IJICTR International Journal of Information & Communication Technology Research
ITRC
Volume 8 - Number 3- Summer 2016 (71-81)

Hiding Fuzzy Sequential Patterns by a Multi-Objective Scheme

O. Behbahani
Department of Electrical & Computer Engineering,
Faculty of Engineering, Kharazmi University,
Tehran, Iran

M.M. Pedram
Department of Electrical & Computer Engineering,
Faculty of Engineering, Kharazmi University,
Tehran, Iran

K. Badie
Info Society Department,
Iran Telecom Research Center,
Tehran, Iran

Received: February 11, 2016- Accepted: June 7, 2016

***Abstract*—Improvements in technology have led to generating and mining huge amounts of data. Sequence mining is a research area of data mining that aims to extract useful knowledge from raw databases. Although transactions with numerical information are usually seen in real-world applications like bank records, customer baskets and network traffic records, and efforts for mining sequential patterns with quantitative values are reported in the technical literature, the issue of privacy of sensitive knowledge in the form of fuzzy sequential patterns has not received much attention. This paper addresses the problem of fuzzy sequential pattern hiding and proposes a multi-objective optimization procedure with few distortions which leads to the preserving of database fidelity. The proposed algorithm can also work with fuzzy databases and more details are handled by fuzzy sets. Distortions, fidelity and time-consumption of the proposed algorithm are three criteria used to compare the performance of it with the existing algorithms.**

Keywords - sequential patterns; sensitive knowledge; quantitative value;fuzzy sequential pattern hiding.

I. INTRODUCTION

Sequence mining methods are employed for extracting useful knowledge hidden in the raw data needed for better decision-making in various applications including business, the drug industry and web-design purposes. The disadvantage is that it can endanger the privacy of people or organizations by exposing important information in a public domain. For example, [1] proposed an algorithm to extract fuzzy sequential patterns from the group members of social networks. In this paper, authors use association rules for analyzing the posting messages into quantitative values and

finding out sequential patterns among them by modifying the Apriori algorithm and finally according to the transactions of users, the algorithm can predict fuzzy sequential patterns which are suited for the thinking of human subjects. This prediction may be attributes and behavior of the leaders in social network and revealing this knowledge may endanger several persons' live. Thus, *Privacy-preserving data mining* methods protect the privacy of individuals and organizations. Sequential activities are common in real world applications, like *web usage logs, biomedical patient data, spatio-temporal geo-referenced traces* and *baskets of customer purchasing*. For example, publishing *the data of a customer purchase basket* may result in the other competitors using the knowledge obtained to outstrip the data owner.

Another issue is that in sequential databases, sequences with numerical data (prices, quantities, …) are commonly seen in real-world applications and as mentioned in [2], fuzzy knowledge representation can improve interacting between an expert system and its users. Several algorithms have been proposed to mine databases to extract fuzzy sequential patterns and fuzzy association rules [2,3,4,5,6,7] but these generalized data may be abused again by rivals. A lot of work has also been done to hide sensitive association rules [8,9,10,11,12,13], but very little research has been carried out to hide sequential patterns. This problem was first addressed in [14]. To achieve the goal of hiding sequential patterns, first the matching set is defined, which is the set of all sets with the size of sensitive pattern. The algorithm then finds occurrences of items with sensitive patterns in each sequence and sorts the database in ascending order according to matching set size, and finally removes all matchings in the top $|SDB| - \lambda$ input sequences. [15] introduces a method with few distortions and less infidelity than the other one. In this work candidate trees for each sequence are constructed containing all the solutions related to the multi-objective sequence-selection framework defined by the user. The algorithm then finds the best solution in the database which will sanitize the sequence. While the support of sensitive patterns is greater than a defined hiding threshold, this process is iterated. In [16] an improvement of [15] is proposed which leads to less memory usage and lower computational time. In this study, the enhanced algorithm prunes the candidate tree which is proposed in [15] from second level of tree based on a multi-objective scheme proposed in [15]. In [16] a comparative study was conducted to compare the improved algorithm with the original one (proposed in [15]) in computational processing time in worst case. Paper [17] proposes a two steps approach, where sensitive items will be identified in the first phase by generating FP tree. In this phase all of the transaction of dataset will be scanned and the items of the transactions will be placed as nodes of the FP tree, then by applying anti-monotone and monotone constraints, sensitive items will be identified. In the second phase, the database is fuzzified with respect to the proposed membership functions, then the original values are replaced by the classified fuzzy values and finally the new dataset is released. In [18] authors proposed 2 algorithms, i.e. HHUSP and MSPCF, which are expansion of USpan and focus on hiding high utility sequential patterns on quantitative sequence database, not transactional database [18]. The HHUSP algorithm, first mines all high utility sequential patterns. It uses U set for storing all high utility sequential patterns and Q for sorting all the q-sequences which each high utility sequential pattern in U belongs to. Then in hiding phase, for each high utility sequential pattern, it calculates the utility that needs to be decreased, then selects the item whose total utility is the maximum value and then modifies the quantity of the item. The first phase of the MSPCF algorithm is just like HHUSP but in the second phase, the goal is to find the item with the maximum conflict count among items in U, then the high utility sequential patterns that contain this item are modified [18]. Studies focused on hiding the fuzzy sensitive association rule are few, with the possible exception of [19,20]. The authors in [19] propose an algorithm to hide the critical fuzzy association rules from quantitative data. For this purpose, "they increase the support value of the LHS of the rule to be hidden". In [20], the authors propose a fuzzy association rules hiding algorithm for hiding the rules discovered in a quantitative database. "The algorithm integrates the fuzzy set concepts with an a priori mining algorithm to find useful fuzzy association rules and then hides them using a privacy-preserving technique. For hiding purpose, the algorithm decreases the support of the rule to be hidden by decreasing the support value of the item on either the Left Hand Side (L.H.S.) or the Right Hand Side (R.H.S) of the rule."

In view of the above discussion, this paper proposes an algorithm for *hiding* such numerically sensitive patterns before the data is published. The input to the proposed algorithm is a fuzzy dataset which is derived by fuzzification of the original dataset beforehand using the knowledge of field experts. Then the hiding process, which is based on the algorithm proposed in [16], is applied on the fuzzy dataset. A multi-objective scheme helps to maintain most of the information and data quality. Details about the hiding process will be presented in section3. The advantages of the proposed algorithm are:

- It can hide quantitative knowledge by applying a fuzzy concept to the dataset.
- It maintains data fidelity with less memory usage and lower computational time, and ensures few distortions by pruning the candidate tree in comparison with the no pruning case.

The organization of the paper is as follows. In section 2 the notations used in the rest of the paper is introduced. In Section 3 the proposed algorithm is presented and the fuzzy sequential pattern hiding problem is described. The numerical issues involving the computing time and memory usage of the algorithm are discussed in section 3 and a methodology for improving the computational performance of the algorithm is presented in this section. Section 4 provides test results based on two different datasets showing the efficiency of the new algorithm. Finally, the conclusion is presented in Section 5.

II. PROBLEM FORMULATION

In this section, some basic definitions of fuzzy sequence data mining and a discussion about the problem of fuzzy sequential pattern hiding are presented .

Definition1. Fuzzy Element and Fuzzy Itemset **:** "A fuzzy element [*x,a*] is related to another element and the fuzzy set function. *x* is the element and *a* is the fuzzy set" [4]. As an example, [*cheese, few*] is a fuzzy element and *few* is the fuzzy set defined with the respect to the membership function of buying cheese. "A fuzzy itemset, (*X*, *A*), is a set of fuzzy elements where *X* is the element set and *A* is a set of corresponding fuzzy sets" [4]. (*X*, *A*)=([*cheese, few*] [*Water, little*]) is a fuzzy itemset which can be presented as ((*cheese*, *water*) (*few*, *little*)).

Definition2. Fuzzy Sequence**:** A sequence $\alpha = a_1 a_2 \dots a_n$ is called a subsequence of another sequence $\beta = b_1 b_2 \dots b_m$ and β is called a super-sequence of α, denoted as $\alpha \sqsubseteq \beta$, if there exist integers $1 \le j_1 < j_2 < \cdots < j_n \le m$ such that $a_1 \subseteq b_{j1}$, $a_2 \subseteq b_{j2}$, …, $a_n \subseteq b_{jn}$. "A fuzzy sequence is an ordered list $S = < s_1 s_2 \dots s_l >$where each $s_i (1 \le i \le l)$ is a fuzzy itemset like s = (*X*, *A*)" [4]. In addition, a fuzzy sequence database *FSDB* contains a set of fuzzy sequences.

Definition3. Support of a Sequence**:** The support of a sequence α in an *FSDB* is the number of sequences in the *FSDB* that are super-sequences of α: $sup_{FSDB}(\alpha) = |\{S \in FSDB | \alpha \sqsubseteq S\}|$. A sequence α is called a sequential pattern in *FSDB* if $sup_{FSDB}(\alpha) \ge$ *min-sup*.

Given a sequence $S = s_1 \dots s_n$, and a subsequence $S' = s'_1 \dots s'_m$ a set of positions $\{i_1, i_2, \dots, i_m\}$ is called an occurrence of S' in S, if $1 \le i_1 < \cdots < i_m \le n$ and $s'_k = s_{ik}$ for each $1 \le k \le m$.

Problem definition1. Fuzzy Sequential Pattern Mining**:** Given a fuzzy sequence database and a minimum support threshold, the fuzzy sequential pattern mining problem is to find the complete set of fuzzy sequential patterns in the database.

Definition4. related occurrence set of an item in a sequence**:** Given a fuzzy sequence *S*, a fuzzy sequential pattern *SP*, and an fuzzy item [*x,a*] which is the *i*th item of *SP*, the related occurrence set of [*x,a*] from *SP* in *S* encircles the item numbers in *S* which correspond to item [*x,a*] and is denoted as $ros_{sp \to s}([x: i, a])$. As an example, consider *s* = < [*a,medieum*] [*a,little*] [*a,high*] ([*c,medium*] [*d,little*]) ([*a,high*] [*b,medium*] [*d,little*]) [*b,medium*] [*b,medium*] ([*c,little*] [*b,medium*] [*c,little*]) > and *sp* = ([*c,medium*] [*d,little*]) [*b,medium*] [*b,medium*], then the related occurrence set of the 3rd item from *SP*, i.e. *b*, in *S* is $ros_{(\text{cd})\text{bb} \to s}([b: 3, medium]) = \{7, 9, 10\}$.

Definition5. Fuzzy Sensitive Patterns**:** Fuzzy sensitive patterns are fuzzy sequential patterns which are considerd as sensitive patterns by field experts and should be hidden.

Problem definition2. Fuzzy Sequential Pattern Hiding**:** Given a fuzzy sequence database *FSDB*, a fuzzy sensitive pattern set *SPS*, and a hiding threshold λ, the goal is to change the *FSDB* in order at least to hide all the fuzzy sensitive patterns in it by reducing their support for λ.

There are some important issues in the last definition. First, the sequences of the database should be changed. The number of items to be changed, are called *distortions* which should be as few as possible, due to the fact that a distortion decreases the quality of the data. Second, the support of the sensitive patterns must reduce exactly to λ because the more the support decreases, the lower the quality of the database.

III. THE PROPOSED ALGORITHM

In this section the background of the new algorithm (*HFSP*), which is based on the previous work [16], is reviewed.

3.1.EMOSS algorithm

First it should be mentioned that *EMOSS* is the improvement of the algorithm proposed in [15] in computational burden time and memory usage, so we first review *EMOSS* to prepare mind for *HFSP* algorithm. When *EMOSS* algorithm (proposed in [16]) is introduced, we first need to know how a

sequential pattern vanishes from a sequence. To this end, all the occurrences of the sequential pattern must be sanitized from the sequence. Consider the sequence *s* = *da*(*cd*)*b*(*cd*)*bebedc* and the sequential pattern *sp* = (*cd*)*bb* and *ebe*. The sanitized *s* = *da*(*cd*)*b*(*cd*)?*e*?*edc* is obtained by two distortions, because $ros_{(cd)bb\to s}(b{:}\,4) = \{8,10\}$, $ros_{ebe\to s}(b{:}\,2) = \{10\}$

Here, the sequence should be sanitized with the least distortions. "It is useful to mention that an optimal sanitization, which is the hiding of all occurrences of sensitive patterns in a sequence, is NP-Hard" [14].

A sequence selection process that overcomes the hiding of the sensitive pattern problem, was introduced in [19], "It finds the best candidate solution for each sequence of the dataset and then by comparing all best candidates in the database, it chooses the best overall candidate for the dataset and applies it to the selected sequence, so the support of some sensitive patterns will have a one unit decrement. The iteration of this process is continued until all sensitive pattern supports descend to the exact value of λ" [16].

The process of selecting a candidate solution was fully presented in [16]. The authors considered the following factors:

- the number of sensitive patterns (*nSP*) [which have a positive effect should be maximized by a factor].
- the number of distortions (*nD*) of the candidate solution [which have a negative effect, need to be minimized by a factor] ,
- the number of non-sensitive patterns (*nNSP*) [which also have a negative effect need to be minimized by a factor].

Thus, "the problem of hiding all the sequences $s \in SPS$ in a sequential database *SDB* is defined as: find $s' \sqsubseteq s$ to be hidden and sanitize *SDB* into *SDB'* so that" [16]:

$$\begin{aligned} &max\ nSP(s'), min\ nD(s'), min\ nNSP(s') \\ &s.t. \quad s' \sqsubseteq s,\ for\ all\ s \in SPS \end{aligned} \quad (1)$$

$$sup_{SDB'}(s) = \lambda, for\ all\ s \in SPS$$

To solve the above multi-objective optimization puzzle, a weighted summation is proposed [16]:

$$F\big(nSP(s'), nD(s'), nNSP(s')\big) = \alpha * nSP(s') - \big(\gamma * nD(s') + \delta * nNSP(s')\big)\ \ \alpha, \gamma, and\ \delta \in [0,1] \quad (2)$$

where α, γ and δ are scaling factors. Finding the best solution is managed by constructing a *candidate tree* discussed in depth in [15]. The candidate tree is comprised of all possible solutions where each of its nodes is a solution to the hiding of sensitive patterns.An important point about the candidate tree is that the algorithm constructs the first level of the tree, then it merges pairs of solutions with a sensitive item in the next level. "Note that solutions which construct the i^{th} level of the candidate tree must be a combination of *i* items from *i* different sensitive patterns" [16].

The objective of [16] is to reduce the computing time and the memory usage of [15]. As mentioned before, to hide sensitive patterns, a candidate tree should be constructed by the algorithm." The height of the tree depends on the number of sensitive patterns which should be hidden" [16]. Pruning the tree reduces the computational burden which results in better processing time and memory usage with respect to the work in [15]. The objective function (2) is calculated for each solution and the result provides a criterion for ranking the solutions as candidates for sanitization.

It should be noted that the deeper solution in the tree offers more distortions, so the result of the objective function might be smaller. Thus, during the construction of the candidate tree, the subtree starting from a solution will be pruned if the objective function value for the solution is lower than the current best objective value for some theresholds.

Definition6. Measure of Pruning Or M-pruning[mine]:It is a real value as a threshold to prune the candidate tree. The current best objective function value will be compared with the objective function value of each solution, then the subtree (starting from the solution) is pruned if the difference is greater than *M-pruning*. In other words:

If (current best objective value – objective value for the solution)>M-pruning, then prune the subtree starting from the current solution.

The pruning mechanism works as follows [16]:

1. The first level of the tree is constructed and the best objective function value is saved as the current best objective value.
2. For the second level or higher, the process of tree construction continues as follows :
   a) If the difference between the current best objective value and the new solution is less than or equal to *M-pruning*, the solution will be added to the tree.

b) If the difference is bigger than *M-prunnig*, then the subtree starting at the solution will be pruned.
c) If the objective value of the solution is better than the current best objective value, the current best objective value is updated.

It is worth mentioning that comparison of time complexity of *MOSS* and *EMOSS* in worst case is discussed in [16].

3.2 Hiding Fuzzy Sequential Pattern (HFSP)

HFSP aims to hide numerically sensitive patterns before the data is published. The input to the proposed algorithm is a fuzzy dataset. Then the hiding process is applied on the fuzzy dataset. Like *EMOSS*, *HFSP* also constructs candidate tree, calculates the objective functions, computes the support of each sensitive pattern and then prunes the candidate tree. Note that in order to determine the support of fuzzy sequential patterns, "fuzzy cardinality of a fuzzy subset B should be computed by counting all elements whose membership degree is not null" [4].

The steps of *HFSP* are presented in Figure1. In the algorithm *DBSeqsForCheck*, which includes all sequences at the beginning, is an array used to determine which *FSDB* sequences should be checked in the next iteration.

The algorithm will be iterated until all sensitive patterns become hidden. In steps 3 through 9, the algorithm finds the best candidate solution for each sequence, and then finds the best solution for the entire *FSDB*.

The supports of the corresponding sensitive patterns in that sequence are decreased by one unit. The sensitive patterns whose supports are equal to λ, will be deleted from the *SPS*. Steps 10 and 11 are *Sanitization* steps in which the algorithm applys the best overall solution to the corresponding sequence and updates the *SDB*.

Table 1: The mappings between letters and goods' names

B	Butter
C	Cheese
E	Eraser
M	Milk
FP	Fountain pen
CP	Correction pen
P	Pencil
Y	Yogurt
Med	'Medium' membership function
Low	'Low' membership function
Hi	'High' membership function

In order to illustrate the proposed algorithm, consider sequence (noticed to Table1) s = < [*M*,*Med*] ([*C*,*Low*] [*B*,*Hi*]) [*P*,*Low*] [*E*,*Low*] ([*Y*,*Low*] [*FP*,*Med*]) [*CP*,*Med*] [*Y*,*Low*] > and its candidate tree in Figure2 with a sensitive pattern set *SPS* = {[*P*,*Low*] [*E*,*Low*] [*CP*,*Med*], [*M*,*Med*] [*E*,*Low*] [*Y*,*Low*]}, the first fuzzy sensitive pattern can be described: the pattern, the *customer buys a pencil (with few numbers), an eraser (with few numbers) and a correction pen (with Medium (not high, not few) numbers)*, is important for the dataset's owner. For the sake of simplicity, the effect of *NSP* is ignored, i.e. $\delta = 0$. The other parameters are $\alpha = 1, \gamma = 1$, and *M-Pruning* = 0.

In Figure2, *HFSP* is applied to the sample sequence and the candidate tree is discussed. Each solution is in the form of (*nSP*,*D*),*Obj* . *nSP* is the number of sensitive patterns, *D* is the number of distortions, and "*Obj*" is the objective function value for the solution, respectively.

NSP is ignored for simplicity. In the first level of the tree, the best objective value is zero, thus it is saved as the current best objective value, and all the solutions in level two of the tree are evaluated at step 1.2.1 and those underlined are pruned and the current best objective value is updated to 1. The final best solution of this candidate tree is [*E*,*Low*] [*E*,*Low*] (*2*,*1*),*1*, which is marked with an asterisk and has a value of 1.

The result of the applying *EMOSS* is the sanitized sequence s = < [*M*,*Med*] ([*C*,*Low*] [*B*,*Hi*]) [*P*,*Low*] [*?*,*?*] ([*Y*,*Low*] [*FP*,*Med*]) [*CP*,*Med*] [*Y*,*Low*] >, in which the two sensitive patterns are hidden solely by one distortion.

Suppose the number of sensitive patterns is greater than two, then the candidate tree will deepen more than 2 levels and there will be opportunities for pruning, resulting in less computational burdens and memory usage.

It can be shown that the complexity of the algorithm, after fuzification part, in the worst case is:[16]

$$[(NSP*LSP)+(NSP-2)*(NSP\hat{}2)]*DBSize*NSP \quad (3)$$

where, LSP and NSP are Length of Sensitive Pattern and Number of Sensitive Patterns which the sequence supports, respectively.

In section 4, the new algorithm is described in detail and its performance is invstigated for different values of parameters. Further illustration will be presented through experimental results which are presented in the next section.

Algorithm: HFSP
INPUTS: SDB, SPS, λ, α, γ, δ, M-pruning
OUTPUT: sanitized SDB (SDB')

1 FSDB = Fuzzification of SDB
2 DBSeqsToCheck ← all s-ids
While |SPS| > 0
3 For each s-id in DBSeqsToCheck
3.1 Level1: Generate solutions that are considered sensitive items for deletion, save the best objective value of the first level as the current best objective value,
3.2 Level2 and more : Combine solutions to generate candidate tree:
3.2.1 If (current best objective value – objective value for the solution)>M-pruning, then prune the subtree starting from the current solution.
3.2.2 If the objective value of the solution is better than the current best objective value, then update the current best objective value.
3.3 bestSolutions ← highest ranked solution in the candidate tree
4 Find best overall solution of FSDB
5 Reduce the support of affected sensitive patterns one unit
6 For each sp in SPS
6.1 If sup(sp) = λ then delete sp from SPS
7 Empty DBSeqsToCheck
8 DBSeqsToCheck ← updated sequence's s-id
9 DBSeqsToCheck ← DBSeqsToCheck + s-id of sequences which contain removed sensitive patterns
10 For each s-id in DBSeqsToCheck
11 Delete the corresponding bestSolution
12 Apply best overall solution to the corresponding sequence
1.1 Replace each selected item with" ?" and update FSDB and SDB

Figure 1: *HFSP*Algorithm

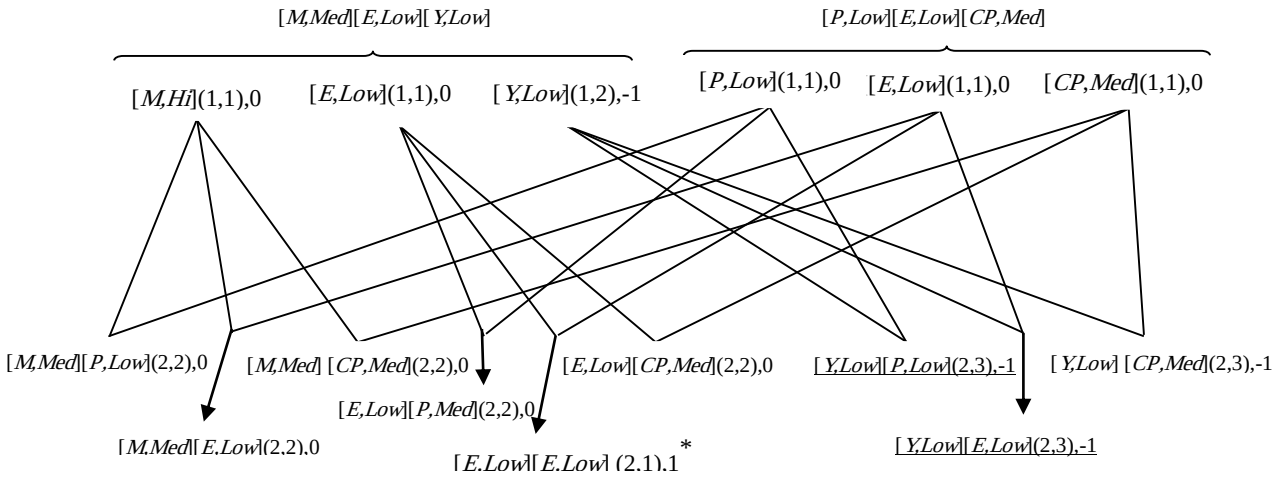

Figure 2: Candidate tree for the sample sequence with pruning

IV. EXPERIMENTAL RESULTS

In this section the performance of *HFSP* is investigated based on two datasets. The first dataset are the synthetic customer-basket sequences used in [21].

The second one is real sequences of a customer-basket of oil products. These datasets are denoted as *CB*, *OilC* respectively.

The proposed algorithm, i.e. *HFSP*, was implemented in MATLAB R2008b and all the experiments were conducted on a system equipped with a 2.66GHz Intel core duo processor and a 3MB physical memory, running the Windows XP operating system.

These comparative studies were performed using the following criteria: the *number of distortions* imposed on the dataset, running-time and *infidelity*. It is well-worth mentioning that "*infidelity* is a measure that encompasses those non-sensitive patterns whose support falls below the support threshold after sanitization which was introduced in the previous work of the authors" [15].

Information regarding the datasets is shown in Table2. The support threshold used to find frequent patterns for each miner algorithm dataset sequence is shown in Column two of Table2, and the third column of the table shows the number of frequence patterns.

Each Figure has its own legend in the form of ""*algorithm-name*,(α,γ,δ,*M-pruning*)".

The *algorithm-name* refers to the algorithm used in the test. (α,γ,δ,*M-pruning*) are the parameters used in the test of the algorithm. It should be noted that the parameter *M-pruning* is defined for the *HFSP* algorithm. Experiments showed that *M-pruning* = 0.7 is a proper value.

Table 2: Test Datasets

Dataset	Support threshold	Patterns
CB	40	245
OilC	60	188

In this section, performance of *HFSP* is tested with/without pruning in computing-time, distortion, infidelity, and the total number of solutions produced during a run which gives a measure of memory usage of each algorithm. The result of *HFSP* without pruning in the legend is shown with the"*algorithm-name*,(α,γ,δ,-)", where "–" means no pruning is applied.

In Figures 3(a)-3(d), the experiments are performed on a *CB* dataset, for the sets of sensitive patterns, i.e. 4 SPs (*S*ensitive *P*atterns).

All experiments were performed for different values of the hiding threshold (λ). Figures3(a) and 3(b) show that *HFSP* with pruning, performs much better than *HFSP* without pruning in terms of running-time and memory usage, while their distortion and infidelity are the same, according to Figures 3(c)-3(d). Note that for better presentation we zoomed time and total solution space in 3(a') and 3(b').

Similar experiments were conducted on the OilC dataset, and the same results are confirmed by Figures 4(a)-4(d) as well as Figures 5(a)-5(d), respectively.

It should be noted that the number of non-sensitive patterns affected by the algorithms was also considered in Figures 5(a)-5(d) by δ =1. This leads to increased infidelity. Note that by selecting an improper value for *M-Pruning* in Figure 4, the results for infidelity and distortion are not satisfacory.

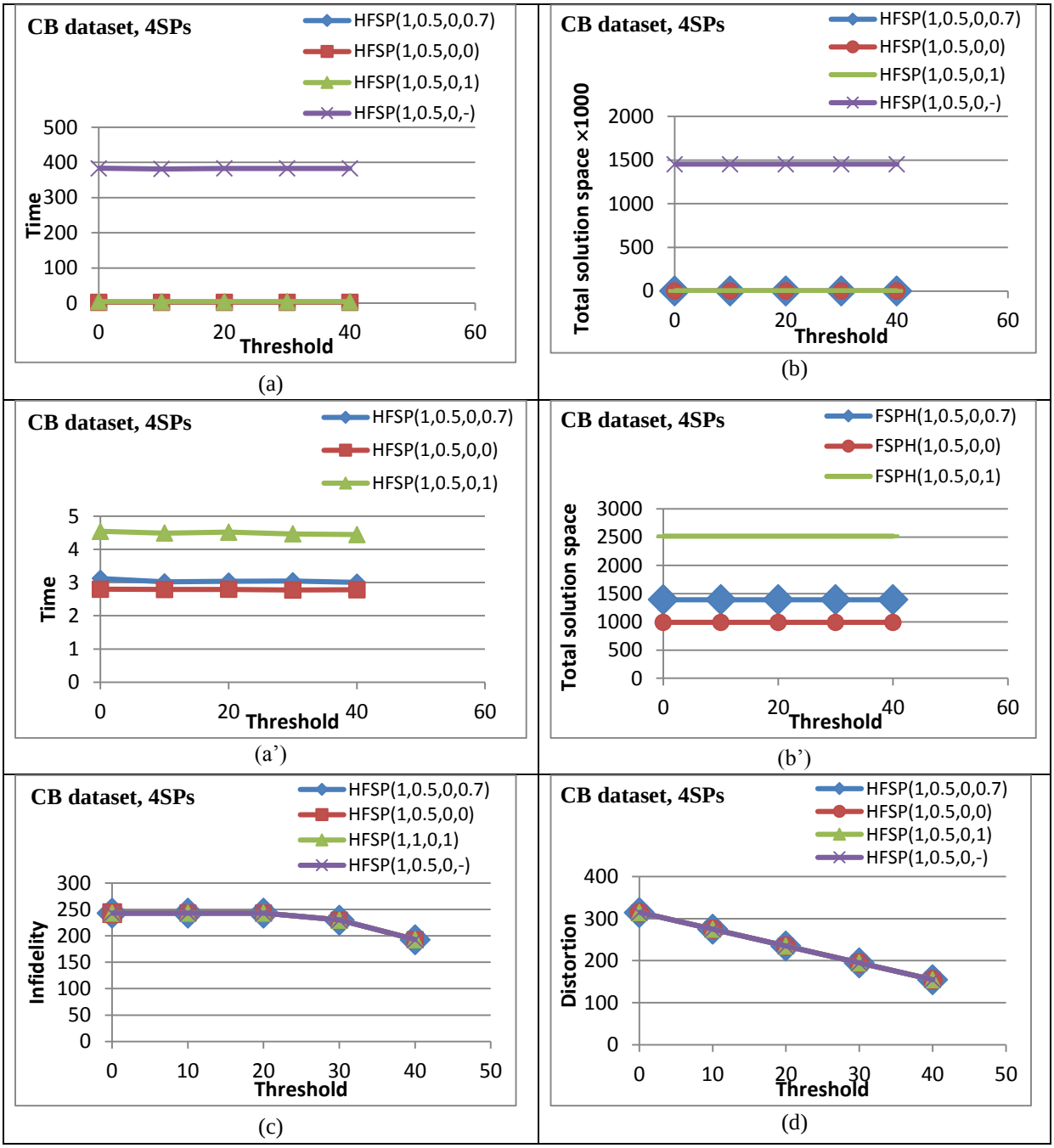

Figure3: CB dataset experimental results :(a) time for 4SPs,(b)number of total solutions for 4SPs,(a') zoom of time for 4SPs, (b')zoom of number of total solutions for 4SPs, (c) infidelity for 4 SPs,(d)distortion for 4SPs

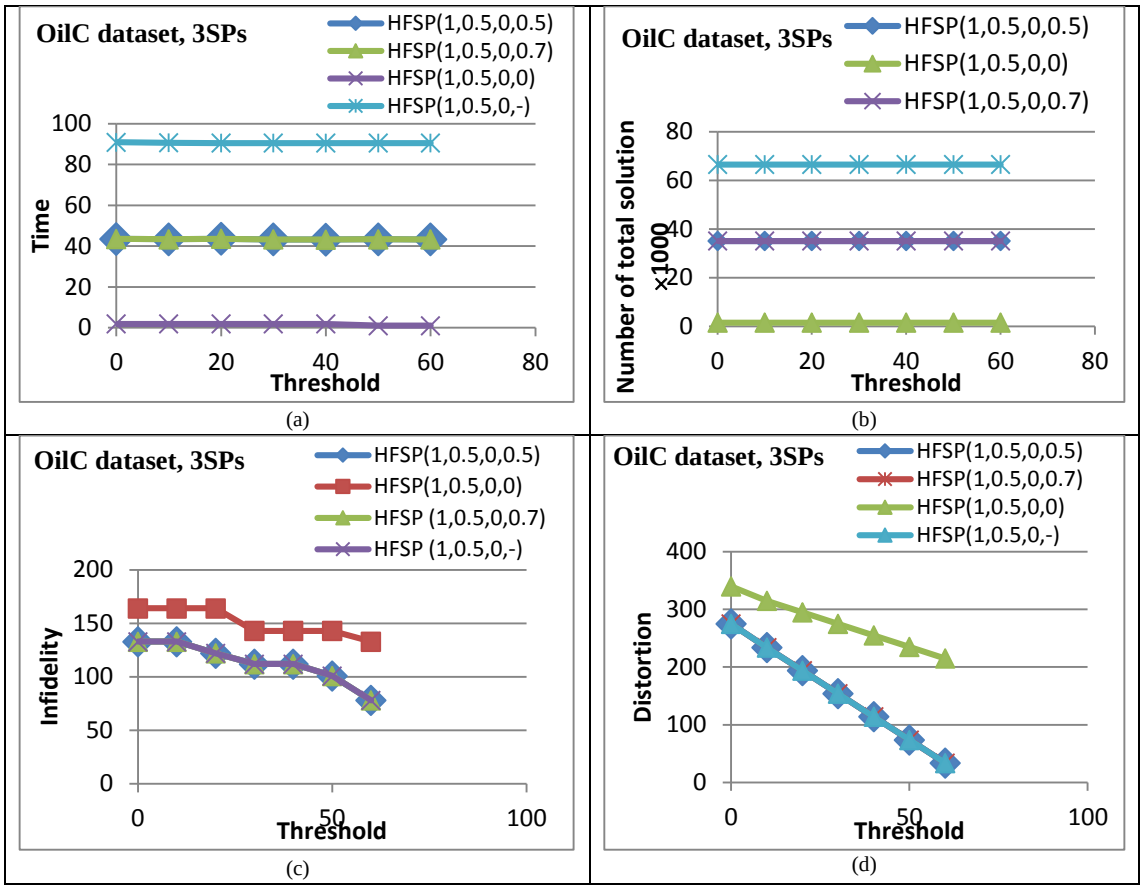

Figure4: OilC dataset experimental results : (a) time for 4SPs,(b)number of total solutions for 4SPs, (c) infidelity for 4 SPs,(d) distortion for 4SPs.

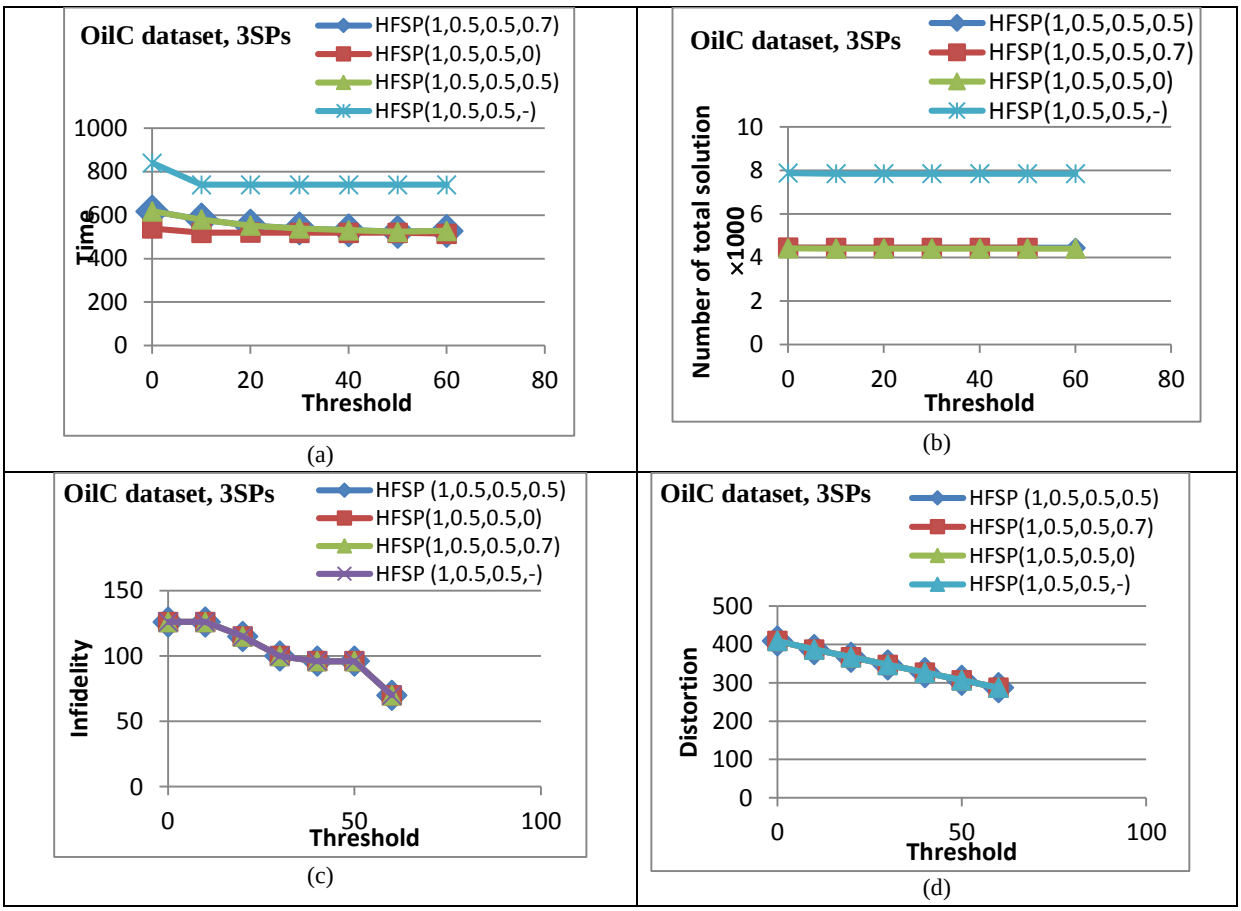

Figure5: OilC dataset experimental results with δ=1: (a) time for 4SPs,(b)number of total solutions for 4SPs, (c) infidelity for 4 SPs,(d) distortion for 4SPs.

5. CONCLUSION& FUTURE STUDIES

Sequential fuzzy pattern mining is a useful and active research area for discovering fuzzy sequential patterns. Maintaining privacy with regard to important information mined by these methods is an issue. In this paper an algorithm is proposed for hiding fuzzy sequential patterns with several advantages. First of all, previous research distorts the dataset by using unknowns,"?" or removing sensitive items, but *HFSP* uses crisp taxonomies to replace sensitive items in order to give the miner more generalized patterns. The next point is that the proposed algorithm hides sensitive patterns from a sequence database by using a highly flexible weighted objective function to find the best solution among all candidate solutions in order to sanitize sequences. This results in fewer distortions and lower infidelity in the database. Data quality can therefore be better preserved. The weights in the objective function of the algorithm and the M-pruning threshold, however, are needed to preserve higher efficiency.

Further studies include devising an efficient method to find the best value for *M*-Pruning for different types of data, number of distortions, number of solutions and other parameters of the objective function, which the authors are working on.

ACKNOWLEDGEMENT

This research was partially supported by The Research Institute for Information and Communication Technology of Iran.

REFERENCES

[1] W. Romsaiyud, W. Premchaiswadi, "Applying mining fuzzy sequential patterns technique to predict the leadership in social networks," In Proceeding of Ninth International Conference on ICT and Knowledge Engineering, pp.134-137, 2011.

[2] Y.-C. Hu, R.-S. Chen, G.-H. Tzeng, J.-H. Shieh, "A Fuzzy Data Mining Algorithm for Finding Sequential Pattehrns," International Journal of Uncertainty Fuzziness and Knowledge-Base Systems, vol.11(2), pp. 173-194, April 2003.

[3] T.P. Hong, K.Y. Lin, S.L. Wang, "Fuzzy data mining for interesting generalized association rules," Fuzzy Sets and Systems, vol. 138, pp. 255-269, 2003.

[4] C. Fiot, F. Masseglia, A. Laurent, M. Teisseire," From Crispness to Fuzziness: Three Algorithms for Soft Sequential Pattern Mining," IEEE Transaction on Fuzzy Sets, vol. 15, no 6, pp. 1263-1277, 2007.

[5] T.P. Hong, K.Y.Lin, S.L. Wang, "Mining fuzzy sequential patterns from quantitative transactions," Soft Computing - A Fusion of Foundations, Methodologies and Applications, vol. 10, pp. 925-932, 2006.

[6] T.P. Hong, K.Y. Lin, B.C. Chien, "Mining fuzzy multiple-level association rules from quantitative data," Applied Intelligence, vol. 18, pp. 79-90, 2003.

[7] Y.L. Chen, T.C.K. Huang, "A novel knowledge discovering model for mining fuzzy multi-level sequential patterns in sequence databases," Data & Knowledge Engineering, vol. 66, pp. 349-367, September 2008.

[8] E. Dasseni, V. Verykios, A. Elmagarmid, E. Bertino, "Hiding association rules by using confidence and support," Springer, pp. 369-383, 2001.

[9] V.S. Verykios, A.K. Elmagarmid, E. Bertino, Y. Saygin, E. Dasseni, "Association rule hiding," IEEE Transactions on Knowledge and Data Engineering, vol. 16, pp. 434-447, 2004.

[10] E. Pontikakis, A. Tsitsonis, V. Verykios, "An experimental study of distortion-based techniques for association rule hiding," Research Directions in Data and Applications Security XVIII, pp. 325-339, 2004.

[11] G. Lee, C.Y. Chang, A.L.P. Chen, "Hiding sensitive patterns in association rules mining," 28th Annual International Computer Software and Applications Conference (COMPSAC'04), IEEE Computer Society, pp. 424-429 2004.

[12] Y. Saygin, V.S. Verykios, C. Clifton, "Using unknowns to prevent discovery of association rules," ACM SIGMOD Record, vol. 30, pp. 45-54, 2001.

[13] S.R.M. Oliveira, O.R. Zaïane, "Protecting sensitive knowledge by data sanitization," Third IEEE International Conference on Data Mining (ICDM), pp. 613-616, 2003.

[14] O. Abul, M. Atzori, F. Bonchi, F. Giannotti, "Hiding sequences," the IEEE 23rd International Conference on Data Engineering Workshop (ICDEW 2007), pp.147-156, 2007.

[15] B. Rahbarinia, M.M. Pedram, H. Arabnia, Z. Alavi, "A multi-objective scheme to hide sequential patterns," The 2nd International Conference on Computer and Automation Engineering (ICCAE), vol.1, pp. 153-158, 2010.

[16] O. Behbahani, M.M. Pedram, K. Badie, B. Rahbarinia, "EMOSS: An Efficient Algorithm to Hide Sequential Patterns", IJICTR (International Journal Of Information & Communication Technology Research), vol.8,Number4,pp.65-80 ,Autumn 2015.

[17] F. Shahzad, S. Asghar, K. Usmani, "A Fuzzy Based Schem For Sanitizing Sensitive Sequential Patterns," The Intenational Arab Journa of Information Technology, Vol. 12, No.1, January 2015.

[18] T. Dinh, M. N. Quang, B. Le, "A Novel Approach for Hiding High Utitlity Sequential Patterns," In Proceeding of Sixth International Symposium on Information and Communication Technology, pp. 121-128, 2015.

[19] T. Berberoglu, M. Kaya, "Hiding Fuzzy Association Rules in Quantitative Data," The 3rd International Conference on Grid and Pervasive Computing Workshops (GPC Workshops '08) , pp. 387-392,2008.

[20] M. Gupta, R. C. Joshi, "Privacy Preserving Fuzzy Association Rules Hiding in Quantitative Data," International Journal of Computer Theory and Engineering, vol. 1, pp. 382-388, October, 2009.

[21] F. Zabihi, M.M. Pedram, M. Ramezan, A. Memariani, "Fuzzy sequential pattern mining with sliding window constraint," 2nd International Conference on Education Technology and Computer (ICETC), vol. 5, pp. 396-400, 2010.

Olya Sadat Behbahani received her M.Sc. degree in Artificial Intelligent from the Kharazmi University, Tehran, Iran, 2011, and Bachelor's in Hardware Computer Engineering from Sanati Babol University, Mazandaran, Babol, Iran, 2004. She is currently a Teacher in the Department of Computer Engineering at Islamic Azad University North Tehran Branch. Her Research areas are Expert Systems, Machine Learning, Data Mining and Operating Systems problems.

Kambiz Badie received all his degrees from Tokyo Institute of Technology, Japan, majoring in pattern recognition. Within the past years, he has been actively involved in cognitive modeling & systemic knowledge processing in general and analogical knowledge processing, and modeling interpretation process in particular, with emphasis on creating new ideas, techniques and contents. Dr. Badie is an active researcher, in the areas of interdisciplinary and transdisciplinary studies in Iran. At present, he is a member of scientific board of IT Research Faculty, and in the meantime, Deputy Director for Research Affairs in ICT Research Institute.

Mir Mohsen Pedram received his P.hD. in Electrical Engineering from the Tarbiat Modarres University, Tehran, Iran, 2003, M.S in Electrical Engineering from Tarbiat Modarres University, Tehran, Iran, 1994 and B.S in Electrical Engineering from Isfahan University of Technology, Isfahan, Iran, 1990. He is currently Assistant Professor in the Department of Electrical and Computer Engineering at Kharazmi University. He is also the head of the Data Mining and Cognitive Science research laboratories at Kharazmi University. His main areas of research are Intelligent Systems, Machine Learning, Data Mining and Cognitive Science.

This Page intentionally left blank.