Applied Computing and Informatics (2016) 12, 134–141

Saudi Computer Society, King Saud University

Applied Computing and Informatics

(http://computer.org.sa)
www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

# Discovery of temporal association rules with hierarchical granular framework

**Tzung-Pei Hong [a,b,*], Guo-Cheng Lan [c], Ja-Hwung Su [d], Pei-Shan Wu [b], Shyue-Liang Wang [e]**

[a] *Department of Computer Science and Information Engineering, National University of Kaohsiung, 811, Taiwan, ROC*
[b] *Department of Computer Science and Engineering, National Sun Yat-Sen University, 804, Taiwan, ROC*
[c] *Department of Computer Science and Information Engineering, National Cheng Kung University, 701, Taiwan, ROC*
[d] *Department of Information Management, Cheng Shiu University, 833, Taiwan, ROC*
[e] *Department of Information Management, National University of Kaohsiung, 811, Taiwan, ROC*

Received 4 November 2014; revised 24 November 2015; accepted 28 January 2016
Available online 2 February 2016

**KEYWORDS**

Data mining;
Association-rule mining;
Temporal association rules;
Item lifespan;
Time granules

**Abstract** Most of the existing studies in temporal data mining consider only lifespan of items to find general temporal association rules. However, an infrequent item for the entire time may be frequent within part of the time. We thus organize time into granules and consider temporal data mining for different levels of granules. Besides, an item may not be ready at the beginning of a store. In this paper, we use the first transaction including an item as the start point for the item. Before the start point, the item may not be brought. A three-phase mining framework with consideration of the item lifespan definition is designed. At last, experiments were made to demonstrate the performance of the proposed framework.


## 1. Introduction

Data mining can help derive useful knowledge from databases. Among its technology, association-rule mining [1,3,28] considers frequency relationship among items and is commonly applied to many applications. A transaction usually includes the items bought and the time of its occurrence. Besides, the periods for items to be exhibited are also important. Some researches about temporal data mining were thus presented [27]. For example, the time period for an item may be the entire time interval of a database [5], the duration from the first occurring time of the item to the end of a database [20], or the on-shelf time periods of the item [8]. However, an infrequent item for the entire time interval may be frequent within part of the time.

* Corresponding author at: Department of Computer Science and Information Engineering, National University of Kaohsiung, 811, Taiwan, ROC.
E-mail addresses: tphong@nuk.edu.tw (T.-P. Hong), rrfoheiay@gmail.com (G.-C. Lan), bb0820@ms22.hinet.net (J.-H. Su), grani1987@gmail.com (P.-S. Wu), slwang@nuk.edu.tw (S.-L. Wang).
Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.aci.2016.01.003
2210-8327 

In this paper, we thus organize time into granules and consider temporal data mining for different levels of granules. We use the first transaction including an item as the start point for the item. We propose a three-phase mining framework with consideration of the above item lifespan definition to mine temporal association rules with time granules from a temporal database. According to the definition of item lifespan, in the first phase, each elementary time interval is processed. The temporal frequent itemsets within the above intervals are first found, and then the itemsets are identified as candidate temporal frequent ones in all the time granules of the upper level of the hierarchy. These candidates are then judged for being temporal or not at each level of granules. Additional database scans may be needed to find the actual supports of the candidates. In the third phase, the possible candidate association rules are derived from the temporal frequent itemsets at each level. Their confidence values are then calculated and compared with the minimum confidence value to get the final temporal association rules.

The organization of the paper is stated below. Related works are given in Section 2. The problem to be solved is described in Section 3. The proposed algorithm with consideration of the first transaction appearance period is presented in Section 4. The performance of the proposed approach is shown in Section 5. Conclusions and future works are finally given in Section 6.

## 2. Review of related works

Temporal data mining is popular in recent years. It analyzes temporal data to get patterns or regularities. There are many techniques included in temporal data mining. Sequential association mining [2], cyclic association mining [22], stock trading rule mining [11], patent mining [12], clinical mining [25], image time series mining [15], software adoption and penetration mining [23], temporal utility mining [9,29], fuzzy temporal mining [6,16,17], and calendar association mining [21] all belong to it. There are also a variety of applications for temporal data mining. For example, Patnaik et al. used temporal data mining to efficiently manage the cooling system in data centers [24], and Rashid et al. adopted it for finding the correlation among sensor data [26].

Chang et al. considered the temporal mining problem of products exhibited in a store [5]. They proposed the concept of common exhibition to find patterns. In a common exhibition period, all the items in an itemset need to be on the shelf at the same time. Lee et al. then used it to discover general temporal association rules for publication databases [20]. Ale and Rossi then considered the transaction periods of products [4], instead of their exhibition periods, for finding temporal association rules. Besides, different products may have different on-shelf properties. For example, a popular product may be sold out quickly, and then be supplied and on shelf soon. It is thus intermittently on-shelf and off-shelf in the entire time [18].

As to hierarchical temporal mining, Li et al. proposed an approach to discover calendar-based temporal association rules [21]. That approach could mine rules according to different calendar constraints including years, months and days. Chen et al. proposed a hierarchical strategy for video event detection from video databases [7]. They divided the frequent actions into two types, namely pre-actions and post-actions by pre- and post-temporal windows. Fang and Wu used granules of features to speed up the mining process of association rules [10].

In this paper, we consider the phenomenon that an itemset may not be frequent in the entire time interval, but may be frequent in a partial time interval. We thus organize the time into different levels of granules and find the temporal association rules at each level. This paper is extended from our previous work [19] with different consideration of effective time intervals. Here we use the first occurring transaction of an item as the start point for the item. Before the start point, the item may not be brought since it is not ready. This definition is of the benefit that it is not necessary to require the exact on-shelf time of each item in advance.

## 3. Problem statement and definitions

To describe the problem of hierarchical temporal association rule mining clearly, assume a temporal database (abbreviated as *TDB*) in Table 1 is given. Four items are included in the transactions, denoted $A$ to $D$.

In addition, there is a pre-defined hierarchy with time granules in three levels, in which there are four basic time periods, denoted as $p_1$ to $p_4$, and the time granules are in three levels in the hierarchy, as shown in Fig. 1. Based on Fig. 1 and Table 1, $\{C\} \rightarrow \{D\}$ is one of hierarchical temporal association rules occurring in the time granule $p_{12}$. The goal of this paper was to mine such temporal association rules, and the detailed definitions and examples will be described as follows.

The terms related to the hierarchical temporal mining under the first occurring transaction periods of items are explained below.

**Definition 1.** $P = \{p_1, p_2, \ldots, p_j, \ldots, p_n\}$ is a set of mutually disjoint time periods, where $p_j$ denotes the $j$-th time period in the whole set of periods, $P$.

**Definition 2.** Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items appearing in a database. If $X \subseteq I$, then $X$ is called an itemset.

**Definition 3.** Let $X$ be an itemset and $t$ be a time stamp. A transaction $T$ is a pair ($X$, $t$).

**Table 1** An example of a temporal database.

| Period | TID | Items |
|---|---|---|
| $p_1$ | $Trans_1$ | $D$ |
| | $Trans_2$ | $C, D$ |
| | $Trans_3$ | $C$ |
| | $Trans_4$ | $D$ |
| $p_2$ | $Trans_5$ | $A, C, D$ |
| | $Trans_6$ | $A, B, C, D$ |
| | $Trans_7$ | $B, C, D$ |
| | $Trans_8$ | $A, D$ |
| $p_3$ | $Trans_9$ | $B$ |
| | $Trans_{10}$ | $A, C$ |
| | $Trans_{11}$ | $A, B, C$ |
| | $Trans_{12}$ | $B, C$ |
| $p_4$ | $Trans_{13}$ | $B, D$ |
| | $Trans_{14}$ | $B, C, D$ |
| | $Trans_{15}$ | $B$ |
| | $Trans_{16}$ | $B, C, D$ |

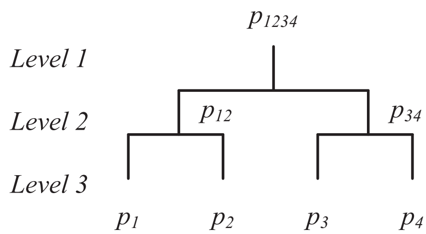


**Figure 1** An example of time granules.

**Definition 4.** A temporal transaction database $TDB = \{Trans_1, Trans_2, \ldots, Trans_y, \ldots, Trans_z\}$, where $Trans_y$ is the $y$-th transaction in $TDB$.

**Definition 5.** The maximal time period of an item $i$, $MTP(i)$, is from the time period of the first occurring transaction of the item to the last time period of the temporal database.

**Definition 6.** The maximal time period of an itemset $X$, $MTP(X)$, represents the common time period of the maximal time periods of all items in $X$ in a temporal database $TDB$.

**Definition 7.** A hierarchy of time granules, $HTG$, is composed of a set of basic time periods. In addition, a time granule $pg_{l.g}$ represents the $g$-th time granule in the $l$-th level of the hierarchy, and it consists of the basic time periods contained by the time granule $pg_{l.g}$.

**Definition 8.** The count $c(i, p)$ of item $i$ in a basic time period $p$ is the number of transactions with $i$ in $p$.

**Definition 9.** The relative support $rsup(i, pg)$ of item $i$ in a hierarchical time granule $pg$ is the number of transactions with $i$ in its maximal time period of $pg$ over the number of all transactions within its maximal time period of $pg$.

**Definition 10.** The relative support $rsup(X, pg)$ of itemset $X$ in a hierarchical time granule $pg$ is the number of transactions including the itemset $X$ in its maximal time period of $pg$ over the number of all transactions in its maximal time period of $pg$.

**Definition 11.** Let $min\_rsup$ be a given minimum relative support threshold. If $rsup_{pg}(X) \geqq min\_rsup$, $X$ is called a hierarchical temporal frequent itemset (abbreviated as $HTFI$).

**Definition 12.** Assume $X$ is a hierarchical temporal frequent $q$-itemset with items $(x_1, x_2, \ldots, x_q)$, $q \geqslant 2$. The relative confidence $rconf(R, pg)$ of a hierarchical temporal association rule within a time granules $pg$, which is denoted as $\{x_1 \wedge \ldots \wedge x_{k-1} \wedge x_{k+1} \wedge \ldots \wedge x_q\} \rightarrow \{x_k\}$, is shown below:

$$rconf(\{x_1 \wedge \ldots \wedge x_{k-1} \wedge x_{k+1} \wedge \ldots \wedge x_q\}, \{x_k\}, pg) = \frac{rsup(X)}{rsup(\{x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_q\})}$$

**Definition 13.** Let $min\_rconf$ be a given minimum relative confidence threshold. For a rule $R$, if $rconf(R, pg) \geqq min\_rconf$, $R$ is called a hierarchical temporal association rule (abbreviated as $HTAR$).

Table 1 is a simple example showing that, the fifth transaction $\{A, C, D\}$ contains three items, $A$, $C$, and $D$, and the time stamp of the transaction is $p_2$. In Table 1, the first time period is represented as $p_1$, and $P$ includes four time periods, $p_1$, $p_2$, $p_3$, and $p_4$. In this example, the itemset $\{AB\}$ containing two items is called a 2-itemset. Since the first transaction including the 1-itemset $\{B\}$ is the sixth transaction in $TDB$, and the first time period of the transaction and the last time period of the database are $p_2$ and $p_4$, respectively, the maximal time period $MTP(\{B\})$ of the item $B$ is $p_2$ to $p_4$. Also, the maximal time period of $\{BCD\}$, $MTP(\{BCD\})$, is from $p_2$ to $p_4$ based on the maximal time periods of the three items, $B$, $C$ and $D$. By considering Fig. 1, the hierarchy is composed of four basic time periods in the temporal database, $p_1$, $p_2$, $p_3$, and $p_4$, and the second time granule $pg_{2.2}$ in the second level of the hierarchy is composed of $p_3$ and $p_4$. Since item $B$ appears in $Trans_6$ and $Trans_7$, within the first basic time period $p_2$, the count value $c(\{B\}, p_2)$ of the item in $p_2$ is the value of 2. Accordingly, the $rsup(\{B\}, pg_{2.1}) = 2/4 = 50\%$. In this example, the maximal time period of the item $B$ is set as $pg_{2.1}$ and only $p_2$ contains the item $B$. That is, the number of transactions containing $B$ and all the transactions in $p_2$ are 2 and 4, respectively. Also, the $rsup(\{AB\}, pg_{2.1}) = 1/4 = 25\%$ since the maximal time period of the itemset $\{AB\}$ in $pg_{2.1}$ only includes $p_2$, and the number of transactions including $\{AB\}$ and all the transactions in $p_2$ are 1 and 4, respectively. Further, the $rsup(\{CD\}, pg_{2.1}) = 50\%$. If the $min\_rsup = 30\%$, then the itemset $\{CD\}$ is a hierarchical temporal frequent itemset within the time granule $pg_{2.1}$. Since the $rsup(\{C\}, pg_{2.1}) = 62.5\%$, the $rconf(\{C\} \rightarrow \{D\}, pg_{2.1}) = 50\%/62.5\% = 80\%$. It is then compared with $min\_rconf$.

Based on the above definitions, the problem to be solved is to find the hierarchical temporal association rules with their actual relative support and confidence values within the maximal time period of the itemset of a time granule being larger than or equal to a predefined minimum relative support threshold $min\_rsup$ and a predefined minimum relative confidence threshold $min\_rconf$, respectively.

## 4. The proposed algorithm

The proposed approach considers the first occurring transaction period information of products and is processed in three phases. It also adopts a predicting strategy which can reduce the number of data scan by the upper-bound support. Basically, the proposed method is a level-wise algorithm which mines the frequent itemsets level by level and period by period. The main contribution of the proposed method is to reduce the number of data scanning, which can be approved by the experimental results later. The mining procedures of the proposed algorithm are stated as follows.

The $TPPF$ algorithm (three-phase algorithm with predicting strategy considering the first occurring transactions of items) is as follows:

INPUT: A temporal database $TDB$ with $n$ transactions, each of which consists of transaction identification, transaction occurring time and items purchased, $m$ items in $TDB$, a hierarchy with time granules $HTG$, the minimum relative support threshold $min\_rsup$, and the minimum relative confidence threshold $min\_rconf$.

OUTPUT: A final set of all hierarchical temporal association rules, $HTAR$.

Phase 1: Find temporal frequent itemsets.

STEP 1: Initialize the $PTT$ (Periodical Total Transaction) table as a zero table, in which the row number is the time period number of the bottom level in the hierarchy of time granules, and each entry in the table is set as 0.

STEP 2: Find the periodical total transaction number $ptt_j$ within each time period $p_j$ of the bottom level in $HTG$ as the number of transactions in $p_j$, and put it in the $PTT$ table.

STEP 3: Initialize the first appearance period $FAP$ table as an empty table, in which each tuple consists of two fields: an item and the time period $p$ of the first transaction including it in $TDB$.

STEP 4: Find the time period $p$ of the first transaction including the item $I$ in $TDB$, and then put the item and its first time period $p$ in $FAP$.

STEP 5: Find the temporal frequent itemsets within each time period $p$ by the *Finding-Individual-TFI* procedure. Let the set of returned temporal frequent itemsets for the $j$-th time period $p_j$ of the bottom level of $HTG$ be denoted as $TFI_j$.

Phase 2: Find all hierarchical temporal frequent itemsets.

STEP 6: Initially set the set of hierarchical temporal frequent itemsets ($HTFI$) as empty.

STEP 7: For each time period granule $pg$ in each of all the other levels in $HTG$ other than the bottom one, do the following substeps.

(a) Get the union of all $TFI_j$'s in $pg$, and denote them as possible itemsets, $PI_{pg}$.

(b) For each itemset $X$ in the set of $PI_{pg}$, find the maximum common period $MCP_X$ of all the items in $X$ within the time granule $pg$ by using the $FAP$ table and then calculate the relative support upper-bound $rsub_{pg,X}$ of $X$ within the time granule $pg$ as:

$$rsub_{pg,X} = \left( \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} c^{actual}_{j,X} + \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} c^{ub}_{j,X} \right) \Bigg/ \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} ptt_j,$$

where $c^{actual}_{j,X}$ is the actual count of $X$ within the $j$-th time period $p_j$ of the time granule $pg$ by the sets of all $TFI_j$ of the time granule $pg$, and $c^{ub}_{j,X}$ is the upper-bound $(= \lambda * ptt_j - 1)$ of $X$ within $p_j$ of $pg$ by the $PTT$ table.

(c) For each itemset $X$ in set of $PI_{pg}$, calculate the relative support lower-bound $rslb_{pg,X}$ of $X$ within the time granule $pg$ as:

$$rslb_{pg,X} = \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} c^{actual}_{j,X} \Bigg/ \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} ptt_j.$$

(d) Store each $X$ in the set of $PI_{pg}$ whose $rslb_{pg,X}$ exceeds the minimum relative support threshold *min_rsup* into the set of hierarchical temporal frequent itemsets ($HTFI$) and set $PI_{pg} = PI_{pg} - X$.

(e) For each itemset $X$ remaining in the current set of $PI_{pg}$, scan the transactions to calculate the relative support value $rsup_{pg,X}$ within the time granule $pg$ as:

$$rsup_{pg,X} = \sum_{p_j \in MCP_x \wedge p_j \subseteq pg} C_{j,X} \Bigg/ \sum_{p_j \subseteq pg} ptt_j,$$

(f) Store each $X$ in the set of $PI_{pg}$ whose relative support exceeds the minimum relative support threshold *min_rsup* into the set of hierarchical temporal frequent itemsets ($HTFI$); otherwise, set $PI_{pg} = PI_{pg} - X$.

Phase 3: Find all hierarchical temporal association rules.

STEP 8: Initially set the set of hierarchical temporal frequent sub-itemsets ($HTFS$) as empty.

STEP 9: For each itemset $X$ in the $HTFI$ set, do the following substeps:

(a) Generate all possible sub-itemsets of the itemset $X$.

(b) For each sub-itemset $s$, check whether the sub-itemset $s$ with the same common period exists in the $HTFI$ set. If it does, put the sub-itemset $s$ in the $HTFS$ set and use the relative support value of $s$ in the $HTFI$ set as the relative support value of $s$ in the $HTFS$ set; otherwise, scan the transactions of the required time periods to find the relative support value of $s$, and then put $s$ with its relative support value in the $HTFS$ set.

STEP 10: For each itemset $X$ with items $(x_1, x_2, \ldots, x_r)$ in the $HTFI$ set, generate all possible hierarchical temporal association rules and calculate the relative confidence value $rconf_{pg,R}$ of each possible rule $R$.

STEP 11: Output the final set of hierarchical temporal association rules ($HTAR$) exceeding the minimum relative confidence *min_rconf*.

After STEP 11, all the rules in the set of $HTAR$ have been found from the temporal database. The *Finding-Individual-TFI* procedure used in STEP 5 is described below. Here, the traditional *Apriori* algorithm is adopted to derive frequent itemsets from the transactions within a time period.

The Finding-Individual-TFI procedure is as follows:

Input: A set of transactions $TDB_j$ within a time period $p_j$.

Output: The temporal frequent itemsets $TFI_j$ in $p_j$.

PSTEP 1: Set $r = 1$ and $C_{jr}$ to include all the items in the time period $p_j$.

PSTEP 2: For each temporal candidate $r$-itemset in the set of $C_{jr}$ within $TDB_j$, scan $TDB_j$ to store the itemset whose count exceeds the threshold of $\lambda * ptt_j$ into $TFI_{jr}$.

PSTEP 3: Generate the temporal candidate set $C_{j(r+1)}$ from $TFI_{jr}$ in the current time period $p_j$. The $r$-sub-itemsets of each candidate in $C_{j(r+1)}$ must exist in $TFI_{jr}$.

PSTEP 4: If $C_{j(r+1)}$ is not null, set $r = r + 1$ and repeat PSTEPs 2 to 3; otherwise, set $TFI_j = \bigcup_{k=1}^{k=r} TFI_{jk}$ and return $TFI_j$.

## 5. Experimental results

In this section, the experimental results for showing the pruning effects and efficiency of the proposed *TPPF* approach are presented. As a comparison, the basic three-phase algorithm without consideration of the predicting strategy (named *TP-HTAR*, Three-Phase algorithm for Hierarchical Temporal Association Rules) is derived from the proposed *TPPF* approach. The experimental environment included a personal computer with 3.0 GHz CPU and 2 GB memory, running J2SDK 1.6.0. The two methods were performed on the same machine using the same program language, data and parameter settings. The execution time included data input, generation of frequent itemsets and result output.

### 5.1. Experimental datasets

Two datasets including synthetic data and real data were used to conduct the comprehensive empirical study. In terms of the synthetic data, it was generated by the public IBM data generator [14]. The temporal database was generated by the model used in [18]. The detailed information of the synthetic data is shown in Table 2.

To attack the insufficiency of the synthetic data, we also adopted a real dataset *Foodmart* as the other experimental data. The *Foodmart* database is a well-known dataset from Microsoft SQL Server 2000. It includes 21,556 transactions and 1600 items.

### 5.2. Experimental results on synthetic data

The synthetic T10I4N4KD100KP16 dataset was first used in the experiments. It was divided into 16 basic time periods, which were organized into a hierarchy of 4 levels. Fig. 2 shows the pruning effects of the two approaches, *TP-HTAR* and *TPPF*, for the T10I4N4KD100KP16 dataset for different thresholds within 0.3–0.4%.

**Table 2** Parameter values of the synthetic data.

| Parameter | Description | Default value |
|---|---|---|
| *T* | The average length of items per transaction | 10 |
| *I* | The average length of maximal potentially frequent itemsets | 4 |
| *N* | The total number of items | 4000 |
| *D* | The total number of transactions | 100,000 |
| *P* | The number of basic periods | 16 |

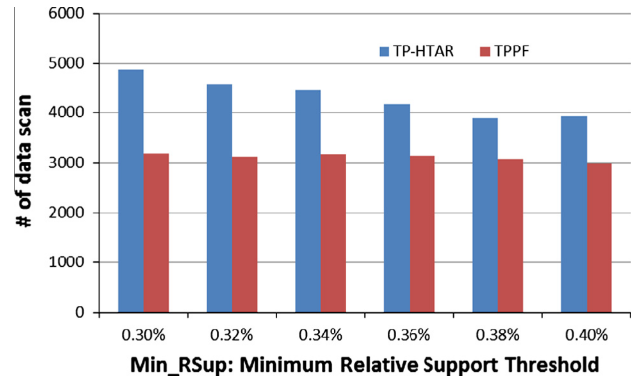


**Figure 2** The pruning effects of the two approaches on the synthetic data.

From the results in Fig. 2, it can be observed that *TPPF* needed less database scans than *TP-HTAR*. It was because *TP-HTAR* purely used the level-wise technique to handle the problem of hierarchical temporal issues. In addition, if all the frequent itemsets in each basic period were identified as possible hierarchical temporal itemsets, then the transactions in the time periods, in which the relative supports of the possible itemsets were unknown, had to be scanned to find the actual relative supports for itemsets. Thus, the *TP-HTAR* performed worse than the proposed *TPPF* in terms of avoiding unnecessary data scans.

The experiments were then conducted to evaluate the efficiency of the two algorithms, *TPPF* and *TP-HTAR*, for the hierarchical temporal mining issue, and Fig. 3 shows the results of the two algorithms working on the T10I4N4KD100KP16 dataset with 16 basic periods and 4 levels for the synthetic datasets with the thresholds varying from 0.3% to 0.4%.

The results clearly show that the execution time of the *TPPF* for the hierarchical temporal mining issue performed better than the other algorithm, *TP-HTAR*. The reason was the same as that mentioned above. Since *TPPF* obviously needed less data scans than *TP-HTAR*, the time cost of unnecessary data scans could effectively be saved by the *TPPF*.

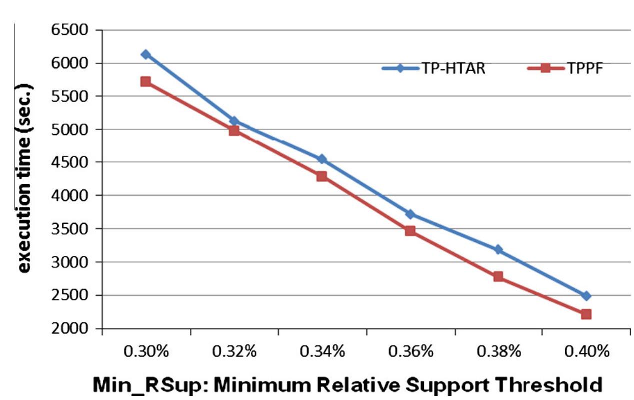


**Figure 3** The execution time of the two approaches on the synthetic data.

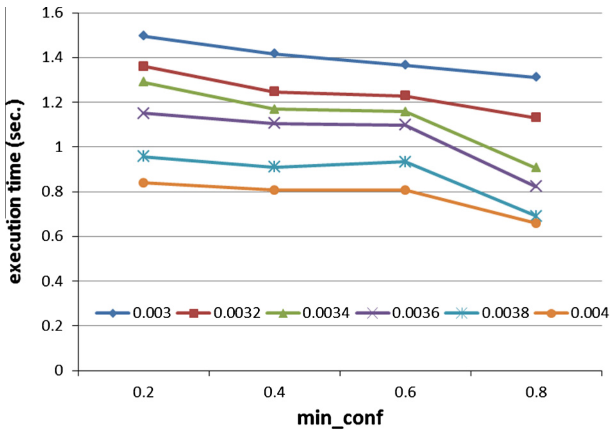


**Figure 4** Execution time of generating association rules under different minimum confidences on the synthetic data.

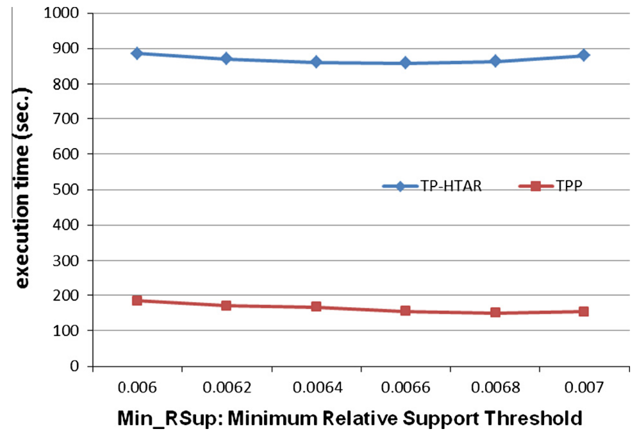


**Figure 5** The execution time of the two approaches on the real dataset *Foodmart*.

Accordingly, *TPPF* could be more efficient than *TP-HTAR* for the synthetic dataset.

In addition to the above experimental results of discovering the frequent itemsets, we also conducted an empirical study for the efficiency of generating association rules based on the discovered frequent itemsets. Fig. 4 shows the experimental results of evaluating the rule generation using the frequent itemsets yielded by different minimum relative support set {0.3%, 0.32%, 0.34%, 0.36%, 0.38%, 0.4%}. That is, six sets were employed to generate associations. The minimum confidence values ranged from 0.2 to 0.8.

From Fig. 4, the experimental discovery can be summarized as follows. First, all of the execution time is quite close, which is within one second. It means the rule generation time is very small when compared with that of generating frequent itemsets. The reason is that, the rule generation is simpler and takes much less time than generating frequent itemsets. Second, whatever the minimum confidence is, the larger the minimum relative support, the smaller the execution time. The reason is when the minimum relative support becomes larger, less frequent itemsets will be generated and thus the rule generation cost will be less as well. Third, for each set, the differences of execution time for generating rules under different minimum confidence values are very slight. The reason is that the confidence checking time depends on the number of frequent itemsets generated, but not on the confidence thresholds. Besides, larger minimum confidence values will get more rules and thus need more time to generate them out. But rule generation is very quick and thus there is no significant difference for different thresholds.

In general, the values of the two parameters *min_rsup* and *min_rconf* affect the performance of the proposed approach. When *min_rsup* is set lower, more candidate itemsets are generated and thus the needed computational time becomes more as well. Similarly, when *min_rconf* is set lower, more rules are generated which needs more computational time. These characteristics can be easily observed from Fig. 4 as well. Besides, the minimum support and confidence values are usually determined according to the data characteristics and user requirement. There are some studies focusing on this issue, but it is beyond our discussion here. Some scholars [13,30,31] adopt the to *p–k* mining approach to find the results, instead of setting the two thresholds.

### 5.3. Experimental results on real data

In addition to synthetic data, a real dataset *Foodmart* was tested in the experiments. The transactions were divided into 10 time periods and the time hierarchy was organized in three levels, with 1, 5 and 10 time periods, respectively. Fig. 5 shows the differences in the execution time needed by the two algorithms for different thresholds, varying from 0.6% to 0.7%. The experimental results show that the algorithm *TPPF* performed much better than *TP-HTAR* since the number of data scans of *TPPF* was much fewer than those of *TP-HTAR*. The results are an echo of Figs. 2 and 3.

For showing the performance of generating association rules, similar to Fig. 4, six sets with different minimum relative support values yielded by the proposed methods were adopted in the experiments, which are {0.6%, 0.62%, 0.64%, 0.66%, 0.68%, 0.7%}. Fig. 6 shows the experimental comparisons for rule generation, which delivers some discovery. First, the execution time for each set is very close to each other even using different minimum confidence values. The reason is the same as above. That is, the confidence checking time depends on the numbers of frequent itemsets generated, but not on the confidence thresholds. Second, the performance for smaller confidence value is worse than that for larger confidence values. The reason is that the former will get more rules and thus need more time to generate them out.

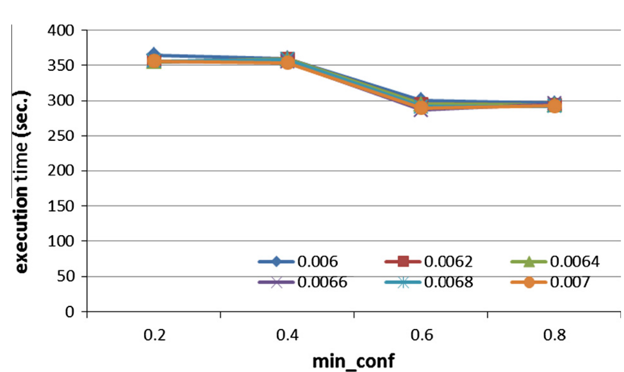


**Figure 6** The execution time of generating association rules under different minimum confidences on the real data.

## 6. Conclusions

In this paper, we introduce a new concept of temporal association rule mining with a hierarchy of time granules to find hierarchical temporal association rules in temporal databases, and we also present the effective approach (abbreviated as *TPPF*) to find such rules. In particular, an effective strategy is designed to predict the upper-bound of support values for itemsets. The strategy can be used to remove unpromising itemsets at an early stage in the process, and the proposed *TPPF* can effectively reduce the computational cost of scanning a temporal database. Experiments were also made, with results showing the proposed *TPPF* outperformed the other one *TP-HTAR* in reducing database scan and computational time.

The future research directions of this work are as follows. First, we will attempt to investigate the incremental problem of hierarchical temporal association rule mining. That is, based on this work, we will design a method to mine the new result without performing the whole mining procedure at database modification. Second, the optimal minimum support and confidence will be approximated by machine learning techniques. Third, actually, this work is the beginning of hierarchical temporal association rule mining. In the future, more efficient mining algorithms such as FP-growth will be adopted as the solutions to accelerate the mining process and more mining consideration such as utility mining will be studied to extend its applications.

## Acknowledgment

This research was supported by the Ministry of Science and Technology of the Republic of China under contract MOST103-2221-E-390-014 and MOST 104-2221-E-230-019.

## References

[1] R. Agrawal, R. Srikant, Fast algorithm for mining association rules, in: The International Conference on Very Large Data Bases, 1994, pp. 487–499.
[2] R. Agrawal, R. Srikant, Mining sequential patterns, in: The 11th International Conference on Data Engineering, 1995, pp. 3–14.
[3] R. Agrawal, R. Srikant, Q. Vu, Mining association rules with item constraints, in: The 3th International Conference on Knowledge Discovery in Databases and Data Mining, Newport Beach, California, 1997.
[4] J.M. Ale, G.H. Rossi, An approach to discovering temporal association rules, in: The 2000 ACM Symposium on Applied Computing, 2000, pp. 294–300.
[5] C.Y. Chang, M.S. Chen, C.H. Lee, Mining general temporal association rules for items with different exhibition periods, in: The Third IEEE International Conference on Data Mining, 2002, pp. 59–66.
[6] C.H. Chen, G.C. Lan, T.P. Hong, S.B. Lin, Mining fuzzy temporal association rules by item lifespans, Appl. Soft Computing 41 (2016) 265–274.
[7] M. Chen, S.C. Chen, M.L. Shyu, Hierarchical temporal association mining for video event detection in video databases, in: The Second IEEE International Workshop on Multimedia Databases and Data Management in The IEEE 23rd International Conference on Data Engineering, 2007, pp. 137–145.
[8] Y.L. Chen, K. Tang, R.J. Shen, Y.H. Hu, Market basket analysis in a multiple store environment, Decision Support Syst. 40 (2) (2005) 339–354.
[9] C.J. Chu, Vincent S. Tseng, T. Liang, Mining temporal rare utility itemsets in large databases using relative utility thresholds, Int. J. Innovative Computing, Inf. Control 4 (8) (2008) 2775–2792.
[10] G. Fang, Y. Wu, A general framework based on composite granules for mining association rules, Int. J. Artificial Intelligence Tools 23 (5) (2014).
[11] A.A. Galib, B. Dhaka, M. Alam, N. Hossain, R.M. Rahman, Stock trading rule discovery based on temporal data mining, in: The International Conference on Electrical and Computer Engineering, 2010, pp. 566–569.
[12] P. Hu, M. Huang, P. Xu, W. Li, A.K. Usadi, X. Zhu, Finding nuggets in IP portfolios: core patent mining through textual temporal analysis, in: The 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 1819–1823.
[13] F.-V. Philippe, Cheng-Wei Wu, Vincent S. Tseng, Mining top-K association rules, in: Canadian Conference on AI, 2012, pp. 61–73.
[14] IBM Quest Data Mining Project, Quest synthetic data generation code, <http://www.almaden.ibm.com/cs/quest/syndata.html>, 1996.
[15] A. Julea, N. Méger, C. Rigotti, E. Trouvé, R. Jolivet, P. Bolon, Efficient spatio-temporal mining of satellite image time series for agricultural monitoring, Trans. Mach. Learning Data Mining 5 (1) (2012) 23–44.
[16] G.C. Lan, C.H. Chen, T.P. Hong, S.B. Lin, A fuzzy approach for mining general temporal association rules in a publication database, in: The Eleventh International Conference on Hybrid Intelligent Systems, 2011, pp. 611–615.
[17] G.C. Lan, C.H. Chen, T.P. Hong, S.B. Lin, Mining fuzzy temporal association rules with effective lifespans of items, in: The Cross-Strait Conference on Information Science and Technology and iCube, 2011.
[18] G.C. Lan, T.P. Hong, Vincent S. Tseng, Discovery of high utility itemsets from on-shelf time periods of products, Expert Syst. Appl. 38 (5) (2011) 5851–5857.
[19] G.C. Lan, T.P. Hong, P.S. Wu, S. Tsumoto, Mining hierarchical temporal association rules in a publication database, in: The 12th IEEE International Conference on Cognitive Informatics & Cognitive Computing, 2013, pp. 503–508.
[20] C.H. Lee, C.R. Lin, M.S. Chen, On mining general temporal association rules in a publication database, in: The 2001 IEEE International Conference on Data Mining, 2001, pp. 337–344.
[21] Y. Li, P. Ning, X.S. Wang, S. Jajodia, Discovering calendar-based temporal association rules, Data Knowledge Eng. 44 (2) (2003) 193–218.
[22] B. Ozden, S. Ramaswamy, A. Silberschatz, Cyclic association rules, in: The 14th International Conference on Data Engineering, 1998, pp. 12–421.
[23] E.E. Papalexakis, T. Dumitras, D.H. Chau, B.A. Prakash, C. Faloutsos, Spatio-temporal mining of software adoption & penetration, in: The 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2013, pp. 878–885.
[24] D. Patnaik, M. Marwah, R. Sharma, N. Ramakrishnan, Temporal data mining approaches for sustainable chiller management in data centers, ACM Trans. Intelligent Syst. Technol. 2 (4) (2011) 34.
[25] A. Rajak, K. Saxena, Modeling clinical database using time series based temporal mining, Int. J. Comput. Theory Eng. 2 (2) (2010) 1793–8201.
[26] M.M. Rashid, I. Gondal, J. Kamruzzaman, Mining associated patterns from wireless sensor networks, IEEE Trans. Comput. 64 (7) (2015) 1998–2011.

[27] J.F. Roddick, M. Spiliopoilou, A survey of temporal knowledge discovery paradigms and methods, IEEE Trans. Knowledge Data Eng. 14 (4) (2002) 750–767.

[28] R. Srikant, R. Agrawal, Mining generalized association rules, in: The 21st International Conference on Very Large Data Bases, Zurich, Switzerland, 1995, pp. 407–419.

[29] Vincent S. Tseng, C.J. Chu, T. Liang, Efficient mining of temporal high utility itemsets from data streams, The ACM KDD Workshop on Utility-Based Data Mining, 2006, pp. 1105–1117.

[30] J. Wang, J. Han, Y. Lu, P. Tzvetkov, TFP: an efficient algorithm for mining top-K frequent closed itemsets, IEEE Trans. Knowledge Data Eng. (2005) 652–664.

[31] C.-W. Wu, B.-E. Shie, V.S. Tseng, P.S. Yu, Mining top-K high utility itemsets, in: ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2012, pp. 78–86.