Appl Intell (2014) 41:439–452
DOI 10.1007/s10489-014-0530-4

# An efficient approach for finding weighted sequential patterns from sequence databases

Guo-Cheng Lan · Tzung-Pei Hong · Hong-Yu Lee

Published online: 6 April 2014

**Abstract** Weighted sequential pattern mining has recently been discussed in the field of data mining. Different from traditional sequential pattern mining, this kind of mining considers different significances of items in real applications, such as cost or profit. Most of the related studies adopt the maximum weighted upper-bound model to find weighted sequential patterns, but they generate a large number of unpromising candidate subsequences. In this study, we thus propose an efficient approach for finding weighted sequential patterns from sequence databases. In particular, a tightening strategy in the proposed approach is proposed to obtain more accurate weighted upper-bounds for subsequences in mining. Through the experimental evaluation, the results also show the proposed approach has good performance in terms of pruning effectiveness and execution efficiency.

**Keywords** Data mining · Sequential pattern · Weighted sequential pattern · Weighted frequent patterns · Upper bound

G.-C. Lan
Department of Mathematics and Computer Sciences, Fuqling Branch of Fujian Normal University, Fuzhou, Fujian, China
e-mail: rrfoheiay@gmail.com

T.-P. Hong (✉) · H.-Y. Lee
Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan
e-mail: tphong@nuk.edu.tw

H.-Y. Lee
e-mail: staryculturesky@gmail.com

T.-P. Hong
Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan

## 1 Introduction

The main purpose of data mining on knowledge discovery is to extract useful rules or patterns from a set of data. In the field of data mining, sequential pattern mining [2] has been widely applied to trend analysis from a set of long-term event sequence data. The traditional sequential pattern mining, however, only considers the occurrence of items, and then it does not reflect any other factors, such as price or profit. Besides, the same significance is assumed for all items in a set of sequences. Thus, the actual significance of a pattern cannot be easily recognized. Some events with low-frequency are quite important, for example products with high-profit in a transaction database or attacked events in a long-term network. Such events may not be easily found by using traditional sequential pattern mining techniques. To handle this, Yun et al. proposed a new research issue, namely weighted sequential pattern mining [21], in which different weights were assigned to items by the importance of each item. In addition, Yun et al. designed an average-weight function to evaluate the weight value of a pattern in a sequence. Based on the average-weight function, Yun et al. [21] also developed an upper-bound model, in which the maximum weight among items in a sequence database was used as the upper-bound of weight value of each sequence, to construct a downward-closure property in the problem of weighted sequential pattern mining. However, although Yun et al.'s *WSpan* algorithm [21] can avoid information losing in mining, a large number of candidate subsequences were still generated due to the upper-bounds of overestimated weighted values for the candidates. It is thus a critical issue to develop a suitable model for weighted sequential pattern mining.

To address the above reasons, the work presents an effective model to reduce a large number of candidates for finding weighted sequential patterns in sequence databases. The major contributions of this work are summarized as follows.

1. This work proposes an effective sequence maximum weight (abbreviated as *SMW*) model to tighten upper-bounds of weighted supports for subsequences in mining. In addition, an efficient projection-based mining approach with the model, namely *IUA* (an improved upper-bound approach), is presented to speed up the execution efficiency in finding weighted sequential patterns.
2. Based on the *SMW* model, an effective pruning strategy in the proposed *IUA* approach is designed to reduce the number of unpromising subsequences in the recursive mining process, thus avoiding unnecessary evaluation. The efficiency in finding weighted sequential patterns can thus be raised.
3. Through a series of experimental evaluation, the results show that the number of weighted frequent upper-bound patterns is less than that needed by the traditional *WSpan* algorithm [21], when working on the synthetic datasets generated by the *IBM* data generator [12] and the real *Kosarak* dataset [11]. The experimental results also show the proposed algorithm executes faster than the *WSpan* algorithm.

The remaining parts of this paper are organized as follows. Some related works are briefly reviewed in Section 2. The problem to be solved and the definitions are described in Section 3. The proposed *IUA* algorithm with a pruning strategy for finding weighted sequential patterns from a set of sequence data is stated in Section 4. An example is given to illustrate the execution procedures of the proposed algorithm in Section 5. The experimental evaluation is showed in Section 6. Conclusions and future works are finally given in Section 7.

## 2 Review of related works

In this section, some related studies on weighted itemset mining and weighted sequential pattern mining are briefly reviewed.

### 2.1 Weighted frequent itemset mining

The main purpose of data mining in knowledge discovery is to extract desired rules or patterns in a set of data. One common type of data mining is to derive association rules from a transaction dataset, such that the presence of certain items in a transaction will imply the presence of some other items. To address this, Agrawal et al. proposed several mining algorithms based on the concept of large itemsets to find association rules from transaction data [1, 3–5]. The a priori algorithm on association-rule mining was the most well-known of existing algorithms. The process of association-rule mining could be divided into two main phases. In the first phase, candidate itemsets were generated and counted by scanning transaction data. If the count of an itemset in the transactions was larger than or equal to the pre-defined threshold value (called the minimum support threshold), the itemset was identified as a frequent one. Itemsets containing only one item were processed first. Frequent itemsets containing only single items were then combined to form candidate itemsets with two items. The above process was then repeated until no candidate itemsets were generated. In the second phase, association rules were derived from the set of frequent itemsets found in the first phase. All possible association combinations for each frequent itemset were formed, and those with calculated confidence values larger than or equal to a predefined threshold (called the minimum confidence threshold) were output as association rules.

An itemset in association-rule mining only considers the frequency of the itemset in databases, and the same significant are assumed for all items in the itemset. In reality, however, the importance of items in a database may be different according to different factors, such as profit and cost of items [17]. For example, LCD TVs may not have high frequency but is a high-profit product when compared to food or drink in a database. Thus, some useful item products may not be discovered by using traditional frequent itemset mining techniques. To handle the problem, Yun et al. then proposed a new research issue, called weighted itemset mining [19] to find weighted frequent itemsets in transaction databases. The weights of items in a database for weighted itemset mining could be flexibly given by users, and the average-weight function in Yu et al.'s study [20] was designed to evaluate the weight of an itemset in a transaction. Different from frequent itemsets with only consideration of frequency, the found itemsets with high-weight values might be used as managers' auxiliary information in terms of making decisions. However, the downward-closure property in association-rule mining cannot be kept in the problem of weighted frequent itemset mining with the average-weight function. To address this, Yun et al. proposed an upper-bound model to construct a new downward-closure property [19] which adopted the maximum weight of a database as the weight upper-bound of each transaction, the *FP-growth*-based was also developed to find weighted frequent itemsets in transaction databases, and the algorithm in their study had a good performance in terms of handling the problem of weighted frequent itemset mining. Afterward, several studies [7–10, 13, 22, 23] related to weighted itemset mining are based on or extended from Yun et al.'s proposed weighted function to deal with various data applications, such as improvement efficiency applications [13], data stream applications [7], incremental data applications [9], weighted sequential pattern mining [10, 22, 23], etc.

### 2.2 Weighted sequential pattern mining

In general, the transaction time (or time stamp) of each transaction for real-world applications is usually recorded in databases. The transactions can then be listed as a time-series data (called a sequence data) in the occurring time order of the transactions. To handle such data with time, a new issue, namely sequential pattern mining, was first developed to achieve the goal [2], and the three algorithms, *AprioriAll*, *AprioriSome*, and *DynamicSome*, were also proposed to find sequential patterns in a sequence data. However, the three algorithms, which were the level-wise techniques, had to execute multiple data scans to complete sequential pattern mining tasks. Afterward, several algorithms for sequential pattern mining were published to improve efficiency, such as *GSP* [18] and *PrefixSpan* [15]. Besides, the principle of sequential pattern mining [2] was applied to different domain applications, such as time-gap sequential pattern mining [19].

As mentioned in weighted itemset mining, the similar problem for the same significance of items also existed in sequential pattern mining. Afterward, some studies considered different significance of items in various applications, such as high utility sequential pattern mining [6], weighted sequential pattern mining, and so on. Since high utility sequential pattern mining [6] considered not only the transaction time order of items in sequences but also the quantities and profits of items, it is useful in supermarket promotion applications. Different from utility sequential pattern mining [6, 16], however, in some applications (e.g. stock trend analysis applications), the weighted sequential pattern mining is more suitable than high utility sequential pattern mining due to different activity significance, such as purchasing activity or at-home activity. Through the weight concept, some importance sequential patterns may be found when compared with the traditional sequential pattern mining and high utility sequential pattern mining. To deal with this, Yun et al. thus proposed a new research issue, named weighted sequential pattern mining [21] to find weighted sequential patterns from the a sequence database. Similarly, different weights were given items by referring to factors, such as their profits, their costs, or users' preferences, and then the actual importance of a pattern could be easily recognized when compared with the traditional sequential pattern mining. Different from the function in weighted itemset mining, the time factor was considered to develop a new average-weight function [21], and the new function could be applied to identify the weight value of a pattern in a sequence. Based on the function, however, the downward-closure property in traditional sequential pattern mining could not be kept on weighted sequential pattern mining. To address the problem, an upper-bound model [21], which the maximum weight in a sequence database was regarded as the upper-bound of each sequence, was directly derived from Yun et al.'s proposed model in weighted itemset mining [21]. However, it was observed that a huge amount of unpromising subsequences still had to be generated by using the traditional upper-bound model [21] for mining, and its performance was thus not good.

Based on the above reasons, this motivates our exploration of the issue of effectively and efficiently mining weighted sequential patterns from a set of sequences.

## 3 Problem statement and definitions

To describe the problem of weighted sequential pattern mining clearly, assume a sequence database is given in Table 1, in which each sequence consists of two features, sequence identification (*SID*) and items purchased (or events appeared). There are eight items in the sequences, respectively denoted as *A* to *H*. Also, assume the predefined weight value of each item is shown in Table 2.

For the formal definitions of weighted sequential pattern mining, a set of terms related to the problem of weighted sequential pattern mining is then defined as follows.

**Definition 1** An itemset $X$ is a subset of items, $X \subseteq I$. If $|X| = r$, the itemset $X$ is called an $r$-itemset. Here $I = \{i_1, i_2, ..., i_m\}$ is a set of items, which may appear in sequences. Note that the items in an itemset are sorted in alphabetical order of the items.

**Definition 2** A sequence $Seq$ is composed of a set of itemsets by time order of the itemsets, and the size of the sequence $Seq$, $|Seq|$, is the number of itemsets in $Seq$. In addition, if the number of items in a sequence, $l_{Seq}$, is $l$, the sequence $Seq$ with the length $l$ is called an $l$-sequence. For simplicity, here we assume that bracket of an itemset can be removed as there is only one item in the itemset.

**Table 1** The set of five sequences in this example

| SID | Sequences |
|---|---|
| $Seq_1$ | <BCB> |
| $Seq_2$ | <*DECHF*> |
| $Seq_3$ | <*ACF*(*DE*)*F*> |
| $Seq_4$ | <(*FG*)*H*> |
| $Seq_5$ | <(*CD*)*ACEF*> |

**Table 2** The weight table in this example

| Item | Weight |
|---|---|
| *A* | 0.1 |
| *B* | 0.15 |
| *C* | 0.2 |
| *D* | 0.3 |
| *E* | 0.45 |
| *F* | 0.55 |
| *G* | 0.65 |
| *H* | 0.95 |

**Definition 3** Let $\alpha$ and $\beta$ be two sequences, $<X_1, X_2, \ldots, X_n>$ and $<Y_1, Y_2, \ldots, Y_m>$. If there exist an integer $1\leq i_1 < \ldots < i_n < m$ such that $X_1 \subseteq Y_{i_1}, \ldots, X_n \subseteq Y_{i_n}$, the sequence $\alpha$ is called the sub-sequence of another sequence $\beta$, and the sequence $\beta$ is called the super-sequence of the sequence $\alpha$.

For example, the sequence $<ABC>$ is the sub-sequence of $<(A)(AB)(CD)>$, and the sequence $<(A)(AB)(CD)>$ is the super-sequence of $<ABC>$.

**Definition 4** A sequence database *SDB* is composed of a set of sequences. That is, $SDB = \{Seq_1, Seq_2, \ldots, Seq_y, \ldots, Seq_z\}$, where $Seq_y$ is the $y$-th sequence in *SDB*.

**Definition 5** The weight value of an item $i$, $w_i$, ranges from 0 to 1.

**Definition 6** The weight value of an itemset $X$, $w_X$, is the summation of weight values of all items in $X$ over the number of all items in $X$. That is,

$$w_X = \frac{\sum\limits_{i \in X \wedge X \subseteq I}^{|X|} w_i}{l_X},$$

where $l_X$ is the number of items in the itemset $X$.

For example, in Table 2, since the weight values of the two items in the itemset $\{AB\}$ are 0.05 and 0.15, respectively, and the number of items in $\{AB\}$ is 2, $w_{\{AB\}} = (0.05 + 0.15)/2 = 0.1$.

**Definition 7** The weight value of a subsequence $S$, $w_S$, is the summation of weight values of all itemsets in $S$ over the number of all itemsets in $S$. That is,

$$w_S = \frac{\sum\limits_{X \in Seq}^{|Seq|} w_X}{|S|},$$

where $|S|$ and $w_X$ are the number of itemsets in the subsequence $S$ and the weight value of the itemset $X$ in $S$, respectively.

For example, in Tables 1 and 2, since the fourth sequence $<(FG)H>$ includes two itemsets, $(FG)$ and $(H)$, and the weights of the two itemsets are 0.6 and 0.95, respectively. Then, $w_{<(FG)H>} = (0.6 + 0.95)/2 = 0.775$.

**Definition 8** The sequence maximum weight value of a sequence $S$, $smw_S$, is the maximal weight among all items in the sequence $S$.

**Definition 9** The total sequence maximum weight of a sequence database *SDB*, *tsmw*, is the summation of the sequence maximum weight values of all sequences in *SDB*. That is,

$$tsmw = \sum_{Seq_y \subseteq SDB} smw_y.$$

For example, in Table 1, the sequence maximum weights of the five sequences are 0.20, 0.95, 0.55, 0.95, and 0.55, respectively. Then, *tsmw* $= 0.20 + 0.95 + 0.55 + 0.95 + 0.55 = 3.20$.

**Definition 10** The weighted support value of a subsequence $S$, $wsup_S$, is the summation of the weight values of the subsequence $S$ in the sequences including $S$ in *SDB* over the total sequence maximum weight *tsmw* of *SDB*. That is,

$$w\text{sup}_S = \frac{\sum\limits_{S \subseteq Seq_y \wedge Seq_y \subseteq SDB} w_S}{tsmw}.$$

For example, in Table 1, the weight of the subsequence $<DF>$ is 0.425 $(= (0.3 + 0.55)/2)$ in accordance with the seventh definition, and it appears in the three sequences, $Seq_2$, $Seq_3$, and $Seq_5$. Besides, the total sequence maximum weight is 3.20. The weighted support of $<DF>$ is then calculated as $(0.425{+}0.425{+}0.425)/3.2$, which is 39.84 %.

**Definition 11** Let $\lambda$ be a pre-defined minimum weighted support threshold. A subsequence $S$ is called a weighted sequential pattern (abbreviated as *WS*) if $wsup_S \geq \lambda$.

Here it is observed that the downward-closure property in traditional pattern mining is not kept in the problem of weighted sequential pattern mining. Take item $D$ in Table 1 as an example. There are three sequences including the item $D$ in Table 1, and the weight of the item $D$ in Table 2 is 0.3. Then, the weighted support value of the sequence $<D>$ can be calculated as $(0.3 + 0.3 + 0.3)/3.2$, which is 28.13 %. If $\lambda = 30$ %, then the sequence $<D>$ is not a weighted sequential pattern, but its super-sequence $<DF>$4 is weighted sequential pattern. As this example describes, the problem of weighted sequential pattern mining is more difficult than the traditional sequential pattern mining. To efficiently handle this, in this study, we proposed an effective sequence maximum weight (abbreviated as *SMW*) model to reduce the number of unpromising subsequences and then speed up the execution efficiency in finding weighted sequential patterns. The relevant terms used in our proposed *SMW* model are defined as follows.

**Definition 12** The sequence-weighted upper-bound of a subsequence *S*, $swub_S$, is the sum of sequence maximum weights of the sequences including *S* in a sequence database over the total sequence maximum weight *tsmw* of the sequence database *SDB*. That is,

$$swub_S = \frac{\sum\limits_{S \subseteq Seq_y \wedge Seq_y \subseteq SDB} smw_y}{tsmw}.$$

For example, in Table 1, the item *D* appears in the three sequences, $Seq_2$, $Seq_3$, and $Seq_5$, and the sequence maximum weights of the three sequences are 0.95, 0.55, and 0.55, respectively. Then, $swub_{<D>} = 2.05/3.20 = 64.06\,\%$.

**Definition 13** Let λ be a pre-defined minimum weighted support threshold. A subsequence *S* is called a weighted frequent upper-bound pattern (abbreviated as *WFUB*) if $swub_S \geq \lambda$.

Based on the definitions above, a weighted sequential pattern considers the individual weights of items in sequence data. The problem to be solved in the paper is to efficiently find all the frequent weighted sequential patterns, which their actual weight values are larger than or equal to a predefined minimum weighted support threshold λ, in a given sequence database. The details of the proposed *IUA* algorithm are described in the next section.

## 4 The proposed algorithm

In this paper, a new projection-based mining algorithm is proposed to effectively handle the problem of finding weighted sequential patterns in a sequence database. The improved model and the pruning strategy used in the proposed algorithm are developed to help its execution. The improved upper-bound model is first described below.

### 4.1 The improved upper-bound model

A new weight upper-bound model is proposed here to enhance the traditional weight upper-bound model [21], thus tightening upper-bounds of weight values for patterns in the mining process. In the traditional upper-bound model [21], the maximum weight in a sequence dataset is used as the upper-bound of weight value for each sequence, and then the downward-closure property can be held on weighted sequential pattern mining. However, it is observed that the maximal weight in a sequence is also used to achieve the same goal. That is, the value can be regarded as the upper-bound of weight value for any subsequence in that sequence. To illustrate the completeness of the sequence maximum weight (abbreviated as *SMW*) model adopted, two lemmas are given below to prove that no weighted sequential patterns are skipped in any weighted sequential pattern mining case. First, Lemma 1 is stated to prove the downward-closure property of the proposed *SMW* model.

**Lemma 1** *The sequence-weighted upper-bound of a pattern x keeps the downward-closure property.*

*Proof* Let *x* be a weighted frequent upper-bound pattern and $d_x$ be the set of sequences containing *x* in a sequence database *SDB*. If *y* is a super-sequence of *x*, then *y* cannot exist in any sequence where *x* is absent. Therefore, the sequence-weighted upper-bound $swub_x$ of *x* is the maximum upper-bound of weight value of *y*. Accordingly, if $swub_x$ is less than a predefined minimum weighted support threshold, then *y* cannot be a weighted frequent upper-bound pattern. □

Next, Lemma 2 proves that all weighted sequential patterns in a database are included in the set of weighted frequent upper-bound patterns.

**Lemma 2** *For a sequence database SDB and a predefined minimum weighted support threshold, the set of weighted sequential patterns WS is a subset of weighted frequent upper-bound patterns WFUB.*

*Proof* Let *x* be a weighted sequential pattern. According to Definitions 12 and 13, the actual weighted support $aws_x$ of *x* must be less than or equal to its sequence-weighted upper-bound $swub_x$. Accordingly, if *x* is a weighted sequential pattern, then it must be a weighted frequent upper-bound pattern. As a result, *x* is a member of the set *WFUB*. □

Based on Lemmas 1 and 2, it can be known that all weighted sequential patterns in a sequence database can be discovered, and the proposed model can be used to effectively tighten upper-bounds of weight values for subsequences when compared with the traditional upper-bound model [21]. Below, an example is given to illustrate how to improve upper bounds of weight values for subsequences by using the model.

For example, according to the traditional upper-bound model [21], the maximum weight value in Table 1 is 0.95, and the value of 0.95 is regarded as upper-bound of any subsequence in each sequence in the dataset. Take item *C* as an example, the item appears in the four sequences, $S_1$, $S_2$, $S_3$and $S_5$, and the weight upper-bounds of the four sequences are all 0.95. Then, the upper-bound of weight value for *C* can be calculated as $(0.95 + 0.95 + 0.95 + 0.95)$, which is 3.8.

Based on the proposed upper-bound model, however, the upper-bound of weight value for item *C* can thus be further tightened. First, the maximal weight in a sequence has to be found. Take the first sequence $Seq_1$:<*BCB*> in Table 1 as an example. The sequence includes two distinct items, *B* and *C*, and their weights are 0.15 and 0.20, respectively. The maximal weight value between the weights of *B* and *C* is 0.20, and then the value of 0.20 is regarded as the upper-bound of weight value for the sequence, $Seq_1$. All the other four sequences in Table 1 can be similarly processed, and the results for maximum weight values of the five sequences are found as 0.20, 0.95, 0.55, 0.95, and 0.55, respectively. The sequence-weighted upper-bound of *C* can be then calculated as $0.20 + 0.95 + 0.55 + 0.55$, which is 2.25.

As this example illustrates, the value (= 2.25) obtained by the proposed model is obviously less than that (= 3.8) obtained by the traditional model. Hence, the sequence-weighted upper-bounds of subsequences in the mining can be effectively tightened by using the proposed model when compared with the traditional upper-bound model [21].

### 4.2 The pruning strategy for unpromising items

In this section, a simple pruning strategy based on both of the proposed model and the projection-based technique is designed to effectively reduce the number of unpromising subsequences for mining. According to Lemmas 1 and 2, the downward-closure property for the problem of weighted sequential pattern mining can then be held by the proposed model. Based on the model, any sub-pattern of a weighted frequent upper-bound pattern must be a weighted frequent upper-bound pattern. On the contrary, if there exists a weighted infrequent upper-bound sub-pattern for a pattern, then the pattern must not be a weighted frequent upper-bound pattern; indeed, the pattern also must not be a weighted sequential pattern. In this case, the pattern can be skipped early since it is impossible to be a weighted frequent upper-bound pattern. In the study, the above concept is applied in the pruning strategy to reduce unpromising subsequences in the recursive process.

The procedure of the strategy in the proposed projection-based algorithm is described as follows. First, when all the weighted frequent upper-bound *r*-patterns with *r* items are found, all items in the set of weighted frequent upper-bound *r*-patterns are gathered as the pruning information for each a prefix *r*-pattern to be processed. Next, the additional $(r+1)$-th item of each generated $(r+1)$-pattern in the next recursive process will be checked for whether it appears in the set of gathered items. If it is, the generated $(r + 1)$-subsequence will be put in the set of $(r + 1)$-sequences; otherwise, it is pruned. An example is given below to illustrate the pruning of unpromising subsequences in the recursive process.

For example, an assumed sequence is <(*AB*)*DEF*>, where symbols represent items, and assume the two patterns <(*AB*)> and <*AC*> are included in the current set of weighted frequent upper-bound *2*-patterns $WFUB_{2,<A>}$ with <*A*> as their prefixes. In this case, only the three distinct items, *A*, *B*, and *C*, are gathered from the set $WFUB_{2,<A>}$ as the pruning information, and the next prefix subsequence to be processed is the pattern <(*AB*)>. For the sequence <(*AB*)*DEF*>, the sequence is the projected sequences of <(*AB*)>, but the three items in the sequence, *D*, *E*, and *F*, are not shown in the set of gathered items. Thus, the three items can be removed from the sequence, and the modified sequence is then <(*AB*)>. The reason is that the super-patterns consisting of the three items and the prefix <*A*> must not be weighted frequent upper-bound patterns. Moreover, since the number of items kept in the modified sequence is less than the value of 3, which is the number of items in the *3*-patterns to be generated, the modified sequence can be removed directly from the projected sequences of <(*AB*)>. As the example describes, the strategy can be applied to effectively speed up the efficiency of the proposed algorithm in terms of handling the problem of weighted sequential pattern mining.

As mentioned previously, the superiority of our proposed approach can be explained as follows. Yun et al.'s approach developed an upper-bound model [21], in which the maximum weight in a sequence database was regarded as the upper-bound of each sequence to keep the downward-closure property in the problem of weighted sequential pattern mining. Afterward, most of the studies [7–10] related to weighted sequential pattern mining adopted the upper-bound model [21] to handle various kinds of weighted data mining issues, such as the problems of weighted sequential pattern mining in stream environments [7], of weighted sequential pattern mining with the consideration of dynamic item weights in different time periods [8], of the incremental weighted sequential pattern mining [9], and of sequential pattern mining with the consideration of different time-interval weights [10].

However, the item with the maximum weight value in a sequence database does not always appear in each sequence. Due to this reason, the proposed *SMW* model uses the maximum weight in a sequence as the upper-bound of any subsequences in that sequence to avoid information lost, thus tightening the upper-bounds of weighted supports for subsequences in mining. Based on the proposed *SMW* model, this work also presents a projection-based *IUA* approach to deal with this problem, and an effective pruning strategy is also embedded to further obtain lower upper-bounds of weighted supports of subsequences in the recursive mining process. Hence, the proposed *IUA* approach based on the improved model could speed up the execution efficiency in mining when compared with Yun et al.'s approach [21].

### 4.3 The proposed projection-based mining algorithm with the improved model

The proposed *IUA* algorithm based on the improved upper-bound model is stated below.

Input: A set of items, each with a weight value; a sequence database *SDB*, in which each sequence includes a subset of items; a minimum weighted support threshold λ.

Output: A final set of weighted sequential patterns, *WS*.

Step 1: For each sequence $Seq_y$ in *SDB*, find the sequence maximum weight $smw_y$ of the sequence $Seq_y$ as:

$$smw_y = max\left\{w_{y1}, w_{y2}, \ldots, w_{yj}\right\},$$

where $w_{yj}$ is the weight value $w(i_{yj})$ of the *j*-th item $i_{yj}$ in $Seq_y$.

Step 2: Find the total sequence maximum weight *tsmw* of the sequence database *SDB* as:

$$tsmw = \sum_{Seq_y \subseteq SDB} smw_y.$$

Step 3: For each item *I* in *SDB*, do the following substeps.

(a) Calculate the sequence-weighted upper-bound $swub_I$ of the item *I* as:

$$swub_I = \frac{\sum_{S \subseteq Seq_y \wedge Seq_y \subseteq SDB} smw_y}{tsmw}.$$

where $smw_y$ is the sequence maximum weight of each $Seq_y$ in *SDB*. Note that an item in a sequence may appear multiple times, but the frequency of the item in the sequence $Seq_y$ has to be seen as 1.

(b) Calculate the actual weighted support count $wsup_I$ of the item *I* as:

$$w\sup_{I} = \frac{\sum_{I \subseteq Seq_y \wedge Seq_y \subseteq SDB} w(I)}{tsmw}.$$

where *w*(*I*) is the weight value of the item *I* in *SDB*.

Step 4: For each candidate *1*-subsequence *I* in *SDB*, do the following substeps.

(a) If the sequence-weighted upper-bound value $swub_I$ of the *1*-subsequence *I* is larger than or equal to the minimum weighted support threshold λ, put it in the set of weighted frequent upper-bound *1*-patterns, $WFUB_1$.

(b) If the actual weighted support count value $wsup_I$ of the *1*-subsequence *I* is larger than or equal to the minimum weighted support threshold λ, put it in the set of weighted sequential *1*-patterns, $WS_1$.

Step 5: Set $r = 1$, where *r* represents the number of items in the processed subsequences.

Step 6: Gather the items appearing in the set of $WFUB_1$, and put them in the set of possible items, $PI_r$.

Step 7: For each *y*-th sequence $Seq_y$ in *SDB*, do the following substeps.

(a) Get each item *I* located after *x* in $Seq_y$.

(b) Check whether *I* appears in $PI_r$ or not. If it does, then keep the item *I* in the sequence $Seq_y$; otherwise, remove the item *I* from $Seq_y$.

(c) If the number of items kept in the modified sequence $Seq_y$ is less than the value ($= r + 1$), then remove the modified sequence $Seq_y$ from *SDB*; otherwise, kept it in *SDB*.

Step 8: Process each item *I* in the set of $WFUB_1$ in the alphabetical order by the following substeps.

(a) Find the relevant sequences including *I* in *SDB*, and put the sequences in the set of projected sequences $sdb_I$ of the item *I*.

(b) Find all the weighted sequential patterns with *I* as their prefix item by the *Finding-WS(I, $sdb_I$, r)* procedure. Let the set of returned weighted sequential patterns be $WS_I$.

Step 9: Output the set of weighted sequential patterns in all the $WS_I$.

After STEP 9, all the weighted sequential patterns are found. The *Finding-WS(x, $sdb_x$, r)* procedure finds all the weighted sequential patterns with the *r*-pattern *x* as their prefix patterns and is stated as follows.

The Finding-WS(x, $sdb_x$, r) procedure:

Input: A prefix r-pattern x and its corresponding projected sequences $sdb_x$.

Output: The weighted sequential patterns with x as its prefix pattern.

Pstep 1: Initialize the temporary sequence $TS_x$ table as an empty table, in which each tuple consists of three fields: sequence, sequence-weighted upper-bound (*swub*) of the sequence, and the actual weighted support (*wsup*) of the sequence.

Pstep 2: For each *y*-th sequence $Seq_y$ in $sdb_x$, do the following substeps.

(a) Get each item I located after x in Seqy, and then generate the ($r$ + 1)-subsequence S composed of the prefix r-pattern x and I; put the new ($r$ + 1)-subsequences in the temporary subsequence table. Here if the subsequence S has not appeared in the temporary subsequence table, then put it in the table; otherwise, omit the subsequence S.

(b) For each unique ($r$ + 1)-subsequence in the temporary set of subsequences, add the sequence maximum weight smwy of the sequence Seqy and the weight w(S) of the subsequence S in the corresponding fields in the TSx table.

Pstep 3: For each ($r$ + 1)-subsequence in the $TS_x$ table, do the following substeps.

(a) If the sequence-weighted upper-bound $swub_S$ of the ($r$ + 1)-subsequence $S$ is larger than or equal to the minimum weighted support threshold λ, put it in the set of weighted frequent upper-bound ($r$ + 1)-patterns with the $x$ as their prefix subpatterns, $WFUB_{(r+1),x}$.

(b) If the actual weighted support $wsup_S$ of the ($r$ + 1)-pattern $S$ is larger than or equal to the minimum weighted support threshold λ, put it in the set of weighted sequential ($r$ + 1)-patterns, $WS_{(r+1),x}$.

Pstep 4: Acquire the items appearing in the set of $WFUB_{(r+1),x}$ of $x$, and put them in the set of possible items, $PI_{(r+1),x}$.

Pstep 5: Set $r = r + 1$, where $r$ represents the number of items in the processed subsequences.

Pstep 6: For each *y*-th sequence $Seq_y$ in $sdb_x$, do the following substeps.

(a) Check whether each item $I$ in $Seq_y$ appears in $PI_{r,x}$ or not. If it does, then keep the item $I$ in $Seq_y$; otherwise, remove the item $I$ from in $Seq_y$.

(b) If the number of items kept in the modified sequence $Seq_y$ is less than $r$ + 1, remove the modified sequence $Seq_y$ from $sdb_x$; otherwise, keep it in $sdb_x$.

Pstep 7: Process each pattern $S$ in the set of $WFUB_r$ in the alphabetical order by the following substeps.

(a) Find the relevant sequences including $S$ from $sdb_x$, and then put the sequences including $S$ in the set of projected sequences $sdb_S$ of $S$.

(b) Find all weighted sequential patterns with $S$ as their prefix pattern by the *Finding-WS(S, $sdb_S$, $r$ + 1)* procedure. Let the set of returned frequent weighted sequential patterns be $WS_S$.

Pstep 8: Return the set of weighted sequential patterns in all the $WS_x$.

## 5 An example of IUA

In the section, a simple example is given to illustrate how to find weighted sequential patterns from a sequence database by the proposed *IUA* algorithm. Assume there are five sequences in a sequence database, as shown in Table 1, and eight items in the sequences, respectively denoted as $A$ to $H$. In addition, assume the individual weights of the eight items are given in Table 2. In this example, the minimum weighted support threshold$\lambda$is set as 30 %. The detailed process of the proposed algorithm is then stated below.

Step 1: The sequence maximum weight for each sequence in *SDB* can be found. Take the first sequence $Seq_1$ in Table 1 as an example. The sequence $Seq_1$ includes three items, $B$, $C$, and $B$, and their weight values are 0.15, 0.20, and 0.15, respectively. The maximum value for the weight values is 0.20, and the value is regarded as the sequence maximum sequence weight of the sequence $Seq_1$. The same process can be done for all the other four sequences in Table 1. The results for the sequence maximum weights of all sequences are showed in Table 3.

Step 2: According to sequence maximum weight (*smw*) of each sequence in Table 3, the total sequence maximum weight (*tsmw*) can be calculated as

**Table 3** The sequence maximum weights of the five sequences in this example

| SID | Sequences | smw$_y$ |
|---|---|---|
| 1 | *<BCB>* | 0.20 |
| 2 | *<DECHF>* | 0.95 |
| 3 | *<ACF(DE)F>* | 0.55 |
| 4 | *<(FG)H>* | 0.95 |
| 5 | *<CDACEF>* | 0.55 |

**Table 4** The sequence-weighted upper-bounds and the weighted supports of all *1*-subsequences in this example

| Subsequence | swub | wsup |
|---|---|---|
| <A> | 34.37 % | 6.25 % |
| <B> | 6.25 % | 4.68 % |
| <C> | 70.31 % | 25 % |
| <D> | 64.06 % | 28.12 % |
| <E> | 64.06 % | 42.18 % |
| <F> | 93.75 % | 62.5 % |
| <G> | 29.68 % | 20.31 % |
| <H> | 59.37 % | 59.37 % |

0.20 + 0.95 + 0.55 + 0.95 + 0.55, which is 3.20.

Step 3: The sequence-weighted upper-bound (*swub*) and weighted support (*wsup*) of each possible item in *SDB* are found simultaneously. Take item *A* in Table 3 as an example. Item *A* appears in the two sequences, $Seq_3$ and $Seq_5$, and both the sequence maximum weights of the two sequences are 0.55. In addition, the weight of item *A* in Table 2 is 0.10, and the total sequence maximum weight *tsmw* is 3.20. The sequence-weighted upper-bound $swub_{<A>}$ of the item *A* can be then calculated as (0.55 + 0.55) / 3.2, which is 34.37 %, and its weighted support $wsup_{<A>}$ can be calculated as (0.10 + 0.10) / 3.2, which is 6.25 %. All the other possible items in *SDB* can be processed in the same fashion. The results for the sequence-weighted upper-bounds and the weighed supports of all possible *1*-subsequences in *SDB* are showed in Table 4.

Step 4: The weighted frequent upper-bound *1*-patterns ($WFUB_1$) and the weighted sequential *1*-patterns ($WS_1$) in Table 4 can be found simultaneously. Take the *1*-subsequence *<D>* in Table 4 as an example. The sequence-weighted upper-bound and the weighted support values of *<D>* in Table 4 are found as 64.06 % and 28.12 %, respectively. Since the sequence-weighted upper-bound of *<D>* is larger than or equal to the minimum weighted support threshold$\lambda$(= 30 %), *<D>* is a weighted frequent upper-bound *1*-pattern. But, *<D>* is not a weighted sequential pattern due to its weighted support (= 28.12 %). The other seven *1*-subsequences in Table 4 can be processed in the same way. After the step, the set of the weighted frequent upper-bound *1*-patterns ($WFUB_1$) includes *<A>*, *<C>*, *<D>*, *<E>*, *<F>*, and *<H>*, and only the three *1*-subsequences, *<E>*, *<F>*, and *<H>* are put in the set of the weighted sequential *1*-patterns ($WS_1$), as shown in Tables 5 and 6.

**Table 5** The set of the weighted frequent upper-bound *1*-patterns in the example

| Subsequence | swub |
|---|---|
| <A> | 34.37 % |
| <C> | 70.31 % |
| <D> | 64.06 % |
| <E> | 64.06 % |
| <F> | 93.75 % |
| <H> | 59.37 % |

**Table 6** The set of the weighted sequential *1*-patterns in the example

| Subsequence | wsup |
|---|---|
| <E> | 42.18 % |
| <F> | 62.5 % |
| <H> | 59.37 % |

Step 5: The variable *r* is initially set to 1, where *r* represents the number of items in the subsequences to be processed.

Step 6: In this example, the six items, *A*, *C*, *D*, *E*, *F*, and *H*, are collected from the six *1*-patterns in Table 5, and the possible items are then denoted as $PI_1$.

Step 7: For each sequence in Table 3, the items not appearing in the set of $PI_1$ are removed from the sequence. Take the first sequence $Seq_1$ in Table 3 as an example. The first sequence includes the three items, *B*, *C*, and *B*, and the sequence maximum weight of $Seq_1$ is 0.2. In this example, since the first and the third items in $Seq_1$ are not shown in the set of $PI_1$, only the item *C* in $Seq_1$ can be kept, and then the sequence is modified as *<C>*. The sequence maximum weight of the modified sequence is still 0.20. However, the modified sequence *<C>* can be removed from Table 3 because no 2-subsequences can be generated from the sequence. All the other four sequences in Table 3 can similarly be processed. The results for all the modified sequences and their sequence maximum weight values are showed in Table 7.

**Table 7** All the modified sequences with the sequence maximum weights in this example

| Sequences | $smw_y$ |
|---|---|
| *<DECHF>* | 0.95 |
| *<ACF(DE)F>* | 0.55 |
| *<FH>* | 0.95 |
| *<CDACEF>* | 0.55 |

**Table 8** The sequence-weighted upper-bound and the actual weighted support values of all 2-subsequences with prefix <*A*> in this example

| Subsequence | swub | wsup |
|---|---|---|
| <AC> | 34.37 % | 9.37 % |
| <AD> | 17.18 % | 6.25 % |
| <AE> | 34.37 % | 17.18 % |
| <AF> | 34.37 % | 20.31 % |

**Table 10** The final set of all the weighted sequential patterns (*WS*) in this example

| Pattern | wsup |
|---|---|
| <*E*> | 42 % |
| <*F*> | 63 % |
| <*H*> | 59 % |
| <*CF*> | 33 % |
| <*DF*> | 35 % |

Step 8: Each *1*-pattern in the set of $WFUB_1$ is sequentially processed in alphabetical order of the patterns. The variable *r* is initially set to 1, where *r* represents the number of items in the *r*-subsequences to be processed. The *1*-pattern <*A*> in the set of $WFUB\ _1$ is first processed for the example. Since the two sequences, $Seq_2$ and $Seq_4$, contain <*A*> in Table 7, the three sequences, <*ACF*(*DE*)*F*> and <*CDACEF*>, are projected and put in the projected sequences $sdb_{<A>}$ of <*A*>. Note that only the items located after the item *A* for each sequence in $sdb_{<A>}$ are kept. Take the second sequence <*CDACEF*> in $sdb_{<A>}$ as an example. Since only the three items, *C*, *E* and *F*, are located after item *A* in the sequence, the projected sequence is then <*ACEF*>. After this, $sdb_{<A>}$ includes the following two projected sequences, <*ACF*(*DE*)*F*> and <*ACEF*>, and their sequence maximum weights are 0.55 and 0.55, respectively.

Next, all the weighted sequential patterns with the prefix <*A*> are found by using the *Finding-WS*(*x*, $sdb_x$, *r*) procedure with the parameters *x* = <*A*> and *r* = 1. The details of the *Finding-WS*(*x*, $sdb_x$, *r*) procedure are described below.

Pstep 1: The temporary sequence table, $TS_{<A>}$, is initialized as an empty table, in which each tuple consists of three fields: subsequence, sequence-weighted upper-bound (*swub*) of the subsequence, and actual weighted support (*wsup*) of the subsequence.

Pstep 2: For each projected sequence in $sdb_{<A>}$, all possible 2-subsequences with the prefix <*A*> in it are produced. Take the second projected sequence <*ACEF*> as an example. Three unique 2-subsequences, <*AC*>, <*AE*> and <*AF*>, can be generated from the sequence <*ACEF*>, and the three subsequences are put in the $TS_{<A>}$ table. In addition, the weight values of the three subsequences are 0.15, 0.275 and 0.325, and the sequence maximum weight (= 0.55) of the sequence are put in the suitable field values of the 2-subsequences in the $TS_{<A>}$ table. The other two sequences in $sdb_{<A>}$ can be similarly processed. The results for all 2-subsequences with the prefix <*A*> in $sdb_{<A>}$ are shown in Table 8.

**Table 9** All the modified sequences in $sdb_{<A>}$ in this example

| Sequences | $smw_y$ |
|---|---|
| <*ACFEF*>4 | 0.55 |
| <*ACEF*>4 | 0.55 |

Pstep 3: As mentioned in STEP 4, the weighted frequent upper-bound 2-patterns ($WFUB_{2,<A>}$) and the weighted sequential 2-patterns ($WS_{2,<A>}$) with the prefix <*A*> in Table 8 can be found simultaneously. After the step, the three 2-subsequences, <*AC*>, <*AE*> and <*AF*> are put in the set of $WFUB_{2,<A>}$, and none of the subsequences in Table 8 is put in the set of $WS_{2,<A>}$.

Pstep 4: In this example, only the four items, *A*, *C*, *E*, and *F*, are collected from the set of $WFUB_{2,<A>}$ with the prefix <*A*>, and they are then denoted as $PI_{2,<A>}$.

Pstep 5: The value of the variable *r* is updated as 2.

Pstep 6: The items not appearing in each sequence in $sdb_{<A>}$ are removed from the sequence, as mentioned in STEP 7. After that, the results for the modified sequences in $sdb_{<A>}$ are shown in Table 9.

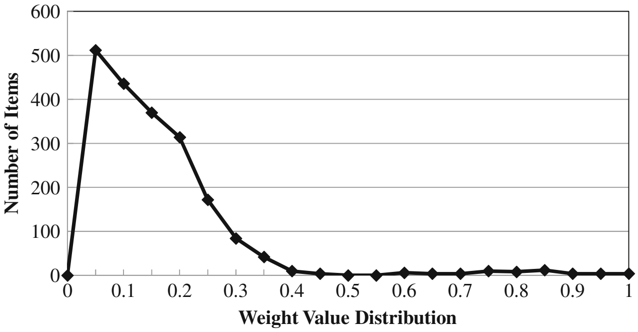


**Fig. 1** The weight-value distribution in the generated sequence datasets

**Table 11** Data characteristics

| Dataset | S | N | D |
|---|---|---|---|
| S8T6I2N2KD200K | 8 | 2,000 | 200 K |
| SXT6I2N2KD200K | 4∼12 | 2,000 | 200K |
| S8T6I2N2KDXK | 8 | 2,000 | 100K∼500K |
| Kosarak | 4.23 | 41,270 | 990,002 |

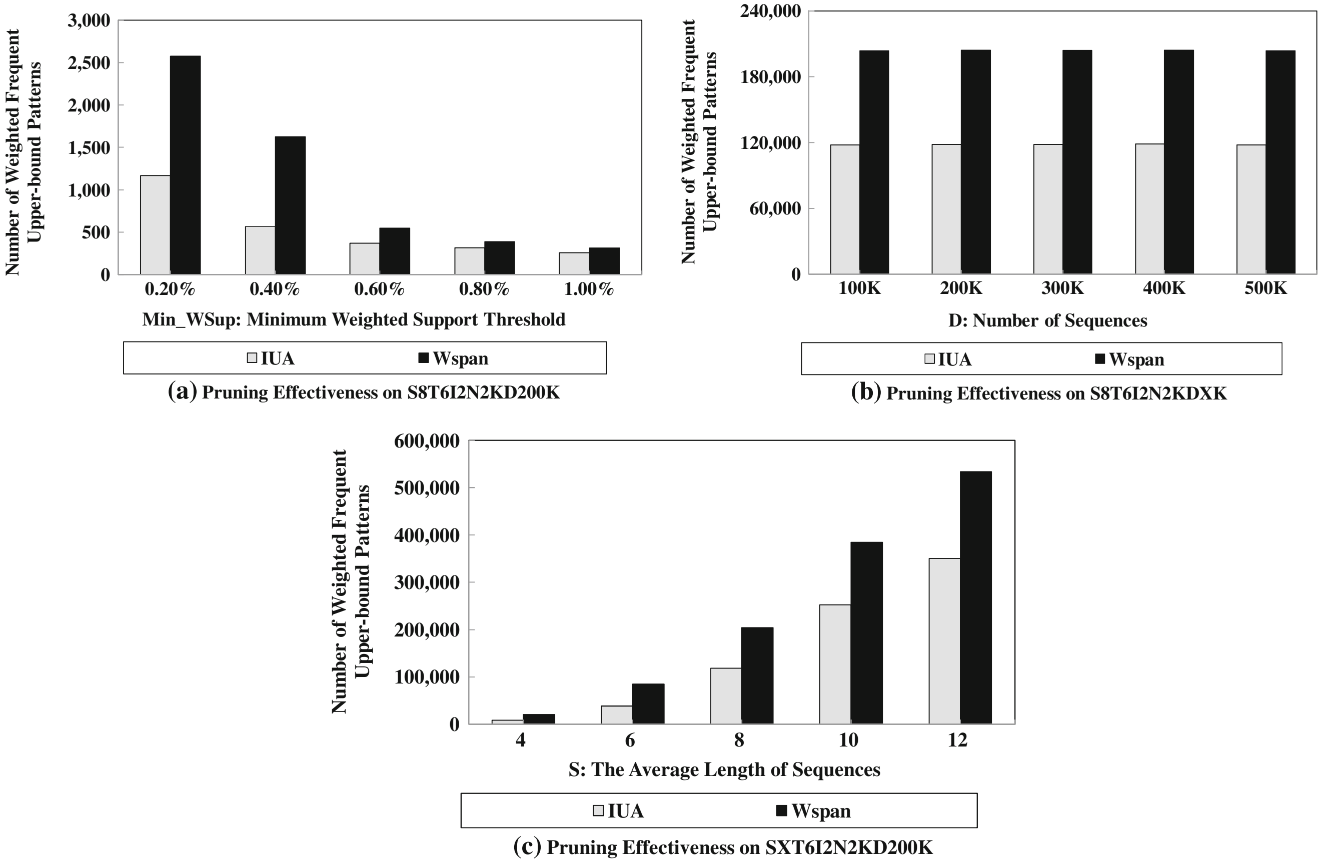
Pstep 7: Each 2-pattern in the set of $WFUB_{2,<A>}$ is processed in alphabetical order of the patterns. The 2-pattern *<AC>* in the set of $WFUB_{2,<A>}$ is thus processed first. The first and second projected sequences in $sdb_{<A>}$, *<ACFEF>* and *<ACEF>*, are put in $sdb_{<AC>}$, and their sequence maximum weights are all 0.55. Next, the weighted sequential patterns with the prefix *<AC>* are then found by recursively invoking the *Finding-WS*($x$, $sdb_x$, $r$) procedure with the parameters $x$ =*<AC>*, $sdb_x = sdb_{<AC>}$ and $r = 2$. The above process is recursively executed until all the *1*-patterns in $WFUB_1$ have been processed. All the weighted sequential patterns in this example are then found, as shown in Table 10.

Pstep 8: In this example, the five weighted sequential patterns in Table 10 are output to users as auxiliary information in terms of making decisions.

As shown in this example, based on the proposed sequence maximum weight (*SMW*) model, the pruning strategy can be effectively used to tighten the upper-bounds of weighted supports of subsequences and then reduce a lot of unpromising subsequences in the recursive process when compared with traditional upper-bound model. The execution efficiency can thus be improved in finding weighted sequential patterns.

## 6 Experimental evaluation

A series of experiments were conducted to compare the performance of the proposed improved upper-bound approach (abbreviated as *IUA*) and the traditional weighted sequential pattern mining approach (named *WSpan*) [21] with different parameter values. They were implemented in J2SDK 1.6.0 and executed on a PC with 3.30 GHz CPU and 4 GB memory.

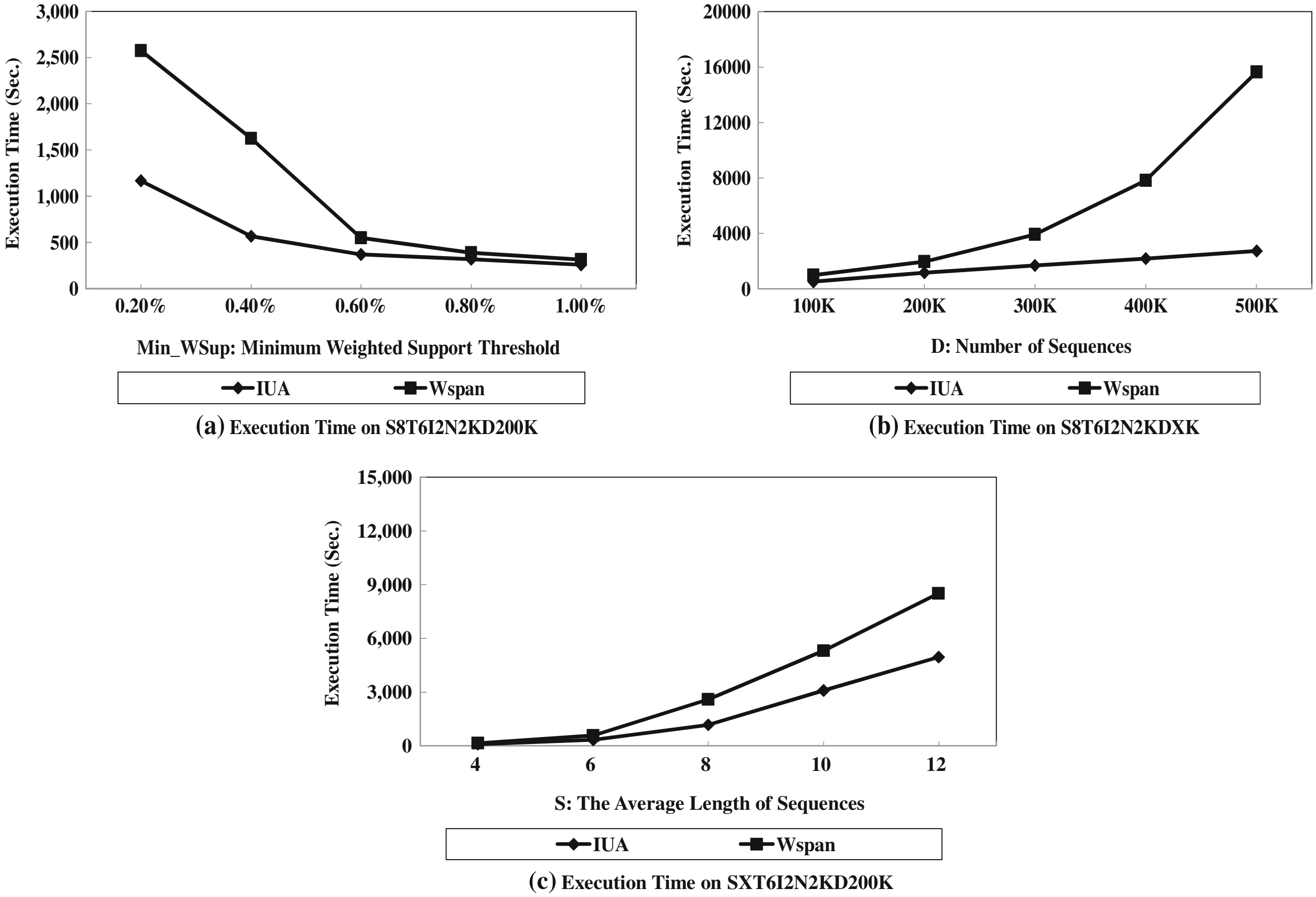


(a) Pruning Effectiveness on S8T6I2N2KD200K

(b) Pruning Effectiveness on S8T6I2N2KDXK

(c) Pruning Effectiveness on SXT6I2N2KD200K

**Fig. 2** Numbers of *WFUBs* generated by the two algorithms on the datasets



**(a)** Execution Time on S8T6I2N2KD200K

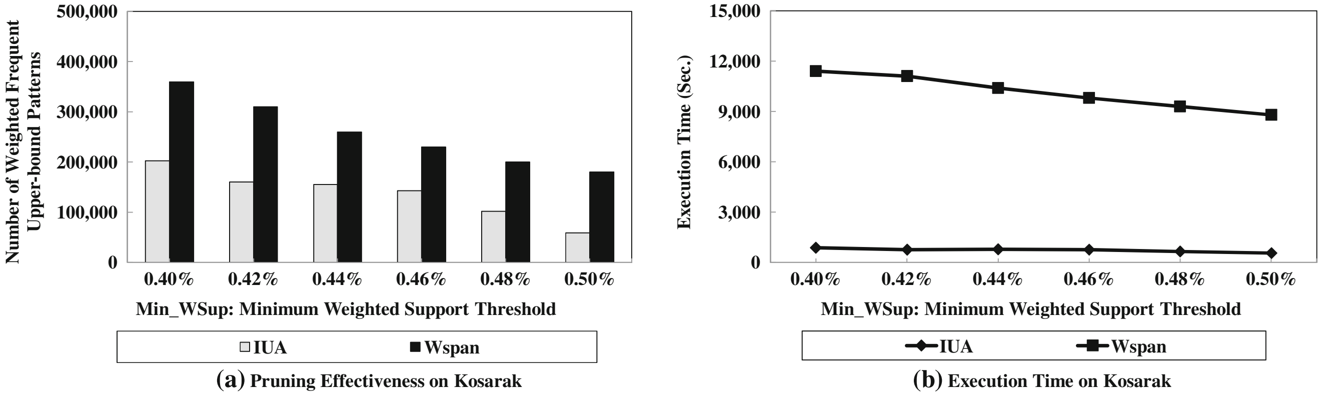
**(b)** Execution Time on S8T6I2N2KDXK

**(c)** Execution Time on SXT6I2N2KD200K

**Fig. 3** Efficiency comparison of the three algorithms on the datasets

### 6.1 Experimental datasets

In the experiments, several synthetic and real datasets are used to evaluate the performance of the algorithms. First, a publicly available *IBM* data generator [11] was adopted to generate the required datasets. To find weighted sequential patterns [21], we thus developed a simulation model, which was derived from the model used in the issue of utility mining [14] to generate the profits of the items. The parameters used in the *IBM* data generator [11] were *S*, *T*, *I*, *N* and *D*, which represented the average length of transactions per sequence, the average length of items per transaction, the average length of maximal potentially frequent itemsets, the total number of distinct items,



**(a)** Pruning Effectiveness on Kosarak

**(b)** Execution Time on Kosarak

**Fig. 4** Performance of the two algorithms on the real *Kosarak* dataset

and the total number of sequences, respectively. Moreover, for each sequence dataset generated, a corresponding weight table was also produced in which a weight value in the range from 0.0 to 1.0 was randomly assigned to an item. Note that the way described in [14] was considered to simulate the profits of items in a store environment, and then the profits of items were further normalized the weight values of the items in weight table. Figure 1 showed the weight-value distribution of all the items generated by the simulation model in the weight table.

In addition, a real dataset *Kosarak* could be obtained from the FIMI Repository [11]. The dataset *Kosarak* [11] was the click-stream data of a Hungarian on-line news portal. The synthetic and real datasets were used to evaluate the performance of the algorithms under various parameter settings. Table 11 showed the characteristics of the datasets used in the experiments.

### 6.2 Evaluation on the improved upper-bound strategy

Experiments were first made on the synthetic datasets to evaluate the difference in the number of weighted frequent upper-bound patterns generated by the two algorithms, *IUA* and *WSpan*. Figure 2 showed the comparisons of the numbers between weighted frequent upper-bound patterns (*WFUBs*) required by the two algorithms for the datasets with various minimum weighted support thresholds (*Min_WSup*), data sizes (*D*), and average length of sequences (*S*).

As seen in the figures, the number of weighted frequent upper-bound patterns required by the proposed *IUA* algorithm was obviously less than that of the existing *WSpan* algorithm, especially when the minimum weighted support threshold (*Min_WSup*) decreased, data sizes increased, and the sequence average length increased. The main reason for this is that the maximal weight in a sequence was more suitable as upper-bound of any subsequence in a sequence when compared with the traditional upper-bound model used in the *WSpan* algorithm [21]. Note that when the number of sequences increased, the minimum weighted support threshold value also increased, but kept the same ratio. Thus, the numbers of weighted frequent upper-bounds subsequences are nearly stable in Fig. 2 (b) when the data size increased and the other parameters settings were fixed. Due to the above results, using the proposed model could be effectively used to reduce more unpromising subsequences in mining under different parameter settings.

### 6.3 Efficiency evaluation

Next, the experiments were made to evaluate the execution efficiency of the algorithms. Figure 3 showed the efficiency comparisons of the two algorithms for the synthetic datasets with various minimum weighted support thresholds (*Min_WSup*), data sizes (*D*), and average length of sequences (*S*).

As shown in the figures, it could be observed that the efficiency of the proposed *IUA* algorithm was better than that of the *WSpan* algorithm under the lower minimum weighted support threshold (*Min_WSup*), the larger data size *D*, and the larger sequence average length *S*. In addition, the proposed *IUA* approach could still keep the flexible performance in terms of scalability under larger data sizes. The main reason was the same as that mentioned in Section 6.2. Thus, the proposed upper-bound model and pruning strategy could be effectively used to speed up the execution efficiency in mining weighted sequential patterns.

### 6.4 Evaluation on real dataset

In the experiment, the real dataset, *Kosarak* [11], was also used to evaluate the performance of the two algorithms. Figure 4 showed the performance comparisons of the two algorithms for the real *Kosarak* dataset with various *Min_WSup*, *D*, and *S*.

It could be observed from the figures that the performance of the proposed *IUA* approach for the real *Kosarak* dataset under different parameter settings still exceeded the existing *WSpan* approach in terms of the number of weighted frequent upper-bound subsequences, scalability and execution efficiency.

## 7 Conclusions

This work presents an efficient projection-based algorithm with an improved strategy (named *IUA*) for weighted sequential pattern mining. In particular, an effective upper-bound model in the proposed algorithm is developed to tighten the upper-bounds of weighted supports for subsequences in mining, and also the pruning strategy is designed to reduce more unpromising subsequences for mining. In addition, the experimental results on the synthetic and real datasets show the number of weighted frequent upper-bound patterns is obviously less than that required by the *WSpan* algorithm, and the proposed *IUA* algorithm outperforms the *WSpan* algorithm in terms of execution efficiency.

In the future, we will apply the proposed approach to some practical applications, such as data streams and supermarket promotion applications. Moreover, we will attempt to handle the maintenance problem of weighted sequential pattern mining when the sequences are inserted, deleted or modified.

## References

1. Agrawal R, Srikant R (1994) Fast algorithm for mining association rules. In: Proceedings of the international conference on very large data bases, pp 487–499
2. Agrawal R, Srikant R (1995) Mining sequential patterns. In: Proceedings of the ieee international conference on data engineering, pp 3–14
3. Agrawal R, Srikant R, Vu Q (1997) Mining association rules with item constraints. In: Proceedings of the 3rd international conference on knowledge discovery in databases and data mining, pp 66–73
4. Agrawal R, Imielinksi T, Swami A (1993) Mining association rules between sets of items in large database. In: Proceedings of The ACM SIGMOD international conference on management of data, pp 207–216
5. Agrawal R, Imielinski T, Swami A (1993) Database mining: a performance perspective. IEEE Trans Knowl Data Eng 5(6):914–925
6. Ahmed CF, Tanbeer SK, Jeong BS (2010) A novel approach for mining high-utility sequential patterns in sequence databases. ETRI J 32(5):676–686
7. Ahmed CF, Tanbeer SK, Jeong BS (2009) Efficient mining of weighted frequent patterns over data streams. In: Proceedings of IEEE international conference on high performance computing and communications (HPCC), pp 400–406
8. Ahmed CF, Tanbeer SK, Jeong BS, Lee YK (2008) Handling dynamic weights in weighted frequent pattern mining. Inst Electr Inf Commun Eng (IEICE) 91-D(11):2578–2588
9. Ahmed CF, Tanbeer SK, Jeong BS, Lee YK, Choi HJ (2012) Single-pass incremental and interactive mining for weighted frequent patterns. Expert Syst Appl 39(9):7976–7994
10. Chang JH (2011) Mining weighted sequential patterns in a sequence database with a time-interval weight. Knowl Based Syst 24(1):1–9
11. Frequent itemset mining implementations repository, Available at http://fimi.cs.helsinki.fi/
12. IBM Quest Data Mining Project, Quest Synthetic Data Generation Code, Available at http://www.almaden.ibm.com/cs/quest/syndata.html
13. Le B, Nguyen H, Vo B (2010) Efficient algorithms for mining frequent weighted itemsets from weighted items databases. In: Proceedings of international conference on computing & communication technologies, research, innovation, and vision for the future (RIVF), pp 1–6
14. Liu Y, Liao W, Choudhary A (2005) A fast high utility itemsets mining algorithm. In: Proceedings of the utility-based data mining workshop, pp 90–99
15. Pei J, Han J, Asi BM, Wang J, Chen Q (2004) Mining sequential patterns by pattern-growth: the prefixspan approach. IEEE Trans Knowl Data Eng 16(11):1424–1440
16. Shie BE, Yu PS, Tseng VS (2013) Mining interesting user behavior patterns in mobile commerce environments. Appl Intell 38(3):418–435
17. Song W, Liu Y, Li J (2014) Mining high utility itemsets by dynamically pruning the tree structure. Appl Intell 40(1):29–43
18. Srikant R, Agrawal R (1996) Mining sequential patterns: generalizations and performance improvements. In: Proceedings of the 5th international conference extending database technology, pp 3–17
19. Yen SJ, Lee YS (2013) Mining non-redundant time-gap sequential patterns. Appl Intell 39(4):727–738
20. Yun U, Leggett JJ (2005) WFIM: weighted frequent itemset mining with a weight range and a minimum weight. In: Proceedings of the 5th SIAM international conference on data mining, pp 636–640
21. Yun U, Leggett JJ (2006) WSpan: weighted sequential pattern mining in large sequence databases. In: Proceedings of the 3rd international ieee conference on intelligent systems, pp 512–517
22. Yun U (2008) A new framework for detecting weighted sequential patterns in large sequence databases. Knowl Based Syst 21(2):110–122
23. Yun U (2009) On pushing weight constraints deeply into frequent itemset mining. Intell Data Anal 13(3):359–383