WIS: 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. Most sequential pattern mining algorithms use a minimum support threshold to prune the combinatorial search space. This strategy provides basic pruning; however, it cannot mine correlated sequential patterns with similar support and/or weight levels. If the minimum support is low, many spurious patterns having items with different support levels are found; if the minimum support is high, meaningful sequential patterns with low support levels may be missed. We present a new algorithm, weighted interesting sequential (WIS) pattern mining based on a pattern growth method in which new measures, sequential s-confidence and w-confidence, are suggested. Using these measures, weighted interesting sequential patterns with similar levels of support and/or weight are mined. The WIS algorithm gives a balance between the measures of support and weight, and considers correlation between items within sequential patterns. A performance analysis 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 is used to find frequent subsequences as patterns in a sequence database. Several sequential pattern mining algorithms have been developed, such as constraint-based sequential pattern mining [1]-[4], closed sequential pattern mining [5]-[7], approximate sequence mining [8], multi-dimensional sequence pattern mining [9], sequence mining in a noisy environment [10], biological sequence mining [11], [12], incremental sequence mining [13], and sequence indexing [14]. 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 [15], [16] such as the generation and testing of all candidates and the repeated scanning of a large proportion of the sequence database, sequential pattern growth approaches [17]-[19] 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 [20] has been suggested. Most algorithms use a support threshold to prune the search space. This strategy provides basic pruning but 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.

Manuscript received Mar. 20, 2006; revised Feb. 14, 2007.

This work was supported by the IT R&D program of MIC/IITA [2007-S001-01, VDMS (Vehicle & Driver Management System) Technology Development].

Unil Yun (phone: + 82 42 869 1627, email: yunei@etri.re.kr) was with Department of Computer Science, Texas A&M University, Texas, USA, and is now with Telematics & USN Research Division, ETRI, Daejeon, Korea.

- Motivating Examples

Here, we give an example of how this work can be applied to market basket data for marketing planning purposes. In sequential pattern mining, a sequential pattern {(bread, milk) (diaper, beer)} can be easily discovered with a 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 considered weak support affinity sequential patterns. For instance, {(gold ring, bronze ring) (vodka, beer)} is a possible weak support affinity sequential pattern 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 item sets 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 [21]. The expensive items within the itemsets have low frequencies so the sequential patterns including such itemsets are not detected with 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. Moreover, giving weights of items according to their priority or importance, the sequential weight affinity patterns with similar weight levels can be found.

In real business, it is useful for marketing managers to know which items in a list have similar profit or frequency levels within an acceptable error range. Trend analyzers are interested in analyzing itemsets with similar levels of profit or selling prices, and customers want to find items with similar price levels so that they can buy interesting items within their budgets. According to the requirement of real applications, data analysis should be performed. From the results of the data analysis, marketing decisions about item prices can be more effectively determined. Therefore, the comparison and analysis of correlated sequential patterns is essential for planning future marketing. Analyzing correlated sequential patterns with the support/weight affinity (s-affinity/w-affinity) can be useful for grouping customers for marketing purposes, focusing on profitable items, identifying interesting itemsets or sequences with similar support/weight levels, and planning marketing policies more accurately with the association structure of different products.

In this paper, we propose an efficient sequential pattern mining algorithm called weighted interesting sequential (WIS) pattern mining based on the pattern growth approach [18], [19]. Based on the new measures of s-confidence and w-confidence, sequential s-affinity/w-affinity patterns are defined. Here, weight/support affinity means the degree to which items within a sequential pattern have similar characteristic in terms of their weight/support values. The sequential s-confidence measure is used to detect s-affinity patterns, and the sequential w-confidence measure is used to identify w-affinity patterns. We show that the two measures satisfy the anti-monotone property and satisfy the cross support/weight property. With the two properties, weak affinity patterns are effectively eliminated. Using this framework, we developed the WIS algorithm to detect correlated sequential patterns with s-affinity/w-affinity by integrating 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 to comparative analysis queries and can 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.

The remainder of the paper is organized as follows. In section II, we define the problem and give a brief summary of related works. In section III, we present WIS sequential pattern mining. Section IV summarizes extensive experimental results. Finally, possible future research and our conclusions are presented in sections V and VI, respectively.

Table 1. 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 $I = \{i_1, i_2 \cdots, i_n\}$ be a unique set of items. Sequence S is an ordered list of itemsets, denoted as $\langle s_1, s_2, \cdots, s_m \rangle$, where s_j is an itemset which is also called an element of the sequence, and $s_j \subseteq I$. That is, $S = \langle s_1, s_2, \cdots, s_m \rangle$ and s_i is $(x_{i1}x_{i2} \cdots x_{ik})$, where x_{it} is an item in itemset s_i. Brackets are omitted if an itemset has only one item. As shown in Table 1, a sequence database (SDB) = $\{S_1, S_2, \cdots, S_n\}$, 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 the SDB has five itemsets: a, (abc), (ac), d, and (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; therefore, the sequence is called a 9-sequence. A sequence $\alpha = \langle X_1, X_2, \cdots, X_n \rangle$ is called a subsequence ($\alpha \sqsubseteq \beta$) of another sequence, $\beta = \langle Y_1, Y_2, \cdots, Y_m \rangle$ ($n \le m$), and β is called a supersequence of the sequence α if there is an integer $1 \le i_1 < \cdots < i_n \le m$ such that $X_1 \subseteq Y_{i1}$, $X_2 \subseteq Y_{i2}, \cdots, \subseteq X_n \subseteq Y_{in}$. For example, sequence ⟨a (bc) d⟩ is a subsequence of ⟨a (abc) (ac) d (cf)⟩ since a ⊆ a, (bc) ⊆ (abc), and d ⊆ d. A tuple (sid, S) is said to contain a sequence α if the sequence S is a supersequence of α ($\alpha \sqsubseteq$ s). The support of sequence α in the SDB is the number of sequences in the SDB that contain the sequence α (support (α) = |{<sid, S> | (<sid, S> ∈ SDB) ∧ ($\alpha \sqsubseteq$ S)}|). Given a support threshold, min_sup, sequence α is called a frequent sequential pattern in the sequence database if the support of sequence α is no less than the minimum support threshold (support (α) ≥ min_sup). For instance, the sequence <a (bc) d> is a frequent sequential pattern because sequences 10 and 20 contain subsequence S = ⟨a (bc) d⟩, and the support of the sequence is 2, which is equal to the minimum support (2). 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 supersequences or the complete set of maximal frequent sequences. The anti-monotone property [15] 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, the generalized sequential pattern (GSP) algorithm [16] mines sequential patterns based on an *a priori*-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 [17] 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 *a priori*-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 [18], [19], a more efficient pattern growth algorithm, improves the mining process. The main idea of PrefixSpan is to examine only the prefix subsequences and to project only their corresponding suffix subsequences into projected databases. In each projected database, sequential patterns are grown by exploring only local frequent patterns. In the sequential pattern discovery using equivalence classes (SPADE) algorithm [22], a vertical id-list data format was implemented and frequent sequence enumeration is performed by a simple join on id lists. The SPADE algorithm can be considered an extension of vertical-format-based frequent pattern mining. The sequential pattern mining (SPAM) algorithm [23] utilizes depth-first traversal of the search space combined with a vertical bitmap representation to store each sequence. Efficient sequential pattern mining algorithms [24] have been developed such as constraint-based sequential pattern mining [1]-[4], approximate sequential pattern mining with a weighted sequence structure [8], sequential pattern mining without using support thresholds [5], and closed sequential pattern mining [6], [7]. These approaches may mine patterns efficiently and reduce the number of patterns. As given in the motivating example in section I.1, the weight/support affinity sequential pattern can be useful, but affinity sequential patterns cannot be detected with 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 differ in their importance. For this reason, weighted sequential pattern mining. The weighted sequential pattern mining (WSpan) algorithm [20] and weighted frequent pattern mining [25]-[27] have been suggested. In WSpan, the items within a sequence are given different weights in the sequence database. The main concern in WSpan is that the anti-monotone property [15] 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 [18], [19], WSpan uses

approximate weighted support within normalized weights to prune weighted infrequent sequential patterns but maintain the anti-monotone property.

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

III. WIS Pattern Mining

In this section, the WIS algorithm is presented. We give actual examples to illustrate the effect of sequential support/weight confidence and explain 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 1. *Prefix and suffix of a sequence*

Suppose that all the items within itemsets in each sequence are listed in alphabetical order. Given the sequence $\alpha = \langle e_1 e_2 \cdots e_n \rangle$ (in which each e_i means a frequent element in α), the sequence $\beta = \langle e'_1 e'_2 \cdots e'_m \rangle$ ($m \le n$) is called a prefix of sequence α if (1) $e_i = e'_i$ for ($i \le m-1$), (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, the sequence $\gamma \langle e''_m e_{m+1} \cdots e'_n \rangle$ is called a suffix of sequence α with regard to 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 sequence S are frequent in S. In addition, <(abc) (ac) d (cf)> is a suffix of the prefix <a>, <(_bc) (ac) d (cf)> is the suffix of the prefix <aa>, and <(_c) (ac) d (cf)> is the suffix corresponding to the prefix <a (ab)>.

Definition 2. *Projected database*

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

Example 2. Given the 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>, <(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 for market basket data.

Table 2. 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. *Weight of a sequential pattern and weighted frequent pattern*

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

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

The 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 the sequential pattern is no less than a minimum threshold.

Definition 4. *Weight range and maximum weight*

The weight of an item is a non-negative real number which shows the importance of that 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 (WR) is exploited to restrict weights of items. A maximum weight (MaxW) is defined as the value of the maximum weight of items in a sequence database or a projected sequence database.

As previously 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 wide variation. As seen in the retail database in Table 2, the variation in prices is so wide that prices cannot be directly used as weights. Therefore, within a specific WR, a normalization process is needed to adjust for differences among different data items in order to create a common basis for comparison. Based on the normalization process, the final weights of items can be decided. In Table 2, the weights of items are given between 0.01 and 1.2. The maximum weight is 1.2, which is the weight 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 ≤weight ≤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 the SDB in Table 1 and a minimum support of 2, the set of items in the database, namely, 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 the 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)> are 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, the MaxWs 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 5. *Sequential support-confidence* (*s-confidence*)

Support-confidence of a sequential pattern $S = \{s_1, s_2, \cdots, s_m\}$ and s_i is $(x_{i1}x_{i2}\cdots x_{ik})$, where x_{it} is an item in the itemset s_i denoted by sequential s-confidence, is a measure which 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

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

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 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.

There 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 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 considered a weak sequential s-affinity pattern.

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

Given a sequential pattern from definition 5, $\text{Max}_{(1\le m''\le m,\,1\le k''\le \text{length}(s_{m''}))}\{\text{support}(\{x_{m''k''}\subseteq s_{m''}\})\}$ of sequential pattern S is always greater than or equal to that of a sub-sequence of sequential pattern S, and $\text{Min}_{(1\le m'\le m,\,1\le k'\le \text{length}(s_{m'}))}\{\text{support}(\{x_{m'k'}\subseteq s_{m'}\})\}$ of pattern S is always less than or equal to that of a subset of sequential pattern S. Therefore, we know that

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

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

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

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 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 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 pattern S is a 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 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 the 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 5, assume that there is a cross support sequential pattern S = {s_1, s_2,⋯, s_m} which contains at least two items X and Y such that support ({X}) / support ({Y}) < t where 0 < t < 1.

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

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

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

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

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 the sequential s-confidence threshold, t.

B. Sequential W-Affinity Pattern

Definition 7. *Sequential weight-confidence* (*w-confidence*)

Weight-confidence of a sequential pattern S = {s_1, s_2,⋯, s_m} and s_i is ($x_{i1}x_{i2}\cdots x_{ik}$), 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

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

Definition 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 the minimum weight confidence (min_wconf). If not, the sequential pattern is considered a weak w-affinity pattern.

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

From definition 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 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 pattern S is always less than or equal to that of a subset of sequential pattern S. Therefore, we know that

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

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

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

In other words, if w-confidence of a sequential pattern S is no less than the min_wconf, so is that of 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 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 sequential pattern Y. Then, the average weight of sequential pattern S and sequential pattern S′ are 0.425 and 0.55, respectively. The sequential w-confidence (S) is 0.29 (2/7) and sequential w-confidence (S′) is 0.56 (4/7). Therefore, 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 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 the 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 7, assume that there is a cross weight sequential pattern $S = \{s_1, s_2, \cdots, s_m\}$ which contains at least two items, Z and W, such that weight ({Z}) / weight ({W}) < t, where $0 < t < 1$.

$$W-conf(S)=\frac{Min_{1\le m'\le m,1\le k'\le length(s_{m'})}\{weight(\{x_{m'k'}\subseteq s_{m'}\})\}}{Max_{1\le m''\le m,1\le k''\le length(s_{m''})}\{weight(\{x_{m''k''}\subseteq s_{m''}\})\}}$$

$$\le\frac{Min_{1\le m'\le m,1\le k'\le length(s_{m'})}\{\cdots,weight(\{Z\}),\cdots,weight(\{W\}),\cdots\}}{Max_{1\le m''\le m,1\le k''\le length(s_{m''})}\{\cdots,weight(\{Z\}),\cdots,weight(\{W\}),\cdots\}}$$

$$\le\frac{weight(\{Z\})}{Max_{1\le m''\le m,1\le k''\le length(s_{m''})}\{\cdots,weight(\{Z\}),\cdots,weight(\{W\}),\cdots\}}$$

$$\le\frac{weight(\{Z\})}{weight(\{W\})}$$

$$< t\,.$$

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

Example 6. Pruning examples of the anti-monotone and cross weight properties on the w-confidence

From the anti-monotone property of sequential w-confidence, if the w-confidence of a sequential pattern is less than 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 the 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 weights in ascending order: {<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 groups: {e, c, g} and {a, d, h, b, f}, any pattern including items from both groups is the cross weight sequential pattern with 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 <ea>, <ed>, <ch>, <cb>, and <gb> have to be generated as candidate patterns. They are pruned later by computing the w-confidence values of the sequential patterns. Note that 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). Similarly, the anti-monotone property and the cross support sequential pattern property can be used to prune weak s-affinity patterns.

3. WIS Patterns

Definition 9. *WIS 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 another item with a higher weight is added. Using the maximum weighted support, the anti-monotone property can be maintained. In other words, if the maximum weighted support (support (S) * MaxW) of a sequential pattern S is less than the minimum support, no super pattern can be a weighted sequential frequent pattern so the pattern can be pruned. During the mining process, weighted infrequent items are pruned and the weights of the weighted infrequent items are not considered as MaxW although the weights of the items are high. In this way, 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. According to 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. According to 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.*

In WIS, a weight range is chosen 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; 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 pruning condition 1. However, the w-confidences (0.33) of sequential patterns K and K′ are the same despite having different weight ranges. We know that sequential w-confidence is defined as the ratio of the minimum weight of items within a sequential pattern to the maximum weight of items within the sequential pattern. Therefore, if the 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 a weight (support) affinity level which shows how many items within a pattern have similar characteristic in terms of weight (support) values among the items. The weight (support) affinity levels are calculated using w-confidence and s-confidence, respectively.

Observation 2. Lemma 5 states that sequential w-confidence in pruning condition 3 can be applied irrespective of different weight ranges. The same applies in sequential s-confidence in 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 versus W-Confidence

Sequential s-confidence is a support measure used to identify sequential s-affinity patterns; sequential w-confidence is a weight measure which 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 versus Weighted Support Constraint

Although the weighted support constraint considers weight and support, it cannot detect affinity patterns. The use of a weight constraint in WSpan [20] can generate weak affinity patterns containing items with different weight levels or can 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-confidence of these patterns may not satisfy the minimum w-confidence if they are sequential patterns with dissimilar weight levels.

C. Sequential S-Confidence versus Support Constraint

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

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, weak affinity patterns containing items with dissimilar support/weight levels may be useful in other applications. The novelty of our approach is that WIS can identify strong or weak support (weight) affinity patterns by applying the s-confidence (w-confidence) measure. Previously proposed sequential pattern mining algorithms cannot find correlated patterns.

4. Mining WIS Patterns with S-Affinity and/or W-Affinity

We developed the WIS algorithm to detect correlated patterns with s-affinity and/or w-affinity. As a mining example, we show how to mine affinity sequential patterns using a prefix-based projection approach [19] which computes local frequent sequential patterns of a prefix by scanning its projected database. The projection is based on a frequent prefix. We use the 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 WIS, the 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. After the first scan of the sequence database, we know that the 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 support of

each item is greater than or equal to the minimum support (2). Using MaxW (0.8), the weighted support of each item is calculated, and weighted infrequent items are pruned according to pruning condition 1 in definition 9. For example, the weighted support (6 * 0.8) of an item <a> is greater than the minimum support, so the item is a 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 the following prefixes: (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.

Step 3.1. Find affinity sequential patterns with the prefix **<a>**

We only collect the sequences which have <a>. Additionally, in a sequence containing prefix <a>, only the subsequence prefixed with the first occurrence of the prefix <a> should be considered. For example, in the sequence <a (abc) (ac) d (cf)>, only the subsequence <(abc) (ac) d (cf)> is considered, and in the sequence <(ad) abc (bcd) (ae) bcde>, only the suffix sequence <(_d) abc (bcd) (ae) bcde> is collected. The sequences in the 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 1 as the support, are removed by the weighted support constraint since the weighted support (0.8) of multiplying the support (1) of the sequences with MaxW (0.8), is less than the minimum support (2). In addition, a local item, e:5, is pruned by the sequential w-confidence. The candidate pattern, from 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). 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 the 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.

Step 3.1.1. 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. The sequential s-confidence and w-confidence of these patterns are no less than the minimum s-confidence and the minimum w-confidence respectively. Recursively, sequential patterns with the prefix <aa> are partitioned and mined.

Step 3.1.2. 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 the 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.

Step 3.1.3. 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.

Step 3.1.4. 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 the <ad> projected database returns a sequential pattern <adc>:4. Recursively, sequential patterns with the prefix <ad> are partitioned and mined.

Step 3.1.5. 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 9.

Step 3.1.6. 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.

Step 3.2. 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 4 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. 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. In WIS, the prefix projected sequential pattern growth approach is used 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 [23] or *a priori*-based algorithms [15], [16].

Table 4. 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)

5. WIS Algorithm

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

Input:
- sequence database: SDB
- support threshold: min_sup
- w-confidence threshold: min_wconf
- s-confidence threshold: min_sconf

Output: The complete set of weighted sequential patterns.

- Let WSP be the set of weighted sequential patterns that satisfy the constraints. Initialize WSP ← {};
- 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.
- For each weighted frequent item, β, in SDB
  Call WIS (WSP, <β>, 1, SDB)
  End for
  End

Procedure WIS (WSP, α, L, S|α)

Parameters:
- α: a weighted sequential pattern that satisfies the above pruning conditions
- L: the length of α
- $S|_{\alpha}$: the sequence database. SDB if α is null; otherwise, it is the α-projected database

Step 1. Scan S $|_{\alpha}$ 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.

Step 2. For each weighted frequent item β,
add it to α to form a sequential pattern α′ and output α′;

Step 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 the 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 [28], such as analyzing retail data, telecommunications data, financial data, and so on. First, correlated sequential patterns with w-affinity/s-affinity can be applied in analyzing customer buying patterns and planning marketing policies. The association structure of different products may be studied by analyzing the sequential time data. Second, the techniques for mining patterns with different levels of support and/or weight can be applied to find fraudulent users and their usage patterns in crimes, such as money laundering, the purchase of expensive items within a short time (with stolen information), the use of stolen mobile phones, and other financial crimes. The usage (transaction) frequency for each user is usually regular, and customers have unique 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. Pattern mining with s-affinity and/or w-affinity can help determine the kinds of genes that are likely to occur together in target samples.

IV. Performance Evaluation

Using various datasets, we tested the performance of WIS in comparison with recently developed algorithms: PrefixSpan [19], WSpan [20], and SPAM [23]. These were chosen 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 ascertain 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 (Figs. 5 and 13) are checked. Second, we present the efficiency of the WIS algorithm, and the quality of weighted affinity sequential patterns. Finally, we demonstrate that WIS has good scalability against the number of sequences in the datasets.

Table 5. 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 2048 MB of memory. All experiments were performed on a Unix machine. In our experiments, a random generation function was used to generate the weights of items. The IBM dataset generator was 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 5 shows parameters and their meanings in this sequential dataset generation. More detailed information can be found in [15]. To make our experiments fair, the synthetic datasets used

in the experiments are the same as those used in SPAM [23].

2. Experimental Results

A. Comparison of WIS with Other Algorithms

Figures 1 to 6 show the results of performance evaluation based on the D1C10T5S8I5 dataset. Weights of items are set to between 0.3 and 0.6. In WIS, using the s-confidence and w-confidence measures, sequential support/weight affinity patterns can be identified. In Figs. 1 and 2, the effect of sequential w-confidence is shown.

In Figs. 3 and 4, the results of using sequential s-confidence are presented. When the minimum support is fixed, sequential patterns with dissimilar weight or support levels are strongly pruned as the minimum w-confidence or s-confidence becomes higher. 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

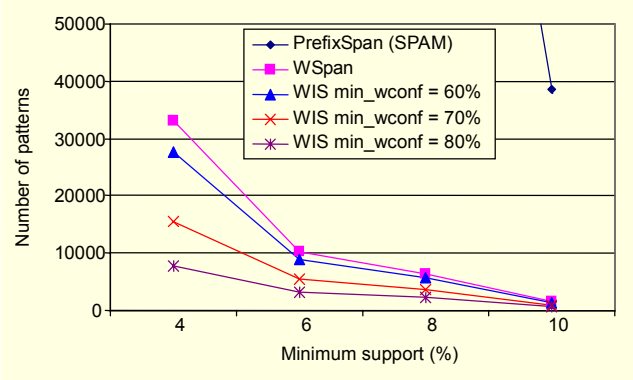

Fig. 1. Number of (w-affinity) patterns (min_wconf: 0.6-0.8).

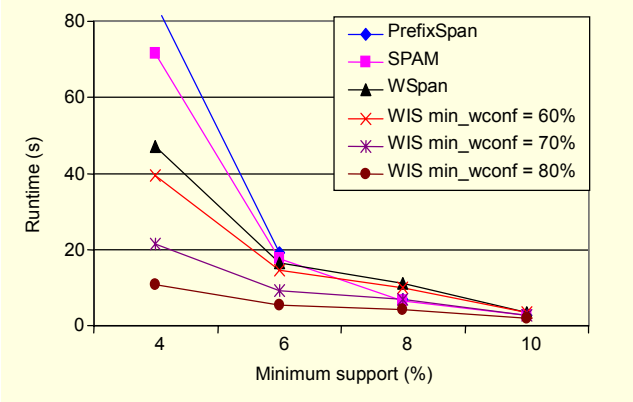

Fig. 2. Runtime (min_wconf: 0.6-0.8).

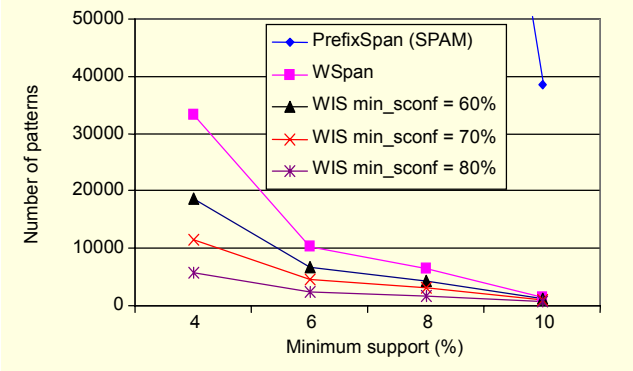

Fig. 3. Number of (s-affinity) patterns (min_sconf: 0.6-0.8).

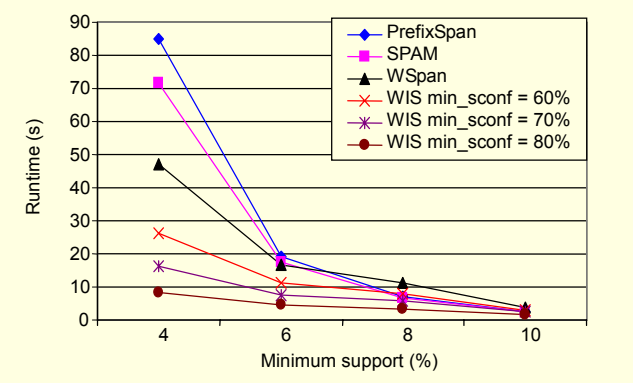

Fig. 4. Runtime (min_sconf: 0.6-0.8).

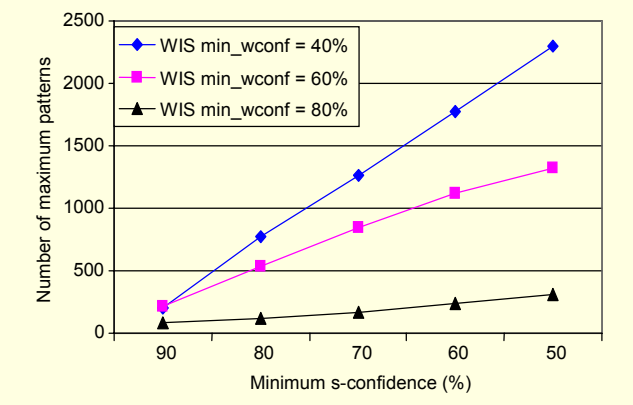

Fig. 5. Number of maximum (w-affinity) patterns (min_sup = 2.0%, min_wconf: 0.4-0.8).

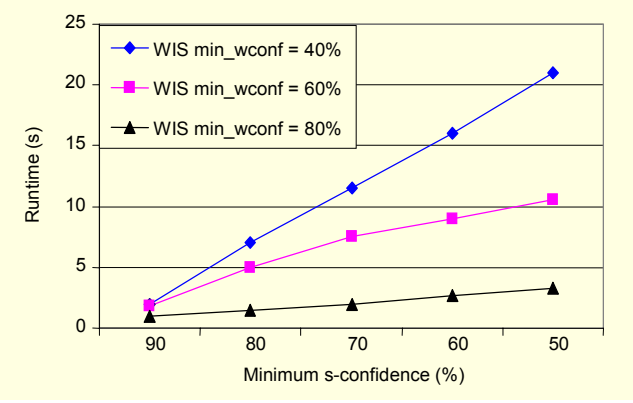

Fig. 6. Runtime (min_sup = 2.0%, min_wconf: 0.4-0.8).

s-confidence/minimum w-confidence rises.

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 Figs. 5 and 6, the minimum support threshold is fixed at 2%. The performance is evaluated as the minimum s-confidence and w-confidence changes. In this test, we check the effect of the combination of the two measures. In particular, we count the number of maximum sequential patterns without any inclusion and count the runtime as the minimum s-confidence and minimum w-confidence changes. Figures 5 and 6 demonstrate that the number of sequential affinity patterns and runtimes can be adjusted by changing the 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 using s-confidence and w-confidence thresholds of 60%. Sequential s-confidence and w-confidence are effectively used to prune weak affinity patterns in terms of support and weight. It is not 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.

Performance Evaluation Using D7C7T7S7I7 Dataset

Figures 7 to 10 show the results of the performance evaluation based on the D7C7T7S7I7 dataset. We set the weights to between 0.1 and 0.3. Using sequential s-confidence and/or w-confidence thresholds, WIS generates correlated sequential patterns with a higher level of affinity as compared to other algorithms. The results demonstrate that the performance when 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%, sequential w-confidence outperforms sequential s-confidence. In Figs. 7 and 8, we cannot show the number of patterns generated by

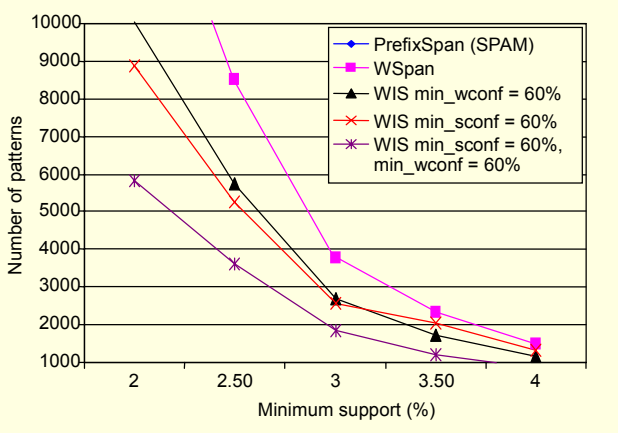

Fig. 7. Number of affinity patterns (min_sconf: 0.6, min_wconf: 0.6).

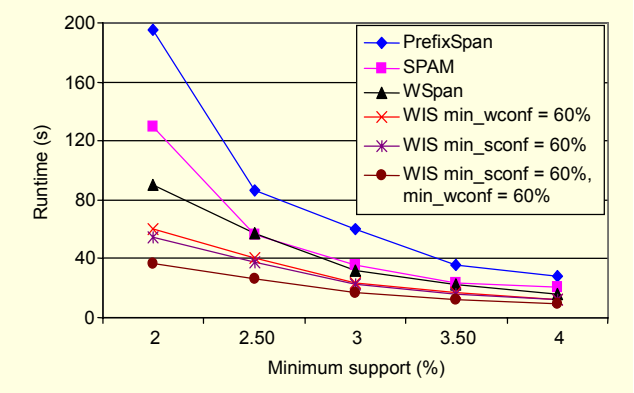

Fig. 8. Runtime (min_sconf: 0.6, min_wconf: 0.6).

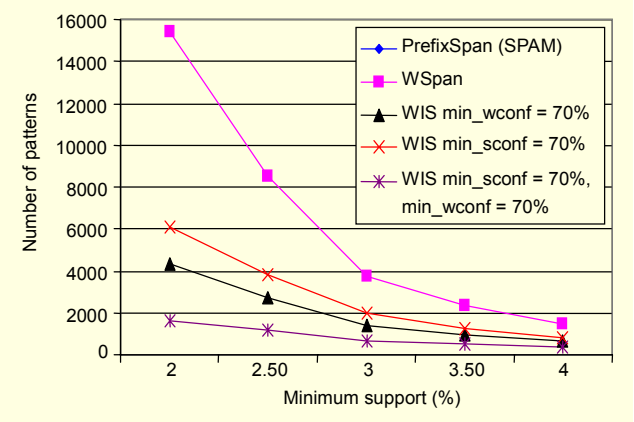

Fig. 9. Number of affinity patterns (min_sconf: 0.7, min_wconf: 0.7).

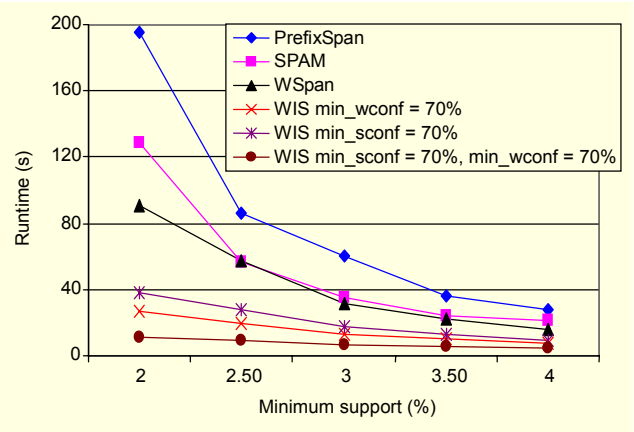

Fig. 10. Runtime (min_sconf: 0.7, min_wconf: 0.7).

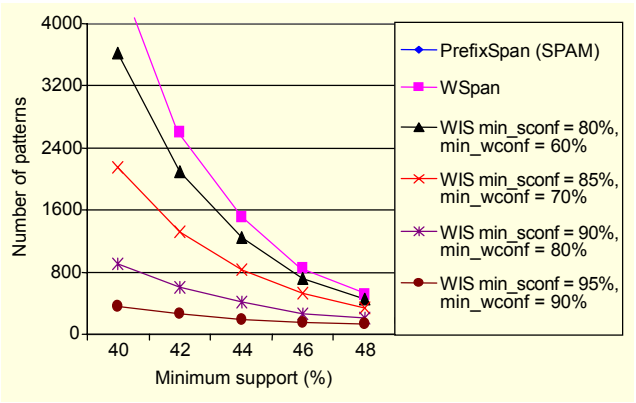

Fig. 11. Number of patterns using combination of two measures.

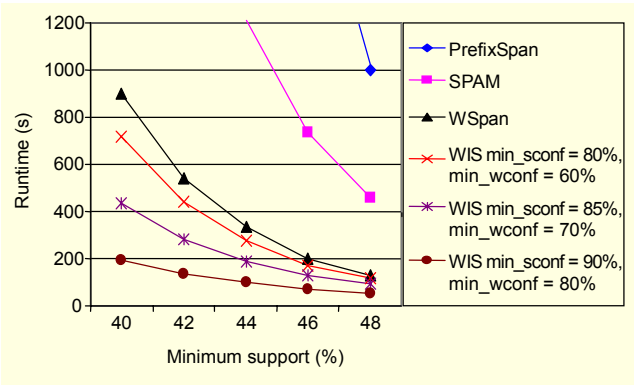

Fig. 12. Runtime using combination of two measures.

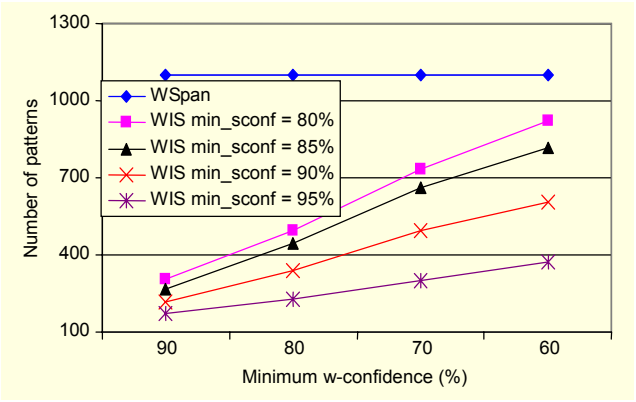

Fig. 13. Number of maximum (s-affinity) patterns (min_sup = 45%).

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%.

Performance Evaluation Using D15C15T15S15I15 Dataset

Figures 11 to 14 demonstrate the results of a performance test using the D15C15T15S15I15 dataset with weights from 0.4 to 0.8. When the w-confidence threshold is lowered, the performance difference of the sequential w-confidence measure becomes larger. At higher weight confidences levels, such as 90%, the performance of WIS improves. The number of (maximum) sequential affinity patterns for WIS decreases as the sequential s-confidence and w-confidence increase. 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 the weight range means more priority is given to the 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.

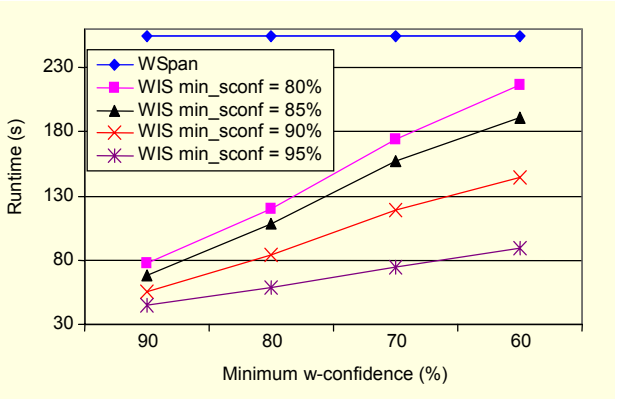

Fig. 14. Runtime (min_sup = 45%).

Performance Evaluation Using Real Gazelle Dataset

The Gazelle dataset is a set of click stream data which is used in KDD Cup-2000. Product pages created by a customer in a session are considered an itemset, and difference sessions created by the customer during one use are considered a sequence (see [7] and [19]). In this experiment, minimum supports are used with normalized weights in a range from 0.1 to 0.9. Figures 15 and 16 show that WIS detects support and weight affinity sequential patterns with sequential s-confidence and w-confidence, respectively. 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

The previously presented evaluations show that sequential s-confidence and w-confidence can be used to detect sequential patterns with s-affinity and/or w-affinity. In all test datasets, items are expressed as integer values, so it is difficult to

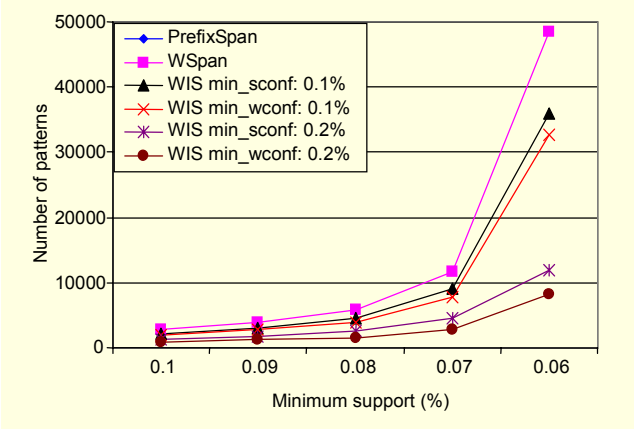

Fig. 15. Number of maximum patterns using real dataset (min_ sconf: 0.001-0.002, min_wconf: 0.001-0.002).

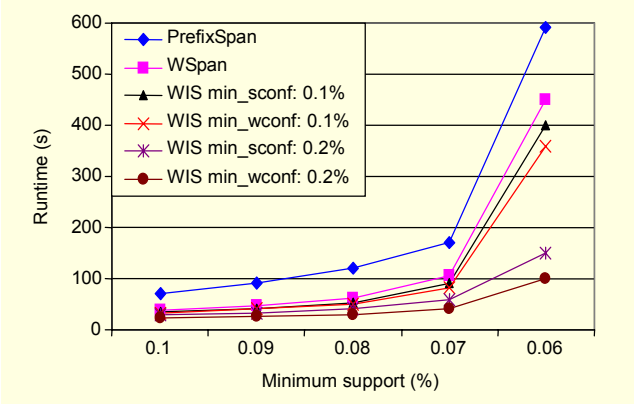

Fig. 16. Runtime using real dataset (min_ sconf: 0.001-0.002, min_wconf: 0.001-0.002).

understand the meaning of items and discovered sequential patterns. In the evaluation presented here, the D7C7T5S4I2.5 dataset is used to illustrate the quality of affinity sequential pattern mining. The minimum support is set to 2.5%, and weights are set between 0.1 and 0.3. We compare the patterns mined by WIS with those mined by 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). 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). Moreover, although the minimum support threshold is increased and/or the weight range is changed, 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. Weak affinity patterns can be effectively pruned by sequential s-confidence and/or w-confidence.

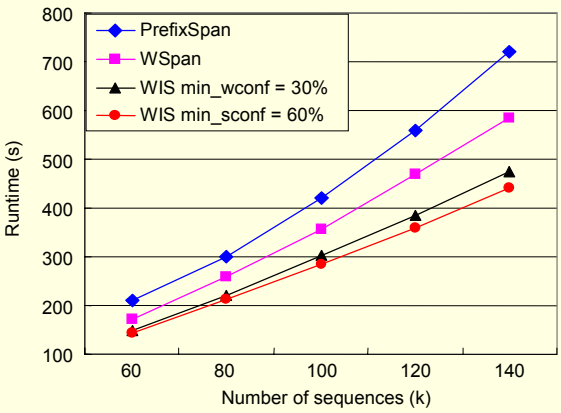

Fig. 17. Scalability test of s-confidence and w-confidence.

C. Scalability Test

Performance Evaluation Using 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 the minimum support to 0.4% and set the weights between 0.1 and 0.5. To clearly show differences among the tested algorithms, the number of sequences in the x-axis was increased up to 140 k and the scalability test was performed. In Fig. 17, the slope differences among the algorithms widen as the number of sequences in the 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 the scalability of WIS is definitely better than that of PrefixSpan. Moreover, WIS shows somewhat better scalability than WSpan.

V. Future Research

WIS is a memory-based sequential pattern mining algorithm but this assumption is limiting when the database is very large or the minimum threshold becomes low. WIS should be extended to be a disk-based method. Second, to set the 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 need to develop a method 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 including minimum support, minimum s-confidence, and minimum w-confidence. Effective setting of thresholds is essential, although this problem is common to all threshold-based mining

algorithms. For example, sequential weak affinity patterns can be strongly pruned by increasing the difference between the maximum weight and the minimum weight in a sequence database, although the minimum w-confidence and/or s-confidence are fixed. Meanwhile, the effect of w-confidence can be reduced by decreasing the difference between the maximum weight and the minimum weight. We need to conduct more research to develop guidelines for efficiently setting thresholds. Finally, improved techniques, such as sequential pattern mining using pseudo projection [19] or bitmap representation [23] have been suggested. In our 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 using the two measures. Sequential s-confidence and/or w-confidence measures can be used to prune weak sequential patterns including items which have dissimilar support and/or weight levels. Extensive performance analysis demonstrated that WIS is efficient and scalable in sequential affinity pattern mining. Moreover, experimental results showed that the WIS algorithm is very effective to detect support and/or weight affinity sequential patterns.

References

[1] M. Garofalakis, R. Rastogi, and K. Shim. "SPIRIT: Sequential Pattern Mining with Regular Expression Constraints," *Proc. of the Twenty-fifth Int'l Conf. on Very Large Data Bases, (VLDB'99)*, Sep. 1999, pp. 223-234.

[2] H. Albert-Lorincz and J.F. Boulicaut, "Mining Frequent Sequential Patterns under Regular Expressions: A Highly Adaptive Strategy for Pushing Constraints," *Proc. of the Third SIAM Int'l Conf. on Data Mining*, May 2003, pp. 316-320.

[3] J. Pei, J. Han, and W. Wang, "Mining Sequential Patterns with Constraints in Large Databases," *Proc. of the 2002 ACM CIKM Int'l Conf. on Information and Knowledge Management*, Nov. 2002, pp. 18-25.

[4] M. Seno and G. Karypis, "SLPMiner: An Algorithm for Finding Frequent Sequential Patterns Using Length-Decreasing Support Constraints," *Proc. of the Second IEEE Int'l Conf. on Data Mining (ICDM 2002)*, Dec. 2002, pp. 418-425.

[5] P. Tzvetkov, X. Yan, and J. Han, "TSP: Mining Top-K Closed Sequential Patterns," *Proc. of the Third IEEE Int'l Conf. on Data Mining (ICDM 2003)*, Dec. 2003, pp. 347-354.

[6] J. Wang and J. Han, "BIDE: Efficient Mining of Frequent Closed Sequences," *Proc. of the Twentieth Int'l Conf. on Data Engineering*, March/April 2004, pp. 79-90.

[7] X. Yan, J. Han, and R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Datasets," *Proc. of the Third SIAM Int'l Conf. on Data Mining*, May 2003, pp. 166-177.

[8] H.C. Kum, J. Pei, W. Wang, and D. Duncan, "ApproxMAP: Approximate Mining of Consensus Sequential Patterns," *Proc. of the Third SIAM Int'l Conf. on Data Mining*, May 2003, pp. 311-315.

[9] H. Pinto, J. Han, J. Pei, and K. Wang, "Multi-Dimensional Sequence Pattern Mining," *Proc. of the 2001 ACM CIKM Int'l Conf. on Information and Knowledge Management*, Nov. 2001, pp. 474-481.

[10] J. Yang, P.S. Yu, W. Wang, and J. Han, "Mining Long Sequential Patterns in a Noisy Environment," *Proc. of the 2002 ACM SIGMOD Int'l Conf. on Management of Data*, June 2002, pp. 406-417.

[11] M. Ester, "A Top-Down Method for Mining Most Specific Frequent Patterns in Biological Sequence Data," *Proc. of the Fourth SIAM Int'l Conf. on Data Mining*, April 2004, pp. 90-101.

[12] K. Wang, Y. Xu, and J.X. Yu, "Scalable Sequential Pattern Mining for Biological Sequences," *Proc. of the 2004 ACM CIKM Int'l Conf. on Information and Knowledge Management*, Nov. 2004, pp. 178-187.

[13] H. Cheng, X. Yan, and J. Han, "IncSpan: Incremental Mining of Sequential Patterns in Large Databases," *Proc. of the Tenth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, Aug. 2004, pp. 527-532.

[14] H. Chung, X. Yan, and J. Han, "SeqIndex: Indexing Sequences by Sequential Pattern Analysis," *Proc. of the Fifth SIAM Int'l Conf. on Data Mining*, Apr. 2005, pp. 601-605.

[15] R. Agrawal and R. Srikant, "Mining Sequential Patterns," *Proc. of the Eleventh Int'l Conf. on Data Engineering*, Mar. 1995, pp. 3-14.

[16] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," *Proc. of the Fifth Int'l Conf. on Extending Database Technology*, Mar. 1996, pp. 3-17.

[17] J. Han, J. Pei, B.M. Asi, Q. Chen, U. Dayal, and M.C. Hsu, "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining," *Proc. of the Sixth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, Aug. 2000, pp. 355-359.

[18] J. Pei, J. Han, B.M. Asi, and H. Pino, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," *Proc. of the Seventeenth Int'l Conf. on Data Engineering*, Apr. 2001, pp. 215-224.

[19] J. Pei, J. Han, B.M. Asi, J. Wang, and Q. Chen, "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan

Approach," *IEEE Trans. on Knowledge and Data Engineering*, vol. 16, Oct. 2004, pp. 1424-1440.
[20] U. Yun and J.J. Leggett, "WSpan: Weighted Sequential Pattern Mining in Large Sequence Databases," *Proc. of the Third Int'l Conf. on IEEE Intelligent Systems*, Sep. 2006, pp. 512-517.
[21] T. Haines, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2001.
[22] M.J. Zaki, "SPADE: An Efficient Algorithm for Mining Frequent Sequences," *Machine Learning*, vol. 42, Jan. 2001, pp. 31-60.
[23] J. Ayres, J. Gehrke, T. Yiu, and J. Flannick, "Sequential Pattern Mining Using a Bitmap Representation," *Proc. of the Eighth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, July 2002, pp. 429-435.
[24] D. Chiu, Y. Wu, and A.L. Chen, "An Efficient Algorithm for Mining Frequent Sequences by a New Strategy without Support Counting," *Proc. of the Twentieth Int'l Conf. on Data Engineering*, Mar./Apr. 2004, pp. 375-386.
[25] U. Yun and J.J. Leggett, "WFIM: Weighted Frequent Itemset Mining with a Weight Range and a Minimum Weight," *Proc. of the Fifth SIAM Int'l Conf. on Data Mining*, Apr. 2005, pp. 636-640.
[26] U. Yun and J.J. Leggett, "WLPMiner: Weighted Frequent Pattern Mining with Length-Decreasing Support Constraints," *Proc. of* the *9th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD`05)*, May 2005, pp. 555-567.
[27] U. Yun, "Mining Lossless Closed Frequent Patterns with Weight Constraints," *Knowledge Based Systems*, vol. 20, Feb. 2007, pp. 86-97.
[28] K.W. Min, K.W. Nam, and J.W. Kim, "Multilevel Location Trigger in Distributed Mobile Environments for Location-Based Services," *ETRI Journal*, vol. 29, no. 1, Feb. 2007, pp. 107-109.

Unil Yun received the MS degree in Computer Science and Engineering from Korea University, Seoul, Korea, in 1997, and the PhD degree in Computer Science from Texas A&M University, Texas, USA, in 2005. He worked at Multimedia Laboratory, Korea Telecom, from 1997 to 2002. After receiving the PhD degree, he worked as a post-doctoral associate for almost one year at the Computer Science Department of Texas A&M University. Currently, he is a senior researcher in the Telematics and USN Research Division, Electronics and Telecommunications Research Institute (ETRI). His research interests include data mining, information retrieval, database systems, artificial intelligence, and digital libraries.