sigma,f(D))} u_{X,Y} = 1$$

$$\sum_{A \in X, B \in Y} u_{A,B} - |X|\, u_{X,Y} \geq 0$$

$$u_{X,Y} \in \{0,1\}$$

Combining the above integer programs it is possible to model the situation where both the frequencies and the structure of the collection are modified.

3 On Feasibility of Deanonymization

The integer programming formulations of Problem 2 and its generalizations consist of polynomial number of variables, equalities and inequalities in $|\mathcal{F}(\sigma, D)|$ and in $|\mathcal{F}(\sigma, f(D))|$. These numbers are clearly bounded above by the product $\mathcal{O}(|\mathcal{F}(\sigma, D)|\,|\mathcal{F}(\sigma, f(D))|)$ when both collections are non-empty. Thus, as integer programming is in NP, also Problem 2 is.

The complexity of the problem can be lower bounded by the complexity of the graph isomorphism problem:

Problem 3 (graph isomorphism). Given a graphs $G = \langle V, E \rangle$ and $G' = \langle V', E' \rangle$, find a bijection $f : V \to V'$ such that $\{u, v\} \in E \iff \{f(u), f(v)\} \in E'$.

On one hand the graph isomorphism is in $NP \cap co-NP$ thus it is not likely that it would be NP-complete, but on the other hand there are no polynomial time algorithms for the problem [17,18].

Theorem 1. *Problem 2 is (polynomially) at least as hard as Problem 3.*

Proof. To show the hardness we construct from a graph $G = \langle V, E \rangle$ a transaction database D_G with polynomial number of transaction in the size of G. The transaction database consists of a transaction $\{A, B\}$ for each edge $\{A, B\} \in E$

and $\max_{C \in V} |B \in V : \{C, B\} \in E| - |B \in V : \{A, B\} \in E|$ copies of transactions $\{A\}$ for each vertex $A \in V$. Then $fr(A, D_G) = fr(B, D_G)$ for all $A, B \in V$, $fr(X, D_G) = fr(Y, D_G)$ for each $X, Y \in E$ and $fr(X, D_G) = 0$ for all other X. Thus, choosing a positive minimum frequency threshold that is small enough $\mathcal{F}(\sigma, D) = V \cup E$. The transaction database D_G consists of at most $|V|\,|E|$ transactions which is clearly polynomial in $|V|$ and $|E|$ and the size of each transaction is at most two.

Thus, both graphs G and G' can be represented by polynomial-size transaction databases in such a way that the frequent itemsets in D_G and $D_{G'}$ can be matched if and only if G and G' are isomorphic. $\square$

The practical vulnerability of the anonymized transaction databases depends on our ability to solve the computational task of finding the mapping in reasonable time. As we have the integer programming formulation of the problem, a natural solution is to solve the integer program. However, this is often computationally too demanding. A standard approach is to relax the integer program to a linear program. For linear programming there exist algorithms efficient in theory and practice [19]. In our case the relaxation of the integer program to a liner program means that the constraints $u_{X,Y} \in \{0, 1\}$ are replaced by constraints $0 \leq u_{X,Y} \leq 1$. A major difficulty in this relaxation is that instead of matching the item names, the solution of the linear program gives for each itemset $Y \in \mathcal{F}(\sigma, f(D))$ values of matching with each $X \in \mathcal{F}(\sigma, D)$ that sum up to one. These values can be translated into integers by means of the following procedure:

1. If $\mathcal{F}(\sigma, f(D)) = \emptyset$ then halt. Otherwise choose Y randomly from $\mathcal{F}(\sigma, f(D))$.
2. Choose a pair $X \in \mathcal{F}(\sigma, D)$ for Y w.r.t. the distribution determined by $u_{Z,Y}$ for all Z.
3. Remove Y from $\mathcal{F}(\sigma, f(D))$ and X from $\mathcal{F}(\sigma, D)$ and set $u_{X,Z} = 0$ and $u_{Z,Y}$ for all Z.

Another possibility to find the matching between $\mathcal{I}$ and $\{1, \ldots, |\mathcal{I}|\}$ is to require that $u_{A,B} \in \{0, 1\}$ for all $A \in \mathcal{I}, B \in \{1, \ldots, |\mathcal{I}|\}$. Usually the number of frequent items is much smaller than the number all frequent itemsets. There exist sophisticated techniques to solve this kind of mixed integer programs [20].

The frequent itemset collection often gives natural constraints that some items cannot be matched with some numbers. Obviously, this information can be used to prune the constraints and setting some variables to be zero in advance.

4 Experiments

To examine the practical feasibility of the integer programming solutions and its relaxations we conducted preliminary experiments with the course enrollment database of Department of Computer Science, University of Helsinki. More specifically, we wanted to study the effect of noise in frequencies to different estimation methods and different minimum frequency thresholds. The item namings

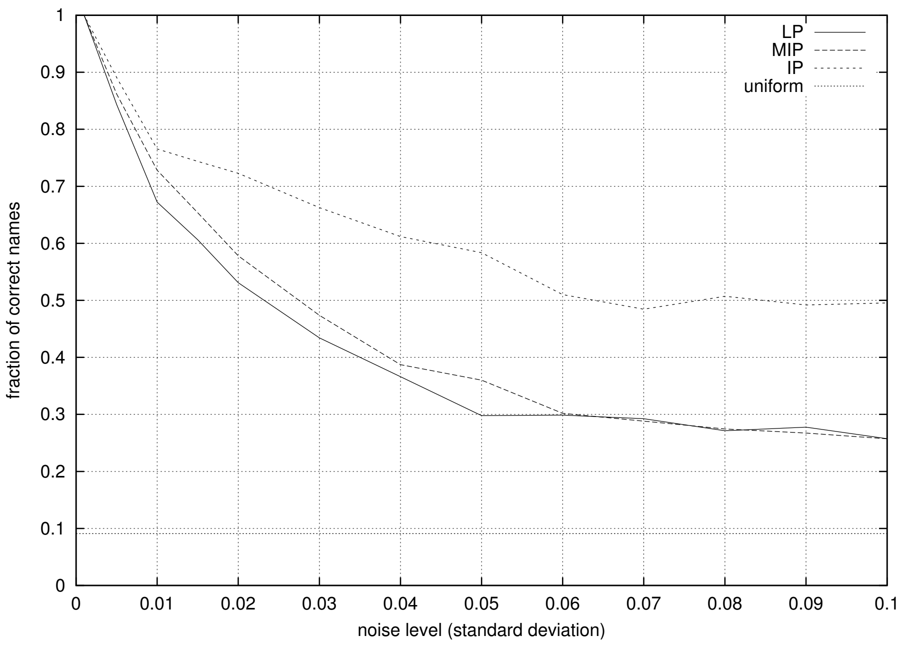

Fig. 1. Deanonymization based on 0.1-frequent itemsets.

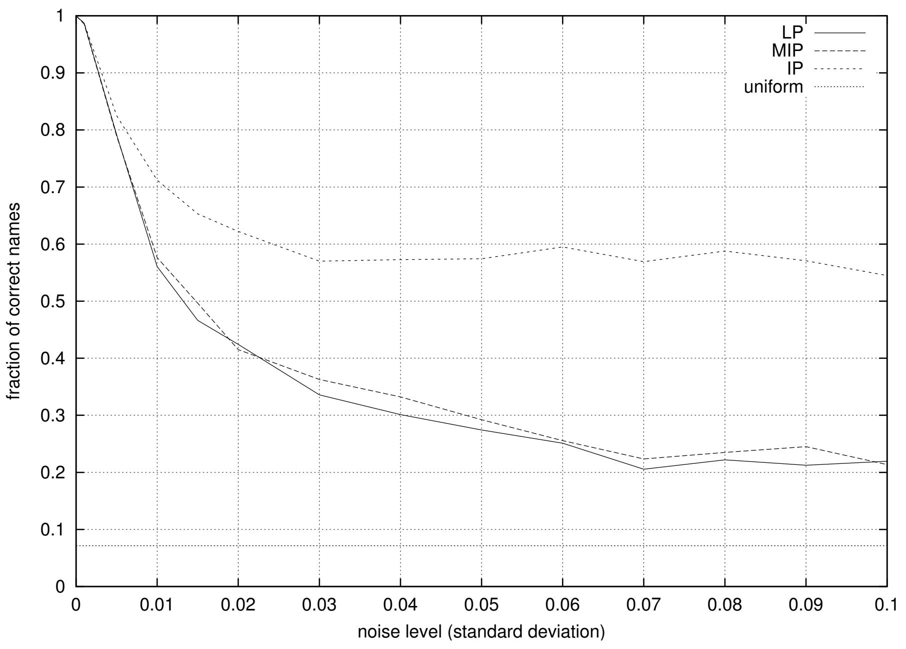

Fig. 2. Deanonymization based on 0.09-frequent itemsets.

were estimated using integer programming where variables were constrained to be either one or zero, mixed integer programming where only the variables corresponding to the singleton itemsets were required to be integer valued, and linear programming where all values were constrained to be between zero and one. The quality of the solution was measured by computing the weight given to the correct item naming of the items divided by the number of frequent items. Thus in the case of integer and mixed integer programming the solutions were readily matchings whereas the linear programming solutions were able to vote several matchings. As the noise were added randomly, all results shown are averages of one hundred experiments.

In all our experiments all methods were always able to find the correct item namings when there were no noise in the frequencies. When the noise level were increased, then the differences between the methods became visible. The integer programming were able to find the most correct item namings. Mixed integer programming and linear programming were approximately equally good. However, the mixed integer programming solution was readily a proper item naming of the items instead of a probability distribution over the possible item names.

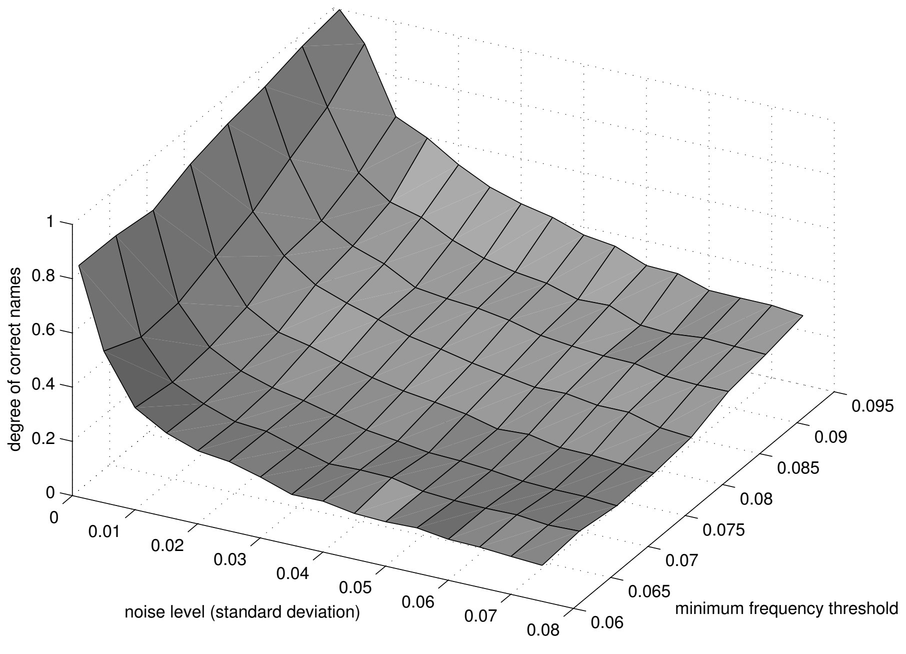

Fig. 3. Deanonymization using linear programming.

In Figure 1 and Figure 2 the results are shown for minimum frequency thresholds 0.1 and 0.09. The uniform voting for all matching pairs is shown as a baseline in both figures. Clearly, even linear programming were able to increase our knowledge about the correct item namings.

In Figure 3 experiments with linear programming for larger variety of different minimum frequency thresholds are shown. The behavior of the method were similar with different thresholds and the linear programming result were always better than the uniform assumption. Overall, the results show that the approach can be used to find namings of the items that are relatively close to the correct ones.

5 Conclusions

In this paper we have studied the privacy problems of renaming items in transaction databases. In particular, we considered the problem of finding the original names of the items using the anonymized transaction database and non-anonymized frequent itemsets. We examined the complexity of the problem and described practical solutions for the problems that can approximate the correct item namings reasonably well. The preliminary experiments on deanonymizing transaction databases shown promising results.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In Buneman, P., Jajodia, S., eds.: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993. ACM Press (1993) 207–216
2. Farkas, C., Jajodia, S.: The inference problem: A survey. SIGKDD Explorations **4** (2002) 6–11
3. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y.: State-of-the-art in privacy preserving data mining. SIGMOD Record **33** (2004) 50–57
4. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth (1984)
5. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Revised second printing edn. Morgan Kaufmann (1988)
6. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Academic Press (2001)
7. Calders, T.: Computational complexity of itemset frequency satisfiability. In: Proceedings of the Twenty-Third ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 13-18, 2004, Maison de la Chimie, Paris, France. ACM (2004)
8. Mielikäinen, T.: On inverse frequent set mining. In Du, W., Clifton, C.W., eds.: Proceedings of the 2nd Workshop on Privacy Preserving Data Mining (PPDM), November 19, 2003, Melbourne, Florida, USA. IEEE Computer Society (2003) 18–23

9. Calders, T., Goethals, B.: Mining all non-derivable frequent itemsets. In Elomaa, T., Mannila, H., Toivonen, H., eds.: Principles of Data Mining and Knowledge Discovery, 6th European Conference, PKDD 2002, Helsinki, Finland, August 19-23, 2002, Proceedings. Volume 2431 of Lecture Notes in Artificial Intelligence. Springer (2002) 74–865
10. Saygin, Y., Verykios, V.S., Clifton, C.: Using unknowns to prevent discovery of association rules. SIGMOD Record **30** (2001) 45–54
11. Oliveira, S.R.M., Zaïane, O.R.: Privacy preserving frequent itemset mining. In Clifton, C., Estivill-Castro, V., eds.: IEEE Workshop on Privacy, Security, and Data Mining. Volume 14 of Conferences in Research and Practice in Information Technology. (2002)
12. Oliveira, S.R.M., Zaïane, O.R.: Protecting confidential knowledge by data sanitation. In Wu, X., Tuzhilin, A., Shavlik, J., eds.: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA. IEEE Computer Society (2003) 613–616
13. Verykios, V.S., Elmagarmid, A.K., Elisa Bertino, F., Saygin, Y., Dasseni, E.: Association rule hiding. IEEE Transactions on Knowledge and Data Engineering **16** (2004) 434–447
14. Atallah, M.J., Bertino, E., Elmagarmid, A.K., Ibrahim, M., Verykios, V.S.: Disclosure limitation of sensitive rules. In: Proceedings of 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX '99). IEEE Computer Society (1999) 45–52
15. Oliveira, S.R.M., Zaïane, O.R., Saygin, Y.: Secure association rule sharing. In Dai, H., Srikant, R., Zhang, C., eds.: Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference, PAKDD 2004, Sydney, Australia, May 26-28, 2004, Proceedings. Volume 3056 of Lecture Notes in Artificial Intelligence. Springer (2004) 74–85
16. Goethals, B., Zaki, M.J., eds.: Proceedings of the Workshop on Frequent Itemset Mining Implementations (FIMI-03), Melbourne Florida, USA, November 19, 2003. Volume 90 of CEUR Workshop Proceedings. (2003) `http://CEUR-WS.org/Vol-90/`.
17. Kreher, D.L., Stinson, D.R.: Combinatorial Algorithms: Generation, Enumeration and Search. CRC Press (1999)
18. Torán, J.: On the hardness of graph isomorphism. In: 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA. IEEE Computer Society (2000) 180–186
19. Padberg, M.: Linear Optimization and Extensions. 2nd edn. Volume 12 of Algorithms and Combinatorics. Springer-Verlag (1999)
20. Martin, A.: General mixed integer programming: Computational issues for branch-and-cut algorithms. In Jünger, M., Naddef, D., eds.: Computational Combinatorial Optimization: Optimal and Provably Near-Optimal Solutions. Volume 2241 of Lecture Notes in Computer Science. Springer (2001) 1–25