Weighted Interesting Sequential Pattern Mining with a similar level of support and/or weight

Unil Yun

Sequential pattern mining has become an essential task with broad applications such as analyzing Web access patterns, customer purchase data, DNA sequences and so on. Most sequential pattern mining algorithms use a minimum support threshold to prune the combinatorial search space. This strategy provides the basic pruning. However, the support based pruning cannot mine correlated sequential patterns with similar support and/or weight levels. In previous sequential pattern mining approaches, if a minimum support is low, many spurious patterns having items with different support levels are found. Meanwhile, if the minimum support is high, meaningful sequential patterns with low support levels may be missed. In this paper, we present a new algorithm, Weighted Interesting Sequential pattern mining (WIS) based on the pattern growth method [13, 15] in which new measures, sequential s-confidence and w-confidence are suggested. By using these measures, weighted interesting sequential patterns with a similar level of support and/or weight are mined. WIS not only gives a balance between the two measures of support and weight, but also considers correlation between items within sequential patterns. To our knowledge, WIS is the first sequential pattern mining algorithm specifically to distinguish a level of support and/or weight between items of sequential patterns by checking ratio of the minimum support (weight) of items within this pattern to the maximum support (weight) of items within the pattern. These sequential affinity patterns can be useful for focusing on the profitable items, identifying interesting itemsets or sequences with similar support / weight levels, and analyzing the sequential time data. A comprehensive performance study shows that WIS is efficient and scalable in weighted sequential pattern mining.

Keywords: Data mining, weighted sequential pattern mining, affinity pattern.

I. Introduction

Sequential pattern mining finds frequent subsequences as patterns in a sequence database and sequential pattern mining algorithms have been extensively developed such as constraint-based sequential pattern mining [7, 11, 12, 14, 17], closed sequential pattern mining [19, 20, 23], approximate sequence mining [10], multi-dimensional sequence pattern mining [16], sequence mining in a noisy environment [24], biological sequence mining [6, 21], incremental sequence mining [4] and sequence indexing [5]. Sequential pattern mining has become an essential task with broad applications such as analyzing Web access patterns, customer purchase data, DNA sequences and so on. To tackle problems of Apriori based sequential pattern mining algorithms [1, 18], such as generation and test of all candidates and repeatedly scanning a large amount of the sequence database, sequential pattern growth approaches [9, 13, 15] have been developed. Sequential pattern growth methods mine the complete set of frequent sequential patterns using a prefix projection growth method to reduce the search space without generating all the candidates. Sequential patterns and items within sequential patterns have been treated uniformly, but real sequences have different importance. For this reason, weighted sequential pattern mining [27] has been suggested. Most algorithms use a support threshold to prune the search space. This strategy provides the basic pruning but the support based pruning is not enough to mine correlated sequential patterns. Previous sequential pattern mining algorithms could not detect sequential patterns with support and/or weight affinity. It is better to prune these weak affinity patterns first when the user wants to reduce the number of sequential patterns at the minimum support. However, no sequential pattern mining algorithm considers levels of support and/or weight.

1. Motivating examples

Let us give a motivating example for this work in market basket data. In sequential pattern mining, a sequential pattern {(bread, milk) (diaper, beer)} can be easily discovered with support threshold because the support (frequency) of the sequential pattern is relatively high. However, if the minimum support is low, many spurious patterns having items with different support levels are found. The spurious patterns are called as weak support affinity sequential patterns. For instance, {(gold ring, bronze ring) (vodka, beer)} is a possible weak support affinity sequential patterns because the support of the expensive item such as “gold ring” is much lower than the support of inexpensive item “bronze ring”. In a similar way, the support of the item “vodka” is lower than that of the item “beer”. Such sequential patterns including these itemsets are weak support affinity patterns. In a reverse case, if the minimum support is high, the interesting patterns which have low support levels may be missed [8]. The expensive items within the itemsets have low frequencies so the sequential patterns including such itemsets are not detected with the high minimum support. Examples of such itemsets are (gold ring, gold necklace), and (TV, DVD player). By considering support levels of items within sequential patterns, correlated sequential patterns can be discovered. As more extension, given weights of items according to the priority or importance, the sequential weight affinity patterns with similar weight levels can be found.

In real business, marketing managers would like to know the item lists which have similar profit or frequency levels with an acceptable error range α% of an interesting item’s profit or frequency. Trend analyzers are interested in analyzing itemsets with similar levels of profits or selling prices and customers want to find the items with similar price levels to buy interesting items within their budgets. According to the requirement of real applications, the needed data analysis should be determined and from the data analysis, the marketing policies about items’ price decision are different. Therefore, the comparison and analysis of correlated sequential patterns is essential to make plan for future marketing. Correlated sequential patterns with the support / weight affinity (s-affinity / w-affinity) can be useful for dividing customers into detailed segments, focusing on the profitable items and identifying interesting itemsets or sequences with similar support / weight levels, and planning marketing policies more accurately with the association structure of different products by analyzing the sequential time data.

In this paper, we propose an efficient sequential pattern mining algorithm called WIS (Weighted Interesting Sequential pattern mining) based on the pattern growth approach [13, 15]. We suggest sequential s-confidence and w-confidence. Based on the measures, sequential s-affinity /w-affinity pattern are defined. Here, weight / support affinity means how much items within a sequential pattern have similar characteristic in terms of weight / support values of the items. The sequential s-confidence measure is used to detect s-affinity patterns and sequential w-confidence measure is utilized to identify w-affinity patterns. We show that the two measures satisfy the anti-monotone property, define cross support / weight property and prove that the s-confidence / w-confidence satisfy the cross support / weight property. With the two properties, weak affinity patterns are eliminated effectively. On the framework, WIS algorithm is developed to detect correlated sequential patterns with the s-affinity / w-affinity by pushing the sequential s-confidence / w-confidence into the prefix projected sequential pattern growth approach. W-affinity and/or s-affinity pattern mining can give answers about the comparative analysis queries and discover interesting patterns which cannot be detected by conventional sequential pattern mining approaches. An extensive performance analysis shows that WIS is efficient and scalable in weighted sequential pattern mining.

2. Our contribution

The main contributions of this paper are as follows.

- ● Introduction of the sequential affinity pattern in terms of support and weight
- ● Definition of new measures, a sequential s-confidence, and a sequential w-confidence
- ●Description of weighted interesting sequential pattern mining by using sequential s-confidence / w-confidence
- ● Implementation of our algorithm, WIS and execution of an extensive experimental study to compare the performance of our algorithm, WIS with SPAM [2], PrefixSpan [15] and WSpan [27]

The remainder of the paper is organized as follows. In section 2, we describe the problem definition and related work. In Section 3, we develop WIS (Weighted Interesting Sequential pattern mining). Section 4 shows extensive experimental results. Finally, future research and conclusion is presented in sections 5 and 6 respectively.

Table 1. A Sequence Database (SDB)

Sequence ID	Sequence
10	⟨a (abc) (ac) d (cf)⟩
20	⟨(ad) abc (bcd) (ae) bcde⟩
30	⟨a(ef) b (ab) c (df) ac⟩
40	⟨ac (bc) eg (af) acb (ch) (ef)⟩
50	⟨ba (ab) (cd) eg (hf)⟩
60	⟨a (abd) bc (he)⟩

II. Problem definition and related work

1. Problem definition

Let I = $\{i_1, i_2... i_n\}$ be a unique set of items. A sequence S is an ordered list of itemsets, denoted as $\langle s_1, s_2, .., s_m \rangle$, where s_j is an itemset which is also called an element of the sequence, and $s_j \subseteq I$. That is, $S = \langle s_1, s_2, \ldots, s_m \rangle$, and s_i is $(x_{i1}x_{i2}\ldots x_{ik})$, where x_{it} is an item in the itemset s_i. The brackets are omitted if an itemset has only one item. As shown in Table 1, a sequence database, SDB = $\{S_1, S_2, .., S_n\}$, is a set of tuples ⟨sid, S⟩, where sid is a sequence identifier and S_k is an input sequence. An item can occur at most one time in an itemset of a sequence but it can occur multiple times in different itemsets of a sequence. Given a sequence database, SDB in Table 1 and a minimum support of 2, the SDB has 8 unique items, and six input sequences. A sequence ⟨a (abc) (ac) d (cf)⟩ in SDB has five itemsets: a, (abc), (ac), d, (cf) where items "a" and "c" appear three times in different itemsets of the sequence. The size |S| of a sequence is the number of itemsets in the sequence. For instance, the size of ⟨a (abc) (ac) d (cf)⟩ is 5. The length, l(S), is the total number of items in the sequence and a sequence with length l is called an l-sequence. For instance, the length of the sequence ⟨a (abc) (ac) d (cf)⟩ is 9. and the sequence is 9-sequence. A sequence $\alpha = \langle X_1, X_2, .., X_n\rangle$ is called a subsequence ($\alpha \sqsubseteq \beta$) of another sequence $\beta = \langle Y_1, Y_2, .., Y_m\rangle$ ($n \leq m$), and β is called a super sequence of the sequence α if there exist an integer $1 \leq i_1 < \ldots < i_n \leq m$ such that $X_1 \subseteq Y_{i1}$, $X_2 \subseteq Y_{i2}$, …, $\subseteq X_n \subseteq Y_{in}$. For example, sequence ⟨a (bc) d⟩ is a sub sequence of ⟨a (abc) (ac) d (cf)⟩ since a $\subseteq$ a, (bc) $\subseteq$ (abc) and d $\subseteq$ d. A tuple (sid, S) is said to contain a sequence α if the sequence S is a super sequence of α ($\alpha \sqsubseteq s$). The support of a sequence α in a sequence database (SDB) is the number of sequences in SDB that contain the sequence α (support (α) = |{<sid, S>| (<sid, S> $\in$ SDB) $\wedge$ ($\alpha \sqsubseteq$ S)}|). Given a support threshold, min_sup, a sequence α is called a frequent sequential pattern in the sequence database if the support of the sequence α is no less than the minimum support threshold (support (α) $\geq$ min_sup). For instance, a sequence <a (bc) d> is a frequent sequential pattern because sequences 10 and 20 contain sub sequence S = ⟨a (bc) d⟩ and the support of the sequence is 2 which is equal to the minimum support (2). Meanwhile, a sequential pattern <(ab) g> is not a frequent sequential pattern since the support (1) of the pattern is less than the minimum support (2). The problem of sequential pattern mining is to find the complete set of all frequent super sequences or the complete set of maximal frequent sequences. The anti-monotone property [1] has been mainly used to prune infrequent sequential patterns. That is, if a sequential pattern is infrequent, all super patterns of the sequential pattern must be infrequent. Based on the anti-monotone property, we can know that all super patterns of the sequential pattern <ag> such as sequential patterns <a (ab) g>, <a (ab) cg>, and <a (ab) (cd) g> are infrequent sequential patterns.

2. Related work

A. Sequential pattern mining

In sequential pattern mining, GSP [1] mines sequential patterns based on an Apriori-like approach by generating and testing all candidate subsequences with multiple scans of the original sequence database. To overcome this problem, an initial projection growth based approach, called FreeSpan [9] was developed. The main idea is to use frequent items to recursively project sequence databases into a set of fewer projected databases and grow subsequence fragments in each projected database. FreeSpan outperforms the Apriori based GSP algorithm. However, FreeSpan may generate any substring combination in a sequence and the projection in FreeSpan keeps all the sequences in the original sequence database without length reduction. PrefixSpan [13, 15], a more efficient pattern growth algorithm, improves the mining process. The main idea of PrefixSpan is to examine only the prefix subsequences and project only their corresponding suffix subsequences into projected databases. In each projected database, sequential patterns are grown by exploring only local frequent patterns. In SPADE [30], a vertical id-list data format was presented and the frequent sequence enumeration was performed by a simple join on id lists. SPADE can be considered as an extension of vertical format based frequent pattern mining. SPAM [2] utilizes depth first traversal of the search space combined with a vertical bitmap representation to store each sequence. Efficient sequential pattern mining algorithms [3, 30] have been developed such as constraint-based sequential pattern mining [7, 12, 14, 17], approximate sequential pattern mining with a weighted sequence structure [10], temporal sequence pattern mining with relational representation [11], sequential pattern mining without using support thresholds [19] and closed sequential pattern mining [20, 23]. These approaches may mine patterns efficiently and reduce the number of patterns. As given in the motivating example in section 1.1, the weight/support affinity sequential pattern can be useful but affinity sequential patterns cannot be detected in previous mining algorithms.

B. Weighted sequential pattern mining

In most of the previous sequential pattern mining algorithms, sequential patterns and items within sequential patterns have been treated uniformly, but real sequences have different importance. For this reason, WSpan (Weighted Sequential pattern mining) [27] and weighted frequent pattern mining [25, 26, 27, 29] have been suggested. In weight based sequential pattern mining, the items within a sequence are given different weights in the sequence database. The main concern in weight based sequential pattern mining is that the anti-monotone property [1] is broken when simply applying weights. In other words, although a sequential pattern is weighted infrequent, super patterns of the sequential pattern may be weighted frequent because super patterns of the sequential pattern with a low weight can get a high weight after adding other items or itemsets with higher weights. With the prefix projected sequential pattern growth method [13, 15], WSpan uses approximate weighted support within normalized weights to prune weighted infrequent sequential patterns but maintain the aniti-monotone property.

Even if WSpan algorithm is effective to identify weighted frequent sequential patterns, it cannot detect sequential correlated patterns with support / weight affinity. On the framework of weighted sequential pattern mining, we study the problem of sequential affinity pattern mining with similar weight and/or support levels. Our strategy is to push w-confidence/s-confidence into the sequential pattern mining algorithm and prune uninteresting patterns with the weak affinity.

III. WIS (Weighted Interesting Sequential pattern mining)

In this section, WIS algorithm is developed to detect correlated patterns with the support affinity (s-affinity) / weight affinity (w-affinity) by pushing the sequential s-confidence/w-confidence into the prefix projected sequential pattern growth approach [13, 15]. We present actual examples to illustrate the effect of sequential support / weight confidence and show our algorithm.

1. Preliminaries

In our approach, a sequence database is recursively projected into a set of fewer projected databases and sequential patterns are grown in each weighted projected database by processing weighted local frequent items. The number of projected databases can be reduced by only considering ordered prefix projection.

Definition 3.1 Prefix and suffix of a sequence

Suppose that all the items within itemsets in each sequence are listed by the alphabetical order. Given a sequence $\alpha = <e_1\ e_2 \ldots e_n>$ (in which each e_i means a frequent element in α), a sequence $\beta = <e`_1\ e`_2 \ldots e`_m>$ ($m \leq n$) is called a prefix of the sequence α if (1) $e_i = e`_i$ for ($i \leq m - 1$), (2) $e`_m \subseteq e_m$ and (3) all the weighted frequent items in ($e_m - e`_m$) are alphabetically listed after those in $e`_m$. Additionally, a sequence $\gamma <e``_m\ e_{m+1} \ldots e`_n>$ is called the suffix of the sequence α with regard to the prefix β, denoted as $\gamma = \alpha/\beta$, where $e``_m = (e_m - e`_m)$ which is also shown as $\alpha = \beta \cdot \gamma$.

Example 1: <a>, <aa>, <a(ab)> and <a (abc)> are prefixes of the sequence S = <a (abc) (ac) d (cf)>. However, <ab> and <a (bc)> are not prefixes if all items of the prefix <a (abc)> of the sequence S are frequent in S. In addition, <(abc) (ac) d (cf)> is the suffix about the prefix <a>, <(_bc) (ac) d (cf)> is the suffix with regard to the prefix <aa> and <(_c) (ac) d (cf)> is the suffix corresponding the prefix <a (ab)>.

Definition 3.2 Projected database

Given a sequential pattern α in a sequence database, α-projected database (S|α) is the collection of suffixes of sequences in S about the prefix α. The support (support (β)) of a sequential pattern β in the α-projected database (S|α) is the number of sequences γ in S|α such that $\beta \sqsubseteq \alpha\cdot\gamma$.

Example 2: Given a sequence database SDB in Table 1, <a>-projected database has six suffix sequences: <(abc) (ac) d (cf)>, <(_d) c (bc) (ae) bc>, <(_b) (df) cb>, <(_f) cbc> <(ab) (cd) e> and <(abd) bc>, and the <(ab)> projected database consists of four suffix subsequences prefixed with <(ab)>: <(_c) (ac) dc>, <dcb>, <(cd)> and <(_d) bc>.

To set up weights of items, attribute values of items of a sequence database can be used. Table 2 shows that prices (profits) of items can be used as a weight factor in market basket data.

Table 2. An example of a retail database

Item	Price	Support (Frequency)	Weight
Laptop Computer	1200$	5000	1.2
Desktop Computer	700$	3000	0.7
Memory stick	200$	20000	0.2
Memory card	150$	10000	0.15
Hard disk	100$	5000	0.1
Mouse	40$	80000	0.04
Mouse pad	10$	100000	0.01

Definition 3.3 Weight of a sequential pattern and weighted frequent pattern

The weight of the sequential pattern is the average value of the weights in items of a sequence. Given a sequence S = {s_1, s_2, …, s_m}, and s_j is ($x_{j1}x_{j2}\ldots x_{jk}$), weight of a sequential pattern S is formally defined as follows.

$$\frac{\sum_{j=1}^{j=m}\sum_{i=1}^{i=s_j} weight\ \ x_{ji}}{\sum_{j=1}^{j=m} length\ \ s_j}$$

A weighted support of a sequential pattern is defined as the resultant value of multiplying the pattern's support with the weight of the pattern. A sequential pattern is called a weighted frequent sequential pattern if the weighted support of a sequential pattern is no less than a minimum threshold.

Definition 3.4 Weight Range (WR) and Maximum Weight (MaxW) A weight of an item is a non-negative real number that shows the importance of each item. The weight of each item is assigned to reflect the importance of each item in the sequence database. Weights of items are given within a specific range (weight range). The weight range is exploited to restrict weights of items. A Maximum Weight (MaxW) is defined as a value of the maximum weight of items in a sequence database or a projected sequence database.

As already mentioned, attribute values such as prices (profits) of items in a sequence database can be used as a weight factor. However, the real values of items are not suitable for weight values because of the big variation. From an example of a real retail database in Table 2, we can know that variation of items' prices is so big that the prices cannot be directly used as weights. Therefore, within a specific weight range, the normalization process is needed which adjusts for differences among data in order to create a common basis for comparison. According to the normalization process, the final weights of items can be decided. That is, the prices or profits of items can be normalized within a specific weight range and the prices based on the definition, items, itemsets and a sequence have their own weights. From this example, weights of items are given between 0.01 and 1.2 and the maximum weight of items is the weight (1.2) of the item “laptop computer”.

Table 3. Sets of items with different weights

Item (min_sup=2)	⟨a⟩	⟨b⟩	⟨c⟩	⟨d⟩	⟨e⟩	⟨f⟩	⟨g⟩	⟨h⟩
Support	6	6	6	5	5	4	2	3
WR_1 : (0.7≤ Weight≤ 1.3)	1.1	1.0	0.9	1.0	0.7	0.9	1.3	1.2
WR_2 : (0.7≤ Weight≤ 0.9)	0.9	0.75	0.8	0.85	0.75	0.7	0.85	0.8
WR_3 : (0.4≤ Weigh≤ 0.8)	0.6	0.8	0.5	0.6	0.4	0.8	0.5	0.6
WR_4 : (0.2≤ Weight≤ 0.6)	0.5	0.2	0.6	0.4	0.6	0.3	0.5	0.3

Example 3: Table 3 shows example sets of items with different weights which are calculated by the normalization process. Given SDB in Table 1, and a minimum support, 2, the set of items in the database, i.e., length-1 subsequences in the form of “<item>:support” is {<a>: 6, <b>: 6, <c>: 6, <d>: 5, <e>: 4, <f>: 3, <g>: 2, <h>: 1}. When WR_1 as weights of items within a sequence is used, the weight of a sequence <a (bc) d (aef)> is 0.957 ((1.1 + (1.0 + 0.9) + 1.0 + (1.1 + 0.7 + 0.9)) / 7). Meanwhile, WR_2 and WR_3 are applied, the weights of the sequence, <a (bc) d (aef)> is 0.807 ((0.9 + (0.75 + 0.8) + 0.85 + (0.9 + 0.75 + 0.7)) / 7) and 0.614 ((0.6 + (0.8 + 0.5) + 0.6 + (0.6 + 0.4 + 0.8)) / 7). Additionally, Maximum Weights (MaxW) within WR_1, WR_2, WR_3 and WR_4 are 1.3, 0.9, 0.8 and 0.6 respectively.

2. Affinity sequential pattern

In this section, we define the sequential s-confidence and w-confidence measures, explain the concept of affinity sequential patterns, and show important properties.

A. Sequential s-affinity pattern

Definition 3.5 ***Sequential support-confidence (s-confidence)***

Support confidence of a sequential pattern $S = \{s_1, s_2, \ldots, s_m\}$, and s_i is $(x_{i1}x_{i2}\ldots x_{ik})$, where x_{it} is an item in the itemset s_i, denoted by sequential s-confidence, is a measure that reflects the overall s-affinity among items within the sequence. It is the ratio of the minimum support of items within this pattern to the maximum support of items within the sequential pattern. That is, this measure is defined as

$$\text{S-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\ 1 \le k' \le \textit{legnth}(s_{m'})} \{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\ 1 \le k'' \le \textit{legnth}(s_{m''})} \{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

To check if items within a sequential pattern have dissimilar support levels, the ratio of the minimum support of items within the pattern to the maximum support of items within the pattern is used. From the definition, sequential patterns with the s-affinity can be detected. From the s-confidence of a pattern, the affinity level can be calculated. For example, if the s-confidence is close to 1, it means that the affinity between items is high whereas if it is close to 0, the affinity is low.

It may be other ways to examine the s-affinity of sequential patterns. More complex definitions may detect more exact support levels. However, based on the definition of the sequential s-confidence, we will use two properties which are effective for identifying sequential s-affinity patterns.

Definition 3.6 ***Sequential s-affinity pattern***

A sequential pattern is a sequential s-affinity pattern if the s-confidence of the sequential pattern is no less than a minimum s-confidence (min_sconf). If not, the sequential pattern is called as a weak sequential s-affinity pattern.

Lemma 1 ***Sequential s-confidence has the anti-monotone property.***

Given a sequential pattern from definition 3.5, Max $_{(1\le m''\le m,\ 1\le k''\le \text{length}(s_{m''}))}$ $\{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}$ of a sequential pattern S is always greater than or equal to that of a sub-sequence of the sequential pattern S and Min $_{(1\le m'\le m,\ 1\le k'\le \text{length}(s_{m'}))}$ $\{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}$ of the pattern S is always less than or equal to that of a subset of the sequential pattern S. Therefore, we know that

$$\text{S-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\ 1 \le k' \le \textit{legnth}(s_{m'})} \{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\ 1 \le k'' \le \textit{legnth}(s_{m''})} \{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m' \le m-1,\ 1 \le k' \le \textit{legnth}(s_{m'})} \{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m-1,\ 1 \le k'' \le \textit{legnth}(s_{m''})} \{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m'-2 \le m,\ 1 \le k' \le \textit{legnth}(s_{m'})} \{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m''-2 \le m,\ 1 \le k'' \le \textit{legnth}(s_{m''})} \{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

That is, if the s-confidence of a sequential pattern is no less than a min_sconf, so is every subset of size m - 1. Therefore, the sequential s-confidence can be used to prune the exponential search space.

Example 4: Consider a pattern S = {⟨AB⟩ ⟨AC⟩ ⟨ABC⟩ ⟨AE⟩} and S` = {⟨BC⟩ ⟨BD⟩ ⟨BCD⟩ ⟨BF⟩}. Assume that a min_sconf is 0.5, support ({A}) = 2, support ({B}) = 5, support ({C}) = 8, support ({D}) = 4, support ({E}) = 5, and support ({F}) = 6, where support (X) is the support value of a sequential pattern X. Then, the sequential s-confidence (S) is 0.25 (2/8) and s-confidence (S`) is 0.5 (4/8). Therefore, sequential pattern S is not a sequential s-affinity pattern but pattern S` is a sequential s-affinity pattern. From the anti-monotone property of the s-confidence, any super pattern of the pattern S is weak s-affinity pattern and is pruned.

Property 1 ***Cross support sequential pattern property***

Given a threshold t, a sequential pattern S is a cross support sequential pattern with respect to t if the pattern S contains two items x and y such that (support ({X}) / support ({Y})) < t, where 0 < t < 1. This means the sequential pattern contains at least two items which have different support levels.

Lemma 2 ***Sequential s-confidence has cross support sequential pattern property***

For any cross support pattern S with a threshold t, it is guaranteed that s-conf (S) < t. That is, given min_sconf as a threshold, if sequential s-confidence has the cross support sequential pattern property, for any cross support sequential pattern S with regard to min_sconf, the value of the sequential s-confidence is less than min_sconf. Given definition 3.5, assume that there is a cross support sequential pattern S = {s_1, s_2, ..., s_m} that contains at least two items X and Y such that support ({X}) / support ({Y}) < t where 0 < t < 1.

$$\text{S-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{support}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{support}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\ldots, \text{support}(\{X\}), \ldots, \text{support}(\{Y\}), \ldots\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\ldots, \text{support}(\{X\}), \ldots, \text{support}(\{Y\}), \ldots\}}$$

$$\le \frac{\text{support}(\{X\})}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\ldots, \text{support}(\{X\}), \ldots, \text{support}(\{Y\}), \ldots\}}$$

$$\le \frac{\text{support}(\{X\})}{\text{support}(\{Y\})} < t$$

Therefore, we know that the value of the sequential s-confidence is less than the min_sconf for any cross support sequential pattern S with regard to a sequential s-confidence threshold, t.

B. Sequential w-affinity pattern

Definition 3.7 ***Sequential weight-confidence (w-confidence)***

Weight confidence of a sequential pattern S = {s_1, s_2, …, s_m}, and s_i is ($x_{i1}x_{i2}\ldots x_{ik}$), where x_{it} is an item, denoted by sequential w-confidence, is a measure that reflects the overall w-affinity among items within the sequential pattern. It is the ratio of the minimum weight of items within this pattern to the maximum weight of items within the pattern. In other words, this measure is defined as

$$\text{W-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{weight}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{weight}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

Definition 3.8 ***Sequential w-affinity pattern***

A sequential pattern is a sequential w-affinity pattern if the w-confidence of the sequential pattern is no less than a minimum weight confidence (min_wconf). If not, the sequential pattern is called as a weak w-affinity pattern.

Lemma 3 ***Sequential w-confidence has the anti-monotone property.***

From definition 3.7, we can see that Max $_{(1 \le m'' \le m, 1 \le k'' \le \text{length}(S_{m''}))}$ {weight ({$x_{m''k''} \subseteq s_{m''}$})} of a sequential pattern S is always greater than or equal to that of a sub-sequence of the sequential pattern S and Min $_{(1 \le m' \le m, 1 \le k' \le \text{length}(S_{m'}))}$ {support ({$x_{m'k'} \subseteq s_{m'}$})} of the pattern S is always less than or equal to that of a subset of the sequential pattern S. Therefore, we know that

$$\text{W-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{weight}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{weight}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m' \le m-1,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{weight}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m-1,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{weight}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m'-2 \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{weight}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m''-2 \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{weight}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

In other words, if w-confidence of a sequential pattern S is no less than a min_wconf, so is every subset of size m - 1. Therefore, the sequential w-confidence satisfies the anti-monotone property and prunes weak w-affinity patterns.

Example 5: consider a pattern S = {⟨AB⟩ ⟨AC⟩ ⟨ABC⟩ ⟨AE⟩} and S` = {⟨BC⟩ ⟨BD⟩ ⟨BCD⟩ ⟨BF⟩}. Assume that a min_wconf is 0.5, weight ({A}) = 0.2, weight ({B}) = 0.4, weight ({C}) = 0.7, weight ({D}) = 0.6, weight ({E}) = 0.4, and weight ({F}) = 0.5, where weight (Y) is the weight value of a sequential pattern Y. Then, the average weight of a sequential pattern S and a sequential pattern S` are 0.425 and 0.55 respectively. The sequential w-confidences (S) is 0.29 (2/7) and sequential w-confidence (S`) is 0.56 (4/7). Therefore, the sequential pattern S is not a sequential w-affinity pattern but pattern S` is a sequential w-affinity pattern.

Property 2 ***Cross weight sequential pattern property***

Given a threshold t, a sequential pattern S is a cross weight sequential pattern with respect to t if the pattern S contains two items Z and W such that (weight ({Z}) / support ({W})) < t, where 0 < t < 1. This means the sequential pattern contains at least two items which have different weight levels.

Lemma 4 ***Sequential w-confidence has cross weight property.***

For any cross weight pattern S with a threshold t, it is guaranteed that w-conf (S) < t. In other words, given min_wconf as a threshold, if sequential w-confidence has the cross weight sequential pattern property, for any cross weight sequential pattern S with regard to min_wconf, the value of the sequential w-confidence is less than min_wconf. Given definition 3.7, assume that there is a cross weight sequential pattern S = {s_1, s_2, ..., s_m} that contains at least two items Z and W such that weight ({Z}) / weight ({W}) < t where 0 < t < 1.

$$\text{W-conf}(S) = \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\text{weight}(\{x_{m'k'} \subseteq s_{m'}\})\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\text{weight}(\{x_{m''k''} \subseteq s_{m''}\})\}}$$

$$\le \frac{\text{Min}_{1 \le m' \le m,\, 1 \le k' \le \textit{legnth}(s_{m'})}\{\ldots, \text{weight}(\{Z\}), \ldots, \text{weight}(\{W\}), \ldots\}}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\ldots, \text{weight}(\{Z\}), \ldots, \text{weight}(\{W\}), \ldots\}}$$

$$\le \frac{\text{weight}(\{Z\})}{\text{Max}_{1 \le m'' \le m,\, 1 \le k'' \le \textit{legnth}(s_{m''})}\{\ldots, \text{weight}(\{Z\}), \ldots, \text{weight}(\{W\}), \ldots\}}$$

$$\leq \frac{\text{weight}(\{Z\})}{\text{weight}(\{W\})} < t$$

Therefore, we know that the value of the w-confidence is less than the min_wconf for any cross weight sequential pattern with regard to a sequential w-confidence threshold, t.

Example 6: The pruning examples of the anti-monotone and cross weight properties on the w-confidence

From the anti-monotone property of the sequential w-confidence, if the w-confidence of a sequential pattern is less than the min_wconf, any super pattern of the sequential pattern is removed. Meanwhile, given an item x, all patterns that contain the item x and at least an item with a weight less than t · weight (x) (for 0 < t < 1) are cross weight patterns and the w-confidences of the sequential patterns are less than t (min_wconf). The cross weight sequential patterns can be directly pruned without calculating the w-confidences. For instance, given a sequence database SDB in Table 1, a weight list for eight items <a:, 0.65, b:0.8, c:0.5, d:0.7, e:0.4, f:0.8, g:0.5, h:0.75>, and the minimum w-confidence of 0.8, the w-confidence (0.67) of a sequential pattern "<ce>" is less than the minimum w-confidence (0.8) so the pattern is pruned. From the anti-monotone property, we can prune the super patterns such as "<(cd)e>" and "<c(ef)>" since these patterns have one subset "<ce>" which is not a w-affinity pattern. Meanwhile, we can prune cross weight patterns by the cross weight property. With a weight ascending order which is: {<e>: 0.4, <c>: 0.5, <g>: 0.5, <a>: 0.65, <d>: 0.7, <h>: 0.75, <b>: 0.8, <f>: 0.8}, we can find an item "e" with weight ("g") = 0.5 < weight ("a") * min_wconf (0.8) = 0.52. If we split the item list into two group {items "e", "c", and "g"} and {items "a", "d", "h", "b" and "f"}, any pattern including items from both groups is the cross weight sequential pattern with the min_wconf because the sequential w-confidence is always less than the minimum w-confidence for cross weight patterns. In this example, without applying the cross weight property, the cross weight patterns such as "<ea>", "<ed>", "<ch>", "<cb>" and "<gb>" have to be generated as candidate patterns and prune them later by computing the w-confidence values of the sequential patterns. Note that those patterns such as the sequential patterns "<ea>", "<ed>", "<ch>", "<cb>" and "<gb>" are not pruned by the anti-monotone property because every subset of the patterns is the w-affinity sequential pattern (w-confidence = 1). In a similar way, the anti-monotone property and the cross support sequential pattern property can be used to prune weak s-affinity patterns.

3. Weighted interesting sequential patterns

In this section, we define weighted interesting sequential pattern mining and show pruning methods.

Definition 3.9 ***Weighted Interesting Sequential pattern***

A sequence is a weighted interesting sequential pattern if the following conditions are satisfied. Note that these conditions can be applied selectively and sequential s-confidence and w-confidence can also be used independently.

Pruning condition 1: (Weighted support constraint) A pattern S is a weighted sequential frequent pattern if and only if |S| > 0 and (support (S) * MaxW) ≥ min_sup.

Observation 1: In weighted sequential pattern mining, the anti-monotone property cannot be directly used. Although a sequential pattern is weighted infrequent, super patterns of the sequential pattern may be weighted sequential frequent because a sequential pattern which has a low weight can get a high weight after adding another item with a higher weight. By using the maximum weighted support, anti-monotone property can be maintained. In other words, if a maximum weighted support (support (S) * MaxW) of a sequential pattern S is less than the minimum support, any super pattern cannot be a weighted sequential frequent pattern so the pattern can be pruned now. During mining process, weighted infrequent items are pruned and weights of the weighted infrequent items are not considered as MaxW although weights of the items are high. By doing so, the MaxW is reduced and the maximum weighted support becomes more accurate.

Pruning condition 2: *(s-confidence ≥ min_sconf)* A sequential pattern S is a sequential s-affinity pattern if and only if |S| > 0 and sconf (S) ≥ min_sconf. In the pruning condition 2, the anti-monotone property and the cross support property are applied to prune weak s-affinity patterns.

Pruning condition 3: *(w-confidence ≥ min_wconf)* A sequential pattern S is a sequential w-affinity pattern if and only if |S| > 0 and wconf (S) ≥ min_wconf. In the pruning condition 3, the anti-monotone property and the cross weight property are applied to prune weak w-affinity patterns.

Lemma 5 ***Sequential w-confidence can be applied irrespective of different weight ranges.***

WIS uses the weight range which can be utilized to calculate a maximum weight and maintain the anti-monotone property efficiently. For example, the weight range WR_k of a sequential pattern K = {<A>, <A, B>, <A, B, C>} is from 1 to 3 and the weight range $WR_{k`}$ of a sequential pattern K` = {<D>, <D, E>, <D, E, F>} is from 0.1 to 0.3. Assume that weight ({A}) = 1, weight ({B}) = 2, weight ({C}) = 3, weight ({D}) = 0.1, weight ({E}) = 0.2, and weight ({F}) = 0.3, where weight is the weight value of a sequential pattern. Then, sequential w-confidence (K) = 0.33 and sequential w-confidence (K`) = 0.33. Using $WR_{k`}$ rather than WR_k generates fewer sequential patterns from the pruning condition 1. However, the w-confidences (0.33) of sequential patterns K and K` are the same in spite of different weight ranges. We know that sequential w-confidence is defined as the ratio of the minimum weight of items within this sequential pattern to the maximum weight of items within the sequential pattern. Therefore, if ratios of the minimum weight to the maximum weight of different weight ranges are the same, the effect is the same. In other words, the w-confidence of a sequential pattern is only decided by a level of w-affinity between items of a sequential pattern, not by a weight range. A level of weight (support) means weight (support) affinity level which shows how much items within a pattern have similar characteristic in terms of weight (support) values among the items. The weight (support) affinity levels are calculated by using w-confidence and s-confidence respectively.

Observation 2: The Lemma 5 gives the information that sequential w-confidence in the pruning condition 3 can be applied irrespective of different weight ranges. It's the same situation in sequential s-confidence of the pruning condition 2 because the w-confidence and s-confidence measures focus on detecting sequential patterns containing items with similar weight (support) levels so two patterns with the same weight (support) ratio can have different weights (supports).

A. Sequential s-confidence VS. w-confidence

Sequential s-confidence is a support measure which is used to identify sequential s-affinity patterns and sequential w-confidence is a weight measure that considers the sequential w-affinity of items within a sequential pattern. Both measures satisfy the anti-monotone property and the cross support / weight sequential pattern property so these measures can be effectively used to prune weak affinity patterns.

B. Sequential w-confidence VS. weighted support constraint

Although weighted support constraint considers weight and support, it cannot detect affinity patterns. The previous use of a weight constraint in WSpan [27] can generate weak affinity patterns containing items with different weight levels or miss interesting low weight patterns. Sequential w-confidence considers only weights of items within patterns. Patterns with a high support and a high weight satisfy the weighted support constraint but the w-confidences of these patterns may not satisfy the minimum w-confidence if they are sequential patterns with dissimilar weight levels .

C. Sequential s-confidence VS. support constraint

Sequential s-confidence and support constraint both use a support measure. Support constraint cannot detect affinity patterns. Although the sequential patterns with a high support satisfy the support constraint, these sequential patterns cannot satisfy the sequential s-confidence when they are sequential patterns including items with different levels of supports.

Observation 3: Recall that our approach focuses on identifying strong affinity sequential patterns in terms of support and weight. The discovered sequential patterns can be useful in processing comparative analysis queries. However, the weak affinity patterns containing items with dissimilar support / weight levels may be also useful in other applications. The novelty of our approach is that WIS can identify strong or weak support (weight) affinity patterns by applying s-confidence (w-confidence) measure. Meanwhile, previous sequential pattern mining algorithms could not find the correlated patterns.

4. Mining weighted interesting sequential patterns with s-affinity and/or w-affinity

On the framework, we develop the WIS algorithm to detect correlated patterns with the s-affinity and/or w-affinity. As a mining example, we show how to mine affinity sequential patterns by using a prefix-based projection approach [15] that computes local frequent sequential patterns of a prefix by scanning its projected database. The projection is based on a frequent prefix. We use the sequence database SDB in Table 1 and apply $0.4 \leq WR_3 \leq 0.8$ as a weight range from Table 3. Assume that min_sup is 2, min_wconf is 0.7 and min_sconf is 0.7. then, the weight list is <a:0.6, b:0.8, c:0.5, d:0.6, e: 0.4, f:0.8, g:0.5, h:0.6> and the maximum weight (MaxW) is 0.8. In the WIS, mining process is performed as follows.

Step 1: ***Find length-1 weighted sequential patterns***.

Scan the sequence database once, count the support of each item, check the weight of each item and find all the weighted frequent items in sequences. First, after the first scan of the sequence database, we know that length-1 frequent sequential patterns (frequent sequential items) are <a> : 6, <b> : 6, <c> : 6, <d> : 5, <e> : 5, <f> : 4, <g> : 2 and <h> : 3 because the supports of the items are greater than or equal to the minimum support (2). Using MaxW (0.8), weighted supports of the items are calculated and weighted infrequent items are pruned according to the pruning condition 1 in definition 3.9. For example, weighted support (6 * 0.8) of an item <a> is greater than the minimum support so the item is weighted frequent sequential item. Meanwhile, an item <g> is not weighted frequent because the weighted support (2 * 0.8) is less than the minimum support. In this way, weighted frequent sequential items are detected and pruned from the condition by the weighted support constraint. After the projected database is generated from the sequence database, WIS mines weighted interesting sequential patterns from the projected databases recursively and the weighted interesting patterns are generated by adding items one by one.

Step 2: ***Divide search space.***

The complete set of weighted sequential patterns can be partitioned into the following seven subsets having prefix: (1) <a>, (2) <b>, (3) <c>, (4) <d>, (5) <e>, (6) <f>, and (7) <h>.

Step 3: ***Find subsets of sequential patterns.***

The subsets of sequential patterns can be mined by constructing the corresponding set of projected databases and mining them recursively.

A. Find affinity sequential patterns with the prefix <a>

We only collect the sequences which have <a>. Additionally, in a sequence containing the prefix <a>, only the subsequence prefixed with the first occurrence of the prefix <a> should be considered. For example, in a sequence <a (abc) (ac) d (cf)>, only the subsequence <(abc) (ac) d (cf)> is considered and in a sequence <(ad) abc (bcd) (ae) bcde>, only the suffix sequence <(_d) abc (bcd) (ae) bcde> is collected. The sequences in the sequence database SDB containing <a> are projected with regards to the prefix <a> to form the <a>-projected database, which consists of six suffix sequences: <(abc) (ac) d (cf)>, <(_d) abc (bcd) (ae) bcde>, <(ef) b (ab) c (df) ac>, <c (bc) e (af) acb (ch) (ef)>, <(ab) (cd) e (hf)> and <(abd) bc (he)>. By scanning the <a> projected database once, its locally frequent items are a:6, b:6, c:6, d:5, e:5, f:4, h:3, (_b):4, (_c):1, (_d):1, (_e):1 and (_f):1. The local items, (_c):1, (_d):1, (_e):1 and (_f):1 which have one as the support, are removed by weighted support constraint since the weighted support (0.8) of multiplying the support (1) of the sequences with MaxW (0.8) is less than a minimum support (2). In addition, a local item "e:5" is pruned by the sequential w-confidence. The candidate pattern, from a local item "e:5" and a conditional prefix "a" is <ae>:5 and the sequential w-confidence (0.67) of the candidate sequential pattern <ae>:5 is less than the minimum w-confidence (0.7). Moreover, the candidate pattern <ah>:3 is pruned by the sequential s-confidence because the s-confidence of the sequential pattern is 0.5 which is less than the minimum s-confidence (0.7). All the length-2 sequential patterns prefixed with <a> are: <aa>:6, <ab>:6, <ac>:6, <ad>:5 <af>:4 and <(ab)>:4. Note that previous sequential pattern mining algorithms only consider a support in each projected database so sequences <(ac)>:1 <(ad)>:1 and <(ae)>:1 are only pruned because they are not frequent. The recently developed WSpan algorithm uses weighted support constraint. However, in WIS, before constructing the next projected database, sequential w-confidence and s-confidence are applied to prune weak affinity sequential patterns. The final <a>-projected database is generated as follows: <(abc) (ac) d (cf)>, <(_d) abc (bcd) abcd>, <fb (ab) c (df) ac>, <c (bc) (af) acbcf>, <(ab) (cd) f> and <(abd) bc>. Recursively, all the sequential patterns with the prefix <a> can be partitioned into six subsets prefixed with: 1) <aa>, 2) <ab>, 3) <ac>, 4) <ad>, 5) <af> and 6) <(ab)>. These subsets can be mined by constructing respective projected databases and mining each recursively as follows.

1) The <aa> projected database consists of six suffix subsequences prefixed with <(_bc) (ac) d (cf)>, <bc (bcd) abcd>, <(_b) c (df) ac>, < (_f) acbcf>, < (_b) (cd) f>, and <(_bd) bc>. By scanning the <aa> projected database once, its local items are a:4, b:3, c:6, d:4, f:4, (_b):4, (_c):1 and (_f):1. The local items, "(_c):1" and "(_f):1", are pruned by the weighted support constraint. The <aa> projected database returns the following sequential patterns: <aaa>:4, <aab>:3, <aac>:6, <aad>:4, <aaf>:4 and <a(ab)>:4. Sequential s-confidence and w-confidence of these patterns are no less than a minimum s-confidence and a minimum w-confidence respectively. Recursively, sequential patterns with the prefix <aa> are partitioned and mined.

2) The <ab> projected database consists of six suffix subsequences prefixed with <ab>: <(_c) (ac) d (cf)>, <c (bcd) abcd>, <(ab) c (df) ac>, <(_c) (af) acbcf>, <(cd) f> and <(_d) bc>. By scanning the <ab> projected database once, we obtain its local items: a:4, b:4, c:6, d:4, f:4, (_c):2, and (_d):1. Local items, (_c):2, and (_d):1, are pruned by weighted support constraints In WIS, the sequential candidate pattern <abf>:4 is removed by the sequential s-confidence since the sequential s-confidence (0.67) of the sequential pattern <abf> is less than a min_sconf (0.7). From the sequential w-confidence, the sequence candidate <abc>:4 is pruned because the w-confidence (0.625) of the sequence candidate <abc>:4 is less than min_wconf (0.7). The final weighted sequential patterns are <aba>:4, <abb>:4 and <abd>:4. Recursively, sequential patterns with the prefix <ab> are partitioned and mined.

3) The <ac> projected database consists of five suffix subsequences prefixed with <ac>: <(ac) d (cf)>, <(bcd) abcd>, <(df) ac>, <(bc) (af) acbcf>, and <(_d) f>. By scanning the <ac> projected database once, its local items are a:4, b:2, c:4, d:3, f:4, (_d):1 and (_f):1. Sequential candidate patterns, <acb>:2, <a(cd)>:1, and <a(cf)>:1, are pruned by weighted support constraint. The weighted sequential patterns <aca>: 4, <acc>:4 <acd>:3 and <acf>:4 are generated. Recursively, sequential patterns with the prefix <ac> are partitioned and mined.

4) The <ad> projected database consists of five suffix subsequences prefixed with <ad>: <(cf)>, <abcd>, <(_f) ac>, <f> and <bc>. By scanning the <ad> projected database once, its local items are a:2, b:2, c:4, d:1, f:2, and (_f):1. Among these candidate patterns, the only weighted frequent item is c:4 which satisfies sequential s-confidence and w-confidence, so <ad> projected database returns a sequential pattern <adc>:4. Recursively, sequential patterns with the prefix <ad> are partitioned and mined.

5) The <af> projected database consists of two suffix subsequences prefixed with <af>: <b (ab) c (df) ac>, and <acbcf>. By scanning the <af> projected database once, its local items are a:2, b:2, c:2, d:1, and f:2. All local items are pruned because they do not satisfy the conditions in definition 3.9.

6) The <(ab)> projected database consists of four suffix subsequences prefixed with <(ab)>: <(_c) (ac) d (cf)>, <c (df) ac>, <(cd) f> and <(_d) bc>. By scanning the <(ab)> projected database once, its local items are a:2, b:1, c:4 d:3, f:3, (_c):1 and (_d):1. Local items "a:2", "b:1" "(_c):1 and "(_d):1" are pruned by the weighted support constraint and the sequential candidate pattern <(ab)c>:4 is pruned by the sequential w-confidence because the w-confidence of the pattern is 0.625 which is less than the minimum w-confidence (0.7). The candidate pattern "(ab)f" is pruned by the sequential s-confidence since it is a weak s-affinity pattern. The sequential s-confidence (0.67) of the candidate sequential pattern <(ab)f>:3 is less than the minimum s-confidence (0.7). Finally, the sequential pattern generated by the <(ab)> projected database is <(ab)d>:3. Recursively, sequential patterns with prefix <(ab)> are partitioned and mined.

B. Mine remaining affinity sequential patterns. This can be done by constructing the <b>, <c>, <d>, <e>, <f> and <h> projected databases and mining them, respectively as shown above.

Step 4: The set of sequential patterns is the collection of patterns found in the above recursive mining process.

Table 5. Examples of pruning candidate patterns.

Candidate patterns	Weighted support	Sequential w-confidence	Sequential s-confidence
<ae> : 5	(0.8 * 5)	Pruned 0.67 (0.4/0.6)	0.83 (5/6)
<ah> : 3	(0.8 * 3)	1 (0.6/0.6)	Pruned 0.5 (3/6)
<acb> : 2	Pruned (0.8 * 2)	Pruned 0.625 (0.5/0.8)	1 (6/6)
<adb> : 2	Pruned (0.8 * 2)	0.75 (0.6/0.8)	0.83 (5/6)
<(ab)c> : 4	(0.8 * 4)	Pruned 0.625 (0.5/0.8)	1 (6/6)
<(ab)f> : 3	(0.8 * 3)	0.75 (0.6/0.8)	Pruned 0.67(4/6)

Table 5 shows examples of pruning candidate patterns in the mining process with a minimum support of 2 and a minimum s-confidence and w-confidence of 0.7 respectively. By using two objective measures, sequential s-confidence and w-confidence, these weak affinity patterns are pruned first when the number of patterns need to be reduced.

Observation 4: WIS used the prefix projected sequential pattern growth approach as a framework. However, note that the main focus of our work is the suggestion of the concept of affinity sequential pattern mining. WIS can be developed by using other frameworks such as depth first traversal algorithms with a vertical bitmap format [5, 6] or Apriori based algorithms [1, 2].

5. WIS algorithm

We show the weighted sequential pattern mining algorithm.

WIS algorithm: Weighted sequential pattern mining with the s-affinity and/or w-affinity.

```
Input: (1) A sequence database: SDB,
       (2) A support threshold: min_sup,
       (3) A w-confidence threshold: min_wconf,
       (4) A s-confidence threshold: min_sconf.
Output: The complete set of weighted sequential patterns.
Begin
  1. Let WSP be the set of Weighted Sequential Patterns that satisfy
     the constraints. Initialize WSP ← {};
  2. Scan SDB once, count the support of each item, check the
     weight of each item and find each weighted frequent item, β, in
     sequences satisfying the following pruning condition: β is a
     weighted sequential item if the weighted support of the item is
     no less than the minimum support.
  3. For each weighted frequent item, β, in SDB
         Call WIS (WSP, <β>, 1, SDB)
     End for
End
```

Procedure WIS (WSP, α, L, S|α)

```
Parameter:
(1) α is a weighted sequential pattern that satisfies the above
pruning conditions,
(2) L is the length of α,
(3) S | $_α$ is the sequence database, SDB if α is null, otherwise, it is
the α-projected database.
 1. Scan S |$_α$ once, count the support of each item, and find each
    weighted frequent item, β in sequences: β is a weighted
    sequential item if the following pruning conditions are satisfied.
      Condition 1: (support * MaxW ≥ min_sup)
      Condition 2: (w-confidence ≥ min_wconf)
      Condition 3: (s-confidence ≥ min_sconf)
    (a) β can be assembled to the last element of α to form a
        sequential pattern or
    (b) <β> can be appended to α to form a sequential pattern.
 2. For each weighted frequent item β,
        Add it to α to form a sequential pattern α`, and output α`;
 3. For each α`,
        Construct α` projected database S| α`;
        Call WIS (α`, L+1, S | α')
    End for
```

After WIS algorithm calls the procedure WIS (WSP, <β>, 0, SDB), WIS (α`, L+1, S | α`) is called recursively after α` projected database S|a` is constructed. Recall that the approximate maximum weighted support (support (S) * MaxW) is used instead of a pattern's real weighted support which does not satisfy the anti-monotone property. Therefore, in final step, we should prune weighted infrequent sequential patterns which satisfy this condition ("support (S) * MaxW ≥ min_sup").

6. Applications of mining weighted sequential patterns with s-affinity and/or w-affinity

Weighted sequential pattern mining with s-affinity and/or w-affinity can be used in several application domains such as analyzing retail data, telecommunications data, and financial data and so on. First, correlated sequential patterns with the w-affinity/s-affinity can be applied in analyzing customer buying patterns and planning marketing policies with the association structure of different products by analyzing the sequential time data. Second, the techniques of mining patterns with different level of support and/or weight can be applied to find previous fraudulent users and their usage patterns in crimes such as money laundering, purchase of expensive items within a short time, use of stolen mobile and other financial crimes. The usage (transaction) frequency for each user is usually regular and customers have purchasing styles so sequential patterns containing products with different levels of frequency (price) may be fraudulent patterns. Therefore, the level of affinity can help catch fraudulent patterns. Third, sequential patterns with s-affinity and/or w-affinity can be applied to identify co-occurring gene sequences in biomedical data and DNA analysis. The pattern mining with s-affinity and/or w-affinity can help determine the kinds of genes that are likely to co-occur together in target samples.

IV. Performance evaluation

In this section, we present our performance study over various datasets. The WIS is the first sequential pattern mining algorithm to consider a level of support and/or weight between items of sequential patterns. We report our experimental results on the performance of WIS in comparison with recently developed algorithms: PrefixSpan [15], SPAM [2] and WSpan [27] because PrefixSpan and SPAM are traditional sequential pattern mining algorithms and WSpan is a weight based sequential pattern mining algorithm. The main purpose of this experiment is to demonstrate how effectively the weighted sequential affinity patterns can be found by using sequential s-confidence and/or w-confidence. First, we show how the number of weighted sequential affinity patterns can be adjusted through user feedback. Specifically, in this performance test, the number of sequential patterns and maximum sequential patterns without inclusions (Fig. 5, and Fig. 13) are checked. Second, we present the efficiency of the WIS algorithm, and quality of weighted affinity sequential patterns. Finally, we illustrate that WIS has good scalability against the number of sequences in the datasets.

Table 4. Parameters for IBM Quest Data Generator.

Symbol	Meaning
D	Number of customers in the dataset
C	Average number of transactions per customer
T	Average number of items per transaction
S	Average length of maximal sequences
I	Average length of transactions within the maximal sequences
N	Number of different items

1. Test environment and datasets

WIS was written in C++. Experiments were performed on a sparcv9 processor operating at 1062 MHz, with 2048MB of memory. All experiments were performed on a Unix machine. In our experiments, a random generation function was used to generate weights of items. The IBM dataset generator is used to generate synthetic sequence datasets. It accepts essential parameters such as the number of sequences (customers), the average number of itemsets (transactions) in each sequence, the average number of items (products) in each itemset, and the number of different items in the dataset. Table 4 shows parameters and their meanings in this sequential dataset generation. More detail information can be found in [1]. To make our experiments fair, the synthetic datasets used in the experiments are the same as those used in SPAM [2].

2. Experimental results

A. Comparison of WIS with other algorithms

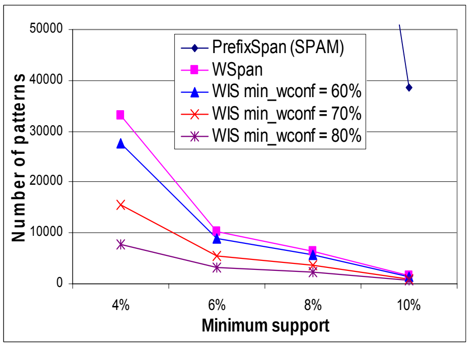


Fig. 1. Num of patterns

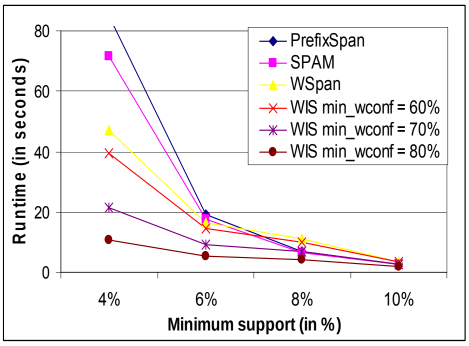


Fig. 2. Runtime

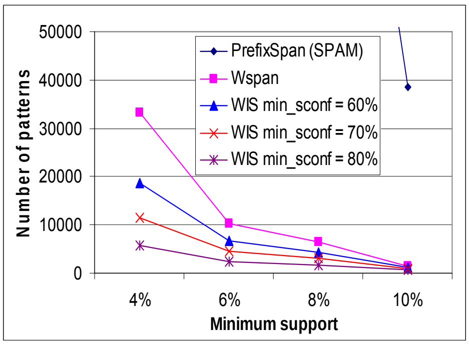


Fig. 3. Num of patterns

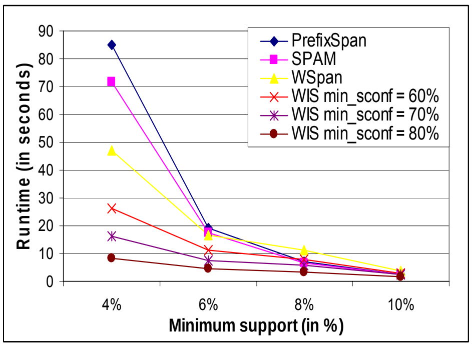


Fig. 4. Runtime

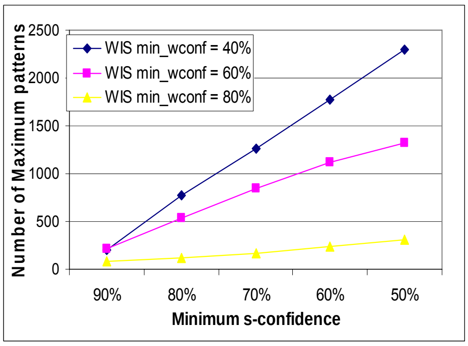


Fig. 5. Num of Maximum patterns (Min_sup = 2.0%)

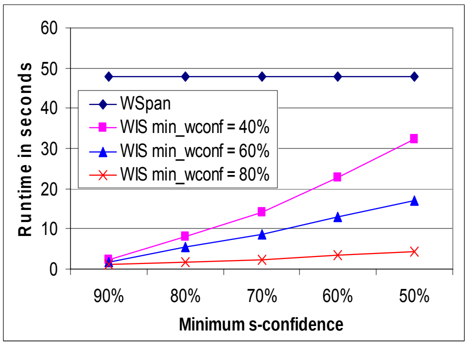


Fig. 6. Runtime (Min_sup = 2.0%)

D1C10T5S8I5 dataset

From Fig. 1 to Fig. 6, we evaluated the performance on the D1C10T5S8I5 dataset. Weights of items are set up between 0.3 and 0.6. In WIS, by using two measures s-confidence and w-confidence, sequential support / weight affinity patterns can be identified. In Fig. 1 and Fig. 2, the effect of sequential w-confidence is shown. Meanwhile, in Fig. 3 and Fig. 4, the results of using sequential s-confidence are presented. When the minimum support fixed, sequential patterns with a dissimilar weight or support levels are much pruned as the minimum w-confidence or s-confidence becomes higher. We can see that the effect of sequential w-confidence / s-confidence is better at lower minimum supports such as 4%. Specifically, the performance gaps increase as the minimum s-confidence / the minimum w-confidence becomes higher. PrefixSpan (SPAM) generates a huge number of sequential patterns with a minimum support of less than 10%. For instance, the numbers of patterns of PrefixSpan (SPAM) are 38,615 with a minimum support of 10%, 160,685 with a minimum support of 8%, and 443,639 with a minimum support of 6%. In Fig. 5 and Fig. 6, a minimum support threshold is fixed at 2% and the performance is evaluated as the minimum s-confidence and w-confidence are changed. In this test, we can check the effect of combination of two measures. In particular, we count the number of maximum sequential patterns without any inclusion and the runtime as the minimum s-confidence and minimum w-confidence are changed. From Fig. 5 and Fig. 6, we can know that the number of sequential affinity patterns and runtimes can be adjustable by changing two thresholds. For instance, with a minimum s-confidence of 80% and a minimum w-confidence of 80%, the runtime is less than 2 seconds. However, the runtimes are more than 10 seconds by using s-confidence and w-confidence thresholds of 60% respectively. Sequential s-confidence and w-confidence are effectively used to prune weak affinity patterns in terms of support and weight. It may not be surprising that the number of sequential patterns and the runtime are reduced. However, in previous sequential pattern mining algorithms such as PrefixSpan and SPAM, sequential weight (support) affinity patterns cannot be detected. Although the weight constraints are used in WSpan, the correlated sequential patterns with similar weight /support levels are not mined.

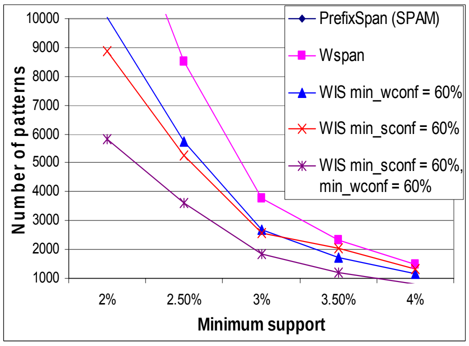


Fig. 7. Num of patterns

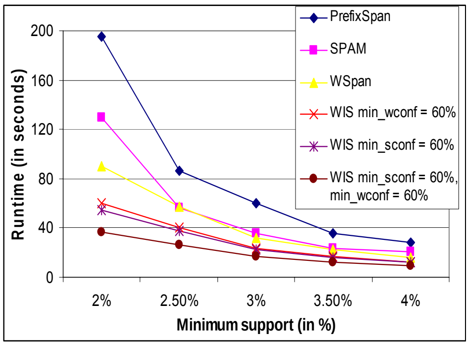


Fig. 8. Runtime

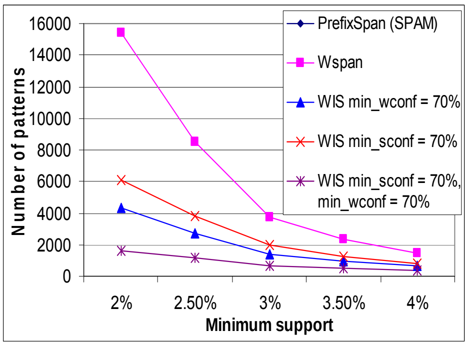


Fig. 9. Num of patterns

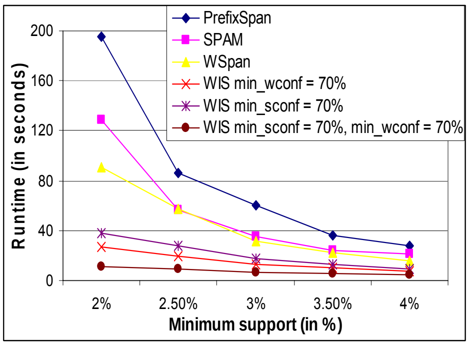
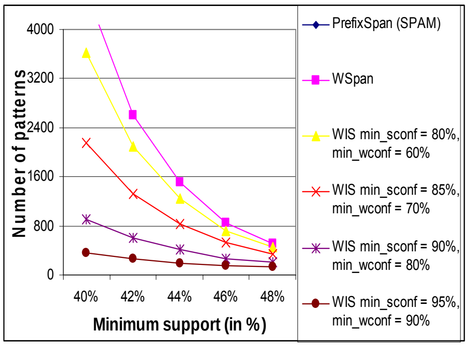


Fig. 10. Runtime

D7C7T7S7I7 dataset

From Fig. 7 to Fig. 10, we report the evaluation results for D7C7T7S7I7 dataset. We set up weights from 0.1 to 0.3 for Fig. 7 to Fig. 10. The main performance difference between WIS and other algorithms such as PrefixSpan, SPAM and WSpan results from using sequential s-confidence and/or sequential w-confidence. By using sequential s-confidence and/or w-confidence thresholds, correlated sequential patterns with a higher level of affinity in terms of support and/or weight are generated. We can also see that the performance of using both sequential s-confidence and w-confidence is better than using either one alone. In addition, given a minimum s-confidence and w-confidence at 60%, the effect of sequential s-confidence is better than that of sequential w-confidence. However, at a threshold of 70%, the performance of sequential w-confidence becomes better than that of sequential s-confidence. In Fig. 7 and Fig. 8, we could not show the number of patterns generated by PrefixSpan (SPAM) because the number of patterns becomes huge at less than 4%. For example, the number of patterns in PrefixSpan (SPAM) are 170,965 with the min_sup of 4%, 292,161 with the min_sup of 3.5%, 439,953 with a minimum support of 3.0%, 701,760 with min_sup of 2.5%, and 1,646,818 with min_sup of 2%.



Fg. 11. Num of patterns

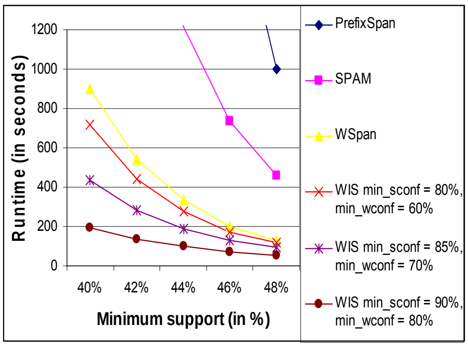


Fig. 12. Runtime

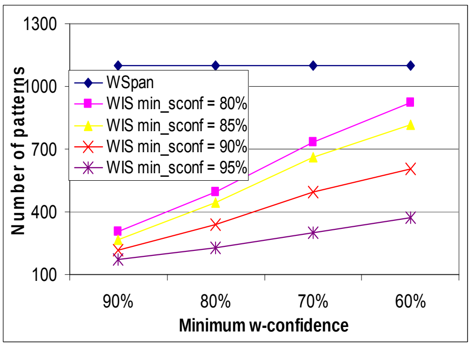


Fig. 13. Num of Maximum patterns (Min_sup = 45%)

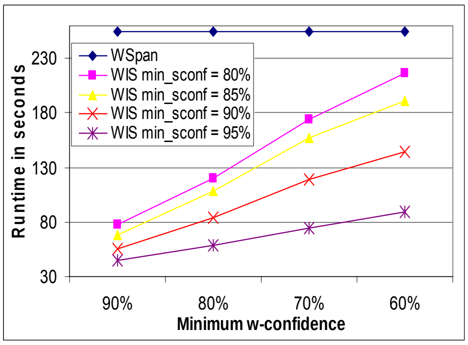


Fig. 14. Runtime (Min_sup = 45%)

D15C15T15S15I15 dataset

Fig.11 to Fig. 14 demonstrates the results of a performance test using the D15C15T15S15I15 dataset with weights from 0.4 to 0.8. When w-confidence threshold is lowered, the performance difference of sequential w-confidence measure becomes larger. At higher weight confidences, such as 90%, the performance of WIS becomes better. We can see that the number of (maximum) sequential affinity patterns for WIS is decreased as the sequential s-confidence and w-confidence are increased. Recall that WSpan can also adjust the number of patterns by resetting the weight range, although we fixed the weight range in these tests. Decreasing a weight range means more priority is given to a support measure. However, WIS prunes the (maximum) sequential patterns with weak s-affinity and/or w-affinity. If users increase the sequential w-confidence threshold, it means they want patterns that involve items with higher w-affinity. Users can choose their level of interest and use a sequential s-confidence and/or w-confidence.

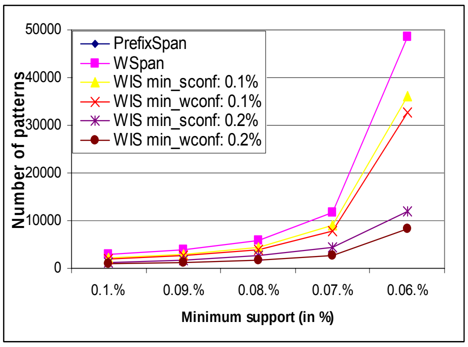


Fig. 15. Num of Maximum patterns

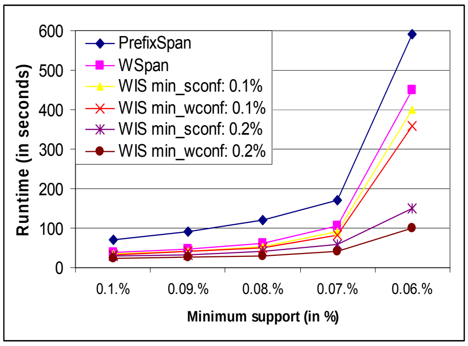


Fig. 16. Runtime

Real Gazelle dataset: We report the evaluation results for Gazelle dataset. The Gazelle dataset is click stream data which is used in KDDCup-2000. Product pages by a customer in a session are considered as an itemset and difference sessions by one use is thought as a sequence. For more detail information, we can refer to [15, 23]. In this experiment, minimum supports are used with normalized weights, 0.1 – 0.9. From Fig. 15 and Fig. 16, we can see that WIS detects support and weight affinity sequential patterns with sequential s-confidence and w-confidence respectively. Moreover, by using higher s-confidence and w-confidence thresholds, strong affinity sequential patterns are mined.

B. Quality of weighted sequential patterns with s-affinity and/or w-affinity

In previous evaluation, we showed that the sequential s-confidence and w-confidence can be used to detect sequential patterns with the s-affinity and/or w-affinity. In all test datasets, items are expressed as integer values so it is difficult to understand the meaning of items and discovered sequential patterns. In this evaluation, the D7C7T5S4I2.5 dataset is used to illustrate the quality of affinity sequential pattern mining. A minimum support is set to 2.5% and weights are set as 0.1 – 0.3. We analyzed the patterns discovered by WIS. We compared the patterns mined by WIS with those of PrefixSpan (SPAM) and WSpan. For example, sequential patterns <(2) (45) (27, 91) (17, 70)>:12 and <(1, 61, 91) (27) (91) (70)>:12 are mined by PrefixSpan (SPAM) and sequential patterns <(70) (61) (45, 61)>:40 and <(91) (47) (91) (27, 91)>:47 are discovered by WSpan. However, these patterns are all pruned by s-confidence (min_sconf = 0.6) and w-confidence (min_wconf = 0.6) respectively. In other words, these sequential patterns are weak affinity patterns. Although the minimum support is increased, these weak affinity patterns such as <(2) (45) (27, 91) (17, 70)>:12 and <(1, 61, 91) (27) (91) (70)>:12 are found by PrefixSpan (SPAM). In addition, although the minimum support threshold is increased and/or the weight range is changed, the weak affinity patterns such as <(70) (61) (45, 61)>:40 and <(91) (47) (91) (27, 91)>:47 are still discovered in result patterns in WSpan. The weak affinity patterns can be effectively pruned by sequential s-confidence and/or w-confidence.

C. Scalability Test

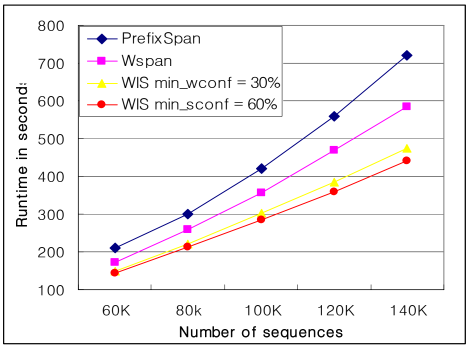


Fig. 17. Scalability test of s-confidence and w-confidence

DxC2.5T5S4I2.5 dataset

The DxC2.5T5S4I2.5 dataset was used to test scalability with the number of sequences in the database. In this test, we set a minimum support as 0.4% and weights as 0.1 to 0.5. To show differences clearly among algorithms, the number of sequences in X-axis is increased up to 140k and the scalability test is performed. In Fig. 17, the slope differences among the algorithms become bigger as the number of sequences in x-axis is increased. We can see that the slope ratio of WIS is lower than those of PrefixSpan and WSpan. When WIS is compared with PrefixSpan, definitely, the scalability of WIS is better than that of PrefixSpan. Moreover, WIS shows somewhat better scalability than WSpan. As a result, WIS shows better scalability than other two algorithms in terms of number of sequences.

V. Future research

As future work, there are a few things to be researched. First, WIS is a main memory based sequential pattern mining algorithm but this assumption give a limitation when the database is very large or the minimum threshold becomes low. WIS should be extended as a disk based method. Second, to set up weights of items, prices of items can be used as a weight factor in market basket data and the prices of items can be normalized into a weight range. However, we should think of ways to assign weights to items in other types of datasets such as web log data, biomedical data, DNA data and data used in other applications. Third, WIS uses three thresholds which are the minimum support, the minimum s-confidence and/or w-confidence. Effective settings of thresholds are essential although it is the common problem of all threshold-based mining algorithms. For example, the sequential weak affinity patterns can be much pruned by increasing the difference between a maximum weight and a minimum weight in a sequence database although the minimum w-confidence and/or s-confidence are fixed. Meanwhile, the effect of the w-confidence can be reduced by decreasing the difference between the maximum weight and the minimum weight. We need to have more research and experiment to give guidance of how efficiently to set up the thresholds. Finally, improved techniques such as sequential pattern mining using pseudo projection [15] or bitmap representation [2] have been suggested. In future work, WIS can be extended by using a combination of these techniques.

VI. Conclusion

In this paper, we studied the problem of mining weighted sequential affinity patterns. We introduced sequential s-confidence and w-confidence measures and the concept of weighted interesting sequential patterns by using the two measures. The sequential s-confidence and/or w-confidence measures can be used to prune weak sequential patterns involving items from dissimilar support and/or weight levels. The extensive performance analysis shows that WIS is efficient and scalable in sequential affinity pattern mining. In addition, from the experiments, we showed that WIS algorithm is very effective to detect support and/or weight affinity sequential patterns.

References

[1] R. Agrawal, and R. Srikant, “Mining Sequential Patterns”, *ICDE*, 1995.

[2] J. Ayres, J. Gehrke, T. Yiu, and J. Flannick, “Sequential Pattern Mining using A Bitmap Representation”, *SIGKDD’02*, 2002.

[3] D. Y. Chiu, Y. h. Wu, and A. L.P. Chen, “An Efficient Algorithm for Mining Frequent Sequences by a New Strategy without Support Counting”, *ICDE’04*, 2004.

[4] H. Cheng, X. Yan, and J. Han, “IncSpan: Incremental Mining of Sequential Patterns in Large Databases”, *SIGKDD’04*, 2004.

[5] H. Chung, X. Yan, and J. Han, “SeqIndex: Indexing Sequences by Sequential Pattern Analysis”, *SDM’05*, 2005.

[6] M. Ester, “A Top-Down Method for Mining Most Specific Frequent Patterns in Biological Sequence Data”, *SDM’04*, 2004.

[7] M. Garofalakis, R. Rastogi, and K. shim, “SPIRIT: Sequential pattern mining with regular expression constraints”, *VLDB’99*, 1999.

[8] T. Haines, R. Tibshirani, and J. Friedman, “The elements of statistical learning: Data mining, inference, and prediction, Springer, 2001.

[9] J. Han, J. Pei, B. Mortazavi-Asi, Q. Chen, U. Dayal, and M. C. Hsu, “FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining”, *SIGKDD’00*, 2000.

[10] H. Kum, J. Pei, W. Wang, and D. Duncan, “ApproxMAP: Approximate Mining of Consensus Sequential Patterns”, *SDM’03*, 2003.

[11] S. Lee, and L. D. Raedt, “Constraint Based Mining of First Order Sequences in SeqLog”, Database Support for Data Mining, 2004.

[12] H. A. Lorincz, and J. F. Boulicaut, “Mining frequent sequential patterns under regular expressions: a highly adaptive strategy for pushing constraints”, *SDM’03*, 2003.

[13] J. Pei, J. Han, B. Mortazavi-Asi, and H. Pino, “PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth”, *ICDE’01*, 2001.

[14] J. Pei, J. Han, and W. Wang, “Mining Sequential Patterns with Constraints in Large Databases”, *CIKM’02*, 2002.

[15] J. Pei, J. Han, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.C. Hsu, “Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach”, *IEEE Transactions on Knowledge and Data Engineering*, Oct, 2004.

[16] H. Pinto, J. Han, J. Pei, and K. Wang, “Multi-domensional Sequence Pattern Mining”, *CIKM’01*, 2001.

[17] M. Seno and G. Karypis, “SLPMiner: An Algorithm for Finding Frequent Sequential Patterns Using Length-Decreasing Support Constraints”, *ICDM’02*, 2002.

[18] R. Srikant, and R. Agrawal, “Mining Sequential Patterns: Generalizations and Performance Improvements”, *EDBT*, 1996.

[19] P. Tzvetkov, X. Yan, and J. Han, “TSP: Mining Top-K Closed Sequential Patterns”, *ICDM’03*, 2003.

[20] J. Wang, and J. Han, “BIDE: Efficient Mining of Frequent Closed Sequences”, *ICDE’04*, 2004.

[21] K. Wang, Y. Xu, and J. X. Yu, “Scalable Sequential Pattern Mining for Biological Sequences”, CIKM’04, 2004.

[22] H. Xiong, P. N. Tan and V. Kumar, “Mining Strong Affinity Association Patterns in Data Sets with Skewed Support Distribution”, *ICDM`03*, 2003.

[23] X. Yan, J. Han, and R. Afshar, “CloSpan: Mining Closed Sequential Patterns in Large Datasets”, *SDM’03*, 2003.

[24] J. Yang, P.S. Yu, W. Wang and J. Han, “Mining long sequential patterns in noisy environment”, *SIGMOD’02*, 2002.

[25] U. Yun, and J. J. Leggett, “WFIM: Weighted Frequent Itemset Mining with a weight range and a minimum weight”, *SDM`05*, April 2005.

[26] U. Yun, and J. J. Leggett, “WLPMiner: Weighted Frequent Pattern Mining with Length-decreasing support constraints”, *PAKDD`05*, May 2005.

[27] U. Yun, and J. J. Leggett, “WSpan: Weighted Sequential pattern mining in large sequence databases”, *IEEE IS`06*, 2006.

[28] K. W. Min, K. W. Nam, and J. W. Kim, “Multilevel Location Trigger in Distributed Mobile Environments for Location-Based Services”, *ETRI Journal, Feb. 2007*.

[29] U. Yun, “Mining lossless closed frequent patterns with weight constraints”, Knowledge based systems, 210 (2007) 86-97.

[30] M. Zaki, SPADE: “An efficient algorithm for mining frequent sequences”, *Machine Learning*, 2001.