Perspectives in Science (2016) 7, 74—80

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/pisc

# High utility-itemset mining and privacy-preserving utility mining☆

Jerry Chun-Wei Lin[a,*], Wensheng Gan[a], Philippe Fournier-Viger[b], Lu Yang[a], Qiankun Liu[a], Jaroslav Frnda[c], Lukas Sevcik[c], Miroslav Voznak[c]

[a] *School of Computer Science and Technology Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China*
[b] *School of Natural Sciences and Humanities, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China*
[c] *Department of Telecommunications, Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, 17. listopadu 15, 708 00 Ostrava-Poruba, Czech Republic*

Received 27 October 2015; received in revised form 27 October 2015; accepted 11 November 2015
Available online 10 December 2015

**KEYWORDS**
Data mining;
High-utility itemset;
Privacy preserving;
PSO algorithm;
GA algorithm;
Evolutionary algorithms

**Summary** In recent decades, high-utility itemset mining (HUIM) has emerging a critical research topic since the quantity and profit factors are both concerned to mine the high-utility itemsets (HUIs). Generally, data mining is commonly used to discover interesting and useful knowledge from massive data. It may, however, lead to privacy threats if private or secure information (e.g., HUIs) are published in the public place or misused. In this paper, we focus on the issues of HUIM and privacy-preserving utility mining (PPUM), and present two evolutionary algorithms to respectively mine HUIs and hide the sensitive high-utility itemsets in PPUM. Extensive experiments showed that the two proposed models for the applications of HUIM and PPUM can not only generate the high quality profitable itemsets according to the user-specified minimum utility threshold, but also enable the capability of privacy preserving for private or secure information (e.g., HUIs) in real-word applications.


## Introduction

With the rapid growth of information techniques and various applications, the Knowledge Discovery in Database (KDD) which is also called data mining, has become a powerful technique and commonly be used to discover interesting and useful knowledge from massive data. The discovered

☆ This article is part of a special issue entitled ''Proceedings of the 1st Czech-China Scientific Conference 2015''.

* Corresponding author.
*E-mail address:* jerrylin@ieee.org (J.C.-W. Lin).

http://dx.doi.org/10.1016/j.pisc.2015.11.013

knowledge from data mining can be generally classified as frequent patterns (FIs) or association rules (ARs) (Agrawal et al., 1993a,b; Han et al., 2004), high-utility patterns (HUPs) (Ahmed et al., 2009; Chan et al., 2003; Liu et al., 2005; Yao et al., 2004), sequential patterns (SPs) (Agrawal and Srikant, 1995; Pei et al., 2004), clustering (Berkhin, 2006) and classification (Kotsiantis, 2007), among others. Among them, association-rule mining (ARM) (Agrawal et al., 1993a,b; Han et al., 2004) is the fundamental knowledge which can be commonly used to analyze the purchase data of customers. However, ARM only considers whether or not the item or itemset is present in a transaction. The other factors in real-life applications, such as weight, profit, quantity, risk or other measures, are not considered in ARM. Thus, high-utility itemset mining (HUIM) (Ahmed et al., 2009; Chan et al., 2003; Liu et al., 2005; Yao et al., 2004) has emerging a critical issue in recent years since it can be used to reveal profitable itemsets (high-utility itemsets) by considering both purchase quantity and distinct profit factors.

Due to quick proliferation of massive data from government, corporations and organizations, the discovered knowledge may, however, implicitly contain confidential, private or secure information (i.e., personal identification numbers, address information, social security numbers, or credit card numbers), which leads to privacy threats if they are misused (Agrawal and Srikant, 2000; Verykios et al., 2004). Generally, collaboration among industries can work together to share information for achieving higher benefits and profits in business. However, the shared information can be extracted and analyzed by the other collaborators or competitors, thus decreasing its own benefits and causing the security threats. Privacy-preserving data mining (PPDM) was thus proposed to address the above limitations by perturbing the original database and producing a sanitized one (Amiri, 2007). Many algorithms have been extensively proposed to hide the private or secure information (i.e., FIs or ARs) from the different type databases (Dunning and Kresman, 2013; Han and Ng, 2007). As the similar consideration of PPDM, privacy preserving for high-utility itemset mining (PPUM) has also become an important topic in recent years. Fewer studies have addressed the issue of PPUM and most of them are processed to reduce the quality or delete transactions for hiding sensitive high-utility itemsets (SHUIs). Several related algorithms for PPUM have been extensively studied, such as the Hiding High-Utility Itemset First (HHUIF) algorithm and Maximum Sensitive Itemsets Conflict First (MSICF) algorithm to hide the SHUIs (Yeh and Hsu, 2010), the GA-based algorithm for hiding SHUIs through transaction insertion (Lin et al., 2014a), and the Fast Perturbation algorithm Using a Tree structure and Tables (FPUTT) algorithm (Kannimuthu and Premalatha, 2015) to speed the sanitization process with an aided tree structure and the associated index table. The above approaches, however, are insufficient since whether PPDM or PPUM belongs to the NP-hard problem. It is necessary to hide the sensitive information but discover the required information in decision making.

In this paper, we address the research issues by proposing efficient algorithms applied in HUIM and PPUM. We firstly propose a PSO-based algorithm for HUIM, and secondly propose a GA-based approach to evaluate the effectiveness and efficiency of PPUM. Key contributions of this paper are present below. (1) We present two evolutionary algorithms, the PSO-based algorithm for efficiently mining HUIs and the GA-based privacy preserving algorithm in PPUM. (2) Not only the mining performance for HUIM, but also a trade-off between mining performance and its privacy preserving for the SHUIs can be ensured. (3) Extensive experiments conducted on several real-life and synthetic datasets showed that the two proposed models for HUIM and the applications of PPUM have better results than the previous works whether in HUIM or PPUM.

## Background

### High-utility itemset mining

The concept of high-utility itemset mining was first proposed by Chan et al. (2003), and the mathematical mode was formed by Yao et al. (2004). The problem of high-utility itemset mining (HUIM) was defined to find the rare frequencies itemsets but with high profits (Yao et al., 2004). Since the downward closure property is no longer kept for HUIM, a Two-Phase model (Liu et al., 2005) was presented to keep the transaction-weighted utilization downward closure (TWDC) property for discovering HUIs. Several tree-based algorithms were also proposed, such as the HUP-tree-based IHUP algorithm for mining HUIs in incremental databases (Ahmed et al., 2009), the HUP-growth algorithm (Lin et al., 2011) to find HUIs without candidate generation, and the efficient UP-tree-based two mining algorithms, UP-growth (Tseng et al., 2010) and UP-growth+ (Tseng et al., 2013). Liu et al. then proposed the HUI-Miner algorithm (Liu and Qu, 2012) to build utility-list structures and to develop a set-enumeration tree to directly extract HUIs without either candidate generation or an additional database rescan. The improved FHM algorithm was further designed by enhancing the HUI-Miner for analyzing the co-occurrences among 2-itemsets (Fournier-Viger et al., 2014).

Besides, many interesting other issues on HUIM have also been extensively studied, such as up-to-date HUIs mining which aims at discovering recent HUIs which may be more useful and interesting (Lin et al., 2015d); HUIs mining in dynamic environment (e.g., data insertion (Lin et al., 2014b), data deletion (Lin et al., 2015a), and data modification (Lin et al., 2015c)), HUIs mining in stream data environment (Li et al., 2008); on-shelf HUIs mining (Lan et al., 2011); top-*k* HUIs mining (Wu et al., 2012); HUIs mining with multiple minimum utility thresholds (Lin et al., 2015b); HUIs mining with or without negative unit profit value (Fournier-Viger, 2014).

### Privacy preserving for high-utility itemsets mining

Although data mining various techniques can be used to find the implicit information, the confidential or secure information is, however, required to be hidden before it is published in the public place or shared among the collaborators. Privacy-preserving data mining (PPDM) has thus arisen as an important topic in recent years (Agrawal and Srikant, 2000; Han and Ng, 2007; Verykios et al., 2004). A novel reconstruction procedure was presented to accurately estimate the distribution of original data values, thus building the classifiers to compare the accuracy between the sanitized data and the original ones (Agrawal and Srikant, 2000). Several algorithms for hiding the sensitive information of PPDM are still designed in progress, such as (Dunning and Kresman, 2013; Han and Ng, 2007). There are three side effects commonly used in PPDM, such as the side effects of artificial cost (*AC*), missing cost (*MC*), and hiding failure (Dunning and Kresman, 2013; Han and Ng, 2007). Since more useful information in high-utility itemsets than in that of the frequent itemsets or sequential patterns, privacy preserving for high-utility itemsets mining (PPUM) is more realistic and critical than PPDM (Lin et al., 2014a; Kannimuthu and Premalatha, 2015; Yeh and Hsu, 2010) The main task of PPUM is to hide the sensitive high-utility itemsets (SHUIs). Yeh et al. first designed the Hiding High Utility Itemset First (HHUIF) algorithm and Maximum Sensitive Itemsets Conflict First (MSICF) and Maximum Sensitive Itemsets Conflict First (MSICF) algorithms to hide SHUIs in PPUM (Yeh and Hsu, 2010). In the past, Lin et al. first developed a GA-based method to hide the user-specified SHUIs by inserting the dummy transactions to the original databases (Lin et al., 2014a). Yun et al. then developed a tree-based algorithm called Fast Perturbation algorithm Using a Tree structure and Tables (FPUTT) for hiding SHUIs in PPUM (S Kannimuthu and Premalatha, 2015).

## Proposed evolutionary algorithm for HUIM

In this section, we propose an efficient PSO-based high-utility itemset mining model, and compared with the state-of-the-art $\mathrm{HUPE_{umu}}$-GRAM algorithm (Kannimuthu and Premalatha, 2014) to evaluate the efficiency of the developed algorithm. Details are described below.

### Binary PSO for HUIM

There are four processes to mine HUIs based on the binary PSO model (Kennedy and Eberhart, 1997), which are pre-processing, particle encoding, fitness evaluation and the updating process. In the pre-processing process, it first discovered the high transaction-weighted utilization 1-itemsets (1-HTWUIs) based on the TWU model (Liu et al., 2005). To avoid the invalid generations of the particles producing in the updating process, it also compresses the valid combinations of the itemsets in the database as a maximal-pattern tree (MP-tree). It determines whether the combined itemsets can generate the valid combination or not in the databases based on the OR and NOR operators. A simple MP-tree is shown in Fig. 1.

The items of 1-HTWUIs are sorted in alphabetic-ascending order corresponding to the *j*th position of a particle in the second particle encoding process. Each particle is encoded as the set of binary variables corresponding to the sorted order of 1-HTWUIs. Then the particles calculate their own fitness to find the *pbest* and *gbest* particles in the fitness evaluation. For the last updating process, the particles are correspondingly updated by velocities, *pbest*, *gbest*, and the sigmoid function. The MP-tree structure is used to generate valid combinations of the particles, which can greatly reduce the computations of multiple database scans. If the fitness value of the particle is no less than the minimum utility value, the itemset corresponding

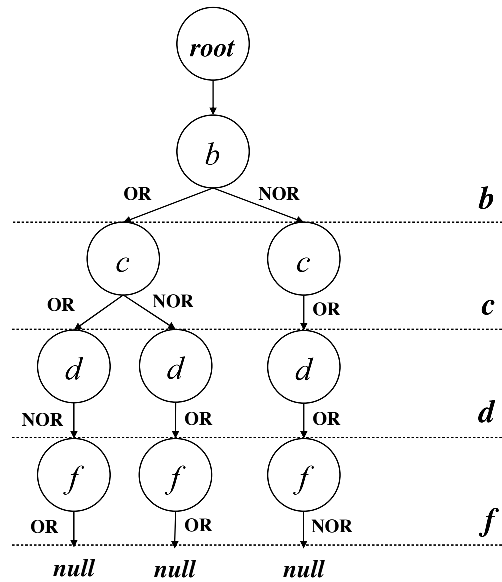


**Figure 1** The developed MP-tree structure.

to the particle will be concerned as a HUI and put into the set of HUIs. The fitness function value of particle is the same as the utility value of corresponding itemset as: $fitness(p_i) = u(X)$, in which $X$ represent the itemset corresponding to the particle $p_i$. The iteration repeated until reaching the termination criteria and then the set of HUIs will be discovered. Details of the proposed PSO-based HUIs mining algorithm are shown in Algorithm 1.

**Algorithm 1** *(Proposed algorithm for HUIM).*

**Input**: *D*, a quantitative database; *ptable*, a profit table; $\delta$, the minimum utility threshold; *M*, the number of particles of each iteration.
**Output**: *HUIs*, a set of high-utility itemsets

```
find 1-HTWUIs;
  set m := |1-HTWUIs|;
initialize p(t) := either 1 or 0;
initialize v(t) := rand( );
  initialize pbest(t) and gbest(t);
while termination criteria is not reached do
    update the v(t+1) of M particles;
    update the p(t+1) of M particles;
    for i ← 1, M particles do
if fitness(p_i(t+1)) ≥ δ then
HUIs ← GItems(p_i(t+1)) ∪ HUIs;
      find pbest(t+1) of each M particle;
      find gbest(t+1) among M;
set t ← t+1;
  return HUIs;
```

## Experimental results of binary PSO for HUIM

The algorithms in the experiments were implemented in C++ language, performing on a PC with an Intel Core2 i3-4160 CPU and 4GB of RAM, running the 64-bit Microsoft Windows 7 operating system. Two real-world datasets called chess and mushroom (http://fimi.ua.ac.be/data/ (2012)), are used in the experiments.

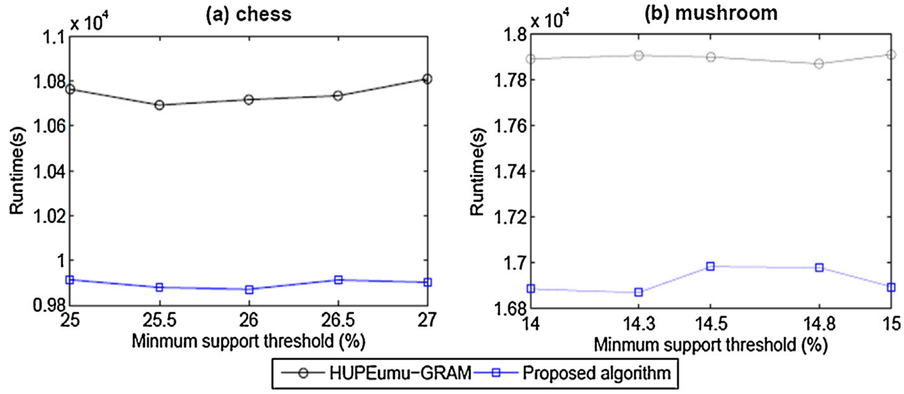


**Figure 2** Runtime of the compared algorithms.

The proposed algorithm was compared with the state-of-the-art evolutionary algorithm $\text{HUPE}_{\text{umu}}$-GRAM (Kannimuthu and Premalatha, 2014) in terms of runtime and the number of HUIs. Note that the performed algorithms were all performed for 10,000 iterations and the population size is set as 20. The experiments were executed five times and the results were the average values of them. The parameters of the proposed algorithm are set as: $c_1 = c_2(=2)$ and $w(=0.9)$.

The results of the conducted experiments of runtime in two datasets are shown in Fig. 2. It can be seen that the proposed algorithm has better performance than the $\text{HUPE}_{\text{umu}}$-GRAM algorithm in performance of runtime. The reason is that the proposed algorithm only generates the valid combinations of itemsets existing in the database, which can greatly avoid the computational problem in the evolution process. The lower the minimum utility threshold is set, the more 1-HTWUIs are discovered and the more combinations to find the promising HUIs in the evolution process.

## Proposed evolutionary algorithm for PPUM

As it was mentioned above, PPUM is a critical task to secure the sensitive high-utility itemsets. For the purpose of PPUM in various applications, it is necessary to develop a novel PPUM approach than the conventional methods. An efficient GA-based algorithm for PPUM is presented below.

### Proposed PPUMGAT+ algorithm

The pseudo-code of the designed PPUMGAT+ algorithm is described in Algorithm 2.

**Algorithm 2** *(Proposed PPUMGAT+ algorithm).*

**Input**: $D$, a quantitative database with its high-utility sensitive itemset ($HS$); *ptable*, a profit table; $S_u$, the upper (minimum) utility threshold; *MDU*, the maximal deleted utility; $M$, the number of chromosomes in a population; $N$, the number of iteration in the evolution process.
**Output**: $DB'$, a sanitized database.
Calculate the item utility value $u(i_j)$, the transaction utility value $tu(T_i)$, and the total utility value *TUD* of the whole dataset.

1 set *Cand_trans* = null;
2 **for** each $T_q \in D$ **do**
3 **for** each $si \in HS$ **do**
4 **if** $si \in T_q$ and $tu(T_q) \leq MDU$ **then**
5 $Cand\_trans \leftarrow Cand\_trans \cup T_q$;
6 sort the transactions in *Cand_trans* in ascending order according to their *tu*;
7 set $m = 0$, $sum = 0$;
8 **for** each $T_q$ in *Cand_trans* **do**
9 $sum{+}{=}tu(T_q)$;
10 **if** $sum \leq MDU$ **then**
11 $m = m+1$;
12 set the size of a chromosome as $m$;
13 generate $M$ chromosomes randomly from *Cand_trans*;
14 calculate the lower utility threshold $S_l$ according to $S_u$ and *MDU* and find *HUIs* and *PUIs*;
15 **while** termination criteria is not reached **do**
16 **for** each chromosome $c_i$ among $M$ chromosomes in a population **do**
17 perform *crossover* operation;
18 perform *mutation* operation;
19 evaluate $fitness(c_i)$;
20 select top $M/2$ chromosomes in a population;
21 generate $M/2$ chromosomes randomly from *Cand_trans*;
22 obtain the optimal chromosome $c_i$ with minimal fitness value from $M$;
23 delete $T_q$ of *ci* from *DB* as *DB*';
24 return *DB*';

Firstly, the utility of each item, the utility of each transaction, and the utility of the whole dataset are calculated by original dataset and the corresponding utility table. After that, each transaction is then determined and projected if it contains any of sensitive high-utility itemsets and its transaction utility *tu* is smaller than the given maximal deleted utility *MDU* as the *Cand_trans* (Lines 1—5). Then sorting the transactions of *Cand_trans* in ascending order by their *tu* (Line 6). Summing up the sorted transactions by their *tu*, and terminating the number of *m* transactions where the summed value is no longer small than *MDU* and the number of *m* is set as the size of a chromosome (Lines 7—12). After that, generating a population with *M* individuals randomly from *Cand_trans* and calculating the lower support threshold $S_l$

of pre-large concept according to the upper support threshold $S_u$ and *MDU* (Lines 13–14). After those operations, the crossover and mutation operations are performed to update the chromosomes (Lines 17–18). The fitness of each chromosome is then evaluated by the designed fitness function (Line 19). The half of the current chromosomes with lowest fitness values is selected as the chromosomes and another half chromosomes are generated randomly for next generation (Lines 20 and 21). The iteration procedure is processed repeatedly until the termination criterion is achieved (Lines 15–21). After that, the optimal chromosome with the lowest fitness function is obtained and the transaction *IDs* in this chromosome is selected as the victim transactions for deletion, thus hiding the sensitive high-utility itemsets (Lines 22 and 23). Finally, the database is sanitized into a non-sensitive database (Line 24).

## Pre-large concept

To speed up the evaluating process and reduce the cost of time and space in the evolution process, the pre-large concept (Hong et al., 2001) is adopted in the designed algorithm to avoid the multiple database scans each time. It uses the upper ($S_u$) and the lower ($S_l$) support thresholds to keep the unpromising itemsets which may have highly probability to be large itemsets as the buffer to avoid the multiple database scans for transaction deletion. This strategy is simple but can be used efficiently to update the discovered information. For the purpose of PPUM in this paper, the sensitive high-utility itemsets are required to be hidden through transaction deletion in the sanitization process. The pre-large concept for transaction deletion is shown below.

**Definition 1.** A safety deleted utility bound ($f$) of pre-large concept indicates that an itemset cannot be large after some transactions are deleted from the original database without database rescan as:

$$f = \left( \frac{(S_u - S_l) \times |D|}{S_u} \right) \tag{1}$$

in which $|D|$ is the size of the original database, $S_u$ is the upper support threshold and $S_l$ is the lower support threshold which both are defined by users' preference.

In the designed PPUMGAT+ algorithm, the maximal deleted utility (*MDU*) which was firstly given by users can be considered as the safety bound in the pre-large concept and total whole database utility (*TUD*) can be considered as the database size in the original database. Based on the pre-large concept, we can calculate the $S_l$ value to keep the set of unpromising pre-large utility itemsets (*PUIs*).

**Definition 2.** Let $S_u$ be the upper support threshold which can be defined by users' preference, and *MDU* is the maximal deleted utility obtained from the sensitive high-utility itemsets, and $TU^D$ is the total utility in the original database. The $S_l$ in the designed approach is modified from the pre-large concept and defined as:

$$S_l = S_u \times \left( 1 - \frac{MDU}{TU^D} \right) \tag{2}$$

After delete the transactions of one chromosome, the deleted utility of each HUI and PUI can be obtained, and the real utility of each HUI and PUI can be calculated. The missing cost (*MC*) only appears in HUIs and the artificial cost (*AC*) only occurs in PUIs. Thus, it is easy to determine whether each itemset of HUIs and PUIs is a HUI.

## Experimental results

Substantial experiments were conducted on real-life chess dataset (http://fimi.ua.ac.be/data/ (2012)) and synthetic T10I4D100K dataset (which was generated by IBM Quest Synthetic Data Generation Code, http://www.Almaden.ibm.com/cs/quest/syndata.html (1994)) to verify the comprehensive performance of the proposed algorithm. The implemented algorithm adopts the pre-large concept and an improved strategy to reduce the rescanning of origin database in the evolution process, and the comparison between our technique and the naive GA-based algorithm (Holland, 1992) also is given to present the advantage of the pre-large concept in the proposed algorithm in terms of running time. Note that the percentage of sensitive itemsets is denoted as *sen_per*. Runtime of the compared approaches on two datasets under different minimum utility thresholds with a fixed *sen_per* of HUIs is shown as Fig. 3.

From Fig. 3, it can be observed that the proposed algorithm which adopts the pre-large concept outperforms the

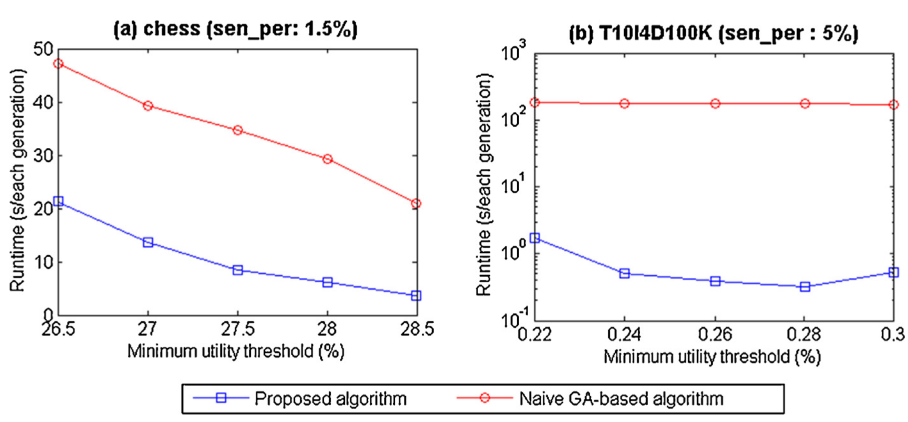


**Figure 3** Runtime w.r.t. various minimum utility thresholds.

naive GA-based algorithm w.r.t. various minimum support thresholds in all datasets. From Fig. 3(a), it can also be obviously seen that the proposed algorithm is twice or three times faster than the naive GA-based algorithm. For the sparse T10I4D100K dataset shown in Fig. 3(b), the designed approach is up to two orders of magnitude faster than the compared technique. The reason is that the pre-large concept is used in the proposed algorithm to reduce the rescanning of origin database, the designed approach thus only scans the origin database once in the whole evolution process. Besides, at the situation of a fixed sensitive percentage, the number of HUIs is decreased when the minimum support threshold is increased, less computations are required due to less number of SHUIs should be hidden.

## Conclusion

The first part of this work is to present a novel evolutionary algorithm for HUIM. Based on the designed MP-tree, the PSO-based algorithm can consume less runtime than the previous $HUPE_{umu}$-GRAM algorithm. The second part of this work is to present a GA-based PPUM algorithm. The pre-large concept was also extended to reduce the computations. The designed approach was up to two orders of magnitude faster than the compared naive GA-based technique.

## Conflict of interest

The authors declare that there is no conflict of interest.

## Acknowledgements

This publication has been created within the project Support of VŠB-TUO activities with China with financial support from the Moravian-Silesian Region and partially was supported by the grant SGS reg. no. SP2015/82 conducted at VSB-Technical University of Ostrava, Czech Republic, and was partially supported by the Shenzhen Peacock Project, China, under grant KQC201109020055A, by the Tencent Project under grant CCF-TencentRAGR20140114, by the Natural Scientific Research Innovation Foundation in Harbin Institute of Technology under grant HIT.NSRIF.2014100, by the Shenzhen Strategic Emerging Industries Program under grant ZDSY20120613125016389, and by National Natural Science Foundation of China (NSFC) under grant No. 61503092.

## References

Agrawal, R., Imielinski, T., Swami, A., 1993a. Database mining: a performance perspective. IEEE Trans. Knowl. Data Eng. 5 (6), 914—925, http://dx.doi.org/10.1109/69.250074.

Agrawal, R., Imielinski, T., Swami, A., 1993b. Mining association rules between sets of items in large database. In: The ACM SIGMOD International Conference on Management of Data, pp. 207—216.

Agrawal, R., Srikant, R., 1995. Mining sequential patterns. Paper presented at the International Conference on Data Engineering.

Agrawal, R., Srikant, R., 2000. Privacy-preserving data mining. ACM SIGMOD Rec. 29 (2), 439—450, http://dx.doi.org/10.1145/335191.335438.

Ahmed, C.F., Tanbeer, S.K., Jeong, B.S., Le, Y.K., 2009. Efficient tree structures for high utility pattern mining in incremental databases. IEEE Trans. Knowl. Data Eng. 21 (12), 1708—1721.

Amiri, A., 2007. Dare to share: protecting sensitive knowledge with data sanitization. Decis. Support Syst. 43 (1), 181—191, http://dx.doi.org/10.1016/j.dss.2006.08.007.

Berkhin, P., 2006. A survey of clustering data mining techniques. Paper presented at the Grouping Multidimensional Data, http://dx.doi.org/10.1007/3-540-28349-8_2.

Chan, R., Yang, Q., Shen, Y.D., 2003. Mining High Utility Itemsets. In: IEEE International Conference on Data Mining, pp. 19—26.

Dunning, L.A., Kresman, R., 2013. Privacy preserving data sharing with anonymous ID assignment. IEEE Trans. Inf. Forensics Secur. 8 (2), 402—413, http://dx.doi.org/10.1109/TIFS.2012.2235831.

Fournier-Viger, P., 2014. FHN: efficient mining of high-utility itemsets with negative unit profits. Adv. Data Min. Appl., 16—29.

Fournier-Viger, P., Wu, C.W., Zida, S., Tseng, V.S., 2014. FHM: faster high-utility itemset mining using estimated utility co-occurrence pruning. Found. Intell. Syst. 8502, 83—92.

Han, J., Pei, J., Yin, Y., Mao, R., 2004. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov. 8 (1), 53—87.

Han, S., Ng, W.K., 2007. Privacy-preserving genetic algorithms for rule discovery. Paper presented at the International Conference on Data Warehousing and Knowledge Discovery, Regensburg, Germany.

Holland, J.H., 1992. Adaptation in Natural and Artificial Systems. MIT Press.

Hong, T.P., Wang, C.Y., Tao, Y.H., 2001. A new incremental data mining algorithm using pre-large itemsets. Intell. Data Anal. 5 (2), 111—129.

Kannimuthu, S., Premalatha, K., 2015. A fast perturbation algorithm using tree structure for privacy preserving utility mining. Expert Syst. Appl. 42 (3), 1149—1165, http://dx.doi.org/10.1016/j.eswa.2014.08.037.

Kennedy, J., Eberhart, R.C., 1997. A discrete binary version of the particle swarm algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, pp. 4104—4108.

Kannimuthu, S., Premalatha, K., 2014. Discovery of high utility itemsets using genetic algorithm with ranked mutation. Appl. Artif. Intell. 28 (4), 337—359, http://dx.doi.org/10.1080/08839514.2014.891839.

Kotsiantis, S.B., 2007. Supervised machine learning: a review of classification techniques. Paper presented at the The Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies.

Lan, G.C., Hong, T.P., Tseng, V.S., 2011. Discovery of high utility itemsets from on-shelf time periods of products. Expert Syst. Appl. 38 (5), 5851—5857, http://dx.doi.org/10.1016/j.eswa.2010.11.040.

Li, H.F., Huang, H.Y., Chen, Y.C., Liu, Y.J., Lee, S.Y., 2008. Fast and Memory Efficient Mining of High Utility Itemsets in Data Streams., pp. 881—886, http://dx.doi.org/10.1109/icdm.2008.107.

Lin, C.W., Hong, T.P., Lu, W.H., 2011. An effective tree structure for mining high utility itemsets. Expert Syst. Appl. 38 (6), 7419—7424, http://dx.doi.org/10.1016/j.eswa.2010.12.082.

Lin, C.W., Hong, T.P., Wong, J.W., Lan, G.C., Lin, W.Y., 2014a. A GA-based approach to hide sensitive high utility itemsets. Sci. World J., 2014.

Lin, J.C.W., Gan, W., Hong, T.P., Pan, J.S., 2014b. Incrementally updating high-utility itemsets with transaction insertion. Adv. Data Min. Appl., 44—56.

Lin, C.W., Hong, T.P., Lan, G.C., Wong, J.W., Lin, W.Y., 2015a. Efficient updating of discovered high-utility itemsets for transaction deletion in dynamic databases. Adv. Eng. Inform. 29 (1), 16—27, http://dx.doi.org/10.1016/j.aei.2014.08.003.

Lin, J.C.W., Gan, W., Fournier-Viger, P., Hong, T.P., 2015b. Mining high-utility itemsets with multiple minimum utility thresholds. In: ACM International Conference on Computer Science & Software Engineering, pp. 9—17, http://dx.doi.org/10.1145/2790798.2790807.

Lin, J.C.W., Gan, W., Hong, T.P., 2015c. A fast updated algorithm to maintain the discovered high-utility itemsets for transaction modification. Adv. Eng. Inform. 29 (3), 562—574, http://dx.doi.org/10.1016/j.aei.2015.05.003.

Lin, J.C.W., Gan, W., Hong, T.P., Tseng, V.S., 2015d. Efficient algorithms for mining up-to-date high-utility patterns. Adv. Eng. Inform. 29 (3), 648—661, http://dx.doi.org/10.1016/j.aei.2015.06.002.

Liu, M., Qu, J., 2012. Mining high utility itemsets without candidate generation. In: ACM International Conference on Information and Knowledge Management, pp. 55—64.

Liu, Y., Liao, W.K., Choudhary, A., 2005. A two-phase algorithm for fast discovery of high utility itemsets. Paper presented at the Lecture Notes in Computer Science, http://dx.doi.org/10.1007/11430919_79.

Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., . . . Hsu, M.-C., 2004. Mining sequential patterns by pattern-growth: the PrefixSpan approach. IEEE Trans. Knowl. Data Eng. 16 (11), 1424—1440, http://dx.doi.org/10.1109/tkde.2004.77.

Tseng, V.S., Shie, B.E., Wu, C.W., Yu, P.S., 2013. Efficient algorithms for mining high utility itemsets from transactional databases. IEEE Trans. Knowl. Data Eng. 25, 1772—1786.

Tseng, V.S., Wu, C.W., Shie, B.E., Yu, P.S., 2010. UP-Growth: an efficient algorithm for high utility itemset mining. In: The 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 253—262.

Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y., 2004. State-of-the-art in privacy preserving data mining. ACM SIGMOD Rec. 33 (1), 50—57, http://dx.doi.org/10.1145/974121.974131.

Wu, C.W., Shie, B.E., Tseng, V.S., Yu, P.S., 2012. Mining top-K high utility itemsets. In: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 78—86.

Yao, H., Hamilton, H.J., Butz, C.J., 2004. A foundational approach to mining itemset utilities from databases. In: The SIAM International Conference on Data Mining, pp. 211—225.

Yeh, J.S., Hsu, P.C., 2010. HHUIF and MSICF: novel algorithms for privacy preserving utility mining. Expert Syst. Appl. 37 (7), 4779—4786, http://dx.doi.org/10.1016/j.eswa.2009.12.038.