MINING SEQUENTIAL PATTERNS WITH CONSIDERATION TO RECENCY, FREQUENCY, AND MONETARY

Ya-Han Hu, Department of Information Management, National Chung Cheng University, Chiayi, Taiwan, R.O.C., yahan.hu@mis.ccu.edu.tw

Yu-Hua Kao, Department of Information Management, National Chung Cheng University, Chiayi, Taiwan, R.O.C., yokus26@gmail.com

Abstract

For superior decision making, the mining of interesting patterns and rules becomes one of the most indispensible tasks in today’s business environment. Although there have been many successful customer relationship management (CRM) applications based on sequential pattern mining techniques, they basically assume that the importance of each customer are the same. Many studies in CRM show that not all customers have the same contribution to business, and, to maximize business profit, it is necessary to evaluate customer value before the design of effective marketing strategies. In this study, we include the concepts of RFM analysis into sequential pattern mining process. For a given subsequence, each customer sequence contributes its own recency, frequency, and monetary scores to represent customer importance. An efficient algorithm is developed to discover sequential patterns with high recency, frequency, and monetary scores. Empirical results show that the proposed method is more advantageous than conventional sequential pattern mining.

Keywords: Data mining, sequential patterns, RFM analysis, constraint-based mining.
1 INTRODUCTION

Over the past decade, there have been tremendous interests in analyzing huge amount of data through various data mining techniques. Data mining can be treated as the process of extracting implicit, previously unknown and potentially useful information from databases (Chen et al. 1996; Chen & Huang 2005; Frawley et al. 1991; Han & Kamber 2001). In business applications, sequential pattern mining can be applied for discovering time-related purchasing behaviour in sequence databases (Chen et al. 2003; Huang & Huang 2009; Lin et al. 2006; Mannila et al. 1997). Many other applications have conducted sequential pattern mining technique to discover useful information, including the analysis of customer purchase behavior, process analysis of scientific experiments, web-log analysis, and so on (Cooley et al. 1999; Ester 2004; Vijayalakshmi et al. 2009; Wang et al. 2004).

Although there have been many successful customer relationship management (CRM) applications of sequential pattern mining techniques (Eirinaki & Vazirgiannis 2003; Liu & Shih 2005), they basically assume that the importance of each customer are the same. However, many studies in CRM show that not all customers have the same contribution to business, and, to maximize business profit, it is necessary to evaluate customer value before the design of effective marketing strategies. Therefore, the consideration to customer value is essential in the process of sequential pattern discovery.

Recently, the concept of Recency, Frequency and Monetary (RFM) analysis has been integrated into the mining of valuable sequential patterns (Chen & Hu 2006; Chen et al. 2009; Cheng & Chen 2009; Liu et al. 2009) and it can be divided into two main approaches. The first is on conducting customer segmentation based on RFM analysis and then discovering patterns from valuable customer group (i.e., also known as 1-1-1 customers) (Liu & Shih 2005; Liu et al. 2009; Miglautsch et al. 2002). This approach is a combination of both clustering and sequential pattern mining techniques. On the other hand, the second is directly on regarding RFM features as constraints in the mining of sequential patterns. This approach is far more complicated than the first one since it involves redesigning the definition of sequential pattern as well as its mining procedure.

Although the first approach performs well in some applications, it has limitation for pattern mining in large-scale retail store. Consider the following example, a large grocery retailer contains over ten thousand kinds of goods, and the varieties of both price range and purchase frequency is huge (e.g., a dozen of eggs vs. a LCD TV). Assume that a group of customers who regularly purchases potato chip and chocolate each week. If we consider all customer transactions as a whole, this group of customers will never be classified as 1-1-1 customers in RFM analysis. It is because the total amount of money for this customer is relatively small when compared with all other customers in the store, and patterns behind such kind of customers are eliminated as well. Due to the above observation, the first approach works well only when investigating single item or items with similar price range and purchase frequency.

The second approach solves problem addressed above. Instead of measuring RFM value of customers and pre-filtering out valueless ones, it directly treats the RFM features as constraints in sequential pattern mining process (Chen & Hu 2006; Chen et al. 2009). By setting adequate thresholds of recency, frequency, and monetary, the mining algorithms search complete customer sequence database and filter out worthless patterns. Although this approach somehow includes the concept of RFM features through various constraints, it still follows the basic assumption of sequential pattern mining, that is, the equal weight of customer sequence. Hence, it results in the sequential patterns of 1-1-1 customers cannot be highlighted.

In this study, we push the concepts of RFM analysis deep into sequential pattern mining process. In RFM analysis, each customer has its own recency, frequency, and monetary scores to represent customer value. We briefly describe how this study incorporates the concept of recency, frequency, and monetary into definitions of the sequential pattern as follows. Given a subsequence, the recency score of a customer is determined by the time of the last occurrence of a subsequence in the customer
sequence. The more recent the last occurrence of a subsequence is, the higher recency score this customer gets. Frequency is defined as the number of occurrences of a subsequence in a customer sequence. If a subsequence repeatedly occurs many times in a customer sequence, the frequency score of this customer for the subsequence is considered as high. Monetary denotes the amount of money this customer spent on the subsequence. A customer with high monetary score means that it contributes higher revenue to business. Based on above definitions, in this study, we can evaluate and accumulate the RFM value of each subsequence according to its recency, frequency, and monetary scores from each customer. In addition, we also consider the compactness constraint, which means the time span between the first and the last purchase in a customer sequence must be within a user-specified threshold. This constraint can assure that the purchasing behavior implied by a sequential pattern must occur in a reasonable period.

Follow the above definitions, we define the RFM sequential pattern (RFM-SP), meaning that sequential patterns with their recency, frequency, and monetary scores satisfy the user-specified minimum recency, frequency, and monetary thresholds, respectively. After that, we develop an efficient projected-based algorithm, called RFM-PostfixSpan, for mining complete set of RFM-SPs.

The rest of this paper is organized as follows. Section 2 reviews the concept of RFM model and recent studies on constraint-based sequential pattern mining. Section 3 formally defines the problem of mining RFM-SPs. Section 4 proposes the RFM-PostfixSpan algorithm. Section 5 depicts the evaluation results of our experiments. Section 6 concludes the study.

2 RELATED WORK

2.1 Customer relationship management using sequential pattern mining

Customer relationship management (CRM) has become a main task in today’s business environment. A definition for CRM is a process that manages the interactions between companies and their customers. From the aspect of information technology, CRM can be seen as an integration of technologies and business processes used to satisfy the needs of a customer during any given interaction.

In the past few years, various data mining techniques have been conducted in CRM (Mannila 1998; Shaw et al. 2001; Stone 1989). It applies a series of information technology, such as automated data collection methods, data analysis techniques, and knowledge discovery methods, in order to sell more goods or services, retain old customers, and increase customer profitability. Among all data mining techniques, sequential pattern mining is particularly useful in the analysis of customers, where certain buying patterns could be identified.

There are many successful CRM applications conducting sequential pattern mining techniques. For example, sequential pattern mining techniques can be applied to discover users’ browsing behaviour from web servers. Interesting user access patterns can be extracted from web access logs. We can use this useful information to predict customers’ next visit or enhance systems’ performance (Pei et al. 2000; Srikant & Yang 2001; Tao et al. 2008). For another example, the purchasing behaviour of a customer can be extracted from members’ transaction records (Tsai 2007). In order to gain maximum profit, the seller/retailer can use these patterns to design sales strategy, such as repetitive selling or cross selling.

2.2 Elicitation of sequential patterns with constraints

Frequent pattern mining plays a crucial role in association rule discovery (Agrawal & Srikant 1995). A sequence or an itemset is considered as a frequent pattern if its support satisfies a user-specified frequency constraint. In past decade, there exist numerous follow-up researches and applications on
frequent pattern mining. Their experimental evaluations reveal that, instead of uncovering thousands of patterns by considering frequency only, incorporating additional constraints into the mining process can yield more promising results (i.e. interesting patterns) (Kiran & Reddy 2009; Pei et al. 2002; Seno & Karypis 2005; Yun 2008).

Srikant and Agrawal (1996) first propose a generalized sequential pattern mining with item constraint and taxonomy. To meet user’s interest, it provides flexibility to discover frequent patterns containing specific items at multiple levels of abstraction. Ng et al. (1998) propose CAP algorithm to include syntactic constraints (e.g. min, max, count, and sum etc.) to reduce computation cost of finding frequent patterns. Recently, some researchers conduct systematic studies or provide thorough experimental evaluations on constraint-based frequent pattern mining. In Bonchi and Lucchese (2007); Han et al. (1999), the problem constraint-based mining is divided into five categories, including antimonotone, monotone, succinct, convertible, and inconvertible, based on the interactions between constraints and mining algorithms. For constraint-based sequential pattern mining, Pei, et al. (2002) classified the methods into seven categories, which includes both temporal and non-temporal constraints. Among them, non-temporal constraints are also applied to frequent pattern mining. Item constraint and super-pattern constraint specify that the discovered patterns should contain or not contain the specific subset/superset of items. Length constraint is used to specify the threshold of pattern length. Regular expression constraint is to satisfy user-specified templates. Aggregate constraint is to consider pattern aggregation information in the mining process.

3 PROBLEM DEFINITION

A customer’s data-sequence $A$ can be represented by $<(a_1, t_1, q_1),(a_2, t_2, q_2),\ldots,(a_n, t_n, q_n)>$, where $(a_j, t_j, q_j)$ indicates that item $a_j$ was purchased at time $t_j$ with quantity $q_j$, $1\leq j\leq n$, and $t_{j-1} \leq t_j$ for $2\leq j \leq n$. If items occur at the same time in the data sequence, they are ordered alphabetically. Based on this format, we give the following definitions:

**Definition 1.** (Containment of itemset) Let $I$ denote the set of all items in the database. Give a data-sequence $A=<(a_1, t_1, q_1),(a_2, t_2, q_2),\ldots,(a_n, t_n, q_n)>$ and an itemset $I_q = (i_1, i_2,\ldots,i_m)$, where $I_q \subseteq I$ and $m \leq n$. We say $I_q$ is contained in $A$ if there are $m$ integers $1 \leq k_1 < k_2 < \ldots < k_m \leq n$ such that $i_1 = a_{k_1}, i_2 = a_{k_2},\ldots, i_m = a_{k_m}$ and $t_{k_1} = t_{k_2} = \ldots = t_{k_m}$.

**Definition 2.** (Subsequence) Let $B = <I_1, I_2,\ldots, I_q >$ be a sequence of itemsets, where each $I_q \subseteq I$, $(1 \leq p \leq q \leq s)$. Sequence $B$ said to be contained in $A$, or a subsequence of $A$ if the following conditions are satisfied: (1) each $I_q$ in $B$ is contained in $A$, and (2) $t_{k_1} < t_{k_2} < \ldots < t_{k_q}$ where $t_{k_q}(1 \leq q \leq s)$ is the time at which $I_q$ occurs in $A$.

**Example 1.** An itemset $<ab>$ is contained in data-sequence $A=<b(1,10),(c,3,5), (a,5,40),(b,5,20),(d,7,30),(a,8,20),(e,8,10)>$, because both items $a$ and $b$ occur in $A$ at time 5. The sequence $<ab>(ae)$ is a subsequence of $A$ since itemsets $(ab)$ and $(ae)$ are contained in $A$ at time 5 and at time 8, respectively, and $t_{(ab)} < t_{(ae)}$.

**Definition 3.** (Compactness constraint) Following definition 2, assume that $B=<I_1, I_2,\ldots, I_q >$ is a subsequence of $A$. Let $ms\text{-}length$ be a user-specified maximum span length, $B$ is called a compact subsequence (c-subsequence) of $A$ if $t_{I_q} - t_{I_{q-1}} \leq ms\text{-}length$.

**Example 2.** Consider the sequence database shown in Table 1a. Assume that $ms\text{-}length$=25, a sequence $BS=<ab>(ae)$ is called a c-subsequence of $sid_{10}$ since (1) itemsets $(ab)$ and $(e)$ are contained in $sid_{10}$ at time 40 and time 60; (2) $t_{(ab)} < t_{(e)}$; and (3) $t_{(e)} - t_{(ab)} \leq ms\text{-}length$=25.
Definition 4. (Cyclically) When we say a c-subsequence $B$ cyclically occurs in $A$, it means that the concatenation of $B$ is also a c-subsequence of $A$, i.e. sequence $<...B_1...B_2...> \text{ is a subsequence of } A$.

We extend the concept of RFM model to give the definitions of recency, frequency and monetary, as following definition 5-8.

Definition 5. (Frequency subsequence) Given a data-sequence $A$ and let $B$ be a c-subsequence of $A$. Assume that $B$ cyclically occurs in $A$. The Frequency score of $B$, denoted as $Fscore(B,A)$, is defined as the sum of frequency score of all data-sequence containing $B$. The total frequency score of $B$, denoted as $TFscore(B)$, is defined as the sum of frequency score of all data-sequences containing $B$.

$$TFscore_{DB}(B) = \sum_{A \in DB} Fscore(B,A)$$

Example 3. Follow Example 2. Consider a data-sequence $sid_{30}$ in Table 1a, we find that $B$= $<ab(e)>$ is a c-subsequence of $sid_{10}$ and $sid_{30}$, Besides, $B$ has two occurrences in $sid_{30}$, where the first occurrence is during time interval 30-45 and the second is during 60-85. Thus, $Fscore(B,sid_{30})=2$, $Fscore(B,sid_{10})=1$, and finally the total frequency score of $B$ is equal to 3 (i.e., $TFscore(B)=1+2=3$).

Definition 6. (Recency subsequence) Assume that a c-subsequence $B$ cyclically in $A$, the recency score of $B$ in $A$, denoted as $Rscore(B,A)$, is equal to $(1-\delta)^{t_{current}-t_i^B}$, where $\delta$ is a user-specified decay speed ($\delta \in [0,1]$), $t_{current}$ denotes the current time stamp, and $t_i^B$ denotes the timestamp of the last item(i.e., $T_i$) in $B$. Given a sequence database $SDB$, the total recency score of sequence $B$, denoted as $TRscore(B)$, is defined as the sum of recency score of all data-sequence containing $B$.

$$TRscore_{SDB}(B) = \sum_{A \in SDB} Rscore(B,A)$$

Example 4. Following Example 2 and 3. Assume the current timestamp $t_{current}$=110 and decay speed $\delta$=0.1. Consider a data-sequence $sid_{30}$ shown in Table 1a, since c-subsequence $B$= $<ab(e)>$ has two occurrences in $sid_{30}$, and the most recent one is during 60-85. A c-subsequence $B$’s $Rscore$ in $sid_{30}$ (i.e., $Rscore(B,sid_{30})$ ) is equal to $(1-0.1)^{110-85} = 0.0717898$. Similarly, $Rscore(B,sid_{10})=(1-0.1)^{110-80}=0.0051538$. Hence, $TRscore(B)=0.0717898 + 0.0051538 = 0.0769436$.

Definition 7. (Monetary subsequence) Assume that c-subsequence $B$ cyclically occurs r times in $A$, i.e. sequence $<...B_1...B_2...B_r...> \text{ is a subsequence of } A$. Let $P(a_i)$ denotes the unit profit of item $a_i$, and $q_i^k$ the purchase quantity of $a_i$ in $B_k$ (1 $\leq k \leq r$). Then $B$’s monetary score in $A$, denoted as $Mscore(B,A)$, is defined as follows.

$$Mscore(B,A) = \sum_{i=0}^{B} \sum_{a \in B} P(a_i) \times q_i^k$$
Moreover, the total monetary score of sequence \( B \), denoted as \( TMscore(B) \), is defined as the sum of monetary score of all data-sequences containing \( B \).

\[
TMscore_{SDB}(B) = \sum_{A \subseteq SDB} Mscore(B, A)
\]

**Example 5.** Consider the list of item unit price shown in Table 1b. Following Example 3, we known that \( B = \langle (ab)(e) \rangle \) occurs once in \( \text{sid}_{10} \) and twice in \( \text{sid}_{30} \). For \( \text{sid}_{10} \), \( Mscore(B, \text{sid}_{10}) = 2 \times 10 + 1 \times 150 + 1 \times 80 = 250 \). \( Mscore(B, \text{sid}_{30}) = (3 \times 10 + 4 \times 150 + 2 \times 80) + (6 \times 10 + 1 \times 150 + 7 \times 80) = 1560 \). Finally, the \( TMscore(B) = 1560 + 250 = 1810 \).

**Definition 8.** Given user-specified recency, frequency, and monetary thresholds, denoted as \( Rmimsup, Fminsup, \) and \( Mminsup \), respectively, we say a subsequence \( B \) is a RFM-sequential pattern (RFM-SP) if \( Trscore \geq Rmimsup, TFScore \geq Fminsup \) and \( TMscore \geq Mminsup \).

**Example 6.** As the Example 2~5, we call the sequence \( B = \langle (ab)(e) \rangle \) is a RFM sequential pattern since the sequence \( B \) satisfy \( Trscore = 0.0769436 \geq Rmimsup = 0.05 \), \( TFScore = 3 \geq Fminsup = 2 \), and \( TMscore = 1810 \geq Mminsup = 1500 \).

### 4 THE RFM-POSTFIXSPAN ALGORITHM

The RFM-PostfixSpan algorithm is developed by modifying the well-known PrefixSpan algorithm, which partitions the sequence database into a number of projected databases and discovers only local frequent patterns in each projected database. Instead of exploring \( SDB \) from the prefix of a sequence (i.e., conventional PrefixSpan algorithm), in RFM-PostfixSpan, we partition \( SDB \) from postfix for efficiently retrieving the recency score of a pattern. Below, we first define compact postfix, compact projection and compact prefix, and depict the RFM-PostfixSpan later.

**Definition 9.** *(The compact postfix)* Given a data sequence \( A = \langle (a_1, t_1, q_1), (a_2, t_2, q_2), \ldots, (a_n, t_n, q_n) \rangle \) and a sequence \( B = \langle I_1, I_2, \ldots, I_s \rangle \), \( B \) is a compact postfix of \( A \) if and only if (1) \( B \) is a c-subsequence of \( A \), (2) \( t_1 = t_n \).

**Example 7.** Given a data sequence \( A = \langle (a, 30, 3), (b, 30, 4), (e, 45, 2), (c, 55, 8), (a, 60, 6), (b, 60, 1), (a, 85, 3), (e, 85, 7) \rangle \) and \( ms_length = 30 \). Sequence \( \langle (c)(ae) \rangle \) is a compact postfix of \( A \) since (1) the timestamp of itemsets \( (c) \) and \( (ae) \) in \( A \) are 55 and 85, respectively, and the distance between them is 30 (i.e., \( \leq ms_length \)), and (2) itemset \( (ae) \) is the last itemset of \( A \). Sequence \( \langle (c)(ae) \rangle \) is not a compact postfix of \( A \) since itemset \( (c) \) occur at time 45 and the distance between \( (b) \) and \( (ae) \) (i.e., 85-45=40) doesn’t satisfy \( ms_length \).

**Definition 10.** *(The compact projection)* Given a data sequence \( A = \langle (a_1, t_1, q_1), (a_2, t_2, q_2), \ldots, (a_n, t_n, q_n) \rangle \), let \( B = \langle b_1, b_2, \ldots, b_m \rangle \) be a c-subsequence of \( A \). A sequence \( A = \langle (a_1, t_1, q_1), (a_2, t_2, q_2), \ldots, (a_p, t_p, q_p) \rangle \) of sequence \( A \) is called a compact projection of \( A \) with respect to compact postfix \( B \) if and only if (1) \( A \) has compact postfix \( B \), (2) \( A \) is a c-subsequence of \( A \), and (3) there exists no supersequence \( A' \) of \( A \) such that \( A' \) is a c-subsequence of \( A \) and also has compact postfix \( B \).

**Example 8.** Given a sequence \( A = \langle (a, 30, 3), (b, 30, 4), (e, 45, 2), (c, 55, 8), (a, 60, 6), (b, 60, 1), (a, 85, 3), (e, 85, 7) \rangle \), and let \( ms_length = 40 \). Assume that \( A \) is projected with compact postfix \( B = \langle (a) \rangle \), we have following three compact projections \( A' \), including \( \langle (a, 30, 3) \rangle, \langle (a, 30, 3), (b, 30, 4) \rangle, \langle (e, 45, 2) \rangle, \langle (c, 55, 8) \rangle, \langle (a, 60, 6) \rangle \) and \( \langle (e, 45, 2), (c, 55, 8), (a, 60, 6), (b, 60, 1), (a, 85, 3) \rangle \). Each of the three compact projections satisfy the following conditions: (1) it has compact postfix \( \langle a \rangle \), (2) the time interval between the first and the last item in a projection satisfies is smaller than or equal to 40 (i.e., a c-subsequence of \( A \), and (3) appending additional items in any of the above three compact projections results in violating (2).
Definition 11. (The compact prefix) Let $A = \langle a_1, t_1, q_1 \rangle, \langle a_2, t_2, q_2 \rangle, \ldots, \langle a_p, t_p, q_p \rangle$ be a compact projection of $A$ with respect to the compact postfix $B = \langle I_1, I_2, \ldots, I_s \rangle$. Let $a_m$ be the first item in the itemset $I_i$ and $t_m$ the time at which $a_m$ occurs in $A$ ($1 \leq m \leq p$). Then $C = \langle a_1, t_1, q_1 \rangle, \langle a_2, t_2, q_2 \rangle, \ldots, \langle a_{m-1}, t_{m-1}, q_{m-1} \rangle$ is the compact prefix of $A$ with respect to $B$.

To differentiate these compact prefixes generated from the same data-sequence, the tag [sid:end_time] is attached to each compact prefix, where sid is the identifier of the data-sequence and end_time is the timestamp in $A$ that matches the last itemset of $B$.

Example 9. Following Example 8, the compact prefixes can be easily obtained by directly removing the compact postfix from the compact projection. Since the first compact projection only contain compact postfix (i.e., $\langle(a,30,3)\rangle$), the compact prefix becomes null. Therefore, the remaining two compact prefixes with respect to $B = \langle a \rangle$ are $[\text{sid}=60] \langle(a,30,3), (b,30,4), (e,45,2), (c,55,8)\rangle$ and $[\text{sid}=85] \langle(e,45,2), (c,55,8), (a,60,6), (b,60,1)\rangle$.

Based on Definition 7, we find the downward closure property no longer holds if $TMscore$ is considered in the mining process. While we include more items or itemsets in a non-RFM-SP, its $TMscore$ will increase and its super-sequence is possible to become an RFM-SP. This property is critical in pattern-growth-based methods (i.e., PrefixSpan and RFP-PostfixSpan) due to great decrease of search space. To solve the problem, we first give two definitions as follows.

Definition 12. (The SM value) Given a sequence $A$ and let $B$ be a compact postfix of $A$. The sequence monetary of compact postfix $B$ in $A$, denoted as $SM(B, A)$, is the total amount of monetary gained from all compact prefixes of $A$ with respect to $B$.

Example 10. Consider a list of item unit price shown in Table 1b. Follow example 9, two compact prefixes of $B = \langle a \rangle$ are $[\text{sid}=60] \langle(a,30,3), (b,30,4), (e,45,2), (c,55,8)\rangle$ and $[\text{sid}=85] \langle(e,45,2), (c,55,8), (a,60,6), (b,60,1)\rangle$. The SM values of the two compact projections are 990 and 600.

Definition 13. (The TSM value) The total sequence monetary of compact postfix $B$ is defined as the sum of sequence monetary in SDB:

$$TSM(B) = \sum_{A \subseteq SDB} SM(B, A)$$

Given a sequence $B$, assume that $TMscore(B) < \text{Mminsup}$ (i.e., $B$ is not a RFM-SP), the algorithm will check whether the value of $(TMscore(B) + TSM(B))$ satisfies $\text{Mminsup}$ or not. If $(TMscore(SDB) + TSM(SDB)) \geq \text{Mminsup}$, sequence $B$ will retain for further consideration since its supersets may satisfy $\text{Mminsup}$. On the contrary, if $(TMscore(SDB) + TSM(SDB)) \leq \text{Mminsup}$, sequence $B$ will be discarded directly. Moreover, while cumulating sequence monetary from compact prefixes, the overcounting problem may exist. It is because identical items may simultaneously exist in two or more compact prefixes. To solve the problem, the timestamp of each occurrence in compact prefix will be recorded during the counting process. While proceeding to the next compact prefix, the algorithm will compare the timestamps of each item with those ones recorded in previous occurrence.

Example 11. Consider the list of item unit price shown in Fig. 1b. Follow example 10, two compact prefixes of $B = \langle a \rangle$ are $[\text{sid}=60] \langle(a,30,3), (b,30,4), (e,45,2), (c,55,8)\rangle$ and $[\text{sid}=85] \langle(e,45,2), (c,55,8), (a,60,6), (b,60,1)\rangle$. We find that the subsequences $\langle e, 45, 2, c, 55, 8 \rangle$ in both two compact prefixes are identical, so they will be counted once. Hence, we have $SM(B, \text{sid}=30) = (3 \times 10 + 4 \times 150 + 2 \times 80 + 8 \times 25) + (6 \times 10 + 1 \times 150) = 1200$.

As we mentioned above, the downward closure property no longer holds when we take account of the profit of a pattern, but another two constraints, recency and frequency, hold the property. The RFM-PostfixSpan algorithm can utilize these two constraints to reduce the search space and efficiently discover all RFM-patterns. For ease of explanation of the algorithm, we define RF-sequential pattern.
Definition 14. Given user-specified recency and frequency thresholds, $R_{\text{minsup}}$ and $F_{\text{minsup}}$, we say a sequence $B$ is an RF-sequential pattern (RF-SP) if $TR_{\text{score}}_{\text{SDB}}(B) \geq R_{\text{minsup}}$, $TF_{\text{score}}_{\text{SDB}}(B) \geq F_{\text{minsup}}$ and $TM_{\text{score}}(B)+TSM(B) \geq M_{\text{minsup}}$.

The RFM-PrefixSpan algorithm is shown in Figure 1. We briefly state the main procedure as follows. Initially, we set $a=\text{null}$. For a given $a$-projected database $SDB|_a$ the algorithm first appends each item in $SDB|_a$ to $a$ to form $a'$. After that, it calculates $TR_{\text{score}}_{\text{SDB}}(a')$, $TF_{\text{score}}_{\text{SDB}}(a')$, and $TM_{\text{score}}_{\text{SDB}}(a')$ for each $a'$ to find complete sets of RF-SPs and RFM-SPs. Note that super-sequences of an RF-SP have possibilities to be RFM-SPs. Therefore, for each RF-SP in $a'$, we construct the projected database $SDB|_{a'}$ and recursively call the procedure RFM-PostfixSpan for finding further RFM-SPs. We use the following example to illustrate the major steps of RFM-PostfixSpan in detail.

Input: A sequence database $SDB$, and the maximum span length $ms\_length$ and three support thresholds $R_{\text{minsup}}$, $F_{\text{minsup}}$ and $M_{\text{minsup}}$.

Subroutine: PostfixSpan($a$, $l$, $SDB|_a$)

Parameters: $a$ is a set of RF-SP $l$ is the length of $a$ $SDB|_a$ is the $a$-projected database

Output: The complete set of RF-SPs and RFM-SPs

Method:

Each item in $SDB|_a$ is appended before $a$ as $a'$ or is added into the first itemset of $a$ as $a'$.

Scan the database $SDB|_a$ once and calculate $TR_{\text{score}}_{\text{SDB}}(a')$, $TF_{\text{score}}_{\text{SDB}}(a')$, and $TM_{\text{score}}_{\text{SDB}}(a')$ for each $a'$.

For each $a'$:

If $TR_{\text{score}}_{\text{SDB}}(a') \geq R_{\text{minsup}}$, $TF_{\text{score}}_{\text{SDB}}(a') \geq F_{\text{minsup}}$ and $TM_{\text{score}}_{\text{SDB}}(a') \geq M_{\text{minsup}}$ then

output $a'$ as an RF-SP.

If $TM_{\text{score}}_{\text{SDB}}(a') \geq M_{\text{minsup}}$ then

output $a'$ as an RFM-SP.

Construct $a'$-projected database $SDB|_{a'}$ and call RFM-PostfixSpan($a'$, $l+1$, $SDB|_{a'}$).

End for.

Figure 1. The RFM-PostfixSpan algorithm.

• Find 1-length RF-SPs and RFM-SPs

For the sequence database $SDB$ in Table 1a. Let $t_{\text{current}} = 110$, $ms\_length=40$, $R_{\text{minsup}}=0.05$, $F_{\text{minsup}}=2$, and $M_{\text{minsup}}=1500$. Scan $SDB$ once and count the $TR_{\text{score}}$, $TF_{\text{score}}$ and $TM_{\text{score}}$ for each item to find the complete set of 1-RF-SPs (i.e., RF-SPs with one item only) and 1-RFM-SPs (i.e., RFM-SPs with one item only). As a result, they are ($a$): (0.549867, 8, 2540, 7160); ($b$): (0.080289, 7, 3000, 5895); ($c$): (0.061061, 5, 4750, 1810); ($d$): (0.896466, 7, 3600, 9480); ($e$): (0.094583, 5, 1440, 3060), where the notation "<pattern>: ($TR_{\text{score}}$, $TF_{\text{score}}$, $TM_{\text{score}}$, $TSM$)" represent the pattern and its associated total recency score, total frequency score, total monetary score and total sequence monetary score. Since all five items satisfy both $R_{\text{minsup}}$ and $F_{\text{minsup}}$ and their $TM_{\text{score}}+TSM$ satisfy $M_{\text{minsup}}$, we output all items (i.e., $\{a,b,c,d,e\}$) as 1-RF-SPs. In addition, since the $TM_{\text{score}}$ of the first four items (i.e., $\{a,b,c,d\}$) also satisfy $M_{\text{minsup}}$, we output them as 1-RFM-SPs.

• Divide search space

According to the above five 1-RF-SPs, the algorithm can partition the complete set of sequential patterns into five subsets, including (1) the ones with postfix <a>, (2) with postfix <b>, (3) with postfix <c>, (4) with postfix <d>, and (5) with postfix <e>. As shown in Table 2, suppose we want to build the $a$-projected database. For $sid_{10}$, item $a$ occurs at time 10 and 40, and we have the following two compact prefixes, [10,50,10,10]:$null$ and [10,20,40,40]:<a,10,5),(c,30,4)>, where $[sid, M_{\text{score}}_{\text{postfix}}, \text{start\_time}, \text{end\_time}]$: <compact projection> represents its sequence id, the $M_{\text{score}}_{\text{postfix}}$, the $\text{start\_time}$ and the $\text{end\_time}$ of postfix=<a> in the sequence, and the compact projection. Since the compact projection is $null$, it is removed for further processing. Continuing in this manner yields the entire ($a$)’s projected database as well as the $SM$ value of each compact projection. We can also get the $TSM$ value of postfix=<a> from the all the $SM$ value except the monetary of the overlap.
Table 2. The <a>-projected database.

<table>
<thead>
<tr>
<th>Postfix</th>
<th>TSM</th>
<th>sid</th>
<th>Mscore_postfix</th>
<th>start_time</th>
<th>end_time</th>
<th>Projected (prefix) database</th>
<th>SM</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;(a)&gt;</td>
<td>7160</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>40</td>
<td>&lt;(a,10,5),(c,30,4)&gt;</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>150</td>
<td>50</td>
<td>50</td>
<td>&lt;(d,30,2)&gt;</td>
<td>90</td>
</tr>
<tr>
<td></td>
<td></td>
<td>30</td>
<td>30</td>
<td>85</td>
<td>85</td>
<td>&lt;(e,45,2),(c,55,8),(a,60,6),(b,60,1)&gt;</td>
<td>1200</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40</td>
<td>1300</td>
<td>100</td>
<td>100</td>
<td>&lt;(e,70,5)&gt;</td>
<td>400</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>900</td>
<td>90</td>
<td>90</td>
<td>&lt;(c,70,160),(b,85,7),(d,85,6)&gt;</td>
<td>5320</td>
</tr>
</tbody>
</table>

Table 3. The projected database of all the <(a)>.

<table>
<thead>
<tr>
<th>Postfix</th>
<th>TSM</th>
<th>sid</th>
<th>Mscore_postfix</th>
<th>start_time</th>
<th>end_time</th>
<th>Projected (prefix) database</th>
<th>SM</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;(a)&gt;</td>
<td>360</td>
<td>30</td>
<td>90</td>
<td>60</td>
<td>85</td>
<td>&lt;(e,45,2),(c,55,8)&gt;</td>
<td>360</td>
</tr>
</tbody>
</table>

5 EXPERIMENT EVALUATION

In this section, we perform a simulation study to empirically compare the proposed algorithm with the PrefixSpan algorithm (i.e., conventional sequential pattern mining method) (Pei et al. 2004). All the algorithms are implemented in Java language and tested on Intel core 2 Q8300-2.5GHz Windows XP system with 4 gigabyte of main memory.

5.1 Synthetic and Real-life Datasets

Seven synthetic datasets are generated by applying the synthetic data generation algorithm in Chen et al. (2009). Table 4 lists the parameters used in the data generation algorithm. Several parameters are fixed in data generation, including |S|=4, |I|=1.25, N_s=5,000, N_i=25,000, N=10,000, T=10, H_price=1,000, M_price=500, L_price=100, H_quantity=5, M_quantity=3 and L_quantity=1. We
varied the value of $|D|$ (from 250K to 750K), $|C|$ (from 10 to 20), and $|T|$ (from 2.5 to 4.5), to perform runtime analyses. The parameter settings of seven synthetic datasets are shown in Table 5.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>D</td>
</tr>
<tr>
<td>$</td>
<td>C</td>
</tr>
<tr>
<td>$</td>
<td>T</td>
</tr>
<tr>
<td>$</td>
<td>S</td>
</tr>
<tr>
<td>$</td>
<td>I</td>
</tr>
<tr>
<td>$N_S$</td>
<td>Number of maximal potentially large sequences</td>
</tr>
<tr>
<td>$N_I$</td>
<td>Number of maximal potentially large itemsets</td>
</tr>
<tr>
<td>$N$</td>
<td>Number of items</td>
</tr>
<tr>
<td>$T_I$</td>
<td>Average length of time intervals</td>
</tr>
<tr>
<td>$H_price$</td>
<td>Average price of high price items</td>
</tr>
<tr>
<td>$M_price$</td>
<td>Average price of medium price items</td>
</tr>
<tr>
<td>$L_price$</td>
<td>Average price of low price items</td>
</tr>
<tr>
<td>$H_quantity$</td>
<td>Average purchased quantity of high price items</td>
</tr>
<tr>
<td>$M_quantity$</td>
<td>Average purchased quantity of medium price items</td>
</tr>
<tr>
<td>$L_quantity$</td>
<td>Average purchased quantity of low price items</td>
</tr>
</tbody>
</table>

Table 4. The parameters of synthetic datasets

| Name | $|D|$ | $|C|$ | $|T|$ |
|------|------|------|------|
| SYN-1 | 250K | 10 | 2.5 |
| SYN-2 | 500K | 10 | 2.5 |
| SYN-3 | 750K | 10 | 2.5 |
| SYN-4 | 250K | 5  | 2.5 |
| SYN-5 | 250K | 15 | 2.5 |
| SYN-6 | 250K | 10 | 1.25 |
| SYN-7 | 250K | 10 | 3.75 |

Table 5. The parameter settings of synthetic datasets

We also investigate a real-life dataset in our experiments. This dataset contained all sales data of a supermarket chain in Taiwan. The sales data, called SC-POS, which recorded all transactions from twenty branches between 2001/12/27 and 2002/12/31. Each transaction in SC-POS is a customer’s shopping list, which records the purchase items, prices, and quantities. After we perform all necessary data pre-processing tasks, the dataset contains 17,685 items and 33,509 customer’s data sequence.

5.2 Performance Evaluation

We first compare the total execution time with RFM-PostfixSpan and PrefixSpan based on the seven synthetic datasets. We set the $Rminsup=300$, $Fminsup=(# of patterns)\times0.01$, $Mminsup=6,000,000$, $ms_length=180$, decay speed $\delta=0.01$ and $t_{current}=500$ in all tests. In the first test, we vary the value of $|D|$ from 250K to 750K and the result in Figure 2a show that both RFM-PostfixSpan and PrefixSpan scale up linearly with $|D|$. We also vary the value of $|C|$ from 5 to 15 (as shown in Figure 2b) and the value of $|T|$ from 1.25 to 3.75 (as shown in Figure 2c). Both two tests indicate that the runtime of both algorithms scale up exponentially with $|C|$ and $|T|$. The results satisfy our expectations. It is because increasing $|C|$ and $|T|$ leads to the increase of a customer sequence length as well as the increase of their purchase times and money spent. The method without considering this particular piece of information will generate huge amount of patterns and the runtime increases dramatically as well. Instead, if more constraints are added such as RFM-PostfixSpan, more uninteresting patterns are removed and thus fewer patterns have to be processed and the runtime decreased.
Next, we use several tests to evaluate the scale-up effect for three support threshold unique to RFM-SPs based on SC-POS dataset. The parameters settings on \( ms\_length \), decay speed and \( t_{\text{current}} \) in this test are identical to those in the previous tests. We only vary one support threshold and keep the other two constant. The results are shown in Table 6. Except the result of varying \( F_{\text{minsup}} \), we can find that as \( R_{\text{minsup}} \) or \( M_{\text{minsup}} \) increases, both the runtime and the number of RFM-SPs decrease. The results also show that monetary constraint can significantly reduce the number of patterns. It is quite useful if a business focus on discovering high-profit patterns. The reason why the effect of varying \( F_{\text{minsup}} \) is insignificant may be that our collected dataset has relatively less repeated purchasing behaviour. In summary, through setting adequate recency, frequency and monetary thresholds simultaneously, we can use less time to get more compact, representative and useful patterns.

<table>
<thead>
<tr>
<th>( R_{\text{minsup}} )</th>
<th>( F_{\text{minsup}} )</th>
<th>( M_{\text{minsup}} )</th>
<th>Runtime</th>
<th>#of RFM-SPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.03</td>
<td>3000000</td>
<td>158</td>
<td>39</td>
</tr>
<tr>
<td>150</td>
<td>0.03</td>
<td>3000000</td>
<td>110</td>
<td>32</td>
</tr>
<tr>
<td>200</td>
<td>0.03</td>
<td>3000000</td>
<td>72</td>
<td>21</td>
</tr>
<tr>
<td>100</td>
<td>0.02</td>
<td>3000000</td>
<td>160</td>
<td>39</td>
</tr>
<tr>
<td>100</td>
<td>0.04</td>
<td>3000000</td>
<td>155</td>
<td>39</td>
</tr>
<tr>
<td>100</td>
<td>0.06</td>
<td>3000000</td>
<td>135</td>
<td>39</td>
</tr>
<tr>
<td>100</td>
<td>0.03</td>
<td>2000000</td>
<td>159</td>
<td>74</td>
</tr>
<tr>
<td>100</td>
<td>0.03</td>
<td>4000000</td>
<td>159</td>
<td>22</td>
</tr>
<tr>
<td>100</td>
<td>0.03</td>
<td>6000000</td>
<td>158</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 6. The results of runtime and the # of RFM-SPs using SC-POS.
CONCLUSION

Sequential pattern mining is a useful method to discover customer purchasing behaviour from large sequence databases. In this study, we propose a novel sequential pattern mining technique, which considers recency, frequency and monetary constraints, based on the concept of RFM analysis. A new type of sequential pattern, named RFM-SP, is defined, and an efficient projected-based algorithm, called RFM-PostfixSpan, is proposed to discover complete set of RFM-SPs from a sequence database.

Seven synthetic datasets and a real-life dataset are used in our experiments. The results show that the proposed method is efficient and outperforms the traditional PrefixSpan algorithm in both the runtime and numbers of generated patterns. In practical, the RFM-PostfixSpan can not only significantly reduce the runtime, but also retain more meaningful results to users.

There are some possible extensions of this paper. For example, it is possible to include fuzzy recency, frequency and monetary constraints, which would lead to a more flexible way to uncover other meaningful patterns. Furthermore, developing a desirable maintenance mechanism is critical for user to properly tune the parameters in the mining process.

Acknowledgement

This research was supported by the National Science Council of the Republic of China under the Grant NSC 99-2410-H-194-065.

References


