Progressive Frequent and Infrequent Patterns and Their Significant Milestones

Qian Wan

Aijun An

Technical Report CSE-2007-03

May 22, 2007

Department of Computer Science and Engineering
4700 Keele Street North York, Ontario M3J 1P3 Canada

Progressive Frequent and Infrequent Patterns and Their Significant Milestones

Qian Wan
Department of Computer Science and Engineering
York University, Toronto, Ontario, Canada
qwan@cs.yorku.ca

Aijun An
Department of Computer Science and Engineering
York University, Toronto, Ontario, Canada
aan@cs.yorku.ca

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 progressive patterns, to capture the dynamic behavior of frequent patterns in a transaction database. Progressive patterns include both progressive frequent and progressive infrequent patterns. Their frequencies increase/decrease dramatically at some time point of a transaction database. We introduce the concept of significant milestones for a progressive 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 complete set of progressive patterns along with their significant milestones. Our experimental studies on real-world databases illustrate that mining progressive frequent and infrequent patterns is highly promising as a practical and useful approach for discovering novel and interesting knowledge from large databases.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*Data Mining*

General Terms

Data mining, Frequent patterns, Progressive patterns, Significant milestone

1. INTRODUCTION

A transaction database usually consists of a set of time-stamped transactions. Mining frequent itemsets or patterns from a transaction database is one of the fundamental and essential operations in many data mining applications, such as discovering association rules, strong rules, correlations, multidimensional patterns, and many other important discovery tasks. The problem of mining frequent itemsets is formulated as finding all the itemsets from a transaction database that satisfy a user specified support threshold. Since it was first introduced by Agrawal et al. [2] in 1993, the problem of frequent itemset mining has been studied extensively by many researchers. As a result, a large number of algorithms have been developed in order to efficiently solve the problem, including the most well-known *Aproiri* [3] *FP-growth* [7], and Eclat [14].

In practice, the number of frequent patterns generated from a data set can often become excessively large, and most of them are useless or simply redundant. Thus, there has been recent interest in discovering a class of new patterns, including maximal frequent itemsets [1, 4, 5], closed frequent itemset [10, 11, 15] , emerging patterns [6, 9], and indirect associations [12, 13].

Despite the abundance of previous work, most of the existing frequent pattern mining algorithms (such as *Apriori* [2] and *FP-growth* [7]) do not consider the time stamps associated with the transactions. Therefore, the dynamic behavior of the discovered frequent patterns cannot be revealed. 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 progressive patterns, to represent patterns whose frequency dramatically changes over time. Progressive patterns include both progressive frequent and progressive infrequent patterns (to be defined in Section 3.1). The frequency of a progressive frequent pattern increases dramatically at some time point of a transaction database, while that of a progressive infrequent pattern decreases dramatically at some point of time. We illustrate progressive patterns using an example as follows.

Consider an example database TDB as shown in Table 1, 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

Table 1: An example transaction database *TDB*

TID	List of itemIDs	Time stamp	Time point
001	P_1, P_2, P_3, P_5	Nov. 2005	6.25%
002	P_1, P_2	Dec. 2005	12.5%
003	P_1, P_2, P_3, P_8	Jan. 2006	18.75%
004	P_1, P_2, P_5	Feb. 2006	25%
005	P_1, P_2, P_4	Mar. 2006	31.25%
006	P_1, P_2, P_4, P_5, P_6	Apr. 2006	37.5%
007	P_1, P_2, P_3, P_4, P_6	May. 2006	43.75%
008	P_1, P_4, P_6	Jun. 2006	50%
009	P_4, P_5, P_6	Jul. 2006	56.25%
010	P_1, P_2, P_3, P_4, P_5, P_6	Aug. 2006	62.5%
011	P_1, P_3, P_4, P_6	Sep. 2006	68.75%
012	P_1, P_3, P_5	Oct. 2006	75%
013	P_1, P_2, P_3, P_6, P_7	Nov. 2006	81.25%
014	P_1, P_3, P_4, P_5	Dec. 2006	87.5%
015	P_1, P_3, P_4	Jan. 2007	93.75%
016	P_1, P_2, P_3, P_5	Feb. 2007	100%

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 1. 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 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.

Progressive patterns have a wide range of potential applications. For example, in the market basket scenario, progressive 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 more often can help to identify the drug that causes the problem.

The contributions of this paper are summarized as follows.

- We propose a framework for mining a new class of patterns, called progressive patterns. The frequencies of these patterns change significantly at some time points of a transaction database.

Table 2: Summary of notations and their meaning

$\mathcal{D}$	a database of transactions
$\|\|\mathcal{D}\|\|$	number of transactions in $\mathcal{D}$
TDB	an example transaction database
$sup_{\mathcal{D}}(X)$	the support of pattern X in $\mathcal{D}$
τ	a time point in $\mathcal{D}$
$\top_{\mathcal{D}}$	a range of τ in $\mathcal{D}$
$prog_{\tau}(X)$	progressive ratio of X at τ
t_s	pattern support threshold
t_p	progressive pattern threshold

- We introduce the concept of significant milestone for each progressive pattern, which is the specific time when the frequency of the pattern increases or decreases most significantly.
- An algorithm, called *PP-mine*, is designed to mine the complete set of progressive patterns along with their significant milestones.
- We present a experimental study to verify the usefulness and effectiveness of progressive patterns. Our results illustrate that mining progressive frequent and infrequent patters is highly promising as a practical approach to discovering new and interesting knowledge for large databases.

The remaining of the paper is organized as follows. In Section 2 we review the terminologies used in frequent pattern mining. In Section 3, we introduce the concepts of progressive frequent and infrequent patters and their significant milestones. In Section 4, we present an algorithm for mining progressive patterns and their significant milestones. In Section 5, we present an experimental study to demonstrate the utility of progressive patterns in two real-world datasets. In Section 6, we compare our method with related work. In Section 7, we conclude paper and present some ideas for future work.

2. PRELIMINARIES - FREQUENT PATTERNS

In this session, we review the standard terminology for frequent pattern mining. Table 2 summarizes the notations that will be used throughout this paper and their meanings.

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ be a set of m *items.* A subset $X \subseteq \mathcal{I}$ is called an *itemset.* A k-itemset is an itemset that contains k items. Let $\mathcal{D} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n\}$ be a set of n transactions, called a *transaction database*, where each transaction $\mathcal{T}_j$ ($j \in \{1, 2, \ldots, n\}$) is a set of items such that $\mathcal{T}_j \subseteq \mathcal{I}$. Each transaction is associated with a unique identifier, called its *TID.* A transaction $\mathcal{T}_j$ contains an itemset X if and only if $X \subseteq \mathcal{T}_j$.

The count of an itemset X in $\mathcal{D}$, denoted as $count_{\mathcal{D}}(X)$, is the number of transactions in $\mathcal{D}$ containing X. An itemset X in a transaction database $\mathcal{D}$ has a *support*, denoted as $sup_{\mathcal{D}}(X)$, which is the ratio of transactions in $\mathcal{D}$ containing X. That is,

$$sup_{\mathcal{D}}(X) = \frac{count_{\mathcal{D}}(X)}{||\mathcal{D}||} \qquad (1)$$

where $||\mathcal{D}||$ is the total number of transactions in $\mathcal{D}$.

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 if $sup_{\mathcal{D}}(X) \geq min\_sup$. Accordingly, X is called an infrequent itemset or infrequent pattern if $sup_{\mathcal{D}}(X) < min\_sup$.

3. PROGRESSIVE PATTERNS

In this section, we present the concepts of progressive frequent and infrequent patterns and their significant milestones.

3.1 Progressive Frequent and Infrequent Patterns

In order to provide formal definitions of progressive patterns, we first introduce the concept of *time points*. Suppose that the transactions in a transaction database $\mathcal{D}$ are ordered according to their time stamps. A time point, denoted by τ, represents a position in the transaction database $\mathcal{D}$ that separates $\mathcal{D}$ into two disjoint parts, $\mathcal{D}_\tau^-$ and $\mathcal{D}_\tau^+$. We use $\mathcal{D}_\tau^-$ to represent the set of transactions in $\mathcal{D}$ that occur before τ, and $\mathcal{D}_\tau^+$ to represent the set of transactions in $\mathcal{D}$ that occur after τ. Thus, $\mathcal{D} = \mathcal{D}_\tau^- \cup \mathcal{D}_\tau^+$, and τ can be represented by a percentage to indicate a position in $\mathcal{D}$ as follows:

$$\tau = \frac{||\mathcal{D}_\tau^-||}{||\mathcal{D}||} \times 100\% \qquad (2)$$

For example, when $\tau = 25\%$, TDB_τ^- contains the first 4 transactions and TDB_τ^+ contains the last 12 transactions. Given a dataset, a time point corresponds to a time stamp. Thus, the number of possible time points is the number of different time stamps in the dataset, which can be equal to or less than the number of transactions in the dataset, assuming that some transactions may occur at the same time. The time points for the example dataset TDB is shown in Table 1.

It's easy to see that the value of τ is between 0% and 100%. However, in practice, τ must be in a reasonable range so that both $\mathcal{D}_\tau^-$ and $\mathcal{D}_\tau^+$ have enough data to be considered. This range, denoted as $\top_{\mathcal{D}}$, can be conveniently determined by the users using their own domain knowledge according to their own interest. For instance, in order to find interesting patterns in the example database TDB which occur during the year 2006, $\top_{TDB}$ should be set to [12.50% ... 87.50%].

Give a time point τ in $\top_{\mathcal{D}}$, the supports of a pattern X in $\mathcal{D}_\tau^-$ and $\mathcal{D}_\tau^+$ are denoted as $sup_\tau^-(X)$ and $sup_\tau^+(X)$, respectively.

$$sup_\tau^-(X) = sup_{\mathcal{D}_t^-}(X) = \frac{count_{\mathcal{D}_t^-}(X)}{||\mathcal{D}_t^-||} \qquad (3)$$

$$sup_\tau^+(X) = sup_{\mathcal{D}_t^+}(X) = \frac{count_{\mathcal{D}_t^+}(X)}{||\mathcal{D}_t^+||} \qquad (4)$$

Definition 1. The progressive ratio of pattern X at time point τ is defined as:

$$prog_\tau(X) = \frac{sup_\tau^+(X) - sup_\tau^-(X)}{MAX(sup_\tau^+(X),\ sup_\tau^-(X))} \qquad (5)$$

where X must exist in the database $\mathcal{D}$ so that the denominator cannot be zero.

Consider the relationship between $sup_\tau^-(X)$ and $sup_\tau^+(X)$, $prog_\tau(X)$ has the following three possible cases.

(1) $prog_\tau(X) = 0$, when $sup_\tau^-(X) = sup_\tau^+(X)$, which means X has exactly the same frequency before and after τ;

(2) $prog_\tau(X) > 0$, when $sup_\tau^-(X) < sup_\tau^+(X)$, which means X is more frequent after τ;

(3) $prog_\tau(X) < 0$, when $sup_\tau^-(X) > sup_\tau^+(X)$, which means X is more frequent before τ;

In this paper, we are interested in the last two cases, where pattern x might be one of interesting progressive frequent (infrequent) patterns, which are defined as follows.

Definition 2. A pattern X is a Progressive Pattern (PP) in $\mathcal{D}$, if there exists a time point τ in $\top_{\mathcal{D}}$ such that:

(a) $sup_\tau^-(X) \geq t_s$ and $sup_\tau^+(X) \geq t_s$;

(b) $|pro_\tau(X)| \geq t_p$.

where t_s and t_p are called *pattern support threshold* and *progressive pattern threshold*, respectively. Moreover, X is called a Progressive Frequent Pattern (PFP) when $pro_\tau(X)$ > 0; and X is called a Progressive Infrequent Pattern (PIP) when $pro_\tau(X) < 0$.

For example, if $t_s = 0.05$ and $t_p = 0.5$, pattern P_1P_3 in the example database TDB is a progressive frequent pattern because there exists a time point (such as 37.5% corresponding to the end of April 2006) where the progressive ratio of the pattern is greater than 0.5 and the pattern is frequent on both corresponding splits of the datasets. Similarly, P_1P_2 is a progressive infrequent pattern in TDB.

Note that a pattern can be both progressive frequent and progressive infrequent in the same transaction database if there exist two time points τ_1 and τ_2 so that conditions (a) and (b) are satisfied on both τ_1 and τ_2, $pro_{\tau_1}(X) > 0$ and $pro_{\tau_2}(X) < 0$. For example, in the example database TDB, pattern P_4P_6 is both a progressive frequent pattern and a progressive infrequent pattern because its progressive ratio at time point 37.50% is 66.67% and the one at time point 62.50% is −66.67%, and condition (a) is also satisfied on both time points.

The reason we have condition (a) for a progressive pattern is that, if we don't have this condition, any pattern that does not occur at the beginning of the transaction database has a progressive ratio equal 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 progressive 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 progressive pattern is also a frequent pattern in the database with respect to pattern

Table 3: Example patterns in *TDB* (%)

τ	$prog_{\tau}(P_1)$	$prog_{\tau}(P_1P_2)$	$prog_{\tau}(P_1P_3)$	$prog_{\tau}(P_4P_6)$
25	-8.33	-50	25	100
31.25	-9.09	-54.55	45	100
37.50	-10	-60	58.33	66.67
43.75	-11.11	-66.67	44.9	35.71
50	-12.5	-57.14	57.14	0
56.25	11.11	-44.9	66.67	-35.71
62.50	10	-58.33	60	-66.67
68.75	9.09	-45	54.55	-100
75	8.33	-25	50	-100

support threshold t_s[1]. In other words, we are only interested in frequent patterns whose frequency changes dramatically during the time period $\top_{\mathcal{D}}$ 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 progressive pattern threshold t_p should be set to a value higher than or equal to 0.5.

In order to obtain reliable values for the supports of a pattern in $\mathcal{D}^-_\tau$ and $\mathcal{D}^+_\tau$, $\mathcal{D}^-_\tau$ and $\mathcal{D}^+_\tau$ 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 positive progressive ratio (or a small negative progressive ratio) due to the fact that its support is too high in $\mathcal{D}^-_\tau$ (or in $\mathcal{D}^+_\tau$). Thus, time period $\top_{\mathcal{D}}$ should be set to a reasonable range, say, between 20% and 80%, depending on the total number of transactions and the domain of the dataset.

3.2 Significant milestones

There may be multiple time points at which a progressive pattern satisfies conditions (a) and (b) in Definition 2. People are usually interested in the time points where the frequency of a progressive pattern changes the most significantly. Below we define the concept of *significant milestones* to represent such points. The significant milestones can be classified into frequency-ascending milestones and frequency-descending milestones.

Definition 3. The significant frequency-ascending milestone of a progressive frequent pattern X within a time period $\top_{\mathcal{D}}$ is defined as a tuple, $\langle \mathcal{M}^+, prog_{\mathcal{M}^+}(X)\rangle$, where $\mathcal{M}^+$ is a time point in $\top_{\mathcal{D}}$ such that:

1. $sup^-_{\mathcal{M}^+}(X) \geq t_s$;
2. $\forall\ \tau \in \top_{\mathcal{D}}$, $prog_{\mathcal{M}^+}(X) \geq prog_{\tau}(X)$.

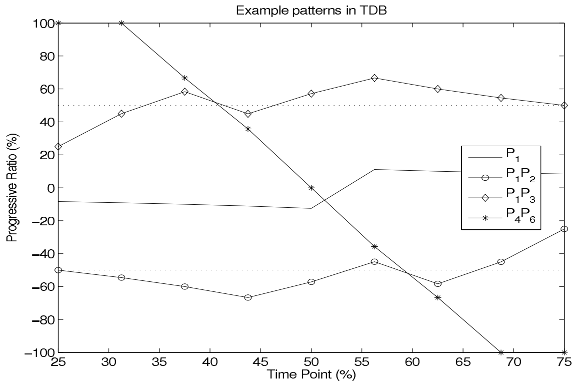

Figure 1: Progressive ratios in *TDB*

Table 3 lists the progressive ratios of four patterns for all the valid time points between 25% and 75% in the example database *TDB*. Figure 1 illustrates how the progressive ratios of these four patterns change along the time points. Assuming that the support threshold is ? and the progressive pattern threshold is 50%, P_1P_3 and P_4P_6 are progressive frequent patterns. The significant milestone for P_1P_3 is $\langle$56.25%, +66.67$\rangle$, and the significant milestone for P_4P_6 is $\langle$37.50%, +66.67$\rangle$. Note that even though the progressive ratio of P_4P_6 is 1 at time points 25% and 31.25%, they are not considered to be milestones because they do not satisfy condition 1 in Definition 3 due to the fact that P_4P_6 does not occur before time point 31.25%.

The reason for having condition 1 in Definition 3 is as follows. Progressive frequent 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 progressive ratios for the corresponding time points may be very high, but these points are not interesting because the sporadic nature of the occurrence. For example, suppose that in a dataset with 1000 transactions, a progressive frequent 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 progressive ratio at the time point corresponding to the 100th transaction is 98.23% and its progressive ratio at the time point corresponding to the 500th transaction is only 98%. But the latter point 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 time (assuming the support threshold is small enough) and we haven't had enough information to see the trend of the pattern yet. Please note that the use of this constraint does not make us miss the significant milestone in the situation where a progressive frequent pattern starts to occur very often right after the first occurrence of the pattern. In this case, the significant milestone may or may not at the place of the first occurrence, but if not, it is not far away from the first occurrence because the support of the pattern in $\mathcal{D}^-$ generally increases quickly as the time point moves forward.

[1] It is trivial to prove that if a pattern is frequent on $\mathcal{D}^-_t$ and $\mathcal{D}^+_t$, it must be frequent on $\mathcal{D}$, where $\mathcal{D} = \mathcal{D}^-_\tau \cup \mathcal{D}^+_\tau$. However, please note that if a pattern is frequent on $\mathcal{D}$ with respect to t_s, there may not exist a time point τ such that $sup^-_\tau(X) \geq t_s$ and $sup^+_\tau(X) \geq t_s$. Therefore, if we want to find all the progressive patterns in a set of frequent patterns discovered with support threshold t, the pattern support threshold t_s for progressive patterns should be set to be smaller than t. We consider this problem to be a different problem from what this paper is concerned about. The pattern support threshold t_s in Definition 2 is only for defining progressive patterns to avoid generalizing over insufficient data.

Similarly, we define the significant frequency-descending milestone for a progressive infrequent pattern as follows.

Definition 4. The significant frequency-descending milestone of a progressive infrequent pattern Y within a time period $\top_{\mathcal{D}}$ is defined as a tuple, $\langle \mathcal{M}^-, prog_{\mathcal{M}^-}(Y) \rangle$, where $\mathcal{M}^-$ is a time point in $\top_{\mathcal{D}}$ such that:

1. $sup^{+}_{\mathcal{M}^-}(Y) \geq t_s$;
2. $\forall\, \tau \in \top_{\mathcal{D}}$, $prog_{\mathcal{M}^-}(Y) \leq prog_\tau(Y)$.

To give an example, patterns P_1P_2 and P_4P_6 in Table 3 are progressive infrequent patterns. Their significant frequency-descending milestones are ⟨43.75%, -66.67⟩ and ⟨62.50%, -66.67⟩, respectively. The reason to have Condition 1 in Definition 4 is similar to that in Definition 3.

Note that a progressive pattern may have both significant frequency-ascending milestones and significant frequency-descending milestones if the pattern is both progressive frequent and progressive infrequent. Also, a progressive frequent (or infrequent) pattern may have more than one significant frequency-ascending (or frequency-descending) milestones.

4. MINING PROGRESSIVE PATTERNS AND THEIR SIGNIFICANT MILESTONES

In this section, we present an algorithm, called *PP-mine*, for mining the complete set of progressive frequent and infrequent patterns and their significant milestones with respect to a pattern support threshold and a progressive pattern threshold. The algorithm is given as follows.

Algorithm: ***PP-mine*** (***Mine*** *the complete set of* ***P****rogressive* ***P****atterns and their significant milestones*)

Input: A transaction database ($\mathcal{D}$), a time period that the user is interested ($\top_{\mathcal{D}}$), pattern support threshold (t_s) and progressive pattern threshold (t_p)

Output: The complete set of progressive frequent and infrequent patterns ($\mathcal{S}_{PFP}$ and $\mathcal{S}_{PIP}$) with their significant milestones

```
 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 transaction right before $\top_{\mathcal{D}}$ to compute the support counts of all the $n$ frequent patterns on this part of the database.
 3: $\mathcal{S}_{PFP} \leftarrow \emptyset$, $\mathcal{S}_{PIP} \leftarrow \emptyset$
 4: for all $i = 1$ to $n$ do
 5:   $MaxProg(P_i) = 0$, $MinProg(P_i) = 0$
 6:   $S_{FAM}(P_i) = \emptyset$, $S_{FDM}(P_i) = \emptyset$
 7: end for
 8: for all $\tau \in \top_{\mathcal{D}}$ do
 9:   for $i = 1$ to $n$ do
10:     if $sup^{-}_{\tau}(P_i) \geq t_s$ and $sup^{+}_{\tau}(P_i) \geq t_s$ then
11:       if $prog_\tau(P_i) \geq t_p$ then
12:         if $P_i \notin S_{PFP}$ then
13:           Add $P_i$ to $\mathcal{S}_{PFP}$
14:         end if
15:         if $Prog_\tau(P_i) > MaxProg(P_i)$ then
16:           $S_{FAM}(P_i) = \{\langle \tau, Prog_\tau(P_i) \rangle\}$
17:           $MaxProg(P_i) = Prog_\tau(P_i)$
18:         else if $Prog_\tau(P_i) = MaxProg(P_i)$ then
19:           Add $\langle \tau, Prog_\tau(P_i) \rangle$ to $S_{FAM}(P_i)$
20:         end if
21:       else if $prog_\tau(P_i) \leq -t_p$ then
22:         if $P_i \notin S_{PIP}$ then
23:           Add $P_i$ to $\mathcal{S}_{PIP}$
24:         end if
25:         if $Prog_\tau(P_i) < MinProg(P_i)$ then
26:           $S_{FDM}(P_i) = \{\langle \tau, Prog_\tau(P_i) \rangle\}$
27:           $MinProg(P_i) = Prog_\tau(P_i)$
28:         else if $Prog_\tau(P_i) = MinProg(P_i)$ then
29:           Add $\langle \tau, Prog_\tau(P_i) \rangle$ to $S_{FDM}(P_i)$
30:         end if
31:       end if
32:     end if
33:   end for
34: end for
35: return $S_{PFP}$ and $S_{FAM}(P_i)$ for each $P_i \in S_{PFP}$
36: return $S_{PIP}$ and $S_{FDM}(P_i)$ for each $P_i \in S_{PIP}$
```

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* [2] or *FP-growth* [7], with t_s as the minimum support threshold. In the second phase (starting from Step 2 to the end), the algorithm finds all the progressive patterns and their significant milestones based on the set of frequent itemsets. As mentioned before, a pattern that is frequent on both $\mathcal{D}^-_\tau$ and $\mathcal{D}^+_\tau$ with respect to support threshold t_s must be frequent on $\mathcal{D}$ with respect to the same threshold, where $\mathcal{D} = \mathcal{D}^-_\tau \cup \mathcal{D}^+_\tau$. 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 progressive 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 $\top_{\mathcal{D}}$ are collected. They are used later in computing $sup^-_\tau(P_i)$, where P_i is a frequent pattern. Step 3 initializes the set of progressive frequent patterns ($\mathcal{S}_{PFP}$) and the set of progressive infrequent patterns ($\mathcal{S}_{PIP}$) to empty. Steps 4-7 initializes the set of significant frequency-ascending milestones for each frequent pattern P_i, $S_{FAM}(P_i)$, and the set of significant frequency-descending milestones for each frequent pattern P_i, $S_{FDM}(P_i)$, to empty. It also initializes the maximal and minimal progressive ratios of P_i, denoted by $MaxProg(P_i)$ and $MinProg(P_i)$, to zero.

After the initializations, the algorithm continues to scan the database $\mathcal{D}$ to find the time points within time period $\top_{\mathcal{D}}$. At each time point τ during the scan, it checks the frequent patterns one by one. For each frequent pattern P_i, it calculates the support of P_i on $\mathcal{D}^-_\tau$, i.e., $sup^-_\tau(P_i)$, and the support of P_i on $\mathcal{D}_\tau^-$, i.e., $sup_\tau^+(P_i)$[2]. If both of them are greater than t_s, the algorithm then checks the progressive ratio of P_i. If the ratio is greater than t_p, then P_i is a progressive frequent pattern and is added into the set $\mathcal{S}_{PFP}$. Then, the algorithm checks whether the progressive ratio of P_i is greater than the current maximal progressive ratio of P_i. If yes, the set of significant frequency-ascending milestones of P_i, i.e., $S_{FAM}(P_i)$, is set to contain $\langle \tau, Prog_\tau(P_i) \rangle$ as its single element. If not but it is equal to the current maximal progressive ratio of P_i, $\langle \tau, Prog_\tau(P_i) \rangle$ is added into $S_{FAM}(P_i)$. Similarly, Steps 21-30 are for finding the set of progressive infrequent patterns and their significant frequency-descending milestones.

If we do not consider the step for generating frequent patterns (i.e., Step 1), the PP-mine algorithm scans the database only once to find all the progressive frequent and infrequent patterns and their significant milestones with respect to a pattern support threshold and a progressive pattern threshold.

5. EXPERIMENTAL STUDIES

To demonstrate the utility of progressive patterns, we have performed two sets of experiments using data sets from two real-world domains: retail market basket and web log data. Table 4 summaries the parameters of each data set along with the threshold values used in our experiments. 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.

Table 4: Database characteristics

Database	# Items	# Trans	# FP	# PFP	# PIP	$\top_{\mathcal{D}}$
Retail	16,470	88,163	580	22	49	[25%, 75%]
LiveLink	38,679	30,586	125	22	22	
$t_s = 5$‰ and $t_p = 50\%$						

5.1 Retail market basket data

The Retail data set was obtained from the Frequent Itemset Mining Dataset Repository[3]. It contains the (anonymized) retail market basket data from an anonymous Belgian supermarket store. Over the entire data collection period, approximately 5 months, the supermarket store carries $16,470$ unique SKU's, and the total amount of receipts being collected equals $88,163$.

Table 5 shows the first 10 progressive frequent patterns in Retail. These patterns are ranked by the progressive ratios at their significant frequency-ascending milestones. For progressive frequent patterns, the greater the ratio, the higher the rank; while for progressive infrequent patterns (shown in Table 6), the less the ratio, the higher the rank.

Table 5: Top 10 Progressive Frequent Patterns in Retail data set

#	PFP	sup_- (‰)	sup_+ (‰)	$\langle \mathcal{M}^+, prog_{\mathcal{M}^+} \rangle$ (%)	sup (‰)	sup #
1	{12925}	5.08	32.95	⟨58.52, +84.59⟩	16.64	72
2	{14098}	5.03	29.72	⟨61.08, +83.07⟩	14.64	88
3	{39, 12925}	5.07	22.96	⟨68.88, +77.91⟩	10.64	149
4	{413}	6.19	26.39	⟨25.08, +76.53⟩	21.32	49
5	{48, 12925}	5.01	19.66	⟨70.93, +74.54⟩	9.27	184
6	{12929}	5.00	18.35	⟨74.41, +72.75⟩	8.42	221
7	{48, 413}	5.01	16.57	⟨31.94, +69.77⟩	12.87	110
8	{39, 413}	5.01	16.30	⟨30.81, +69.28⟩	12.82	112
9	{405}	5.06	15.05	⟨50.85, +66.35⟩	9.97	160
10	{39, 48, 413}	5.00	14.06	⟨57.39, +64.43⟩	8.86	200

The first PFP, product R_{12925}, has a support rank of 72 in the whole Retail dataset, which is a mediocre frequent item. From its significant milestone, we notice that before the time point 58.52%, its frequency is just a little bit greater than the minimum support threshold; but its frequency increases over 6 times after the time point, which is as twice as much of its frequency in the whole Retail data set. This unusual phenomena might be the result of a special even around the time point, such as a new advertisement or a sale promotion. In order to satisfy customers' increasing demands for product R_{12925}, the store has to take actions to enhance the supply of this product. Moreover, the supplies of products R_{39} and R_{48} need to be enhanced as well because of their co-occurrences with product R_{12925} in the 3^{rd} and 5^{th} progressive frequent patterns.

As we can see from the last line of Table 5, there are 3 items R_{39}, R_{48} and R_{413} in the 10^{th} PFP. 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 this significant milestone, these products appear together more frequently after the time point 57.89%. 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 7^th and 8^th progressive frequent patterns.

Table 6: Top 10 Progressive infrequent Patterns in Retail dataset

#	PIP	sup_- (‰)	sup_+ (‰)	$\langle \mathcal{M}^-, prog_{\mathcal{M}^-} \rangle$ (%)	sup (‰)	sup #
1	{1327}	31.82	5.00	⟨56.90, -84.29⟩	20.26	54
2	{39, 1327}	25.51	5.01	⟨39.52, -80.37⟩	13.11	106
3	{48, 1327}	20.80	5.00	⟨37.84, -75.96⟩	10.98	143
4	{32, 39, 41}	45.00	13.04	⟨42.93, -71.02⟩	26.76	36
5	{41, 225}	17.22	5.01	⟨40.44, -70.91⟩	9.95	161
6	{32, 41}	60.82	17.92	⟨42.73, -70.53⟩	36.25	20
7	{38, 39, 41}	57.87	17.19	⟨42.81, -70.29⟩	34.61	22
8	{32, 39, 41, 48}	31.07	9.34	⟨42.93, -69.94⟩	18.67	64
9	{38, 39, 41, 48}	37.63	11.34	⟨42.78, -69.87⟩	22.58	48
10	{41, 65}	18.72	5.69	⟨42.97, -69.61⟩	11.29	137

The first 10 progressive infrequent patterns in Retail are listed in Table 6. The frequency of the 6^th progressive infrequent pattern is very high, ranked 20 out of 580 frequent itemsets. Its frequency is much higher, almost twice as much before the time point 42.73%; but decreases significantly afterwards. This could be the main reason why the frequencies of the 4^{th} and 8^{th} PIPs decrease after almost the same time since product R_{39} has the highest frequency in the Retail

[2] Note that the two supports, $sup_\tau^-(P_i)$ and $sup_\tau^+(P_i)$, can be calculated based on the support count of the pattern on $\mathcal{D}_\tau^-$, which is collected during the database scan, and the supports of P_i over $\mathcal{D}$ computed in Step 1.

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

data set and appears in most of the top PFPs. New marketing strategies should be planned for products R_{32} and R_{41}, such as a new advertisement or price dropping, to resume the sales volume for these two products and other associated products.

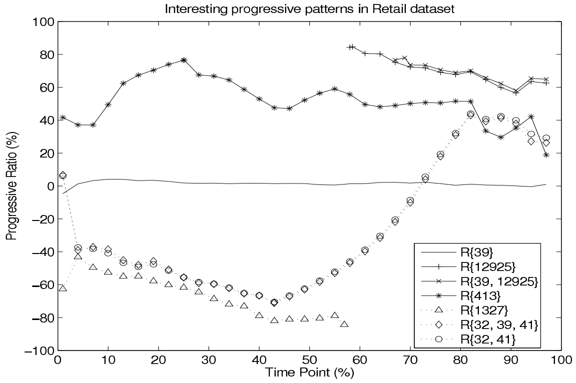

Figure 2: Selected progressive ratios in Retail data set

Another interesting observation is that the significant milestones of most top PIPs occur around 40% to 45%. As shown in Figure 2, even the progressive ratios of some top PFPs are going down during the special period of time. 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.

5.2 Livelink web log data

The Livelink data set was first used in [8] to discovery interesting association rules from Livelink[4] 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 [8].

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 data set is 30,586 and the total number of objects[5] (items) is 38,679.

The top 16 progressive frequent and infrequent patterns in Livelink data set are shown in Table 7 and Table 8, respectively.

As we can see from the first row of Table 7, the object L_{15000} is visited most frequently after the time point 44.17%, its frequency increases about 5 times. This shows that users are very interested in the new information in L_{15000} that are updated after the specific time. Therefore, object L_{15000} should be upgraded to a higher level so that it can be more easily accessed by the users.

Table 7: Top 16 Progressive Frequent Patterns in Livelink data set

#	PFP	sup_{-} (‰)	sup_{+} (‰)	$\langle \mathcal{M}^{+}, prog_{\mathcal{M}^{+}} \rangle$ (%)	sup (‰)	sup #
1	{15000}	5.03	25.12	⟨44.17, +79.97⟩	16.25	25
2	{1375}	5.04	22.72	⟨62.87, +77.79⟩	11.61	35
3	{1859}	5.54	17.92	⟨75.00, +69.10⟩	8.63	58
4	{8106}	5.03	15.60	⟨71.49, +67.75⟩	8.04	65
5	{544}	5.05	15.27	⟨56.96, +66.92⟩	9.45	49
6	{1381}	5.00	15.03	⟨73.24, +66.72⟩	7.68	68
7	{273}	5.53	16.33	⟨57.96, +66.16⟩	10.07	41
8	{1509}	5.03	13.92	⟨45.50, +63.87⟩	9.87	44
9	{545}	5.02	13.80	⟨57.36, +63.66⟩	8.76	56
10	{544, 545}	5.02	13.77	⟨57.98, +63.55⟩	8.70	57
11	{135}	14.83	39.52	⟨74.93, +62.46⟩	21.02	14
12	{135, 136}	12.91	34.05	⟨74.94, +62.07⟩	18.21	18
13	{136}	13.18	34.44	⟨74.94, +61.75⟩	18.51	17
14	{109}	11.32	28.53	⟨43.04, +60.33⟩	21.12	13
15	{1858}	6.63	16.22	⟨75.00, +59.14⟩	9.02	55
16	{2155}	5.04	11.28	⟨73.34, +55.35⟩	6.70	81

Table 8: Top 16 Progressive Infrequent Patterns in Livelink data set

Top 16 Progressive Infrequent Patterns						
#	PIP	sup_{-} (‰)	sup_{+} (‰)	$\langle \mathcal{M}^{-}, prog_{\mathcal{M}^{-}} \rangle$ (%)	sup (‰)	sup #
1	{355}	50.31	7.24	⟨40.42, -85.60⟩	24.65	9
2	{384}	26.56	5.01	⟨52.32, -81.15⟩	16.28	24
3	{11034}	18.60	5.03	⟨32.35, -72.97⟩	9.42	50
4	{434}	33.81	9.76	⟨59.47, -71.14⟩	24.06	10
5	{15001}	17.03	5.04	⟨46.84, -70.39⟩	10.66	38
6	{15000, 15001}	16.62	5.04	⟨46.81, -69.68⟩	10.46	40
7	{1735}	22.00	7.75	⟨60.78, -64.76⟩	16.41	22
8	{396}	14.15	5.00	⟨52.92, -64.66⟩	9.84	45
9	{225, 396}	13.54	5.07	⟨52.90, -62.56⟩	9.55	48
10	{1322}	15.69	5.96	⟨41.26, -62.03⟩	9.97	43
11	{397}	16.78	6.92	⟨60.78, -58.78⟩	12.91	31
12	{225}	87.67	36.80	⟨61.08, -58.03⟩	67.87	3
13	{87}	19.54	8.23	⟨31.29, -57.88⟩	11.77	34
14	{225, 1322}	11.73	5.01	⟨41.26, -57.28⟩	7.78	67
15	{225, 226}	67.00	30.15	⟨60.75, -55.00⟩	52.54	5
16	{383}	10.92	5.00	⟨25.46, -54.20⟩	6.51	85

On the contrary, the frequency of the first progressive infrequent pattern decreased significantly from 57.31% to 7.24% after time point 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 L_{15000}.

Object L_{15000} is also in the 6^{th} progressive infrequent pattern and is frequently visited together with L_{15001} by the users before the time point 46.81%. However, after that time, the frequencies of the 5^{th} and 6^{th} progressive infrequent pattern (L_{15001}) decrease significantly, which means that most of the users who visit L_{15000} do not visit L_{15001} at the same time. Therefore, these two objects should be treated differently. On the other hand, object L_{135} and L_{136} should be in the same category and have links for the user to access from one to the other more easily.

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

[5] 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 [8].

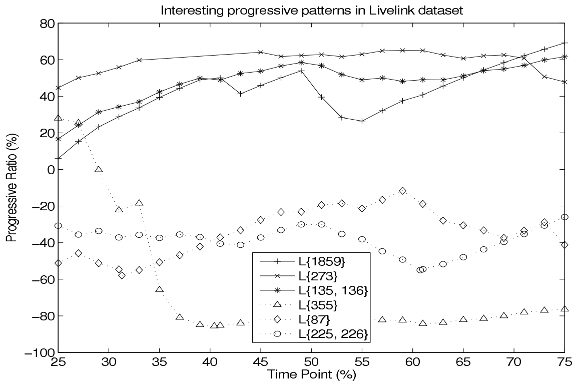

Figure 3: Selected progressive ratios in Livelink data set

6. COMPARISON WITH RELATED WORK

According to our knowledge, the *emerging patterns* proposed in [6] is the only kind of patterns that is similar to progressive patterns. Emerging patterns are defined as itemsets whose support increase significantly from one data set to another. There are two major differences between progressive patterns and emerging patterns:

- Emerging patterns are used to capture the significant difference between two data sets. When applied to time-stamped data sets, emerging patterns are used to find contrasts between two data sets with different time periods, which is separated by a unchangeable time point. Theoretically, emerging patterns can be considered as progressive frequent patterns with time point set to a constant value. As we can see from the above experimental results, the significant milestones of progressive patterns can be at different places in one data set. Thus, at a specific time point, the progressive ratio of a pattern might not reach its greatest value or even close to 0. For example, the progressive ratio of pattern P_4P_6 at time point 50% in TDB is 0 (see Table 3), and the progressive ratio of pattern L_{87} in the Livelink data set is close to 0 at about 60% (see Figure 3). Therefore, with a constant time point value, most of the interesting progressive patterns cannot be identified correctly.
- More specifically, emerging patterns are itemsets whose growth rates are larger than a given threshold; and interest in emerging patterns is mainly on the degree of changes in supports, but not on their actual support values. For example, suppose two patterns X and Y never occur in the first data set but occur once and 10000 times in the second data set, respectively. According to the definition of the growth rate given in [6], both X and Y have the same growth rate of ∞. Therefore, they are considered as emerging patterns with the same significance. This problem is solved in our progressive pattern framework by adding the information of time with a minimum support threshold.

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 progressive 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 involving, 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.

7. 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, progressive frequent and infrequent patterns, to represent frequent patterns whose frequency of occurrences changes significantly at some point of time in the transaction database. We also defined the concepts of significant frequency-ascending milestones and significant frequency-descending milestones to capture the time points where the frequency of patterns changes most significantly. To discover progressive patterns, we proposed the PP-mine algorithm to mine the complete set of progressive frequent and infrequent patterns with respect to a pattern support threshold and a progressive pattern threshold. Our algorithms takes one database scan after mining frequent patterns to find the progressive patterns and their significant milestones.

In our experimental study, we demonstrated the usefulness of progressive patterns in two real-world domains and showed that what is revealed by the progressive 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 progressive ratio would lead to better discovery of progressive patterns and their milestones. Second, we would like to identify more types of patterns (such as periodical patterns) by analyzing the discovered milestones. Moreover, finding progressive sequential patterns is another interesting topic that we would like work on in the future.

8. REFERENCES

[1] R. Agarwal, C. Aggarwal, and V. V. V. Prasad. Depth first generation of long patterns. In *Proceedings of ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[3] R. Agrawal and R. 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, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[4] R. J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the International ACM SIGMOD Conference*, pages 85–93, May 1998.

[5] D. Burdick, M. Calimlim, and J. Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proceedings of the 17th International Conference on Data Engineering*, Heidelberg, Germany, April 2001.

[6] G. Dong and J. Li. Efficient Mining of Emerging Patterns: Discovering Trends and Differences. *Knowledge Discovery and Data Mining*, pages 43–52, 1999.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of ACM-SIGMOD International Conference on Management of Data*, pages 1–12, Dallas, TX, May 2000.

[8] X. Huang, A. An, N. Cercone, and G. Promhouse. Discovery of interesting association rules from livelink web log data. In *Proceedings of IEEE International Conference on Data Mining*, Maebashi City, Japan, 2002.

[9] J. Li and K. Ramamohanarao and G. Dong. Emerging Patterns and Classification. In *Proceedings of the 6th Asian Computing Science Conference on Advances in Computing Science*, pages 15–32, London, UK, 2000.

[10] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. *Lecture Notes in Computer Science*, 1540:398–416, 1999.

[11] J. Pei, J. Han, and R. 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.

[12] P. Tan and V. Kumar. Mining indirect associations in web data. In *Proc of WebKDD2001: Mining Log Data Across All Customer TouchPoints*, August 2001.

[13] P. Tan, V. Kumar, and J. Srivastava. Indirect association: mining higher order dependencies in data. In *Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 632–637, Lyon, France, 2000.

[14] M. Zaki and K. Gouda. Fast vertical mining using diffsets. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2003.

[15] M. J. Zaki and C. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *In Proceedings of the second SIAM International Conference on Data Mining (SIAM 2000)*, 2000.