International Journal of Innovative Computing, Information and Control
Volume **9**, Number **12**, December **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 4811–4824

AN EFFICIENT ALGORITHM FOR MINING SEQUENTIAL RULES WITH INTERESTINGNESS MEASURES

Thi-Thiet Pham[1,2], Jiawei Luo[1,*], Tzung-Pei Hong[3,4] and Bay Vo[5]

[1]School of Information Science and Engineering
Hunan University
Yuelushan, Changsha 410082, P. R. China
*Corresponding author: luojiawei@hnu.edu.cn

[2]Faculty of Information Technology
Industrial University of Ho Chi Minh City
Ho Chi Minh City, Vietnam
phamthithiet@hui.edu.vn

[3]Department of Computer Science and Information Engineering
National University of Kaohsiung
No. 700, Kaohsiung University Rd., Nanzih Dist., Kaohsiung 81148, Taiwan
tphong@nuk.edu.tw

[4]Department of Computer Science and Engineering
National Sun Yat-sen University
No. 70, Lienhai Rd., Kaohsiung 80424, Taiwan

[5]Faculty of Information Technology
Ton Duc Thang University
Ho Chi Minh City, Viet Nam
bayvodinh@gmail.com

Received December 2012; revised April 2013

Abstract. *Mining sequential rules are an important problem in data mining research. It is commonly used for market decisions, management and behaviour analysis. In traditional association-rule mining, rule interestingness measures such as confidence are used for determining relevant knowledge. They can reduce the size of the search space and select useful or interesting rules from the set of the discovered ones. Many studies have examined the interestingness measures for mining association rules, but have not been devoted to mine sequential rules in sequence databases. In this paper, we thus consider and apply several interestingness measures to generate all relevant sequential rules from a sequence database. The prefix tree structure is also used to get the support values of sequential patterns faster and reduce the execution time for mining sequential rules. Our experimental results show that the run time for mining sequential rules with interestingness measures on the prefix tree structure is much faster than that of other algorithms.*

Keywords: Sequential pattern, Sequential rule, Sequence database, Interestingness measure, Prefix tree

1. **Introduction.** Sequential pattern mining plays an important role in data mining research and has a broad range of applications, including customer purchase behaviour analysis [5,6], DNA sequence pattern analysis [8,18], guidance systems [38], web usage behaviour analysis [31], and so on. Agrawal and Srikant [1] first proposed the sequential pattern mining in 1995. The same authors later developed a generalized algorithm based on the apriori property, called GSP [35]. Many other algorithms, which will be mentioned in Section 2, have also been proposed to improve the effect of mining sequential patterns.

However, when the support value is set low, many sequential patterns including irrelevant or spurious patterns, may be obtained. Thus, designing an efficient sequential rule mining process to remove these spurious patterns is important. Sequential rules express the temporal relationships between sequential patterns from a sequence database [34]. Sequential rules can be considered natural extension of original sequential patterns, just as association rules are natural extension of frequent itemsets [35]. Using sequential rules, we can know the series of events that will usually occur after a series of previous ones. Thus, they can help users better understand the chronological order of the sequences present in the sequence database. Like a sequential pattern, a sequential rule is also applied in many application areas, including the stock market [5,6,21], weather observation [16], e-learning [11], trade [9], and software engineering [26,41]. In addition, sequential rule mining is also applied to address the prediction problem [10,12,13,16,17,21]. In the prediction problem, a sequence of events that appear frequently in a database is not sufficient to predict events, while sequential rules allow a better understanding of the prediction. An appropriate sequential rule mining process, instead of mining only sequential patterns, is also desired.

The use of the interestingness measures of a rule can reduce the sizes of search spaces and play an important role in selecting useful or interesting rules from the set of the discovered rules. Many studies have examined interestingness measures to mine rules, including support, confidence, cosine, lift, χ^2, gini-index, Laplace, and phi-coefficient [2,3,14,19,22-25,28,30,32,33,37,40] and so on. However, to the best of our knowledge, these interestingness measures have been used for mining association rules in transaction databases [25,33,37,40] but have not been used to mine sequential rules in sequence databases except the traditional measures of support and confidence. Thus, the main aim of this paper is to apply different interestingness measures to the sequential rule mining problem. In this paper, we focus on some specific interestingness measures, including Conviction [6], Cosine and Lift [37], etc., for mining relevant sequential rules. An algorithm is proposed to generate all sequential rules from a set of sequential patterns in a sequence database based on the prefix tree structure with these interestingness measures. On a prefix tree, each node stores a sequential pattern and its corresponding support value. When a sequential rule $X \Rightarrow Y$ is mined, the prefix tree is directly traversed to obtain the corresponding values, including the support values of X, Y and XY, which are then used to calculate the measured value of the rule. Most interestingness measures of a rule depend on the support value of the left hand side of the rule, and sometimes the support value of the right hand side of the rule is also used to calculate the measured values of a rule. The use of the prefix-tree structure in this paper can help easily get the support values of sequential patterns to fast calculate the measure values of a rule.

The rest of this paper is organized as follows. Section 2 is a summary of related work. Section 3 presents some definitions required for the sequential rule mining problem. Section 4 discusses the proposed algorithm for mining sequential rules using interestingness measures. Section 5 presents the experimental results, and Section 6 gives conclusions and future work.

2. Related Work.

The sequential pattern mining problem was firstly proposed by Agrawal and Srikant [1] in 1995. The same authors later applied the apriori property to develop a generalized algorithm, called GSP [35]. Many other algorithms have also been proposed to improve the effect of mining sequential patterns, including the SPADE [41] algorithm, which was proposed to divide candidate sequences into distinct groups such that each group could be completely stored in the main memory. The SPAM [4] algorithm could speed up the mining process by using a lexicographic sequence tree and a bitmap representation. PrefixSpan [29] examined the prefix subsequences and projected the corresponding postfix subsequences into projected databases. The PRISM [15] algorithm used the primal block encoding approach to represent candidate sequences and joined operations over the primal blocks to determine the frequency of each candidate. Experimental results [15] also showed that PRISM was one of the best methods for mining sequential patterns. It outperformed existing methods by an order of magnitude or more and had a low memory footprint.

Spiliopoulou [34] proposed generating a full set of sequential rules from a set of frequent sequences and adding a post-processing phase to remove some redundant rules. On the basis of description in [34], Lo *et al.* [27] generalized and named for algorithm be the *Full* algorithm to mine a full set of sequential rules. Besides, they also proposed a compressed set of non-redundant rules that were generated from two sequence set types: the set of projected-database-closed patterns (LS-Closed) and the set of closed patterns (CS-Closed). The premise of a rule is a sequence in an LS-Closed set, and the consequence is a sequence in a CS-Closed set. Van *et al.* [39] then improved the *Full* algorithm [27] and proposed an algorithm, called MSR-Full, to find sequential rules between pairs of sequential patterns. The algorithm was improved by sorting all sequential patterns in an ascending order of sizes. Sequences that were prefixes of a sequence X could only appear before X in the list of frequent sequential patterns. The authors [39] also gave another algorithm, called MSR-PreTree. Based on the property of the prefix-tree structure, any sequence X in the tree (except the tree root) is always a prefix of all sequences on the sub trees in which each node was the sequence extension of X. The MSR-PreTree algorithm could directly generate sequential rules. However, all the above algorithms proposed only used the two traditional measures, support and confidence, to generate sequential rules.

In 1991, Piatetsky-Shapiro [30] proposed the statistical independence of rules, which could be thought of as an interestingness measure as well. Many other measures [19,36] have thus been proposed since then. To the best of our knowledge, these measures have been used for mining association rules in transaction databases [25,33,37,40] but have not been used to mine sequential rules in sequence databases except the traditional measures of support and confidence. Thus, in this paper, we attempt to generate sequential rules with the different interestingness measures.

3. **Problem Definitions.** A sequence database *SD* is a set of sequences $S = \{s_1, s_2, \ldots, s_n\}$, where each sequence s_x is an ordered list of itemsets. That is, $s_x = (x_1, x_2, \ldots, x_n)$, where x_1 occurs before x_2, which occurs before x_3, and so on, such that $x_1, x_2, \ldots, x_n \subseteq I$, where I is a set of items $\{i_1, i_2, \ldots, i_n\}$. The size of a sequence is the number of itemsets in the sequence. The length of a sequence is the number of items in the sequence. A sequence with length l is called a *l-pattern*. A sequence with size k is called a k-sequence.

Given two sequences $\alpha = \langle a_1\ a_2 \ldots a_n\rangle$ and $\beta = \langle b_1\ b_2 \ldots b_m\rangle$, where a_i, b_i are itemsets. The sequence α is called a subsequence of β and β is a supersequence of α, denoted $\alpha \subseteq \beta$, if $n \leq m$ and there exist integers $j_1, j_2, \ldots, j_n$ such that $1 \leq j_1 < j_2 < \ldots < j_n \leq m$ and $a_1 \subseteq b_{j1}, a_2 \subseteq b_{j2}, \ldots, a_n \subseteq b_{jn}$. For example, if $\alpha = \langle (ab),\ d\rangle$ and $\beta = \langle (abc),\ (de)\rangle$, where a, b, c, d and e are items, then α is a subsequence of β and β is a supersequence of α. The support of a sequence $\alpha\,\mathrm{sup}(\alpha)$ in a sequence database is the number of sequences in the database containing α. Sequence α is a frequent sequence in a sequence database *SD* if the support $\mathrm{sup}(\alpha)$ of the sequence α is larger than or equal to the given minimum support threshold *minSup*. A frequent sequence is called a sequential pattern.

Sequence α is a prefix of β if and only if $a_i = b_i$ for all $1 \leq i \leq n < m$.

If the prefix part α is removed from sequence β, then the remaining part of β is called a postfix of β. Sequence α is an incomplete prefix of β if and only if $a_i = b_i$ for all $1 \leq i \leq n-1$, $a_n \subset b_n$ and all items in $(b_n - a_n)$ are lexicographically after those in a_n.

From the above definition, it can be derived that a sequence of size k has $(k-1)$ prefixes. For example, a sequence $\langle(A)(BC)(D)\rangle$ has 2 prefixes: $\langle(A)\rangle$ and $\langle(A)(BC)\rangle$. Therefore, $\langle(BC)(D)\rangle$ is the postfix for prefix $\langle(A)\rangle$, and $\langle(D)\rangle$ is the postfix for prefix $\langle(A)(BC)\rangle$. However, neither $\langle(A)(B)\rangle$ nor $\langle(BC)\rangle$ is considered as a prefix of the given sequence, but $\langle(A)(B)\rangle$ is an incomplete prefix of the given sequence.

A prefix tree is similar to a lexicographic tree [15,39], which starts from the tree root at level 0. The root is set with a null sequence ∅, and each child node stores a sequential pattern and its corresponding support value. At level 1, each node is set with a frequent item; at level k, each node is set with a k-sequence. Recursively, the nodes at the $(k+1)$ level will be formed by extending a k-sequence with a frequent item. There are two ways to extend a k-sequence: sequence extension and itemset extension [15]. In sequence extension, an item from the whole set of items I is appended to the sequence as a new itemset, and the size of the extended sequence always increases. Thus, a k-sequence α is a prefix of all the sequence-extended sequences of α, and is also the prefix of all the sub-nodes of the nodes which was sequence-extended from α. In itemset extension, an item from the whole set of items I is appended to the last itemset in the sequence so that the ID number of the item must be greater than the ID numbers of all the items

Table 1. A sequence database

SID	Sequence
1	$\langle(AB)(B)(B)(AB)(B)(AC)\rangle$
2	$\langle(AB)(BC)(BC)\rangle$
3	$\langle(B)(AB)\rangle$
4	$\langle(B)(B)(BC)\rangle$
5	$\langle(AB)(AB)(AB)(A)(BC)\rangle$

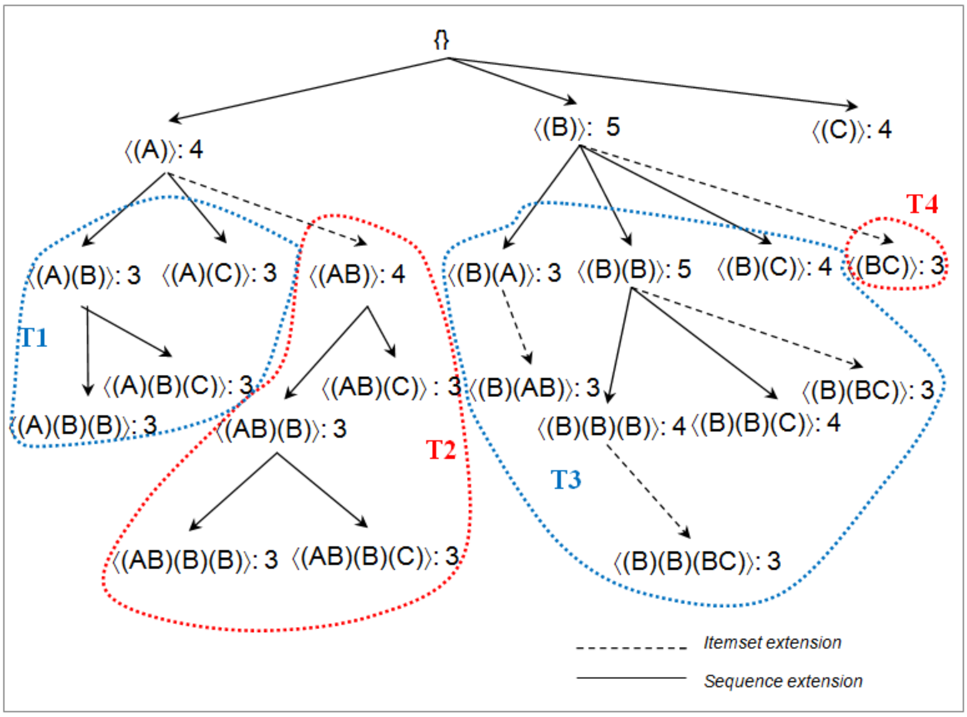


Figure 1. A prefix tree structure storing sequential patterns from Table 1

in the last itemset. The size of itemset-extended sequences does not change, and α is an incomplete prefix of all sub-nodes of the itemset-extended nodes in α.

Figure 1 shows the prefix tree of sequential patterns generated from the sequence database in Table 1 with *minSup* $= 50\%$. Sequences $\langle(A)(B)\rangle$ and $\langle(A)(C)\rangle$ are sequence-extended sequences of $\langle(A)\rangle$, and $\langle(AB)\rangle$ is an itemset-extended sequence of $\langle(A)\rangle$. Sequence $\langle(A)\rangle$ is a prefix of all the sequences in $T1$ and an incomplete prefix of all the sequences in $T2$. Similarly, sequence $\langle(B)\rangle$ has the three sequences-extended sequences $\langle(B)(A)\rangle$, $\langle(B)(B)\rangle$ and $\langle(B)(C)\rangle$, and the one itemset-extended sequence $\langle(BC)\rangle$. Sequence $\langle(B)\rangle$ is a prefix of all the sequences in $T3$ and an incomplete prefix of all the sequences in $T4$.

Given the frequent sequential patterns of X and Y, there is a sequential rule $X \Rightarrow Y$, if its confidence satisfies the minimum confidence threshold. The confidence of a sequential rule $X \Rightarrow Y$ is the ratio of the number of sequences that contain both X and Y against the number of those that contain X. Similar to the association rule mining problem, we also divide the sequential rule mining using interestingness measures from a sequence database into two stages. The first stage is to mine sequential patterns that satisfy the *minSup*. The next stage is to generate all the sequential rules with their interestingness measures from the above sequential patterns.

4. Mining Sequential Rules with Interestingness Measures.

In this section, we describe the sequential rule mining process. The sequential rule mining problem, presented above, contains two stages. The first stage is to mine sequential patterns from a sequence database to satisfy the *minSup* threshold, and the second stage is to mine sequential rules from the set of sequential patterns generated using the interestingness measure values. To efficiently mine sequential patterns in the first stage, the PRISM [15] algorithm is adopted, which uses the prime block encoding approach to represent candidate sequences and the join operations over the prime blocks to determine the frequency for each candidate. All the sequential patterns generated by the PRISM algorithm are stored in a prefix tree structure.

4.1. Sequential rule mining.

A sequential rule has the form $X \Rightarrow Y(q, imv)$, where X and Y are sequential patterns, $X \cap Y = \emptyset$, q is the support of the rule ($q = \sup(X, Y)$), and *imv* is an interestingness measure value of the rule. In the traditional sequential rules, *imv* is the confidence of a rule, and $imv = \sup(X, Y)/\sup(X)$.

A sequential rule can be created by splitting a sequential pattern into two parts: the prefix (*pre*) and the postfix (*post*). If *pre* is concatenated with *post*, denoted $pre + +post$, then the result is the original sequential pattern. A sequential rule r can thus be formed as $pre \Rightarrow post\ (sup, imv)$. The support $\sup(r)$ of r is thus $\sup(pre + +post)$. The interestingness measure value of r is *imv*, and the traditional measure value of r is the confidence measure $conf(r)$ of r. That is, $conf(r) = \sup(pre + +post)/\sup(pre)$. A sequence of size k has $(k-1)$ prefixes, and can thus have $(k-1)$ sequential rules.

4.2. Interestingness measures.

Interestingness measures are important metrics for rule mining in the data mining research. They can reduce the search space size and play an important role in selecting useful or interesting rules from a set of the discovered rules. Recent research has examined the interestingness measures for mining rules. Table 2 shows some interestingness measures.

From the equations in Table 2, it can be easily observed that the terms often used to calculate a measured value of the rule $X \Rightarrow Y$ are the total number of sequences in a sequence database (n), the number of sequences that contain $X(n_X)$, the number of sequences that contain $Y(n_Y)$, the number of sequences that contain both X and

Table 2. Some interestingness measures for a rule $X \Rightarrow Y$

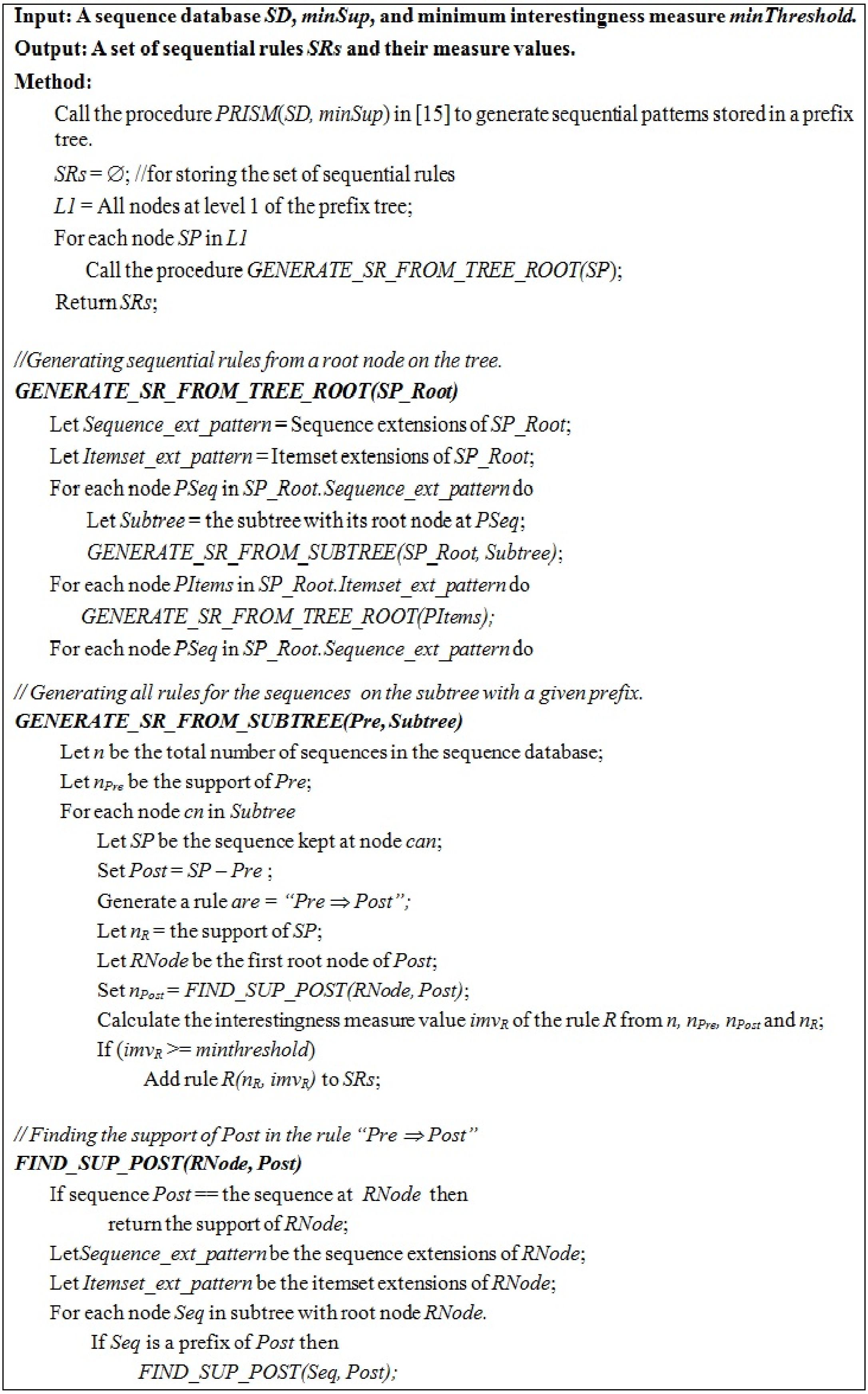
Interestingness measure	Equation	Value	Reference
Confidence	$\frac{n_{XY}}{n_X}$	$\frac{3}{5}$	[22]
Support	$\frac{n_{XY}}{n}$	$\frac{3}{5}$	[22]
Conviction	$\frac{n_X n_{\bar{Y}}}{n n_{X\bar{Y}}}$	$\frac{5*2}{5*2} = 1$	[8,20]
Lift	$\frac{n n_{XY}}{n_X n_Y}$	$\frac{5*3}{5*3} = 1$	[20,24]
Piatetsky-Shapiro	$n_{XY} - \frac{n_X n_Y}{n}$	$3 - \frac{5*3}{5}$	[19,20,27]
Cosine	$\frac{n_{XY}}{\sqrt{n_X n_Y}}$	$\frac{3}{\sqrt{5*3}}$	[19]
Jaccard	$\frac{n_{XY}}{n_X + n_Y - n_{XY}}$	$\frac{3}{5+3-3} = \frac{3}{5}$	[19,23]

$Y(n_{XY})$, the number of sequences that contain X but not $Y(n_{X\bar{Y}})$, and the number of sequences that contain Y but not $X(n_{\bar{X}Y})$. If we know n, n_X, n_Y and n_{XY}, other terms for calculating the measured value in these equations can be easily determined like $n_{X\bar{Y}} = n_X - n_{XY}$, $n_{\bar{X}Y} = n_Y - n_{XY}$, $n_{\bar{X}} = n - n_X$ and $n_{\bar{Y}} = n - n_Y$. Consider the sequence database in Table 1. If $X = \langle(B)\rangle$ and $Y = \langle(B)(BC)\rangle$, then $n = 5$, $n_X = 5$, $n_Y = 3$ and $n_{XY} = 3$. These terms then derive $n_{\bar{X}} = 0$, $n_{\bar{Y}} = 2$, $n_{X\bar{Y}} = 2$. Table 2 also presents the interestingness measure values for the rule $X \Rightarrow Y$.

4.3. **Proposed algorithm.** The previous algorithm PRISM in [15] is first applied to generate sequential patterns stored in the prefix tree structure. An algorithm based on the characteristics of the prefix tree is then proposed to generate sequential rules with interestingness measures. By traversing the prefix tree, the algorithm can then easily identify the components of a rule, such as the *pre* and the *post* parts, and can calculate the measured values of the rule. Figure 2 presents the proposed algorithm to mine sequential rules with interestingness measures.

In Figure 2, the algorithm first calls the *PRISM*(*SD*, *minSup*) procedure to generate all sequential patterns and store these patterns in the prefix tree structure. For each node *SP* at level 1 of the prefix tree, it calls the *GENERATE_SR_FROM_TREE_ROOT*(*SP_Root*) procedure to generate sequential rules from each sub-tree with *SP* as its root node. When the procedure *GENERATE_SR_FROM_TREE_ROOT*(*SP_Root*) is processed, there are two types of nodes: sequence-extended and itemset-extended nodes. Since the size of the itemset-extended nodes set does not change w.r.t the root node size based on the definition of itemset extension, sequential rules are not generated from this itemset-extended node set. Only sequential rules from sequences on the subtrees whose nodes are sequence-extended nodes of the root are generated from the called procedure *GENERATE_SR_FROM_SUBTREE*(*Pre, Subtree*), because the sequence at the root denoted as *pre* will form the prefix of all extended sequences from the sequence-extended nodes of the root. Hence, for each sub–tree, sequential rules from the sequences on the subtree following the prefix *pre* are generated. All the extended nodes of the current root then become prefixes of the subtrees at the next level, and this procedure is recursively called for every extended node of the root. This recursive process is repeated until the last level of the prefix tree is reached.

Besides, in the procedure *GENERATE_SR_FROM_SUBTREE*(*Pre, Subtree*), the input is sequences *pre* and *Subtree* so that *pre* is a common prefix of all the sequences on the subtree. For each sequence *SP* in the subtree, the rule "*pre⇒post*" is generated such that *post* is a postfix of *SP* with respect to the prefix *pre*.



```
Input: A sequence database SD, minSup, and minimum interestingness measure minThreshold.
Output: A set of sequential rules SRs and their measure values.
Method:
    Call the procedure PRISM(SD, minSup) in [15] to generate sequential patterns stored in a prefix
    tree.
    SRs = ∅; //for storing the set of sequential rules
    L1 = All nodes at level 1 of the prefix tree;
    For each node SP in L1
        Call the procedure GENERATE_SR_FROM_TREE_ROOT(SP);
    Return SRs;

//Generating sequential rules from a root node on the tree.
GENERATE_SR_FROM_TREE_ROOT(SP_Root)
    Let Sequence_ext_pattern = Sequence extensions of SP_Root;
    Let Itemset_ext_pattern = Itemset extensions of SP_Root;
    For each node PSeq in SP_Root.Sequence_ext_pattern do
        Let Subtree = the subtree with its root node at PSeq;
        GENERATE_SR_FROM_SUBTREE(SP_Root, Subtree);
    For each node PItems in SP_Root.Itemset_ext_pattern do
        GENERATE_SR_FROM_TREE_ROOT(PItems);
    For each node PSeq in SP_Root.Sequence_ext_pattern do

// Generating all rules for the sequences on the subtree with a given prefix.
GENERATE_SR_FROM_SUBTREE(Pre, Subtree)
    Let n be the total number of sequences in the sequence database;
    Let $n_{Pre}$ be the support of Pre;
    For each node cn in Subtree
        Let SP be the sequence kept at node can;
        Set Post = SP – Pre ;
        Generate a rule are = "Pre ⇒ Post";
        Let $n_R$ = the support of SP;
        Let RNode be the first root node of Post;
        Set $n_{Post}$ = FIND_SUP_POST(RNode, Post);
        Calculate the interestingness measure value $imv_R$ of the rule R from n, $n_{Pre}$, $n_{Post}$ and $n_R$;
        If ($imv_R$ >= minthreshold)
            Add rule R($n_R$, $imv_R$) to SRs;

// Finding the support of Post in the rule "Pre ⇒ Post"
FIND_SUP_POST(RNode, Post)
    If sequence Post == the sequence at RNode then
        return the support of RNode;
    LetSequence_ext_pattern be the sequence extensions of RNode;
    Let Itemset_ext_pattern be the itemset extensions of RNode;
    For each node Seq in subtree with root node RNode.
        If Seq is a prefix of Post then
            FIND_SUP_POST(Seq, Post);
```

Figure 2. The proposed algorithm for generating sequential rules based on a prefix tree

Most of the interestingness measures (*imv*) for a rule depend on the support (n_{Post}) of *Post*. To obtain the support of *Post*, the procedure *FIND_SUP_POST*(*RNode, Post*) is called, where *RNode* is a not-empty and the first root node of *Post* on the prefix tree. The procedure *FIND_SUP_POST*(*RNode, Post*) produces the support of *Post* by traversing the branch of the prefix tree based on the root node *RNode*, which is the prefix of *Post*.

4.4. **An example.** An example is given here to illustrate the above algorithm. Consider the sequence database presented in Table 1, with *minSup* = 50%. All the sequential patterns in the database are stored on the prefix tree shown in Figure 1.

Note that when the minimum interestingness measure threshold *minThreshold* is 0, the numbers of the sequential rules generated from the prefix tree in Figure 1 for all of the different interestingness measures are equal (totally 23 sequential rules), including $\langle(A) \Rightarrow (B)\rangle$; $\langle(A) \Rightarrow (C)\rangle$; $\langle(A) \Rightarrow (B)(B)\rangle$; $\langle(A) \Rightarrow (B)(C)\rangle$; $\langle(A)(B) \Rightarrow (B)\rangle$; $\langle(A)(B) \Rightarrow (C)\rangle$; $\langle(AB) \Rightarrow (B)\rangle$; $\langle(AB) \Rightarrow (C)\rangle$; $\langle(AB) \Rightarrow (B)(B)\rangle$; $\langle(AB) \Rightarrow (B)(C)\rangle$; $\langle(AB)(B) \Rightarrow (B)\rangle$; $\langle(AB)(B) \Rightarrow (C)\rangle$; $\langle(B) \Rightarrow (A)\rangle$; $\langle(B) \Rightarrow (B)\rangle$; $\langle(B) \Rightarrow (C)\rangle$; $\langle(B) \Rightarrow (AB)\rangle$; $\langle(B) \Rightarrow (BC)\rangle$; $\langle(B) \Rightarrow (B)(B)\rangle$; $\langle(B) \Rightarrow (B)(C)\rangle$; $\langle(B) \Rightarrow (B)(BC)\rangle$; $\langle(B)(B) \Rightarrow (B)\rangle$; $\langle(B)(B) \Rightarrow (C)\rangle$; $\langle(B)(B) \Rightarrow (BC)\rangle$. However, when the minimum interestingness measure threshold *minThreshold* is greater than 0, the numbers of sequential rules generated are different. Table 3 shows the results of the sequential rules generated from the prefix tree with the different interestingness measures. For example, if the minimum interestingness measure for *minConfidence*, *minLift* and *minCosine* are set at 0.8, then 10 sequential rules satisfy *minConfidence*, 17 sequential rules satisfy *minLift* and only 6 sequential rules satisfy *minCosine* generated as shown in Table 3. To quickly get the support (n_{Post}) of the right-hand side of the rule, the algorithm only needs to traverse the branch of the prefix tree based on the root nodes that are the prefixes of the post sequence of the rule. For example, consider the process of generating sequential rules from the root node $\langle(A)\rangle$ on the prefix tree in Figure 1 with the *Lift* measure. The sequential rules are generated as follows. The root node $\langle(A)\rangle$ has one itemset-extended sequence $\langle(AB)\rangle$ and two sequence-extended sequences $\langle(A)(B)\rangle$ and $\langle(A)(C)\rangle$. Because $\langle(A)\rangle$ is an incomplete prefix of $\langle(AB)\rangle$ and all sub-nodes of $\langle(AB)\rangle$ which extended from $\langle(AB)\rangle$, the algorithm does not need to generate rules from the nodes with prefix $\langle(A)\rangle$. On the contrary, since $\langle(A)\rangle$ is a prefix of the two sequence-extended sequences $\langle(A)(B)\rangle$ and $\langle(A)(C)\rangle$, the following rules can be generated: $\langle(A) \Rightarrow (B)\rangle$ and $\langle(A) \Rightarrow (C)\rangle$. For the sequential rule $\langle(A) \Rightarrow (B)\rangle$, since the support value of the sequential pattern B is 5 by traversing the prefix tree and the calculated *Lift* measure value of the rule in Table 2 is less than *minLift*, the rule $\langle(A) \Rightarrow (B)\rangle$ is not generated. Similarly for the rule $\langle(A) \Rightarrow (C)\rangle$, since the support value of the sequential pattern C is 4 and the calculated *Lift* measure value in Table 2 is 0.9375, which is greater than *minLift*, the sequential rule $\langle(A)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle(C)\rangle(5,4,4,3)$ is generated. Moreover, $\langle(A)\rangle$ is a prefix of all the sub-nodes of $\langle(A)(B)\rangle$ and $\langle(A)(C)\rangle$, such that the algorithm can generate rules as well from the subnodes in a similar process, the subnodes include sequences $\langle(A)(B)(B)\rangle$ and $\langle(A)(B)(C)\rangle$. The above generating sequential rules process is applied for two these subnodes and only $\langle(A)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle(B)(C)\rangle(5,4,4,3)$ sequential rule is generated. The above process can then be repeated for all the subnodes of $\langle(A)\rangle$ to generate sequential rules. All the remaining nodes on the prefix tree in Figure 1 can be similarly processed, and the results are shown in Table 3.

Table 3. The sequential rules with $minThreshold = 0.8$

Prefix	Rules with the confidence measure $(X \overset{(\sup, imv)}{\Rightarrow} Y(n, n_x, n_y, n_{xy}))$	Rules with the lift measure $(X \overset{(\sup, imv)}{\Rightarrow} Y(n, n_x, n_y, n_{xy}))$	Rules with the cosine measure $(X \overset{(\sup, imv)}{\Rightarrow} Y(n, n_x, n_y, n_{xy}))$
$\langle (A)\rangle$		$\langle (A)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle (C)\rangle(5,4,4,3)$; $\langle (A)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle (B)(C)\rangle(5,4,4,3)$	
$\langle (A)(B)\rangle$	$\langle (A)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (B)\rangle(5,3,5,3)$; $\langle (A)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$	$\langle (A)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (B)\rangle(5,3,5,3)$; $\langle (A)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$	$\langle (A)(B)\rangle \overset{(3,0.886)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$
$\langle (AB)\rangle$		$\langle (AB)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle (C)\rangle(5,4,4,3)$; $\langle (AB)\rangle \overset{(3,0.9375)}{\Rightarrow} \langle (B)(C)\rangle(5,4,4,3)$	
$\langle (AB)(B)\rangle$	$\langle (AB)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (B)\rangle(5,3,5,3)$; $\langle (AB)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$	$\langle (AB)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (B)\rangle(5,3,5,3)$; $\langle (AB)(B)\rangle \overset{(3,1.25)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$	$\langle (AB)(B)\rangle \overset{(3,0.886)}{\Rightarrow} \langle (C)\rangle(5,3,4,3)$
$\langle (B)\rangle$	$\langle (B)\rangle \overset{(5,1)}{\Rightarrow} \langle (B)\rangle(5,5,5,5)$; $\langle (B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$; $\langle (B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (B)(B)\rangle(5,5,5,4)$; $\langle (B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (B)(C)\rangle(5,5,4,4)$;	$\langle (B)\rangle \overset{(5,1)}{\Rightarrow} \langle (B)\rangle(5,5,5,5)$; $\langle (B)\rangle \overset{(4,1)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$; $\langle (B)\rangle \overset{(3,1)}{\Rightarrow} \langle (BC)\rangle(5,5,3,3)$; $\langle (B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (B)(B)\rangle(5,5,5,4)$; $\langle (B)\rangle \overset{(4,1)}{\Rightarrow} \langle (B)(C)\rangle(5,5,4,4)$; $\langle (B)\rangle \overset{(3,1)}{\Rightarrow} \langle (B)(BC)\rangle(5,5,3,3)$	$\langle (B)\rangle \overset{(5,1)}{\Rightarrow} \langle (B)\rangle(5,5,5,5)$; $\langle (B)\rangle \overset{(4,0.894)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$; $\langle (B)\rangle \overset{(4,0.894)}{\Rightarrow} \langle (B)(C)\rangle(5,5,4,4)$;
$\langle (B)(B)\rangle$	$\langle (B)(B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (B)\rangle(5,5,5,4)$; $\langle (B)(B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$;	$\langle (B)(B)\rangle \overset{(4,0.8)}{\Rightarrow} \langle (B)\rangle(5,5,5,4)$; $\langle (B)(B)\rangle \overset{(4,1)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$; $\langle (B)(B)\rangle \overset{(3,1)}{\Rightarrow} \langle (BC)\rangle(5,5,3,3)$	$\langle (B)(B)\rangle \overset{(4,0.894)}{\Rightarrow} \langle (C)\rangle(5,5,4,4)$;

5. **Experimental Results.** Experiments were then made to evaluate the performance of the proposed algorithm for sequential rule mining using different interestingness measures. An algorithm modified from the Full algorithm [27], called modified Full algorithm, for generating only traditional sequential rules by using the confidence measure was also run for comparison. All the experiments were performed on a PC machine with dual-core 2.81 GHz, 2 GBs RAM, running Windows XP professional, and implemented by C#. The synthetic databases were generated by the IBM synthetic data generator to mimic transactions in a retail environment. The synthetic data generation program used the following parameters: C was the average number of itemsets per sequence, T was the average number of items per itemset, S was the average number of itemsets in maximal sequences, I was the average number of items in maximal sequences, N was the number of distinct items, and D was the number of sequences. Two synthetic databases, C6T5S4I4N1kD1k and C6T5S4I4N1kD10k, were used in the experiments. Table 4 shows numbers of sequential patterns, numbers of sequential rules with interestingness measures, and the execution time in the two synthetic databases, corresponding to their minimum supports and different rule measures between the proposed algorithm and the modified Full algorithm.

The experimental results in Table 4 show that sequential rule mining with interestingness measures using the proposed algorithm based on the prefix tree was always much faster than that using the modified Full algorithm. The former only consumed a small amount of time when compared with the latter. The time ratio was calculated as follows: (mining time on the prefix tree/mining time on the modified Full) *100%. For the C6T5S4I4N1kD1k dataset with $minSup = 0.5\%$ and the confidence measure, the mining time based on the Prefix tree was 0.22, and based on the Full algorithm was 265.77, such that the time ratio was (0.22/265.77)*100%, which was 0.08%. If the Lift measure was used, the time ratio was (9.47/588.92)*100%, which was 1.61%. Similarly, the time ratio of the cosine measure was (9.35/595.27) *100%, which was 1.57%, for the Piatetsky-Sharipo measure was (9.39/592.15)*100%, which was 1.59%, for the conviction measure was (9.38/596.41)*100%, which was 1.57%, and for the Jaccard measure was (9.4/594.09)*100%, which was 1.58%. Among all the above time ratios, the one for the confidence measure was the smallest, because it did not need to revisit the prefix tree to determine the support of Y (the right-hand side of rules). According to the results in Table 4, it could be easily seen that for low minimum support values, the number of sequential rules generated from sequence databases was large and the proposed algorithm outperformed the modified Full algorithm much. Though the modified Full algorithm had to scan a set of sequential patterns to determine the support of the right-hand side of each rule, the proposed algorithm only traversed the branch of the prefix tree based on the root nodes that were the prefixes of the sequence on the right-hand side of each rule. Thus, the use of the prefix tree structure is an effective approach for mining sequential rules with all of the different interestingness measures.

Table 4. The experimental return for different interestingness measures

Database	Min sup (%)	Number of Sequential Pattern	Number of Rules	Time for Confidence measure		Time for Lift measure		Time for Cosine measure		Time for Piatetsky-Shapiro measure		Time for Conviction measure		Time for Jaccard measure	
				Modified Full	Prefix Tree	Modified Full	Prefix Tree	Modified Full	Prefix Tree	Modified Full	Prefix Tree	Modified Full	Prefix Tree	Modified Full	Prefix Tree
C6T5S4I4 N1kD10k	0.9	6934	8439	8.54	0.04	17.72	1.25	17.92	1.06	17.89	1.06	17.79	1.06	17.84	1.06
	0.8	8430	10576	12.72	0.05	26.79	1.44	26.88	1.43	26.42	1.43	26.8	1.43	27.12	1.44
	0.7	10480	13582	24.39	0.06	51.56	2.18	52.39	2.2	50.61	2.18	51.3	2.13	51.33	2.11
	0.6	13628	18313	40.01	0.07	87.11	3.08	89.99	3.05	90.38	3.06	89.38	3.05	89.35	3.05
	0.5	18461	25848	83.52	0.11	187.71	4.92	189.41	4.86	188.38	5	198.04	5.09	201.43	4.97
	0.4	27168	39661	194.42	0.18	436.96	8.77	431.13	8.67	430.63	8.66	428.44	8.65	431.47	8.79
	0.3	44585	67808	431.51	0.37	1041.51	12.39	1026.06	12.56	1019.44	12.45	1021.57	12.39	1033.2	12.51
C6T5S4I4 N1kD1k	0.9	8795	11214	13.85	0.04	29	1.45	28.71	1.4	29.43	1.4	28.73	1.39	28.66	1.39
	0.8	11211	14815	25.98	0.06	49.52	2.08	49.48	2.09	49.5	2.04	49.04	2.04	49.86	2.05
	0.7	14802	20224	48.19	0.08	105.54	3.12	107.87	3.1	107.47	3.08	105.86	3.07	106.63	3.09
	0.6	20644	29364	107.8	0.12	244.88	5.17	241.09	5.14	238.67	5.12	240.9	5.13	242.68	5.14
	0.5	31311	46577	265.77	0.22	588.92	9.47	595.27	9.35	592.15	9.39	596.41	9.38	594.09	9.4
	0.4	54566	85846	848.89	0.36	1918.58	21.1	1884.54	21.01	1874.99	21	1877.17	21.23	1889.69	21.3
	0.3	124537	214445	4898.17	1.21	9790.15	73.4	10844.19	72.95	10924.12	73.16	11315.21	73.04	11110.08	73.1

6. **Conclusions and Future Work.** In this paper, we have considered and applied several interestingness measures, which have been used for mining association rules and mining unexpected sequential rules, etc., to mine sequential rules from a set of sequential patterns in sequence databases. In large sequence databases, the determination of measured values becomes difficult, and the time required to compute measure values and generate rules is long. This paper thus uses the prefix tree structure to compute the values fast and to reduce the time for mining sequential rules. By traversing the prefix tree, the proposed approach can immediately determine which sequences are the left- and right-hand sides of a rule as well as their support values to compute the interestingness measure values of the rule from the sequential pattern set.

The experimental results show that the performance of the proposed algorithm for mining sequential rules with different interestingness measures on the prefix tree structure is much better than that of the modified Full algorithm. In the future, we will attempt to apply these interestingness measures for mining non-redundant sequential rules [27] based on the prefix tree structure. Besides, the incremental data mining problem have been proposed in recent years [20]. We will also study incremental data mining for maintaining sequential patterns and sequential rules in the future.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Grant No. 61240046) and the Science and Technology Planning Project of Hunan Province (Grant No. 2011FJ3048).

REFERENCES

[1] R. Agrawal and R. Srikant, Mining sequential patterns, *Proc. of the 11th Int. Conf. Data Engineering*, Taipei, Taiwan, pp.3-14, 1995.

[2] R. Agrawal, T. Imielinski and A. Swami, Mining association rules between sets of items in large databases, *Proc. of 1993 ACM-SIGMOD International Conference on Management of Data*, Washington, DC, USA, pp.207-216, 1993.

[3] J. Azé and Y. Kodratoff, A study of the effect of noisy data in rule extraction systems, *Proc. of the 16th European Meeting on Cybernetics and Systems Research*, pp.781-786, 2002.

[4] J. Ayres, J. E. Gehrke, T. Yiu and J. Flannick, Sequential pattern mining using a bitmap representation, *SIGKDD Conf.*, pp.1-7, 2002.

[5] M. J. Berry and G. S. Linoff, *Data Mining Techniques for Marketing, Sales and Customer Support*, John Wiley & Sons, 1997.

[6] S. Brin, R. Motwani, J. Ullman and S. Tsur, Dynamic itemset counting and implication rules for market basket data, *Proc. of the 1997 ACMSIGMOD Int. Conf. on the Management of Data*, pp.255-264, 1997.

[7] S. Brin, R. Motwani and C. Silverstein, Beyond market baskets: Generalizing association rules to correlations, *ACM SIGMOD/PODS 1997 Joint Conference*, pp.265-276, 1997.

[8] Y.-I. Chang, C.-C. Wu, J.-R. Chen and Y.-H. Jeng, Mining sequential motifs from protein databases based on a bit pattern approach, *International Journal of Innovative Computing, Information and Control*, vol.8, no.1(B), pp.647-657, 2012.

[9] G. Dong and J. Pei, Sequence data mining, *Springer Science + Business Media, LLC*, 2007.

[10] J. S. Deogun and L. Jiang, Prediction mining – An approach to mining association rules for prediction, *Proc. of RSFDGrC 2005 Conference*, Regina, Canada, pp.98-108, 2005.

[11] U. Faghihi, P. Fournier-Viger, R. Nkambou and P. Poirier, Generic episodic learning model implemented in a cognitive agent by means of temporal pattern mining, *Proc. of IEA-AIE 2010*, Cordoba, Spain, pp.438-449, 2010.

[12] P. Fournier-Viger, U. Faghihi, R. Nkambou and E. M. Nguifo, CMRules: An efficient algorithm for mining sequential rules common to several sequences, *Proc. of the 23th Int. Florida Artificial Intelligence Research Society Conference*, Daytona, USA, pp.410-415, 2010.

[13] P. Fournier-Viger, R. Nkambou and V. S. Tseng, RuleGrowth: Mining sequential rules common to several sequences by Pattern-Growth, *SAC*, TaiChung, Taiwan, 2011.

[14] I. J. Good, *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*, The MIT Press, Cambridge, MA, USA, 1965.

[15] K. Gouda, M. Hassaan and M. J. Zaki, PRISM: A primal-encoding approach for frequent sequence mining, *Journal of Computer and System Sciences*, vol.76, no.1, pp.88-102, 2010.

[16] H. J. Hamilton and K. Karimi, The TIMERS II: Algorithm for the discovery of causality, *Proc. of the 9th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Hanoi, Vietnam, pp.744-750, 2005.

[17] S. K. Harms, J. Deogun and T. Tadesse, Discovering sequential association rules with constraints and time lags in multiple sequences, *Proc. of the 13th Int. Symp. on Methodologies for Intelligent Systems*, Lyon, France, pp.373-376, 2002.

[18] D. He, X. Zhu and X. Wu, Mining approximate repeating patterns from sequence data with gap constraints, *Computational Intelligence*, vol.27, no.3, pp.336-362, 2011.

[19] R. Hilderman and H. Hamilton, *Knowledge Discovery and Measures of Interest*, Kluwer Academic Publishers, 2001.

[20] T. P. Hong, C. Y. Wang and S. S. Tseng, An incremental mining algorithm for maintaining sequential patterns using pre-large sequences, *Expert Systems with Applications*, vol.38, no.6, pp.7051-7058, 2011.

[21] Y. L. Hsieh, D.-L. Yang and J. Wu, Using data mining to study upstream and downstream causal relationship in stock market, *Proc. of 2006 Joint Conference on Information Sciences*, Kaohsiung, Taiwan, 2006.

[22] R. A. Huebner, Diversity-based interestingness measures for association rule mining, *Proc. of ASBBS*, Las Vegas, vol.16, no.1, 2009.

[23] H. X. Huynh, F. Guillet, J. Blanchard, P. Kuntz, R. Gras and H. Briand, A graph based clustering approach to evaluate interestingness measures: A tool and a comparative study, *Quality Measures in Data Mining*, vol.43, pp.25-50, 2007.

[24] H. Jeffreys, Some tests of significance treated by the theory of probability, *Proc. of the Cambridge Philosophical Society*, vol.31, pp.203-222, 1935.

[25] P. Lenca, P. Mayer, B. Valliant and S. Lallich, On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid, *European Journal of Operational Research*, vol.184, no.2, pp.610-626, 2008.

[26] D. Lo and S.-C. Khoo, SMArTIC: Toward building an accurate, robust and scalable specification miner, *Proc. of SIGSOFT Symposium on the Foundations of Software Engineering*, pp.265-275, 2006.

[27] D. Lo, S. C. Khoo and L. Wong, Non-redundant sequential rules-theory and algorithm, *Information Systems*, vol.34, no.4-5, pp.438-453, 2009.

[28] K. Pearson, Mathematical contributions to the theory of evolution. III. Regression, heredity and panmixia, *Philosophical Transactions of the Royal Society A*, 1896.

[29] J. Pei et al., Mining sequential patterns by pattern-growth: The prefixspan approach, *IEEE Trans. on Knowledge and Data Engineering*, vol.16, no.10, pp.1424-1440, 2004.

[30] G. Piatetsky-Shapiro, Discovery, analysis, and presentation of strong rules, *Knowledge Discovery in Databases*, pp.229-248, 1991.

[31] G. T. Raju, P. S. Satyanarayana and L. M. Patnaik, Knowledge discovery from web usage data: Extraction and applications of sequential and clustering patterns – A survey, *International Journal of Innovative Computing, Information and Control*, vol.4, no.2, pp.381-389, 2008.

[32] C. J. V. Rijsbergen, *Information Retrieval*, 2nd Edition, Butterworths, London, 1979.

[33] I. N. M. Shaharanee, F. Hadzic and T. S. Dillon, Interestingness measures for association rules based on statistical validity, *Knowledge-Based Systems*, vol.24, no.3, pp.386-392, 2011.

[34] M. Spiliopoulou, Managing interesting rules in sequence mining, *Proc. of European Conference on Principles of Data Mining and Knowledge Discovery*, pp.554-560, 1999.

[35] R. Srikant and R. Agrawal, Mining sequential patterns: Generalizations and performance improvements, *Proc. of the 5th Int. Conf. Extending Database Technology*, pp.3-17, 1996.

[36] P. N. Tan, V. Kumar and J. Srivastava, Selecting the right interestingness measure for association patterns, *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery in Databases*, pp.32-41, 2002.

[37] P.-N. Tan, V. Kuma and J. Srivastava, Selecting the right objective measure for association analysis, *Information Systems*, vol.29, no.4, pp.293-313, 2004.

[38] C.-Y. Tsai and P.-H. Lo, A sequential pattern based route suggestion system, *International Journal of Innovative Computing, Information and Control*, vol.6, no.10, pp.4389-4408, 2010.

[39] T.-T. Van, B. Vo and B. Le, Mining sequential rules based on prefix-tree, *Studies in Computational Intelligence*, vol.351, pp.147-156, 2011.

[40] B. Vo and B. Le, Interestingness measures for association rules: Combination between lattice and hash tables, *Expert Systems with Applications*, vol.38, no.9, pp.11630-11640, 2011.

[41] J. Yang, D. Evans, D. Bhardwaj, T. Bhat and M. Das, Mining temporal API rules from imperfect traces, *Proc. of International Conference on Software Engineering*, pp.282-291, 2006.

[42] M. J. Zaki, SPADE: An efficient algorithm for mining frequent sequences, *Machine Learning Journal*, vol.42, no.1/2, pp.31-60, 2000.