Discovering Transitional Patterns and Their Significant Milestones in Transaction Databases

Qian Wan, *Student Member, IEEE*, and Aijun An, *Member, IEEE*

Manuscript received June 23, 2008; revised December 12, 2008.

Q. Wan and A. An are with the Department of Computer Science and Engineering, York University, Toronto, Ontario, Canada.

Abstract

A transaction database usually consists of a set of time-stamped transactions. Mining frequent patterns in transaction databases has been studied extensively in data mining research. However, most of the existing frequent pattern mining algorithms (such as *Apriori* and *FP-growth*) do not consider the time stamps associated with the transactions. In this paper, we extend the existing frequent pattern mining framework to take into account the time stamp of each transaction and discover patterns whose frequency dramatically changes over time. We define a new type of patterns, called transitional patterns, to capture the dynamic behavior of frequent patterns in a transaction database. Transitional patterns include both positive and negative transitional patterns. Their frequencies increase/decrease dramatically at some time points of a transaction database. We introduce the concept of significant milestones for a transitional pattern, which are time points at which the frequency of the pattern changes most significantly. Moreover, we develop an algorithm to mine from a transaction database the set of transitional patterns along with their significant milestones. Our experimental studies on real-world databases illustrate that mining positive and negative transitional patterns is highly promising as a practical and useful approach for discovering novel and interesting knowledge from large databases.

Index Terms

Data mining, association rule, frequent pattern, transitional pattern, significant milestone.

I. INTRODUCTION

The problem of mining frequent itemsets is to find all the itemsets from a transaction database that satisfy a user specified support threshold. It is one of the fundamental and essential operations in many data mining tasks, such as association rule mining [3], [4], [34], sequential pattern mining [5], [29], structured pattern mining [15], correlation mining [10], and associative classification [23]. Since it was first introduced by Agrawal et al. [3] in 1993, the problem of frequent itemset mining has been studied extensively. As a result, a large number of algorithms have been developed in order to efficiently solve the problem, including the most well-known *Aproiri* [4] *FP-growth* [16], and *Eclat* [40] algorithms.

In practice, the number of frequent patterns generated from a data set are often excessively large, and most of them are useless or simply redundant. Thus, there has been interest in discovering new types of patterns, including maximal frequent itemsets [1], [11], [31], closed frequent itemset [26], [28], [41], indirect associations [36]–[38] and emerging patterns [7], [12],

[21]. Mining for maximal or closed frequent itemsets greatly reduces the number of generated patterns by generating only the largest frequent itemsets with no frequent superset or no superset of higher frequency. Indirect associations are closely related to negative associations in that they both represent itemsets that do not have sufficiently high support. Indirect associations provide an effective way to detect interesting negative associations by discovering only "infrequent itempairs that are highly expected to be frequent" without using negative items or domain knowledge. Emerging patterns are defined as itemsets whose frequency increase significantly from one data set to another. They can capture emerging trends from one database to the other.

A common characteristic of the above learning methods is that they treat the transactions in a database equally and do not consider the time stamps associated with the transactions. Therefore, the dynamic behavior of the discovered frequent patterns cannot be revealed by these methods. In this paper, we extend the traditional frequent pattern mining framework to take into account the time stamp of each transaction, i.e., the time when the transaction occurs. We define a new type of patterns, called transitional patterns, to represent patterns whose frequency dramatically changes over time. Transitional patterns include both positive and negative transitional patterns (to be defined in Section III). The frequency of a positive transitional pattern increases dramatically at some time point of a transaction database, while that of a negative transitional pattern decreases dramatically at some point of time. We illustrate transitional patterns using an example as follows.

Consider an example database TDB as shown in Table I, which has 16 transactions of 8 items. Let's focus on two patterns, P_1P_2 and P_1P_3. Without considering the time information of these transactions, P_1P_2 and P_1P_3 have the same significance in the traditional frequent pattern framework since they have the same frequency 62.50%. However, interesting differences between these two patterns can be found after we consider the time information of each transaction in the database, as shown in the third column of Table I. For simplicity, suppose TDB contains all the transactions from November 2005 to February 2007, one transaction per month. We can easily see that before (and including) May 2006, pattern P_1P_2 appears every month; but after May 2006, P_1P_2 only occurs 3 times in 9 transactions, which is equivalent to a frequency of 33.33%. That is to say that the frequency or support of pattern P_1P_2 decreases significantly after May 2006. On the other hand, after July 2006, the frequency of pattern P_1P_3 increases significantly from 33.33% to 100%.

The above observations have shown that frequent patterns discovered by standard frequent

TABLE I

AN EXAMPLE TRANSACTION DATABASE *TDB*

TID	List of itemIDs	Time stamp
001	P_1, P_2, P_3, P_5	Nov. 2005
002	P_1, P_2	Dec. 2005
003	P_1, P_2, P_3, P_8	Jan. 2006
004	P_1, P_2, P_5	Feb. 2006
005	P_1, P_2, P_4	Mar. 2006
006	P_1, P_2, P_4, P_5, P_6	Apr. 2006
007	P_1, P_2, P_3, P_4, P_6	May. 2006
008	P_1, P_4, P_6	Jun. 2006
009	P_4, P_5, P_6	Jul. 2006
010	P_1, P_2, P_3, P_4, P_5, P_6	Aug. 2006
011	P_1, P_3, P_4, P_6	Sep. 2006
012	P_1, P_3, P_5	Oct. 2006
013	P_1, P_2, P_3, P_6, P_7	Nov. 2006
014	P_1, P_3, P_4, P_5	Dec. 2006
015	P_1, P_3, P_4	Jan. 2007
016	P_1, P_2, P_3, P_5	Feb. 2007

pattern mining algorithms may be different in terms of their distributions in a transaction database. However, such patterns cannot be distinguished with the standard algorithms. The objective of the research presented in this paper is to distinguish such frequent patterns, discover frequent patterns whose frequency changes significantly over time and identify the time points for such significant changes.

Transitional patterns have a wide range of potential applications. For example, in the market basket scenario, transitional patterns allow business owners to identify those products or combinations of products that have recently become more and more popular (or not as popular as before) so that they can adjust their marketing strategy or optimize product placement in retail environments. In medical domains, a significant increase in the occurrence of certain symptom in a group of patients with the same disease may indicate a side effect of a new drug. Finding the time point when this symptom starts to occur may help to identify the drug that causes the problem. These patterns can also be used in Web applications, for example, for dynamically

restructuring Web sites by adding links between those pages whose frequency of being visited together becomes more frequent.

The contributions of this paper are summarized as follows.

- We propose a framework for mining a new class of patterns, called transitional patterns. The frequencies of these patterns change significantly at some time points of a transaction database.
- We introduce the concept of significant milestones for each transitional pattern, which are specific time points at which the frequency of the pattern increases or decreases most significantly.
- An algorithm, called *TP-mine*, is designed to mine the set of transitional patterns along with their significant milestones. We show through experiments that the proposed algorithm is highly scalable.
- We present an experimental study to verify the usefulness and effectiveness of transitional patterns. Our results illustrate that mining positive and negative transitional patters is highly promising as a practical approach to discovering new and interesting knowledge from large databases.

The remaining of the paper is organized as follows. In Section II we review the terminologies used in frequent pattern mining. The concepts of positive and negative transitional patters and their significant milestones are introduced in Section III and Section IV, respectively. In Section V, we present an algorithm for mining transitional patterns and their significant milestones. In Section VI, we present an experimental study to demonstrate the utility of transitional patterns in two real-world datasets and the scalability of the proposed algorithm. In Section VII, we compare our method with related work. Finally, in Section VIII, we conclude paper and present some ideas for future work.

II. Preliminaries and Notations

Mining frequent patterns is one of the fundamental operations in data mining applications for extracting interesting patterns from databases. In this session, we briefly review the basic concepts of frequent pattern mining. Table II summarizes the notations that will be used throughout this paper and their meanings.

TABLE II

SUMMARY OF NOTATIONS AND THEIR MEANING

$\mathcal{D}$	a database of transactions
$\|\|\mathcal{S}\|\|$	the cardinality of set $\mathcal{S}$
TDB	an example transaction database
$cov(X)$	the cover of pattern X in $\mathcal{D}$
$sup(X)$	the support of pattern X in $\mathcal{D}$
$\rho(\mathcal{T})$	the position of transaction $\mathcal{T}$ in $\mathcal{D}$
$\tau^i(X)$	the i^{th} transaction of pattern X in $\mathcal{D}$
$\xi^i(X)$	the i^{th} milestone of pattern X in $\mathcal{D}$
T_ξ	a range of $\xi^i(X)$ in $\mathcal{D}$
$sup^i_-(X)$	the support of pattern X before its i^{th} *milestone* in $\mathcal{D}$
$sup^i_+(X)$	the support of pattern X after its i^{th} *milestone* in $\mathcal{D}$
$tran^i(X)$	the transitional ratio of pattern X at its i^{th} milestone in $\mathcal{D}$
t_s	pattern support threshold
t_t	transitional pattern threshold
PTP	positive transitional pattern
NTP	negative transitional pattern
$\langle \xi^{\mathcal{M}}(X), tran^{\mathcal{M}}(X) \rangle$	significant frequency-descending milestone of X
$\langle \xi^{\mathcal{N}}(Y), tran^{\mathcal{N}}(Y) \rangle$	significant frequency-descending milestone of Y

Let $\mathcal{I}$ = $\{i_1, i_2, \ldots, i_m\}$ be a set of *m* distinct *items*. A subset $X \subseteq \mathcal{I}$ is called an *itemset* or a *pattern*. A *k*-itemset is an itemset that contains *k* items. In this paper, we use AB to represent pattern $\{A, B\}$, where $A \in \mathcal{I}$ and $B \in \mathcal{I}$, for simplicity. A transaction over $\mathcal{I}$ is a couple $\mathcal{T}$ = (tid, I) where tid is the transaction identifier (or time-stamp) and $I \subseteq \mathcal{I}$ is an itemset. A transaction $\mathcal{T}$ = (tid, I) is said to support an itemset $X \subseteq \mathcal{I}$, if and only if X $\subseteq I$. A transaction database $\mathcal{D}$ over $\mathcal{I}$ is a set of transactions over $\mathcal{I}$.

The *cover* of an itemset X in $\mathcal{D}$, denoted as $cov(X, \mathcal{D})$, consists of the set of transactions in $\mathcal{D}$ that support X:

$$cov(X, \mathcal{D}) := \{(tid, I) \mid (tid, I) \in \mathcal{D}, X \subseteq I\}.$$

An itemset *X* in a transaction database $\mathcal{D}$ has a *support*, denoted as $sup(X, \mathcal{D})$, which is the

ratio of transactions in $\mathcal{D}$ containing *X*. That is,

$$sup(X, \mathcal{D}) := \frac{\|cov(X, \mathcal{D})\|}{||\mathcal{D}||},$$

where $\|\mathcal{S}\|$ is the cardinality of set $\mathcal{S}$. In the rest of the paper we omit $\mathcal{D}$ whenever it is clear from the context.

Given a transaction database $\mathcal{D}$ and a user specified minimum support threshold min_sup, an itemset X is called a frequent itemset or frequent pattern in $\mathcal{D}$ if $sup(X) \geq$ ***min_sup***. Accordingly, X is called an infrequent itemset or infrequent pattern if $sup(X) <$ ***min_sup***. The problem of mining frequent patterns is to find the complete set of frequent patterns in a transaction database with respect to a given support threshold.

III. Transitional Patterns

In this section we first define several concepts relevant to our proposed framework, and then present formal definitions of positive and negative transitional patterns.

A. Definition of milestone

In order to provide formal definitions of transitional patterns, we first introduce the concept of a pattern's *milestones*.

Definition 3.1: Assuming that the transactions in a transaction database $\mathcal{D}$ are ordered by their time-stamps, the *position* of a transaction $\mathcal{T}$ in $\mathcal{D}$, denoted as $\rho(\mathcal{T})$, is the number of transactions whose time-stamp is less than or equal to that of $\mathcal{T}$. Thus, $1 \leq \rho(\mathcal{T}) \leq \|\mathcal{D}\|$.

Definition 3.2: The i^{th} *transaction* of a pattern X in $\mathcal{D}$, denoted as $\tau^i(X)$, is the i^{th} transaction in $cov(X)$ with transactions ordered by their positions, where $1 \leq i \leq \|cov(X, \mathcal{D})\|$.

Definition 3.3 (i^{th} milestone): The i^{th} *milestone* of a pattern X in $\mathcal{D}$, denoted as $\xi^i(X)$, is defined as:

$$\xi^i(X) := \frac{\rho(\tau^i(X))}{\|\mathcal{D}\|} \times 100\% \tag{1}$$

where $1 \leq i \leq \|cov(X)\|$.

According to this definition, the i^{th} milestone of pattern X represents the relative position (expressed in a percentage) of the i^{th} transaction of X in $\mathcal{D}$. For instance, in the example database TDB in Table I, we have $\xi^4(P_1P_2) = \frac{4}{16} = 25\%$ and $\xi^4(P_1P_3) = \frac{10}{16} = 62.5\%$.

Definition 3.4: The support of a pattern X before its i^{th} milestone in $\mathcal{D}$, denoted as $sup^i_-(X)$, is defined as:

$$sup^i_-(X) := \frac{i}{\rho(\tau^i(X))} \tag{2}$$

where $1 \leq i \leq \|cov(X)\|$.

Definition 3.5: The support of a pattern X after its i^{th} milestone in $\mathcal{D}$, denoted as $sup^i_+(X)$, is defined as:

$$sup^i_+(X) := \frac{\|cov(X)\| - i}{\|\mathcal{D}\| - \rho(\tau^i(X))} \tag{3}$$

where $1 \leq i \leq \|cov(X)\|$.

For example, $sup^6_-(P_1P_2)$ = 1.0 and $sup^6_+(P_1P_2)$ = 0.4.

B. Transitional Ratio

We define *transitional ratio* below to measure the difference of a pattern's frequency before and after its ith milestone.

Definition 3.6 (Transitional Ratio): The *transitional ratio* of pattern X at its i^{th} milestone in $\mathcal{D}$ is defined as:

$$tran^i(X) := \frac{sup^i_+(X) \; - \; sup^i_-(X)}{MAX(sup^i_+(X), \; sup^i_-(X))} \tag{4}$$

where $1 \leq i \leq \|cov(X)\|$.

It's easy to see that the higher the absolute transitional ratio of a pattern at its i^{th} milestone, the greater the difference between its supports before and after its i^{th} milestone. A nice feature of this definition is that the value of a transitional ratio is between -1 and 1. As for transitional patterns, we are interested in patterns whose absolute values of transitional ratio are large, which are defined below.

C. Positive and negative transitional patterns

Definition 3.7 (Transitional Pattern): A pattern X is a *Transitional Pattern* (*TP*) in $\mathcal{D}$ if there exists at least one milestone of X, $\xi^k(X) \in T_\xi$, such that:

(a) $sup^k_-(X) \geq t_s$ and $sup^k_+(X) \geq t_s$;

(b) $|tran^k(X)| \geq t_t$.

where T_ξ is a range of $\xi^i(X)$ ($1 \leq i \leq \|cov(X)\|$), t_s and t_t are called *pattern support threshold* and *transitional pattern threshold*, respectively. Moreover, X is called a *Positive Transitional Pattern* (*PTP*) when $tran^k(X) > 0$; and X is called a *Negative Transitional Pattern* (*NTP*) when $tran^k(X) < 0$.

Let us explain the range T_ξ first. In order to obtain reliable values for $sup^i_-(X)$ and $sup^i_+(X)$, the numbers of transactions before $\xi^i(X)$ and after $\xi^i(X)$ should not be too small. Otherwise, a "uniformly distributed" frequent pattern that happens to occur in the first few (or the last few) transactions may have a large absolute value of transitional ratio due to the fact that its support is too high before $\xi^i(X)$ (or after $\xi^i(X)$). Thus, $\xi^i(X)$ should be in an appropriate range T_ξ, to allow a reasonable amount of data before $\xi^i(X)$ and after $\xi^i(X)$ in the database. Moreover, in practice, T_ξ can be specified by the user according to their own interest. For instance, in order to find interesting patterns in the example database TDB which occur during the year 2006, T_ξ should be set to [12.50% ... 87.50%], since 12.50% is the starting time point for 2006 and 87.50% is the ending point of 2006.

The reason we have condition (a) for a transitional pattern is that, if we do not have this condition, any pattern that does not occur at the beginning of the transaction database has a transitional ratio very close to 1 when the pattern first occurs in the database (or any pattern that does not occur at the end of the transaction database has a transitional ratio equal to -1 after its last occurrence in the database). However, such a pattern may be just a sporadic pattern that occurs occasionally in the database, which is not interesting at all. By adding condition (a), a transitional pattern is also a frequent pattern in the database with respect to pattern support threshold t_s [1]. In other words, we are only interested in frequent patterns whose frequency changes dramatically before and after one of its milestones in the transaction database. In practice, t_s should be set to a low value for real datasets, as experienced in frequent pattern mining. Intuitively, the transitional pattern threshold t_t should be set to a value higher than or equal to 0.5.

[1]It is trivial to prove that if a pattern X is frequent before $\xi^i(X)$ and after $\xi^i(X)$, it must be frequent in the whole database. However, please note that if a pattern is frequent on $\mathcal{D}$ with respect to t_s, there may not exist a milestone $\xi^i(X)$ such that $sup^i_-(X) \geq t_s$ and $sup^i_+(X) \geq t_s$. Therefore, if we want to find all the transitional patterns in a set of frequent patterns discovered with support threshold min_sup, the pattern support threshold t_s for transitional patterns should be set to be smaller than min_sup. We consider this problem to be a different problem from what this paper is concerned about. The pattern support threshold t_s in Definition 3.7 is only for defining transitional patterns to avoid generalization over insufficient data.

For example, if $t_s = 0.05$ and $t_t = 0.5$, pattern P_1P_3 in the example database TDB is a positive transitional pattern because there exists a milestone of P_1P_3, such as $\xi^5(P_1P_3) = 68.75\%$ corresponding to the end of September 2006, where the transitional ratio of the pattern is greater than 0.5 and the pattern is frequent before and after the milestone. Similarly, P_1P_2 is a negative transitional pattern in TDB.

Note that, theoretically, a pattern X can be both a positive transitional pattern and a negative transitional pattern in the same transaction database if there exist two milestones $\xi^m(X)$ and $\xi^n(X)$ so that conditions (a) and (b) are satisfied at both $\xi^m(X)$ and $\xi^n(X)$, where $tran^m(X) > 0$ and $tran^n(X) < 0$. For example, in the example database TDB, pattern P_4P_6 is both a positive transitional pattern and a negative transitional pattern because its transitional ratio at milestone $\xi^1(P_4P_6)$ is $+66.67\%$ and the one at milestone $\xi^5(P_4P_6)$ is -66.67%, and condition (a) is also satisfied at both milestones.

IV. Significant milestones

There may be multiple milestones at which a transitional pattern satisfies conditions (a) and (b) in Definition 3.7. People are usually interested in the milestones where the frequency of a transitional pattern changes the most significantly. Below we define the concept of *significant milestones* to represent such positions. The significant milestones can be classified into frequency-ascending milestones and frequency-descending milestones.

A. Significant frequency-ascending milestone

Definition 4.1 (Significant frequency-ascending milestone): The significant frequency-ascending milestone of a positive transitional pattern X with respect to a time period T_ξ is defined as a tuple, $\langle \xi^{\mathcal{M}}(X), tran^{\mathcal{M}}(X) \rangle$, where $\xi^{\mathcal{M}}(X) \in T_\xi$ is the $\mathcal{M}^{th}$ milestone of X such that:

1) $sup_-^{\mathcal{M}}(X) \geq t_s$;
2) $\forall\ \xi^i(X) \in T_\xi$, $tran^{\mathcal{M}}(X) \geq tran^i(X)$.

Table III lists the transitional ratios of four patterns in the example database TDB with T_ξ = [25%, 75%]. Figure 1 illustrates how the transitional ratios of these four patterns change along their milestones. Assuming that the support threshold is 5% and the transitional pattern threshold is 50%, P_1P_3 and P_4P_6 are positive transitional patterns. The significant frequency-ascending

TABLE III

MILESTONES AND TRANSITIONAL RATIOS OF EXAMPLE PATTERNS WITH T_ξ = [25%, 75%] IN *TDB* (%)

i	$\xi^i(P_1)$	$tran^i(P_1)$	$\xi^i(P_1P_2)$	$tran^i(P_1P_2)$	$\xi^i(P_1P_3)$	$tran^i(P_1P_3)$	$\xi^i(P_4P_6)$	$tran^i(P_4P_6)$
1							37.50%	+66.67
2							43.75%	+35.71
3					43.75%	+44.90	50.00%	0
4	25.00%	-8.33	25.00%	-50	62.50%	+60	56.25%	-35.71
5	31.25%	-9.09	31.25%	-54.55	68.75%	+54.55	62.50%	-66.67
6	37.50%	-10	37.50%	-60	75.00%	+50	68.75%	-100
7	43.75%	-11.11	43.75%	-66.67				
8	50.00%	-12.5	62.50%	-44.90				
9	62.50%	+11.11						
10	68.75%	+10						
11	75.00%	+9.09						

milestone for P_1P_3 is $\langle$62.5%, +60$\rangle$, and the significant frequency-ascending milestone for P_4P_6 is $\langle$37.50%, +66.67$\rangle$.

The reason for having condition 1) in Definition 4.1 is as follows. Positive transitional patterns usually occur sporadically at the beginning of the transaction database and are more heavily distributed at the latter part of the database. For such sporadic occurrences, the transitional ratios at the corresponding milestones may be very high, but these positions are not interesting because the sporadic nature of the occurrence.

For example, suppose that in a dataset with 1000 transactions, a positive transitional pattern occurs in every transaction in the second half of the database, but sporadically occurs 10 times between the 100th and the 500th transactions. Assume that its first occurrence is at the 100th transaction, its transitional ratio is 98.23% at the 1^{st} milestone (corresponding to the 100th transaction) and its transitional ratio is only 98% at the 10^{th} milestone (corresponding to the 500th transaction). But the latter milestone is much more interesting. By using constraint $sup_{-}^{\mathcal{M}}(X) \geq t_s$, sporadic occurrences of a pattern at the beginning of the database are not considered as significant milestones because the pattern is infrequent at that milestone and we haven't had enough information to see the trend of the pattern yet.

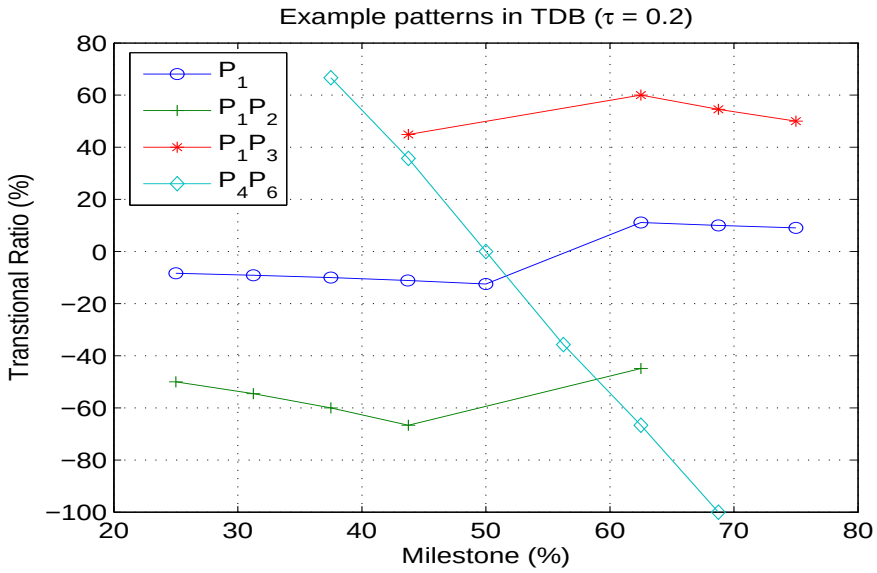

Fig. 1. Transitional ratios in TDB

Please note that the use of this constraint does not make us miss the significant milestone in the situation where a positive transitional pattern X starts to occur very often right after its first milestone. In this case, the significant milestone of X may or may not be at the place of the first occurrence, but if not, it is not far away from the first milestone because the value of $sup^i_-(X)$ generally increases quickly as i gets larger. Please also note that since X is a positive transitional pattern at $\xi^{\mathcal{M}}(X)$, it is easy to see that $sup^{\mathcal{M}}_+(X) > t_s$ since $sup^{\mathcal{M}}_+(X) > sup^{\mathcal{M}}_-(X)$ and $sup^{\mathcal{M}}_-(X) \geq t_s$.

B. Significant frequency-descending milestone

Similarly, the significant frequency-descending milestone for a negative transitional pattern is defined below.

Definition 4.2 (Significant frequency-descending milestone): The significant frequency-descending milestone of a negative transitional pattern Y with respect to a time period T_ξ is defined as a tuple, $\langle \xi^{\mathcal{N}}(Y), tran^{\mathcal{N}}(Y) \rangle$, where $\xi^{\mathcal{N}}(Y) \in T_\xi$ is the $\mathcal{N}^{th}$ milestone of Y such that:

1) $sup^{\mathcal{N}}_+(Y) \geq t_s$;
2) $\forall\ \xi^j(Y) \in T_\xi$, $tran^{\mathcal{N}}(Y) \leq tran^j(Y)$.

To give an example, patterns P_1P_2 and P_4P_6 in Table III are negative transitional patterns. Their significant frequency-descending milestones are $\langle$43.75%, -66.67$\rangle$ and $\langle$62.50%, -66.67$\rangle$, respectively. The reason to have Condition 1) in Definition 4.2 is similar to that in Definition 4.1. Note that even though the transitional ratio of P_4P_6 is -1 at its last milestone 68.75%, it can not be considered as a significant milestone because condition 1 in Definition 4.2 is not satisfied, due to the fact that P_4P_6 does not occur after that milestone.

Theoretically, a transitional pattern may have both significant frequency-ascending and significant frequency-descending milestones if it is both a positive and a negative transitional pattern. Also, a positive (or negative) transitional pattern may have more than one significant frequency-ascending (or frequency-descending) milestones.

The significant milestones capture the most significant changes of a transitional pattern within a time period. In the real world, an evolving pattern may exist a more complicated trend (such as periodically evolving trends). Such a pattern may not be captured as a transitional pattern if its transitional ratio at any time point stays lower than the transitional pattern threshold. Finding such patterns is not a concern of this paper, but can be considered as future work.

V. Mining transitional patterns and their significant milestones

In this section, we present an algorithm, called *TP-mine*, for mining the set of positive and negative transitional patterns and their significant milestones with respect to a pattern support threshold and a transitional pattern threshold. The algorithm is given as follows.

A. TP-mine algorithm

Algorithm:

TP-mine. (***Mine*** *the set of* ***T****ransitional* ***P****atterns and their significant milestones*)

Input:

A transaction database ($\mathcal{D}$), an appropriate milestone range that the user is interested (T_ξ), pattern support threshold (t_s) and transitional pattern threshold (t_t).

Output:

The set of transitional patterns ($\mathcal{S}_{PTP}$ and $\mathcal{S}_{NTP}$) with their significant milestones.

Method:

1: Extract frequent patterns, P_1, P_2, ..., P_n, and their supports using a frequent pattern generation algorithm with *min_sup* = t_s.
2: Scan the transactions from the first transaction to the last transaction before T_ξ to compute the support counts, c_k ($1 \geq k \geq n$), of all the n frequent patterns on this part of the database.
3: $\mathcal{S}_{PTP} = \emptyset$, $\mathcal{S}_{NTP} = \emptyset$
4: **for all** k = 1 to n **do**
5: $MaxTran(P_k) = 0$, $MinTran(P_k) = 0$
6: $S_{FAM}(P_k) = \emptyset$, $S_{FDM}(P_k) = \emptyset$
7: **end for**
8: **for all** transactions $\mathcal{T}_i$ whose position satisfying T_ξ **do**
9: **for** k = 1 to n **do**
10: **if** $\mathcal{T}_i \supseteq P_k$ **then**
11: c_k = c_k + 1
12: **if** $sup_{-}^{c_k}(P_k) \geq t_s$ **and** $sup_{+}^{c_k}(P_k) \geq t_s$ **then**
13: **if** $tran^{c_k}(P_k) \geq t_t$ **then**
14: **if** $P_k \notin S_{PTP}$ **then**
15: Add P_k to $\mathcal{S}_{PTP}$
16: **end if**
17: **if** $tran^{c_k}(P_k) > MaxTran(P_k)$ **then**
18: $S_{FAM}(P_k)$ = $\{\langle \xi^{c_k}(P_k), tran^{c_k}(P_k)\rangle\}$
19: $MaxTran(P_k) = tran^{c_k}(P_k)$
20: **else if** $tran^{c_k}(P_k) = MaxTran(P_k)$ **then**
21: Add $\langle \xi^{c_k}(P_k), tran^{c_k}(P_k)\rangle$ to $S_{FAM}(P_k)$
22: **end if**
23: **else if** $tran^{c_k}(P_k) \leq -t_t$ **then**
24: **if** $P_k \notin S_{NTP}$ **then**
25: Add P_k to $\mathcal{S}_{NTP}$
26: **end if**
27: **if** $tran^{c_k}(P_k) < MinTran(P_k)$ **then**
28: $S_{FDM}(P_k) = \{\langle \xi^{c_k}(P_k), tran^{c_k}(P_k)\rangle\}$

29: $MinTran(P_k) = tran^{c_k}(P_k)$
30: **else if** $tran^{c_k}(P_k) = MinTran(P_k)$ **then**
31: Add $\langle \xi^{c_k}(P_k), tran^{c_k}(P_k) \rangle$ to $S_{FDM}(P_k)$
32: **end if**
33: **end if**
34: **end if**
35: **end if**
36: **end for**
37: **end for**
38: **return** S_{PTP} and $S_{FAM}(P_k)$ for each $P_k \in S_{PTP}$
39: **return** S_{NTP} and $S_{FDM}(P_k)$ for each $P_k \in S_{NTP}$

There are two major phases in this algorithm. During the first phase (Step 1), all frequent itemsets along with their supports are initially derived using a standard frequent pattern generation algorithm, such as *Apriori* [4] or *FP-growth* [16], with t_s as the minimum support threshold. In the second phase (starting from Step 2 to the end), the algorithm finds all the transitional patterns and their significant milestones based on the set of frequent itemsets. As mentioned before, a pattern that is frequent before and after one of its milestones in $\mathcal{D}$ with respect to support threshold t_s must be frequent on $\mathcal{D}$ with respect to the same threshold. Thus, it is safe for us to first mine the frequent itemsets on the entire database using the threshold t_s and then find the transitional patterns based on the set of frequent itemsets.

In Step 2, the support counts of all the frequent patterns on the set from the first transaction to the transaction right before the time period T_ξ are collected. They are used later in computing $sup_{-}^{c_k}(P_i)$, where P_i is a frequent pattern. Step 3 initializes the set of positive transitional patterns ($\mathcal{S}_{PTP}$) and the set of negative transitional patterns ($\mathcal{S}_{NTP}$) to empty. Steps 4-7 initialize the set of significant frequency-ascending milestones for each frequent pattern P_k, $S_{FAM}(P_k)$, and the set of significant frequency-descending milestones for each frequent pattern P_k, $S_{FDM}(P_k)$, to empty. It also initializes the maximal and minimal transitional ratios of P_k, denoted by $MaxTran(P_k)$ and $MinTran(P_k)$, to zero.

After the initializations, the algorithm continues to scan the database $\mathcal{D}$ to find the milestones of P_k within the range T_ξ. At each valid milestone $\xi^{c_k}(P_k)$ during the scan, it calculates the support

of P_k before $\xi^{c_k}(P_k)$, i.e., $sup_-^{c_k}(P_k)$, and the support of P_k after $\xi^{c_k}(P_k)$, i.e., $sup_+^{c_k}(P_k)$[2]. If both of them are greater than t_s, the algorithm then checks the transitional ratio of P_k. If the ratio is greater than t_t, then P_k is a positive transitional pattern and is added into the set $\mathcal{S}_{PTP}$.

Then, the algorithm checks whether the transitional ratio of P_k is greater than the current maximal transitional ratio of P_k. If yes, the set of significant frequency-ascending milestones of P_k, i.e., $S_{FAM}(P_k)$, is set to contain $\{\langle \xi^{c_k}(P_k), tran^{c_k}(P_k)\rangle\}$ as its single element. If not but it is equal to the current maximal transitional ratio of P_k, $\{\langle \xi^{c_k}(P_k), tran^{c_k}(P_k)\rangle\}$ is added into $S_{FAM}(P_k)$.

Similarly, Steps 23-32 are for finding the set of negative transitional patterns and their significant frequency-descending milestones.

B. Database scan and time complexity

If we do not consider the step for generating frequent patterns (i.e., Step 1), the *TP-mine* algorithm scans the database only once to find all the transitional patterns and their significant milestones with respect to a pattern support threshold and a transitional pattern threshold. Suppose the number of frequent patterns generated from Step 1 is *n*, the time complexity of the *TP-mine* algorithm from Step 2 to Step 35 is $\mathcal{O}(\|\mathcal{D}\| + n \times \|T_\xi\|)$, where $\|\mathcal{D}\|$ is the number of transactions in $\mathcal{D}$ and $\|T_\xi\|$ is maximum number of milestones of P_k ($1 \leq k \leq n$).

The number of database scans and time complexity in Step 1 depends on the algorithm used for mining frequent patterns. For example, if *FP-growth* [16] is used, only two database scans are needed in Step 1. The total number of scans for mining transitional patterns is thus 3.

Please note that in our design of the *TP-mine* algorithm, finding transitional patterns is a separate step after generation of frequent patterns. It might be possible to incorporate transitional pattern mining into a frequent pattern mining process. However, it is not a good idea to incorporate it into either *Apriori* [4] or *FP-growth* [16], which are the two most popular algorithms for mining frequent patterns. *Apriori* finds frequent patterns by multi-level candidate generation and testing, which involves multiple database scans. Since transitional patterns do not have the so-called *downward closure property* (i.e., a sub-pattern of a transitional pattern may not

[2]Note that the two supports, $sup_-^{c_k}(P_k)$ and $sup_+^{c_k}(P_k)$, can be calculated based on the support count of the pattern before $\xi^{c_k}(P_k)$, which was collected in Step 2 and Step 11, and the support count of P_k over $\mathcal{D}$ computed in Step 1.

TABLE IV

DATABASE CHARACTERISTICS

Database	number of items	number of transactions	number of Frequent Patterns	number of *PTP*s	number of *NTP*s	T_ξ
Retail	16,470	88,163	580	21	48	[25%, 75%]
Livelink	38,679	30,586	125	22	22	
t_s = 0.5% and t_t = 0.5						

be a transitional pattern), the number of candidates evaluated in *Apriori* cannot be reduced based on the fact that a frequent pattern is found to be a transitional pattern. Thus, we would need to compute the transitional ratio for each candidate, resulting in a much higher time complexity than using the two-phase *TP-mine* algorithm. As for *FP-growth*, since a *FP-tree* does not contain the time information of the transaction database, it is not possible to mine transitional patterns from the *FP-tree* without significant modification of the tree to adapt time information.

Therefore, we choose to design a two-phase algorithm to mine transitional patterns, i.e., mining transitional patterns after mining frequent patterns. A benefit of such a design is that we can make use of the existing efficient and scalable frequent pattern mining algorithms, such as *FP-growth*, to improve the overall efficiency of the process. In the next section, we will show that the second phase of the *TP-mine* algorithm is also highly scalable.

VI. EXPERIMENTAL STUDIES

To demonstrate the utility of transitional patterns and the efficiency of the *TP-mine* algorithm, we have performed two sets of experiments using datasets from two real-world domains: retail market basket and web log data. Table IV summaries the parameters of each dataset along with the threshold values used in our experiments.

A. Retail market basket data

The Retail dataset was obtained from the Frequent Itemset Mining Dataset Repository[3]. It contains the (anonymized) retail market basket data from an anonymous Belgian supermarket

[3]http://fimi.cs.helsinki.fi/data/

TABLE V

TOP 16 POSITIVE TRANSITIONAL PATTERNS IN RETAIL DATASET

rank	*Transitional Pattern* P	$sup_{-}^{\mathcal{M}}(P)$ (‰)	$sup_{+}^{\mathcal{M}}(P)$ (‰)	$\langle\ \xi^{\mathcal{M}}(P),\ tran^{\mathcal{M}}(P)\ \rangle$ (%)	$sup(P)$ (‰)	*support* rank
1	$\{R12925\}$	5.10	32.92	⟨ 58.52, +84.52 ⟩	16.64	72
2	$\{R14098\}$	5.01	29.53	⟨ 60.71, +83.05 ⟩	14.64	88
3	$\{R39, R12925\}$	5.09	22.93	⟨ 68.88, +77.81 ⟩	10.64	149
4	$\{R413\}$	6.24	26.38	⟨ 25.09, +76.34 ⟩	21.32	49
5	$\{R48, R12925\}$	5.0	19.57	⟨ 70.72, +74.43 ⟩	9.27	180
6	$\{R12929\}$	5.01	18.33	⟨ 74.44, +72.64 ⟩	8.42	221
7	$\{R48, R413\}$	5.01	16.56	⟨ 31.92, +69.74 ⟩	12.87	110
8	$\{R39, R413\}$	5.04	16.28	⟨ 30.81, +69.02 ⟩	12.82	112
9	$\{R405\}$	5.09	15.03	⟨ 50.86, +66.15 ⟩	9.97	160
10	$\{R39, R48, R413\}$	5.0	14.05	⟨ 57.38, +64.41 ⟩	8.86	200
11	$\{R10515\}$	5.02	13.68	⟨ 42.47, +63.30 ⟩	10	159
12	$\{R649\}$	5.02	12.54	⟨ 37.49, +59.94 ⟩	9.72	166
13	$\{R389\}$	5.01	12.21	⟨ 38.71, +58.98 ⟩	9.43	171
14	$\{R809\}$	5.22	12.62	⟨ 74.56, +58.65 ⟩	7.1	305
15	$\{R49\}$	6.3	14.86	⟨ 25.21, +57.62 ⟩	12.7	113
16	$\{R441\}$	5.32	11.52	⟨ 74.40, +53.80 ⟩	6.91	323

store [9]. Over the entire data collection period, approximately 5 months, the supermarket store carries $16,470$ unique SKU's[4], and the total amount of receipts being collected equals $88,163$.

Table V shows the first 16 positive transitional patterns in Retail. These patterns are ranked by the transitional ratios at their significant frequency-ascending milestones. For positive transitional patterns, the greater the ratio, the higher the rank; while for negative transitional patterns (shown in Table VI), the less the ratio, the higher the rank.

The first positive transitional pattern, product $\{R12925\}$, has a support rank of 72 in the whole Retail dataset, which represents a mediocre frequency. From its significant milestone, we notice that before the milestone 58.52%, its frequency is just a little bit greater than the minimum support threshold (which is 0.5%); but its frequency increases over 6 times after its significant milestone, which is twice as much as its frequency over the whole Retail dataset.

[4]A Stock Keeping Unit, or SKU, is a unique identifier for each distinct product or service that can be ordered from a supplier.

TABLE VI

TOP 16 NEGATIVE TRANSITIONAL PATTERNS IN RETAIL DATASET

rank	*Transitional Pattern* P	$sup^{\mathcal{N}}_{-}(P)$ (‰)	$sup^{\mathcal{N}}_{+}(P)$ (‰)	$\langle\ \xi^{\mathcal{N}}(P),\ tran^{\mathcal{N}}(P)\ \rangle$ (%)	$sup(P)$ (‰)	*support* rank
1	$\{R1327\}$	31.8	5.03	$\langle$ 56.89, -84.2 $\rangle$	20.26	54
2	$\{R39, R1327\}$	25.51	5.01	$\langle$ 39.52, -80.37 $\rangle$	13.11	106
3	$\{R48, R1327\}$	20.81	5.01	$\langle$ 37.77, -75.91 $\rangle$	10.98	143
4	$\{R32, R39, R41\}$	45	13.04	$\langle$ 42.93, -71.02 $\rangle$	26.76	36
5	$\{R41, R225\}$	17.22	5.01	$\langle$ 40.44, -70.91 $\rangle$	9.95	161
6	$\{R32, R41\}$	60.82	17.92	$\langle$ 42.73, -70.53 $\rangle$	36.25	20
7	$\{R38, R39, R41\}$	57.87	17.19	$\langle$ 42.81, -70.29 $\rangle$	34.61	22
8	$\{R32, R39, R41, R48\}$	31.07	9.34	$\langle$ 42.93, -69.94 $\rangle$	18.67	63
9	$\{R38, R39, R41, R48\}$	37.63	11.34	$\langle$ 42.78, -69.87 $\rangle$	22.58	46
10	$\{R41, R65\}$	18.72	5.69	$\langle$ 42.97, -69.61 $\rangle$	11.29	137
11	$\{R38, R41\}$	73.31	22.37	$\langle$ 42.86, -69.48 $\rangle$	44.2	17
12	$\{R1344\}$	16.35	5.01	$\langle$ 31.16, -69.36 $\rangle$	8.54	215
13	$\{R32, R41, R48\}$	38.71	11.89	$\langle$ 42.93, -69.29 $\rangle$	23.4	42
14	$\{R38, R41, R48\}$	44.55	13.76	$\langle$ 42.78, -69.12 $\rangle$	26.93	35
15	$\{R41\}$	278.53	87.39	$\langle$ 42.97, -68.62 $\rangle$	169.52	6
16	$\{R39, R41\}$	212.74	66.77	$\langle$ 42.95, -68.62 $\rangle$	129.47	7

This unusual phenomena might be the result of a special event around that time point, such as a new advertisement or a sale promotion. In order to satisfy customers' increasing demands for product $\{R12925\}$, the store has to take actions to enhance the supply of this product. Moreover, the supplies of products $\{R39\}$ and $\{R48\}$ need to be enhanced as well because of their co-occurrences with product $\{R12925\}$ in the third and fifth positive transitional patterns.

As we can see from Table V, there are 3 items $R39$, $R48$ and $R413$ in the tenth positive transitional pattern. This pattern can be easily ignored by traditional frequent pattern mining framework since its support is relatively low (ranked 200 out of 580). However, according to the corresponding significant milestone, these products appear together more frequently after the milestone 57.38%. Therefore, putting these products close to each other or starting a package promotion for these products might be very useful in selling more of these products. This idea is also backed up by the seventh and eighth positive transitional patterns.

The first 16 negative transitional patterns in Retail are listed in Table VI. The frequency of

the sixth negative transitional pattern $\{R32, R41\}$ is very high, ranked 20 out of 580 frequent itemsets. Its frequency is much higher before the milestone 42.73%, almost twice as much as its frequency over the whole dataset; but it decreases significantly afterwards. This could be the main reason why the frequencies of the fourth and eighth negative transitional patterns decrease after almost the same time since product $\{R39\}$ has the highest frequency in the Retail dataset and appears in most of the top positive transitional patterns. New marketing strategies should be planned for products $\{R32\}$ and $\{R41\}$, such as a new advertisement or price dropping, to resume the sales volume for these two products and other associated products.

Another interesting observation is that the significant milestones of most top negative transitional patterns occur around 40% to 45%. This information will encourage decision makers to find out the reason and take corresponding actions to prevent the sales of these products from decreasing further more.

B. Livelink web log data

The Livelink dataset was first used in [19] to discover interesting association rules from Livelink[5] web log data. This data set is not publicly available for proprietary reasons. The log files contain Livelink access data for a period of two months (April and May 2002). The size of the raw data is 7GB. The data describe more than 3,000,000 requests made to a Livelink server from around 5,000 users. Each request corresponds to an entry in the log files. The detail of data preprocessing, which transformed the raw log data into the data that can be used for learning association rules, was described in [19]. The resulting session file used in our experiment was derived from the 10-minute time-out session identification method. The total number of sessions (transactions) in the dataset is 30,586 and the total number of objects[6] (items) is 38,679.

The top 16 positive and negative transitional patterns in Livelink dataset are shown in Table VII and Table VIII, respectively. As we can see from the first row of Table VII, the object $\{L15000\}$ is visited most frequently after the milestone 44.17% and its frequency increases about 5 times. This shows that users are very interested in the new information in $\{L15000\}$ that are updated

[5]Livelink is a web-based product of Open Text Corporation.

[6]An object could be a document (such as a PDF file), a project description, a task description, a news group message, a picture and so on [19].

TABLE VII

TOP 16 POSITIVE TRANSITIONAL PATTERNS IN LIVELINK DATASET

rank	*Transitional Pattern* P	$sup_{-}^{\mathcal{M}}(P)$ (‰)	$sup_{+}^{\mathcal{M}}(P)$ (‰)	$\langle\ \xi^{\mathcal{M}}(P), tran^{\mathcal{M}}(P)\ \rangle$ (%)	$sup(P)$ (‰)	*support* rank
1	$\{L15000\}$	5.03	25.12	$\langle$ 44.17, +79.96 $\rangle$	16.25	25
2	$\{L1375\}$	5.04	22.72	$\langle$ 62.87, +77.79 $\rangle$	11.61	35
3	$\{L8106\}$	5.03	15.6	$\langle$ 71.49, +67.75 $\rangle$	8.04	65
4	$\{L544\}$	5.05	15.27	$\langle$ 56.96, +66.92 $\rangle$	9.45	49
5	$\{L273\}$	5.58	16.26	$\langle$ 57.97, +65.65 $\rangle$	10.07	41
6	$\{L1381\}$	5.04	14.51	$\langle$ 72.05, +65.28 $\rangle$	7.68	68
7	$\{L1509\}$	5.03	13.92	$\langle$ 45.5, +63.86 $\rangle$	9.87	44
8	$\{L545\}$	5.02	13.8	$\langle$ 57.36, +63.65 $\rangle$	8.76	56
9	$\{L544, L545\}$	5.02	13.65	$\langle$ 57.37, +63.26 $\rangle$	8.7	57
10	$\{L135\}$	14.88	39.39	$\langle$ 74.94, +62.23 $\rangle$	21.02	14
11	$\{L135, L136\}$	12.96	33.92	$\langle$ 74.94, +61.81 $\rangle$	18.21	18
12	$\{L136\}$	13.22	34.32	$\langle$ 74.94, +61.48 $\rangle$	18.51	17
13	$\{L109\}$	11.39	28.47	$\langle$ 43.05, +59.99 $\rangle$	21.12	13
14	$\{L1858\}$	6.75	15.37	$\langle$ 73.63, +56.09 $\rangle$	9.02	55
15	$\{L2155\}$	5.04	11.28	$\langle$ 73.34, +55.35 $\rangle$	6.7	81
16	$\{L1859\}$	5.39	11.83	$\langle$ 49.7, +54.4 $\rangle$	8.63	58

after the specific time. Therefore, object $\{L15000\}$ should be upgraded to a higher level so that it can be more easily accessed by the users.

On the contrary, the frequency of the first negative transitional pattern decreased significantly from 50.31% to 7.24% after the milestone 40.42%. It is very obvious that the information is out-of-date or the users are not interested in it any more. Thus, this object should be moved to a corresponding lower level in order to give room to other important objects, such as $\{L15000\}$.

Object $\{L15000\}$ is also in the sixth negative transitional pattern ($\{L15000, L15001\}$) and is frequently visited together with $\{L15001\}$ by the users before the milestone 46.81%. However, after that time, the frequencies of the fifth ($L15001$) and sixth negative transitional pattern decrease significantly, which means that most of the users who visit $\{L15000\}$ do not visit $\{L15001\}$ at the same time. Therefore, these two objects should be treated differently.

On the other hand, objects $\{L135\}$ and $\{L136\}$ (see the eleventh positive transitional pattern) should be in the same category and have links for the user to access from one to the other

TABLE VIII

TOP 16 NEGATIVE TRANSITIONAL PATTERNS IN LIVELINK DATASET

rank	*Transitional Pattern* P	$sup^{\mathcal{N}}_{-}(P)$ (‰)	$sup^{\mathcal{N}}_{+}(P)$ (‰)	$\langle\ \xi^{\mathcal{N}}(P), tran^{\mathcal{N}}(P)\ \rangle$ (%)	$sup(P)$ (‰)	*support* rank
1	$\{L355\}$	50.31	7.24	⟨ 40.42, -85.6 ⟩	24.65	9
2	$\{L384\}$	26.56	5.01	⟨ 52.32, -81.15 ⟩	16.28	24
3	$\{L11034\}$	18.6	5.03	⟨ 32.35, -72.97 ⟩	9.42	50
4	$\{L434\}$	33.81	9.76	⟨ 59.47, -71.14 ⟩	24.06	10
5	$\{L15001\}$	17.03	5.04	⟨ 46.84, -70.39 ⟩	10.66	38
6	$\{L15000, L15001\}$	16.62	5.04	⟨ 46.81, -69.68 ⟩	10.46	40
7	$\{L1735\}$	22	7.75	⟨ 60.78, -64.76 ⟩	16.41	22
8	$\{L396\}$	14.09	5.07	⟨ 52.91, -64.03 ⟩	9.84	45
9	$\{L225, L396\}$	13.54	5.07	⟨ 52.9, -62.56 ⟩	9.55	48
10	$\{L1322\}$	15.69	5.96	⟨ 41.26, -62.03 ⟩	9.97	43
11	$\{L397\}$	16.78	6.92	⟨ 60.78, -58.78 ⟩	12.91	31
12	$\{L225\}$	87.67	36.8	⟨ 61.08, -58.03 ⟩	67.87	3
13	$\{L87\}$	19.54	8.23	⟨ 31.29, -57.88 ⟩	11.77	34
14	$\{L225, L1322\}$	11.73	5.01	⟨ 41.26, -57.28 ⟩	7.78	67
15	$\{L225, L226\}$	67	30.15	⟨ 60.75, -55 ⟩	52.54	5
16	{L226}	68.24	31.32	⟨ 60.75, -54.1 ⟩	53.75	4

more easily because $\{L135\}$, $\{L136\}$ and $\{L135, L136\}$ are all positive transitional patterns with similar supports and signifincant milestones.

C. Evaluation on scalability

To study the efficiency and scalability of the proposed *TP-mine* algorithm, another set of experiments is conducted on both Retail and Livelink datasets. For each dataset, we generate a number of subsets with increasing numbers of transactions. On each subset, we run the *TP-mine* algorithm with different support thresholds between 0.5% and 2.5%. All the experiments are performed on a double-processor server, which has 2 Intel Xeon 2.4G CPU and 2G main memory, running on Linux with kernel version 2.6.

Figure 2 illustrates the execution time of the second phase of the *TP-mine* algorithm (i.e., excluding the time for generating frequent patterns) on the differently-sized subsets of the Retail data set for different support threshold values. Figure 3 shows that for the Livelink data set.

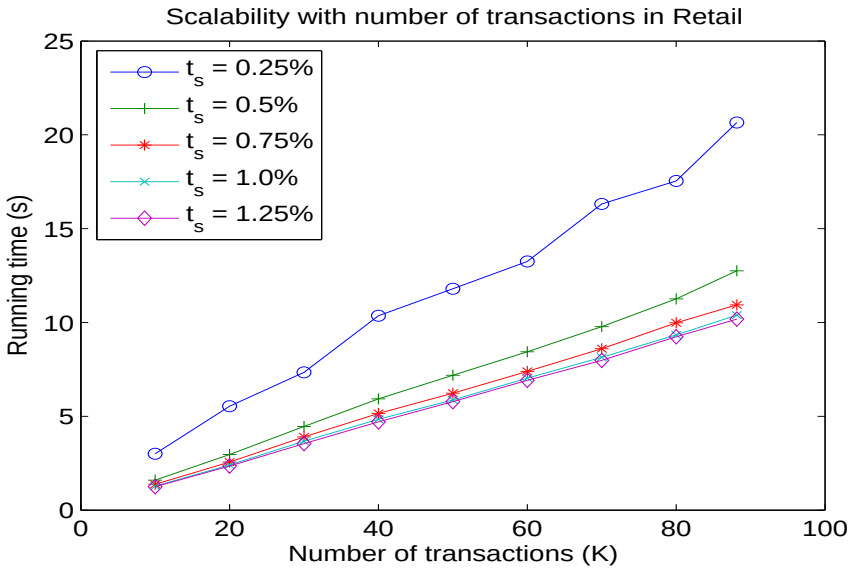

Fig. 2. Scalability on Retail dataset

As we can see from these figures, the second phase of our algorithm has linear scalability against the number of transactions in the data set. We use *FP-growth* to mine frequent patterns in the first phase, which has been shown to be linearly scalable with the number of transactions [16].

D. Comparison to Our Earlier Work on Transitional Patterns

We first introduced the concept of transitional patterns and an algorithm for mining transitional patterns and their significant milestones in [39]. In that algorithm, for each frequent itemset we calculated two supports of the pattern and the transitional ratio (if the two supports satisfy the minimum support threshold) at each *time point* that corresponds to a time stamp in the transaction database, while in the new TP-mine algorithm presented in this paper these values are calculated at each *milestone* that corresponds to the time point where the itemset occurs. As a result, in [39] a transitional pattern was defined as a frequent pattern whose transitional ratio satisfies the transitional pattern threshold at at least one of the time points (i.e., time stamps). While in this paper, a frequent pattern is a transitional pattern only if its transitional ratio passes the threshold at at least one of the time points where the pattern occurs. Similarly, the significant milestones of a transitional pattern defined in this paper only occur at the time points where the pattern

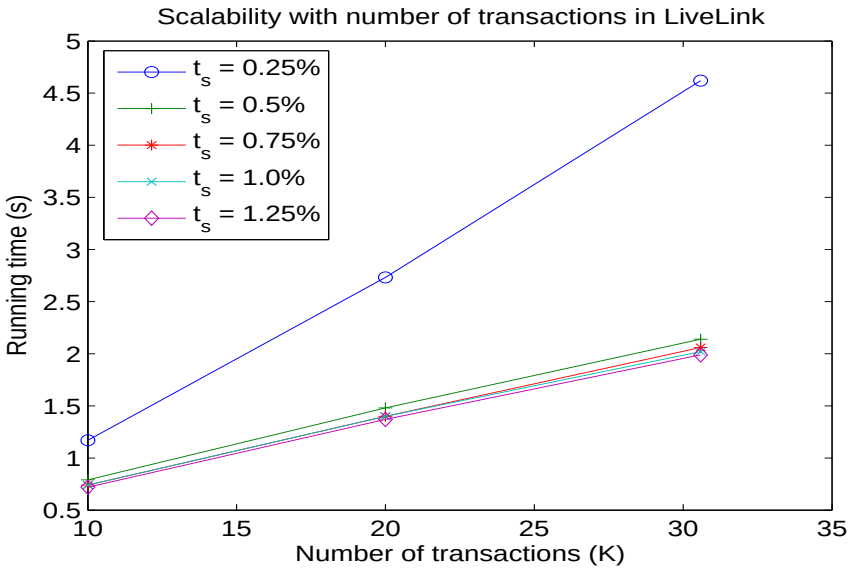

Fig. 3. Scalability on Livelink dataset

occurs, while in [39] a significant milestone can be at a time point where the pattern does not occur. The benefits of such a change are as follows. By only checking the time points where the pattern occurs, the efficiency of transitional pattern mining is improved. In addition, according to [32] that discusses methods for detecting frequency change points in an event sequence, only the time points where an event occurs can be the optimal change points that maximize the likelihood of the event sequence when piecewise constant functions are used to model the density of the event occurrences. Furthermore, the event occurrence time points are usually more interesting when monitoring the event changes. Thus, although the transitional ratio of a pattern may not always peek at the pattern occurrence time points, focusing on the occurrence time points can lead to faster, more interesting and potentially optimal solutions.

To show the speed-up of the new transitional pattern mining algorithm, we compared the new TP-mine algorithm presented in this paper to the one presented in [39] in terms of run-time. Figures 4 and 5 show the comparison on the Retail and Livelink data sets respectively. We can see that the new TP-mine algorithm is faster than the old TP-mine algorithm. The lower the support threshold, the more significant the speed-up is. Since the support threshold should usually be set to a low value for large real data sets, such a speed-up is desirable for real applications.

We also compared the top ten positive/negative transitional patterns generated by the old TP-

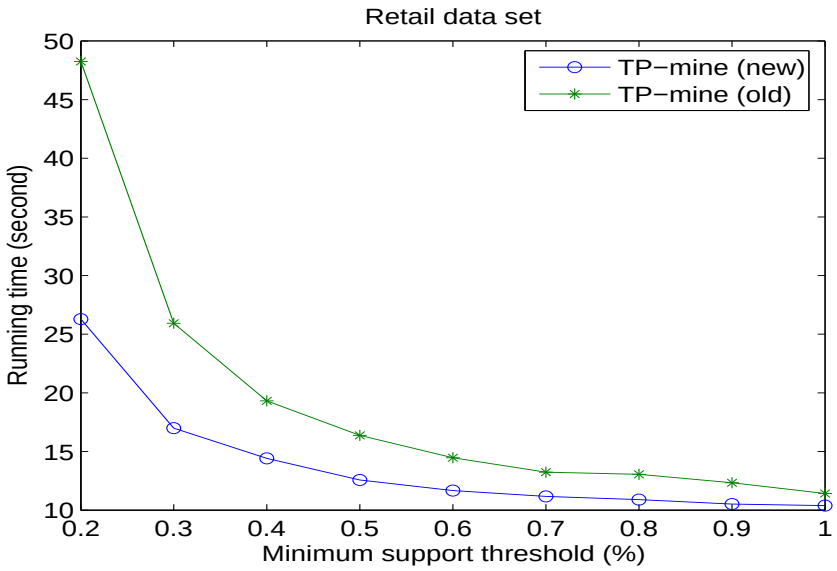

Fig. 4. Run-time comparion on the Retail data set

mine algorithm[7] and the ones generated by the new TP-mine algorithm on both the Retail and Livelink data sets. The transitional patterns on each list are ranked according to the absolute value of the transitional ratio at the significant milestone. We found that for the Retail data set, the lists of top ten positive transitional patterns discovered by the two algorithms are the same, i.e., they contain the same patterns in the same order. Looking into their significant milestones and the transitional ratios at these milestones, we found that the significant milestone (and its corresponding transitional ratio) of a pattern identified by one algorithm is either the same or very close to the one discovered by the other algorithm. The biggest absolute difference in the significant milestone is 0.37% and the biggest absolute difference in the highest transitional ratio is 0.19%. Comparing the lists of top ten negative transitional patterns discovered by the two algorithms from the Retail data set, the similarity is even stronger. Not only are the two lists the same, the significant milestones and their corresponding transitional ratios discovered by the two algorithms are exactly the same for 9 of the 10 patterns. Only for one pattern, a minor difference exists. The same observation holds for the two lists of top ten negative transitional patterns discovered by the two algorithms on the Livelink data set. For the two

[7]The top ten results from the *old* TP-mine algorithm were listed in [39]. Here we only describe the differences between the results from the two algorithms.

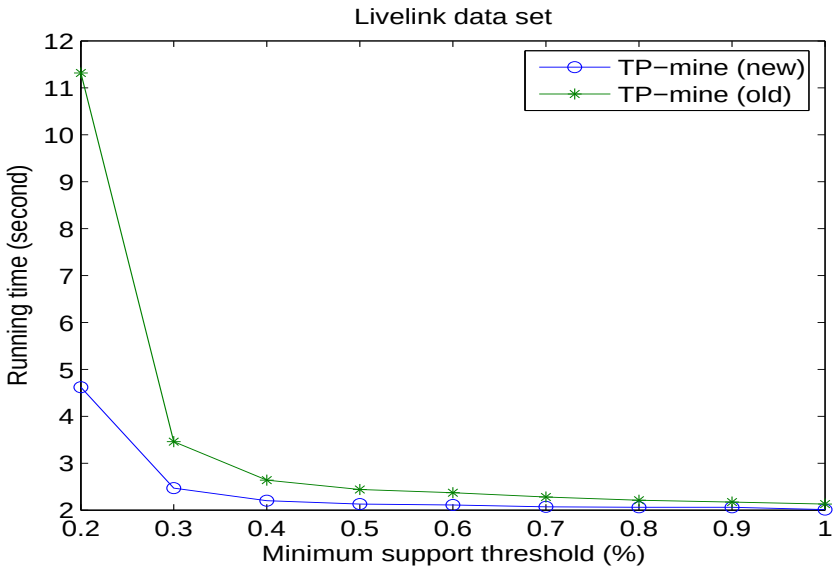

Fig. 5. Run-time comparison on Livelink data set

lists of top ten positive transitional patterns generated by the two algorithms on the Livelink data set, 9 patterns are common, 7 of which have the exactly same significant milestones and corresponding transitional ratios and 2 of them have minor differences in these two values. Only on one pattern which appeared on the top ten list generated by the old algorithms but does not appear on the top ten list by the new algorithm, we see significant difference in the significant milestone and its corresponding transitional ratio. Based on the above result we can say that in almost all the cases, the transitional ratio reaches the maximum or minimum values at or near a time point where the pattern occurs. Thus, the new algorithm speeds up the old one without significantly losing information in terms of finding the maximum or minimum transitional ratios and significant milestones of a transitional pattern.

VII. Related Work

In this section, we discuss existing work related to the transitional pattern mining framework proposed in this paper.

A. Emerging patterns

Emerging patterns proposed in [12] are defined as itemsets whose support increase significantly from one dataset to another. There are two major differences between transitional patterns and

emerging patterns. First, emerging patterns are used to capture the significant difference between two datasets. When applied to time-stamped datasets, emerging patterns are used to find contrasts between two datasets with different time periods, which is separated by a unchangeable time point. Theoretically, emerging patterns can be considered as positive transitional patterns with the time point set to a constant value. As we can see from the above experimental results, the significant milestones of transitional patterns can be at different places in one dataset. Thus, at a specific time point, the transitional ratio of a pattern might not reach its greatest value or even close to 0. For example, the transitional ratio of pattern P_4P_6 at milestone 50% in TDB is 0 (see Table III), and the transitional ratio of pattern L_{87} in the Livelink dataset is close to 0 at about 60%. If the constant time point is set to 50% or 60% in these two datasets respectively, these two interesting transitional patterns cannot be identified. Second, emerging patterns are itemsets whose growth rates are larger than a given threshold. The growth rate of a pattern X with respect to datasets D_1 and D_2 is defined as $GrowthRate(X) = \frac{sup(X,D_2)}{sup(X,D_1)}$. The value of the growth rate ranges from 0 to ∞, while the transitional ratio used in our method is a normalized measure that ranges between -1 and 1. In addition, the value of the transitional ratio is symmetric in the sense that a pattern whose support increases, say, 10 times with respect to a milestone has the same absolute value of the transitional ratio with the pattern whose support decreases 10 time. This feature makes it convenient for us to define positive and negative transitional patterns.

B. Contrast sets

Bay and Pazzani [8] introduced the problem of detecting differences across several contrasting groups as that of finding all contrast-sets, which are conjunctions of attribute-value pairs, that have meaningfully different support levels across the contrasting groups. This allows users to answer queries of the form, “How are History and Computer Science students different?” or “What has changed from 1993 through 1998?”. They proposed the STUCCO algorithm [8], which is based on Bayardo’s Max-Miner [31] rule discovery algorithm. In the level-wise search for contrast sets, formed of conjunctions of attribute-value pairs of length i, the interestingness of the conjunct is estimated by its statistical significance, assessed using a χ^2 test with a Bonferroni correction. In their application, they discovered trends in student admissions to UCI in the years from 1993 to 1998 by analyzing the frequency differences of a pattern across the years. Different from our approach, their approach focused on finding frequency differences of a pattern (which is

a conjunction of attribute-value pairs) between two or more contrasting groups of objects, where time may or may not be the criterion used to assign the objects into different groups. In this paper we focus on discovering patterns (which can be itemsets or conjunctions of attribute-value pairs) whose frequency dramatically changes over the period of time in a database.

C. Temporal association rules

Since it was formulated over a decade ago, the problem of association rule mining has been extended in several ways, among which is the discovery of temporal association rules. A temporal association rule is an association rule that holds during specific time intervals. There are several kinds of meaningful temporal association rules, including cyclic association rules (i.e., the association rules that occur periodically over time) [25], calendar-based temporal association rules (i.e., the association rules w.r.t. precise or fuzzy match for a user-given calendar schema) [22], and temporal association rules over items' lifespan (i.e., the period between the first and the last time the item appears in transactions of a database) [6], etc. Compared to these temporal association rule mining techniques, our proposed research is different in the following critical aspects. First, temporal association rules are based on user-defined time intervals, such as months, years, or other calendar-based constrains, while the transitional pattern mining technique automatically finds significant milestones of the patterns, which are unknown before the mining process. As we can see from the experimental results, significant milestones of transitional patterns are distributed throughout a wide range in the databases, and they can hardly fit into a specific time interval. Second, very strong rules tend to be strong in almost all time intervals. Thus, they are usually considered to be valid temporal association rules. But most of them are not interesting because they can be easily identified by the users with common sense. Transitional patterns, on the contrary, usually do not have very high frequency, and can be easily ignored by the users in the traditional pattern mining model, which has been demonstrated in our experimental results.

D. Sequential Patterns and Frequent Episodes

Many previous studies also consider time stamps in the database when mining frequent patterns. Representative work includes mining sequential patterns [5], [27], [30] and mining frequent episodes [24]. A sequential pattern, defined first in [5], is a sequence of elements whose occurrence frequency in a set of sequences (called a sequence database) is no less than

a minimal support threshold. Early sequential pattern mining algorithms (e.g., in [5], [30]) are based on a level-wise candidate generation and testing process, in which length-k candidates are generated from the frequent sequences of length $k-1$ and then tested by scanning the database to compute the frequency of each candidate. Some later algorithms improve the efficiency of sequential pattern mining by, e.g., using a recursive divide-and-conquer procedure that generates the complete set of frequent sequences without candidate generation and testing [27].

In [24], a framework for discovering frequent episodes in sequential data was proposed. An episode is a collection of events that occur relatively close to each other in a given partial order. An episode can be *serial*, in which events occur in a sequence, or *parallel*, in which no constraints are posed on the relative order of the events. In [24], an algorithm was proposed to find all the frequent episodes in an event sequence, which satisfy a user-specified support threshold. The paper also presented an algorithm for producing rules that describe the associations between the discovered frequent episodes.

The above algorithms make use of the time information in the database to find frequent sequential relationships between events or itemsets. In this paper, we focus on finding the events or itemsets whose own frequency changes significantly over the time period of the database without considering the sequential relationships between different events or itemsets.

E. Change detection in event sequences, time series and data streams

The problem of finding the significant milestones of a transitional pattern is related to the problem of finding the optimal k-partition of an event sequence discussed in [32]. An event sequence is a list of events ordered according to their occurrence times. It is often useful to detect changes in the frequency or density of event occurrences. In [32], approaches to detecting optimal change points in a sequence of events of a single type were proposed and compared. Their approaches, based on dynamic programming or Markov chain Monte Carlo methods, partition an event sequence into k subsequences by finding $k-1$ change points that maximize the Poisson likelihood of the data, and then compute a piecewise constant function to model the intensity of the event, which expresses the instantaneous probability of occurrence of an event as a function of time. Our problem of finding the significant milestones of a transitional pattern can be considered as that of partitioning a transaction database (i.e., a sequence of transactions) into k partitions where $k = 2$. The major differences between our approach and

the approaches in [32] are as follows. First, we use the transitional ratio defined in Section 3 to evaluate time points and identify the ones whose frequency changes the most significantly based on the transitional ratio, while the approaches in [32] find the change points by dynamic programming or stochastic simulation that optimizes the likelihood of the data. Second, we focus on analyzing frequent patterns and finding all the transitional patterns and their significant milestones from a transaction database. Since each pattern (represented by an itemset) in our approach can be considered as an event type in the problem setting in [32], our approach deals with multiple event types. In comparison, the approaches in [32] focus on finding $k-1$ change points in an event sequence that consists of events of a single type, which may or may not be frequent.

Methods have also been proposed for change point detection in a time series in both statistical and data mining literature. Standard methods include the ones in [14], [17], [18], [35]. These methods worked under the assumptions that the number of change points is known apriori and that a stationary known model can be used to fit the subsequence between successive change points. In [13], these assumptions are removed and change points are found in a hierarchical way by repeatedly splitting the time series that maximizes the statistical likelihood of the change points. The splitting process is stopped when the likelihood becomes stable or starts to increase according to a user-defined stability threshold. Our method differs from these methods in that we are not dealing with time series data although the data we deal with may be converted into a set of time series data (i.e., one time series for one frequent itemset) if a good frequency function can be found for each frequent itemset. In addition, we focus on finding the significant milestones for all the transitional patterns, which correspond to finding the most significant change points in the multiple sequences instead of a single sequence.

Change detection is also an important issue in data stream mining. A data stream is a continuous flow of data often generated at a high speed in a dynamic, time-changing environment. It is often required that a data stream be quickly analyzed in an online fashion with only one pass of data. A framework for diagnosing frequency changes in the evolution of fast data streams was presented in [2], in which velocity density estimation is used to create both temporal velocity profiles and spatial velocity profiles at periodic instants in time. The velocity density estimates the rate at which the changes in the data density are occurring at each spatial location based on some user-defined temporal window. Kernel density estimation [33] is used in the definition

of the velocity density. In contrast to the approach we present, the framework was applied to understand changes in multi-dimensional data streams using differential kernel density estimation functions with various window sizes. In evolving data streams the same data spaces are used at different points in time while the data items change. Another approach to analyzing the distribution changes in data streams was proposed in [20]. The approach was based on a two-window paradigm, in which the data in some reference window is compared to the data in a current window which slides forward with each incoming data. The method passes the data once and provides proven guarantees on the statistical significance of detected change. Different from data stream mining, we focus on detecting changes on historical and static data without real time constraints.

F. Histogram

Our work can also be compared with the histogram technique used in statistics. Although a histogram can illustrate the frequency distribution of a variable over a time period, it is only a graphic tool for human to look at the distribution of a variable. When applying to analyzing the frequency distributions of frequent itemsets in a transaction database, we would need to draw a histogram for each of the frequent pattern. When the number of frequent patterns is large (which is usually the case for real applications), the amount of work involved is huge and the user can be easily overwhelmed by too many graphs.

In comparison, with the transitional pattern mining technique proposed in this paper, patterns with interesting distributions can be identified easily. If the user would like to see the distribution of such patterns, he/she can use histograms to look at them in details. But without first identifying such patterns, the user may not have an idea as to which patterns should be looked at.

In addition, when applying histograms to the transaction database, the user needs to discretize the time variable into intervals. Without knowing how the patterns are evolving, it is not an easy job to choose a good discretization. With our technique, we do not need to split the time period into intervals.

VIII. Conclusions and Future work

A limitation of existing frequent itemset mining framework is that it does not consider the time stamps associated with the transactions in the database. As a result, dynamic behavior of

frequent itemsets cannot be discovered. In this paper, we introduced a novel type of patterns, positive and negative transitional patterns, to represent frequent patterns whose frequency of occurrences changes significantly at some points of time in a transaction database. We also defined the concepts of significant frequency-ascending milestones and significant frequency-descending milestones to capture the time points at which the frequency of patterns changes most significantly. To discover transitional patterns, we proposed the *TP-mine* algorithm to mine the set of positive and negative transitional patterns with respect to a pattern support threshold and a transitional pattern threshold. Our algorithms takes one database scan after mining frequent patterns to find the transitional patterns and their significant milestones. Our experimental results showed that the proposed algorithm is highly scalable.

In our experimental study, we demonstrated the usefulness of transitional patterns in two real-world domains and showed that what is revealed by the transitional patterns and their significant milestones would not be found by the standard frequent pattern mining framework. As there are concerns about the practical usefulness of data mining techniques, we hope that the research presented in this paper brings a promising avenue to look at the data from a new angle, which allows us to find new, surprising, useful and actionable patterns from data.

In the future, we would like to extend this work in the following directions. First, we would like to investigate whether other designs of the transitional ratio would lead to better discovery of transitional patterns and their milestones. Second, we would like to identify other types of patterns (such as periodical patterns) by analyzing the discovered milestones. Moreover, finding sequential transitional patterns is another interesting topic that we would like work on in the future.

IX. Acknowledgments

This research is supported by Natural Sciences of Engineering Research Council of Canada (NSERC). We would also like to thank the anonymous reviewers for their constructive comments.

References

[1] Ramesh C. Agarwal, Charu C. Aggarwal, and V. V. V. Prasad. Depth first generation of long patterns. In *KDD'00: Proc. of the 6th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 108–118. ACM, 2000.

[2] Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *SIGMOD '03: Proc. of the 2003 ACM SIGMOD Int. Conf. on Management of data*, pages 575–586. ACM, 2003.

[3] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *SIGMOD '93: Proc. of the 1993 ACM SIGMOD Int. Conf. on Management of data*, pages 207–216. ACM, 1993.

[4] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases,*, pages 487–499. Morgan Kaufmann, 1994.

[5] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In Philip S. Yu and Arbee S. P. Chen, editors, *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.

[6] Juan M. Ale and Gustavo H. Rossi. An approach to discovering temporal association rules. In *SAC '00: Proc. of the 2000 ACM symposium on Applied computing*, pages 294–300. ACM, 2000.

[7] James Bailey, Thomas Manoukian, and Kotagiri Ramamohanarao. Fast algorithms for mining emerging patterns. In *PKDD '02: Proc. of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 39–50, London, UK, 2002. Springer-Verlag.

[8] Stephen D. Bay and Michael J. Pazzani. Detecting change in categorical data: mining contrast sets. In *KDD '99: Proc. of the fifth ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 302–306. ACM, 1999.

[9] Tom Brijs, Gilbert Swinnen, Koen Vanhoof, and Geert Wets. Using association rules for product assortment decisions: a case study. In *Proc. of the 5th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 254–260. ACM, 1999.

[10] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: generalizing association rules to correlations. *SIGMOD Rec.*, 26(2):265–276, 1997.

[11] Doug Burdick, Manuel Calimlim, Jason Flannick, Johannes Gehrke, and Tomi Yiu. Mafia: A maximal frequent itemset algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1490–1504, 2005.

[12] Guo-Zhu Dong and Jin-Yan Li. Efficient mining of emerging patterns: discovering trends and differences. In *KDD '99: Proc. of the fifth ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 43–52. ACM, 1999.

[13] Valery Guralnik and Jaideep Srivastava. Event detection from time series data. In *KDD '99: Proc. of the fifth ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 33–42. ACM, 1999.

[14] S.B. Guthery. Partition regression. *Journal of the American Statistical Association*, 69(348):945–947, 1974.

[15] Jia-Wei Han, Jian Pei, and Xi-Feng Yan. From sequential pattern mining to structured pattern mining: a pattern-growth approach. *Journal of Computer Science and Technology*, 19(3):257–279, 2004.

[16] Jia-Wei Han, Jian Pei, Yi-Wen Yin, and Run-Ying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.

[17] D.M. Hawkins and D.F. Merriam. Optimal zonation of digitized sequential data. *Mathematical Geology*, 5(4):389–395, 1973.

[18] Douglas M. Hawkins. Point estimation of the parameters of piecewise regression models. *The Journal of the Royal Statistical Society Series C (Applied Statistics)*, 25(1):51–57, 1976.

[19] Xiangji Huang, Aijun An, Nick Cercone, and Gary Promhouse. Discovery of interesting association rules from livelink web log data. In *ICDM '02: Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02)*, page 763, Washington, DC, USA, 2002. IEEE Computer Society.

[20] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *VLDB '04: Proc. of the 30th Int. Conf. on Very large data bases*, pages 180–191. VLDB Endowment, 2004.

[21] Jin-Yan Li, Kotagiri Ramamohanarao, and Guo-Zhu Dong. Emerging patterns and classification. In *ASIAN '00: Proc.*

of the 6th Asian Computing Science Conference on Advances in Computing Science, pages 15–32, London, UK, 2000. Springer-Verlag.

[22] Ying-Jiu Li, Peng Ning, X. Sean Wang, and Sushil Jajodia. Discovering calendar-based temporal association rules. In *TIME '01: Proc. of the 8th Int. Symposium on Temporal Representation and Reasoning (TIME'01)*, page 111, Washington, DC, USA, 2001. IEEE Computer Society.

[23] Bing Liu, Wynne Hsu, and Yi-Ming Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[24] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1:259–289, 1997.

[25] Banu Özden, Sridhar Ramaswamy, and Abraham Silberschatz. Cyclic association rules. In *ICDE '98: Proc. of the Fourteenth Int. Conf. on Data Engineering*, pages 412–421, Washington, DC, USA, 1998. IEEE Computer Society.

[26] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT '99: Proceeding of the 7th International Conference on Database Theory*, pages 398–416, London, UK, 1999. Springer-Verlag.

[27] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 2004.

[28] Jian Pei, Jiawei Han, and Run-Ying Mao. CLOSET: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30, 2000.

[29] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jian-Yong Wang, Helen Pinto, Qi-Ming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Mining sequential patterns by pattern-growth: The PrefixSpan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1424–1440, November 2004.

[30] Rakesh Agrawal Ramakrishnan Srikant. Mining sequential patterns: Generalizations and performance improvements. In *the 5th Int. Conf. Extending Database Technology (EDBT'96)*, 1996.

[31] Jr. Roberto J. Bayardo. Efficiently mining long patterns from databases. In *SIGMOD '98: Proc. of the 1998 ACM SIGMOD Int. Conf. on Management of data*, pages 85–93. ACM, 1998.

[32] Marko Salmenkivi and Heikki Mannila. Using markov chain monte carlo and dynamic programming for event sequence data. *Knowledge and Information Systems*, 7(3):267–288, 2005.

[33] B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[34] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. *Future Generation Computer Systems*, 13(2–3):161–180, 1997.

[35] N. Sugiura and T. Ogden. Testing change-point with linear trend. *Communications in Statistics B: Simulation and Compution*, 23:287–322, 1994.

[36] Pang-Ning Tan and Vipin Kumar. Mining indirect associations in web data. In *WEBKDD '01: Revised Papers from the Third International Workshop on Mining Web Log Data Across All Customers Touch Points*, pages 145–166, London, UK, 2002. Springer-Verlag.

[37] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Indirect association: Mining higher order dependencies in data. In *PKDD '00: Proc. of the 4th European Conf. on Principles of Data Mining and Knowledge Discovery*, pages 632–637, London, UK, 2000. Springer-Verlag.

[38] Qian Wan and Ai-Jun An. An efficient approach to mining indirect associations. *Journal of Intelligent Information Systems*, 27(2):135–158, 2006.

[39] Qian Wan and Aijun An. Transitional patterns and their significant milestones. In *Proc. of the 7th IEEE Int. Conf. on Data Mining*, Omaha, NE, USA, 2007.

[40] Mohammed J. Zaki and Karam Gouda. Fast vertical mining using diffsets. In *KDD '03: Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 326–335. ACM, 2003.

[41] Mohammed J. Zaki and Ching-Jui Hsiao. CHARM: An efficient algorithm for closed itemset mining. In *SIAM '02: Proc. of the 2nd SIAM Int. Conf. on Data Mining*, pages 34–43, 2002.

Qian Wan is a Ph.D. candidate in the Department of Computer Science and Engineering at York University, Toronto, Canada. He is expected to obtain his Ph.D. degree in Summer 2009. He received an M.Sc. degree in Computer Science from York University in 2003 and a B.Sc. degree in Computer Science from Wuhan University in China. His research interests are in the area of knowledge discovery and data mining, focusing on discovering novel and useful patterns from transaction databases.

Aijun An is an Associate Professor in the Department of Computer Science and Engineering at York University, Toronto, Canada. She received her Ph.D. degree in Computer Science from the University of Regina in 1997. She held research positions at the University of Waterloo from 1997 to 2001. She joined York University in 2001. She has published more than 80 papers in refereed journals and conference proceedings. Her research interests include data mining, information retrieval and biomedical informatics.