Anonymizing Transaction Data by Integrating Suppression and Generalization

Junqiang Liu[1,2] and Ke Wang[1]

[1] Simon Fraser University, Burnaby, B.C. V5A 1S6, Canada
[2] Zhejiang Gongshang University, Hangzhou, 310018, China
{jjliu,wangk}@cs.sfu.ca

Abstract. Privacy protection in publishing transaction data is an important problem. A key feature of transaction data is the extreme sparsity, which renders any single technique ineffective in anonymizing such data. Among recent works, some incur high information loss, some result in data hard to interpret, and some suffer from performance drawbacks. This paper proposes to integrate generalization and suppression to reduce information loss. However, the integration is non-trivial. We propose novel techniques to address the efficiency and scalability challenges. Extensive experiments on real world databases show that this approach outperforms the state-of-the-art methods, including global generalization, local generalization, and total suppression. In addition, transaction data anonymized by this approach can be analyzed by standard data mining tools, a property that local generalization fails to provide.

Keywords: Anonymity, privacy, information security, transaction data.

1 Introduction

Transaction data, such as shopping transactions [1], web query logs [11], and movie ratings [10], are important sources for knowledge discovery. People are increasingly releasing transaction data to the data mining research community for discovering knowledge that helps improve services. However, transaction data contains significant amount of personal and sensitive information. The release of such data to the public or a third party could breach privacy, as highlighted by recent incidents [2][10]. Transaction data must be anonymized before release.

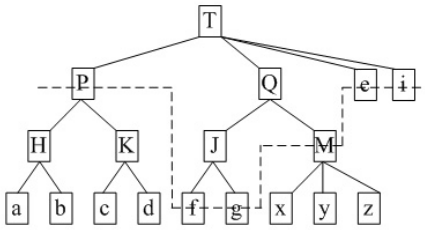
Recently, several works started to address the transaction data anonymization problem [4][5][13][14]. However, these works suffer from a few limitations, namely, incurring high information loss, failing to enable standard data mining tools, and introducing invalid analysis results. Let us examine those prior works using the transaction data in Fig. 1 (a) and the taxonomy in Fig. 1 (b).

Global generalization [13]. The k^m-*anonymity* in [13] requires that every subset of no more than m items is contained in at least k transactions. The global generalization technique (a.k.a. *full subtree generalization* [6]) is employed in [13], which is vulnerable to excessive distortion in the presence of outliers. Let k^∞-*anonymity* denote k^m-*anonymity* with m being the longest transaction length.

M.J. Zaki et al. (Eds.): PAKDD 2010, Part I, LNAI 6118, pp. 171–180, 2010.

TID	transaction database D	*Transactional 2-anonymity*, local generalization	2^∞-*anonymity*, global generalization	2^∞-*anonymity*, suppression	2^∞-*anonymity*, gen. with suppr. [ours]
1	b, c, d	T	T	*, *, d	P
2	a, f, g	P, f, g	T	a, f, g	P, f, g
3	d, f, y, z	K, f, M	T	d, f, *, *	P, f, M
4	c, d, f, x	K, f, M	T	*, d, f, *	P, f, M
5	a, b, c, f, g	P, f, g	T	a, *, *, f, g	P, f, g
6	e, i	T	T	e, *	e, *
7	e	T	T	e	e
8	i	T	T	*	*

(a) Transactional database D and a variety of anonymization solutions



(b) Domain generalization hierarchy H_p

Fig. 1. Transactional database D, anonymizations of D, and taxonomy tree H_P

For example, to achieve 2^∞-*anonymity*, as in the 4^{th} column in Fig. 1 (a), all items are generalized to the top level because of the outlier, {e, i}.

Suppression [14]. The (h, k, p)-*coherence* in [14] demands that every subset of no more than p public items must be contained in at least k transactions and no more than h percent of these transactions contain a common private item. k^m-*anonymity* is its special case with $h = 100\%$ and $p = m$. [14] employs the total item suppression technique to enforce (h, k, p)-*coherence*, which incurs high information loss when the data is sparse. E.g., in the 5^{th} column in Fig. 1 (a), all occurrences of b, c, i, x, y, and z, are removed as indicated by *.

Local generalization [5]. The *transactional k-anonymity* in [5] requires that each transaction has at least k duplicates. Such a requirement is stronger than k^∞-*anonymity* and introduces much more distortion than necessary. The *multi-dimensional generalization* technique [7] is employed in [5]. But, it destroys the *domain exclusiveness* property, e.g., the 3^{rd} column in Fig. 1 (a) shows the data anonymized by [5] where items T, P, and K coexist in the anonymized data, but their domains are not exclusive of each other. The analysis result based on such data are hard to interpret, e.g., according to such data, whenever a transaction contains K, it also contains f. But it is not true with the original data, e.g., the first transaction contains c and d (and hence K), but it does not contain f.

Band matrix method [4]. A method for grouping transactions and permuting the private items in each group to enforce *l-diversity* [9] is presented in [4]. However, invalid analysis results could be derived from the anonymized data. The example in [4] explained this: in the original data, all customers who bought

cream but not *meat* have also bought a *pregnancy test*; while in the data anonymized by [4], only a half of such customers have bought a *pregnancy test*.

This paper, motivated by the limitations of the prior works, proposes to integrate the global generalization technique with the total item suppression technique for enforcing k^m-*anonymity*. Our observation is that suppression can remove outlier items that otherwise will cause substantially generalization of many other items, and generalization can slightly generalize items that otherwise must be suppressed. While a single technique could not perform well, the integration can greatly reduce the overall information loss. Our approach has two strong properties: the anonymized data can be analyzed by standard data mining tools, and results derived from it are true in the original data. This is because both techniques preserve the domain exclusiveness property. For example, the last column in Fig. 1 (a) shows the data anonymized by our approach which suppresses item i and generalizes some other items.

Integrating generalization and suppression is non-trivial because the search space is much larger than only employing one of them. We propose a multi-round, top-down greedy search strategy to address the challenge. Extensive comparative experiments showed that our approach yields better data utility than the prior works and is efficient and scalable in anonymizing real world databases.

The rest of the paper is organized as follows. Section 2 describes the privacy requirement and the anonymization model, Section 3 presents the basic approach that integrates generalization with suppression, Section 4 proposes the key techniques that make our approach efficient and scalable, Section 5 evaluates the applicability of our approach, and Section 6 concludes the paper.

2 Privacy and Anonymization Model

A publisher wants to release a transaction database $D = \{t_1, t_2, \ldots, t_n\}$, where each transaction t_i corresponds to an individual and contains items from an item universe $I = \{i_1, i_2, \ldots, i_q\}$. An adversary tries to link a target individual to his/her transaction with a high probability. To do so, the adversary acquired knowledge from external sources. That is, the adversary knows that the transaction is in the released data and knows some items of the target individual. The publisher wants to prevent such a linking attack.

Definition 1 (Privacy threats and k^m-*anonymity*): A subset of items is called an *itemset*. An itemset X with $|X| \leq m$ is called a *privacy threat* if the number of transactions in D that support X, denoted by $sup(X)$, is less than a user specified anonymity threshold k, i.e., $sup(X) < k$. A transaction t *supports* X if X is a subset of t; D observes k^m-*anonymity* [13] if there is no privacy threat supported by D. ■

Enforcing the privacy notion in Definition 1 assures that the adversary's certainty in making any linking attack is no more than $1/k$.

Anonymization solutions: To enforce the privacy notion, the *full subtree generalization* technique [6][13] and the *total item suppression* technique [14] are

integrated to anonymize D. We assume that a taxonomy tree H_P for generalizing items is available. With the *full subtree generalization* technique, a generalization solution is defined by a cut on H_P. A cut contains *exactly one* item on every root-to-leaf path on H_P, and is denoted by *the set of such items*. E.g., {P, f, g, M, e, i} denotes the cut depicted by a dash line on H_P in Fig. 1 (b). With the *total item suppression* technique, to eliminate privacy threats in the generalized data, some constituent items of *Cut* are totally removed from all transactions. The set of items to be removed is called a *suppression scenario of Cut*.

In other words, an anonymization is defined by *Cut* and *SS*, a generalization cut and the suppression scenario associated with the cut. The anonymized data D'' is derived in two steps: first the original items in D are generalized to their taxonomic ancestors in *Cut* to get $D' = g(D, Cut)$, and then items in *SS* are suppressed from D' to eliminate threats, which results in $D'' = s(D', SS)$.

Running Example: Consider the transaction database D in the 2^{nd} column in Fig. 1 (a) and the taxonomy H_P in Fig. 1 (b). Suppose that we enforce 2^{∞}-*anonymity* . By generalizing D to the cut {P, f, g, M, e, i}, only one privacy threat, {e, i}, exists in the generalized data $D' = g(D, \{P, f, g, M, e, i\})$. If we suppress item i from D', we get the anonymized data $D'' = s(D', \{i\})$ where no privacy threat exists, as shown in the last column in Fig. 1 (a). ■

Anonymization causes information loss. Given *Cut* and *SS*, a generalization cut and its associated suppression scenario, $cost_G(Cut)$ denotes the information loss incurred by generalizing D to get $D' = g(D, Cut)$, and $cost_S(SS)$ denotes that incurred by suppressing items in *SS* from D' to get $D'' = s(D', SS)$. The total cost is $cost(Cut, SS) = cost_G(Cut) + cost_S(SS)$.

Anonymization can be measured by a variety of metrics. As most cost metrics are additive, we can write $cost_G(Cut) = \sum_{x^* \in Cut} O(x^*) \cdot IL_G(x^*)$, and $cost_S(SS) = \sum_{x^* \in SS} O(x^*) \cdot IL_S(x^*)$, where $O(x^*)$ is the total number of occurrences in D of all leaf items that are descendants of x^*, $IL_G(x^*)$ is the generalization cost *per occurrence* of x^*, and $IL_S(x^*)$ is an extra suppression cost in addition to $IL_G(x^*)$ if x^* is suppressed.

We use *LM* [6] in our discussion, and assume that D only contains leaf items on H_P. With *LM*, $IL_G(x^*) = (\#leaves(x^*) - 1)/(\#leaves(H_P) - 1)$, where $\#leaves(x^*)$ and $\#leaves(H_P)$ denotes the number of leaves in the subtree rooted at x^* and that in the taxonomy H_P respectively. If x^* is suppressed, it is deemed that all descendants of x^* are generalized to the top level of H_P, the overall information loss per occurrence of x^* is 1, so the extra suppression cost is $IL_S(x^*) = 1 - IL_G(x^*)$. For example, for D'' in the last column in Fig. 1 (a), *Cut* = {P, f, g, M, e, i}, *SS* = {i}. With LM, $cost_G(Cut) = 3.6$, and $cost_S(SS) = 2$. The total cost is $cost(Cut, SS) = 5.6$.

3 Integrating Generalization and Suppression

An anonymization is defined by (*Cut, SS*), a generalization cut with its associated suppression scenario, and can be found by two nested loops.

As the number of cuts is exponential in the number of items and so is the number of suppression scenarios for a cut, a complete enumeration for either loop is intractable. Therefore, we present a basic approach, *heuristic generalization* with *heuristic suppression*, namely HgHs.

3.1 Top-Down Greedy Search of the Lattice of Cuts

The outer loop of HgHs enumerates generalizations (cuts) by a top-down greedy search of a lattice of all possible cuts [8], where a specific cut (child) is derived from a general cut (parent) by replacing one constituent item of the parent cut by its child items on the taxonomy tree.

Starting from the top-most cut which consists of only the root (item) of the taxonomy tree, the outer loop continues with the most promising child cut of the current cut, which is achieved by evaluating the suppression scenario for each child of the current cut by running the inner loop (*detailed in the next subsection*), and computing the anonymization cost. The outer loop stops when no child cut reduces the anonymization cost.

For example, Fig. 2 (a) describes the searching process. The outer loop starts from cut_1={T} with $cost(cut_1, SS_1) = 23$ where the suppression scenario SS_1 for $cut_1 = \{\}$. The only child of cut_1 is $cut_2 = \{$P, Q, e, i$\}$. There is one privacy threat, {e, i}, in $D' = g(D, cut_2)$. The suppression scenario SS_2 for cut_2 (*computed by the inner loop described in the next subsection*) is {i}. So $cost(cut_2, SS_2) = cost_G(cut_2) + cost_S(SS_2) = 6.6 + 2 = 8.6$. The search continues to evaluate the children of cut_2. The best child is cut_4 since $cost(cut_4, SS_4) = 6.2$ while $cost(cut_3, SS_3) = 10.2$. The search stopped at cut_6 ={P, f, g, M, e, i} with SS_6 = {i} as no child of cut_6 reduces the cost. So, (cut_6, SS_6) defines the anonymized data D'' as shown in the last column in Fig. 1 (a).

3.2 Finding a Good Suppression Scenario for a Cut

The inner loop of HgHs is responsible for finding an item suppression scenario SS to eliminate privacy threats from $D' = g(D, Cut)$ where Cut is the cut currently being enumerated by the outer loop. SS is a subset of Cut, all occurrences of items in SS will be suppressed from D'.

To determine SS, the inner loop greedily searches the so called suppression scenario enumeration tree, *which is built per cut*. Each node on the suppression scenario enumeration tree is denoted by a *headlist* and a *taillist*. The items in *headlist* are to be *kept*, and the items not in *headlist* are to be suppressed. We also use the set notation to represent *headlist* and *taillist*. Thus, the suppression scenario represented by a node N is *Cut* - N.*headlist*. And its suppression cost is $\sum_{x^* \in Cut - N.headlist} O(x^*) \cdot IL_S(x^*)$. For the root node, $headlist = \{\}$ and $taillist = Cut$, i.e., all items are suppressed. The j^{th} child node C of a parent node P is derived based on the j^{th} item, i_j, in P.*taillist* such that C.*headlist* = P.$headlist \cup \{i_j\}$ and C.*taillist* = the suffix of P.*taillist* after i_j.

For example, Fig. 2 (b) is the suppression scenario enumeration for cut_3 in Fig. 2 (a). N_1 is the root with N_1.*headlist* = {} and N_1.*taillist* = cut_3 where

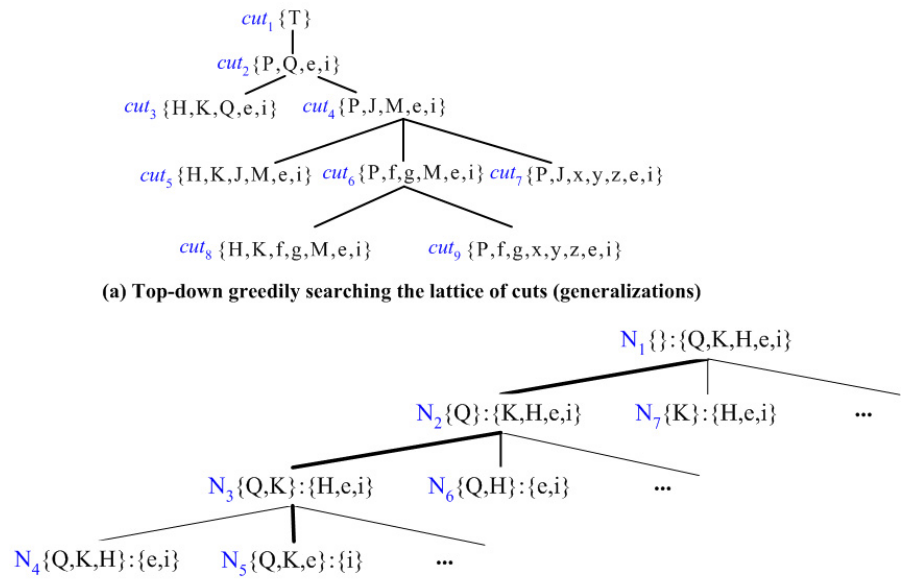


Fig. 2. Searching the cut lattice and finding the suppression scenario for each cut

items are listed in the descending order of suppression costs. N_2 is derived from N_1, by moving the first item Q from *taillist* to *headlist*, which means that all items except Q are suppressed.

Clearly, a suppression scenario represented by a node N is *valid* if and only if no threat in D' is contained by N.*headlist*. Moreover, if a threat X is contained by N.*headlist*, then X is also contained by the *headlist* of any descendant of N. Therefore, if N is invalid, all its descendants are invalid, we can stop searching the subtree rooted at N. If items in *headlist* and *taillist* are in the descending order of suppression costs, the first valid child of any node is the most promising child of the node, as it is valid and reduces the suppression cost most.

For example, Fig. 2 (b) shows how the suppression scenario SS_3 for eliminating the threats, {H,K,Q} and {e,i}, from $D' = g(D, cut_3)$ is found. The inner loop of HgHs starts with N_1 which is valid. N_2 is the first child of its parent and is valid, and so is N_3. And N_4 is the first child of N_3 but it is invalid. The inner loop stoped at N_5 with $N_5.headlist$ = {Q, K, e}. So, the final suppression scenario $SS_3 = cut_3 - N_5.headlist$ = {H, i}.

4 Addressing Efficiency and Scalability Issues

Although our basic approach HgHs presented in Section 3 enumerates a limited number of anonymizations, the work for examining each enumerated anonymization is still non-trivial. In this section, we propose the key techniques to address the efficiency and scalability issues in this regard.

4.1 Minimal Privacy Threats

The outer loop of HgHs needs to know the set of privacy threats in $D' = g(D, Cut)$ for the current *Cut*. If such a set is empty, all threats are already eliminated by generalizing D. If it is not, we have to suppress some generalized items to eliminate all privacy threats from D'. First, we claim that it suffices to generate the set of *minimal privacy threats*.

Definition 2 (Minimal privacy threats): A privacy threat X is a minimal threat if there is no privacy threat that is a subset of X. ■

Since every privacy threat contains some minimal privacy threat, if we eliminate all minimal privacy threats, we also eliminate all privacy threats. However, finding the set of minimal privacy threats on-the-fly is inefficient, since every threat occurs in multiple versions of the generalized data derived by different cuts and hence will be repeatedly generated while the number of cuts to be enumerated is still quite large.

Our approach is to generate the set of the minimal privacy threats supported by all cuts on the taxonomy H_P in an initialization step. For *Cut* being enumerated by the outer loop, we can retrieve privacy threats relevant to *Cut* from that set instead of generating $D' = g(D, Cut)$ and mining D' on-the-fly. Given a suppression scenario *SS* for *Cut*, to see if all threats are removed from $D'' = s(D', SS)$, we check if no relevant threat is contained in *Cut* - *SS*.

For the running example, there are 25 threats in the set of the minimal privacy threats, from which we can retrieve the threats, {H, K, Q} and {e, i}, relevant to cut_3 in Fig. 2 (a), for searching suppression scenarios in Fig. 2 (b).

4.2 A Multi-round Approach

The set of the minimal privacy threats supported by all cuts on the taxonomy H_P could be huge when H_P is of a large scale and the maximum size m of privacy threats is large, which makes HgHs not scalable. We propose a multi-round approach, mHgHs, to address the scalability issue.

To find a solution, mHgHs runs HgHs in m rounds. The 1^{st} round finds (Cut^1_{best}, SS^1_{best}) on the original taxonomy H_P, which defines an anonymization observing k^1-*anonymity*. The *i-th* round finds (Cut^i_{best},SS^i_{best}), which defines an anonymization observing k^i-*anonymity*, on the reduced taxonomy H_P^{i-1} that is derived by removing nodes under Cut^{i-1}_{best}. Clearly, Cut^i_{best} is above Cut^{i-1}_{best}. In other words, mHgHs performs anonymization progressively. Each round works on a reduced taxonomy based on the precedent round, so the set of the minimal privacy threats supported by all cuts for each round is under control.

For the running example, mHgHs first finds Cut^1_{best} = {a, b, c, d, f, g, M, e, i} with $SS^1_{best} = \{\}$, and gets H^1_P by removing nodes under Cut^1_{best}. Then, mHgHs works on H^1_P, and so on. After five rounds, mHgHs finds Cut^5_{best} = {P, f, g, M, e, i} with SS^5_{best} = {i}, which conforms 2^5-*anonymity*.

5 Experimental Evaluation

Our major goal is to investigate if our approach preserves more data utility than others approaches, and if our algorithm is scalable and efficient. We evaluate our algorithm mHgHs by comparing it with several state-of-the-art algorithms, the local generalization algorithm LG [5], the global generalization algorithm AA [13], and the suppression algorithm MM [14]. The executables of AA and MM were provided by the authors. We implemented LG as it is not available.

The POS dataset [15] and the AOL web query log dataset [11] are used in the experiments. The taxonomy tree for the POS dataset was created by [13]. We preprocessed the AOL dataset using WordNet [3] in creating the taxonomy tree. The AOL dataset is divided into 10 subsets. We use the first subset to evaluate the basic features of all algorithms, and use all subsets to evaluate scalabilities. We measure information loss by *NCP*, a variant of *LM* [6], as it was used by AA and LG. Experiments were performed on a PC with a 3.0 GHz CPU and 3.2 GB RAM. In the experiments, the default setting is $k = 5$, $m = 7$.

5.1 Information Loss Evaluation

Fig. 3(a)-(b) show the information loss on the POS dataset. Fig. 3(c)-(d) show that on the AOL dataset. Among all the 4 algorithms, the information loss by MM is the highest for all cases with $m \geq 2$, which is between 7.5% and 70% on POS and between 46% and 96% on AOL. This is consistent with the finding in [14] that MM is not good for sparse datasets as the POS dataset is quite sparse while the AOL dataset is even sparser.

The information loss by AA is the second highest in general. In some cases on POS (with a small m and a large k), LG incurs a little bit more. The information loss by AA on AOL is strikingly high, all around 39% even for m=1. Because the AOL dataset is extremely sparse, a lot of very infrequent items spread over the taxonomy. They have to be generalized to high levels, which brings their siblings to the same ancestors by AA. This situation is similar to our motivation example where as items e and i are infrequent, their siblings, P and Q, although quite frequent, have to be generalized to the top level together with e and i by AA. In such cases, suppressing a few outlier items will reduce information loss. This motivates our approach.

The information loss by LG is the third highest. As we pointed out in Section 1, LG exerts excessive distortion as it enforces the *transactional k-anonymity* principle which is too strong to be necessary. LG does not make use of m. So, the curves of LG with a varying m are all horizontal lines.

Our algorithm, mHgHs, incurs the least information loss which is the advantage of integrating suppression and generalization. The data utility gain of mHgHs over LG is moderate on POS, but it is significant on AOL. All the information loss with mHgHs is under 10% on AOL, while the worst case with LG is 27%, e.g., that by mHgHs is around 7.9% with $k = 5$ and $m \geq 5$ on AOL as in Fig. 3(d), while that with LG is 12%. The gap increases when enforcing a more restrictive privacy requirement, e.g., that by mHgHs with $k = 50$ and $m \geq 5$ on

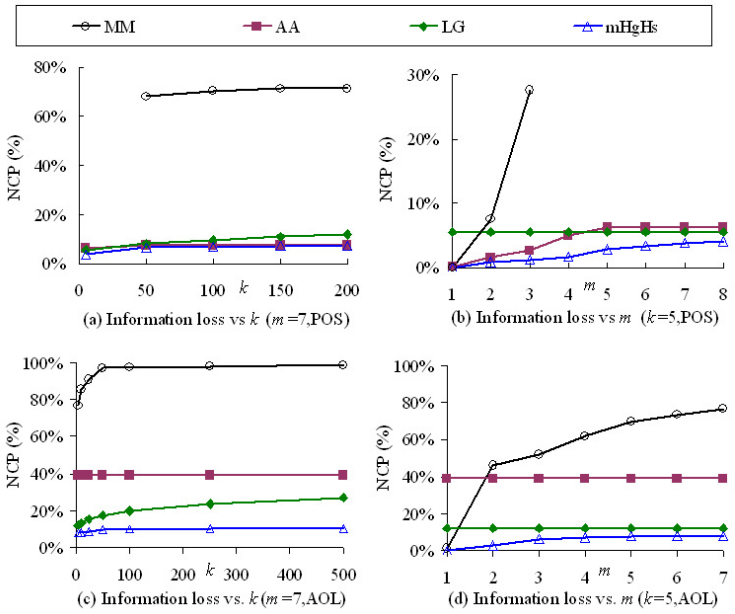


Fig. 3. Information loss of algorithms MM, AA, LG, and mHgHs on POS and AOL

AOL is 9% as in Fig. 3(c), while that by LG is 17%. Notice that the information loss reported for mHgHs is also computed on the *original* taxonomy.

5.2 Efficiency and Scalability Evaluation

We evaluated the efficiencies of all the algorithms. LG is the most efficient because it employs a divide-and-conquer approach. But, it comes with an expense, i.e., the anonymized data does not observe the domain exclusiveness as discussed in Section 1. Our algorithm, mHgHs, is the second most efficient. Although mHgHs is less efficient than LG, the significant gain in data utility by mHgHs over LG is worth the longer runtime. AA and MM are less efficient. This is because the breadth-first search approaches they employ are not efficient in dealing with privacy threats with a large size. One exception is that AA is as efficient as LG on AOL because the search space is greatly pruned by AA, which unfortunately results in the second highest information loss on AOL.

We also evaluated the scalabilities of algorithms on all the 10 subsets of the AOL query logs. The result showed that our algorithm mHgHs is quite scalable.

6 Conclusion

This paper proposed to integrate generalization and suppression to enhance data utility in anonymizing transaction data. We presented a multi-round, top-down greedy search algorithm to address the performance issues. Extensive experiments show that our approach outperforms the state-of-the-art approaches.

Acknowledgements. The research is supported in part by the Natural Sciences and Engineering Research Council of Canada, in part by the Science and Technology Development Plan of Zhejiang Province, China (2006C21034), and in part by the Natural Science Foundation of Zhejiang Province, China (Y105700).

We thank Harshwardhan Agarwal for his help in experimental evaluation.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: SIGMOD 1993 (1993)
2. Barbaro, M., Zeller, T.: A Face Is Exposed for AOL Searcher No. 4417749. New York Times (August 9, 2006)
3. Fellbaum, C.: WordNet, An Electronic Lexical Database. MIT Press, Cambridge (1998)
4. Ghinita, G., Tao, Y., Kalnis, P.: On the Anonymization of Sparse High-Dimensional Data. In: ICDE 2008 (2008)
5. He, Y., Naughton, J.: Anonymization of Set-valued Data via Top-down Local Generalization. In: VLDB 2009 (2009)
6. Iyengar, V.: Transforming Data to Satisfy Privacy Constraints. In: KDD 2002 (2002)
7. LeFevre, K., DeWitt, D., Ramakrishnan, R.: Mondrian Multidimensional k-Anonymity. In: ICDE 2006 (2006)
8. Liu, J., Wang, K.: On Optimal Anonymization for l^+-Diversity. In: ICDE 2010 (2010)
9. Machanavajjhala, A., Gehrke, J., Kifer, D., Venkitasubramaniam, M.: l-Diversity: Privacy beyond k-Anonymity. In: ICDE 2006 (2006)
10. Narayanan, A., Shmatikov, V.: How to Break Anonymity of the Netflix Prize Dataset. ArXiv Computer Science e-prints (October 2006)
11. Pass, G., Chowdhury, A., Torgeson, C.: A Picture of Search. In: The 1st Intl. Conf. on Scalable Information Systems, Hong Kong (June 2006)
12. Samarati, P., Sweeney, L.: Generalizing Data to Provide Anonymity When Disclosing Information. In: PODS 1998 (1998)
13. Terrovitis, M., Mamoulis, N., Kalnis, P.: Privacy Preserving Anonymization of Set-valued Data. In: VLDB 2008 (2008)
14. Xu, Y., Wang, K., Fu, A., Yu, P.S.: Anonymizing Transaction Databases for Publication. In: KDD 2008 (2008)
15. Zheng, Z., Kohavi, R., Mason, L.: Real World Performance of Association Rule Algorithms. In: KDD 2001 (2001)