Engineering Applications of Artificial Intelligence 74 (2018) 242–251

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

An efficient approach for mining sequential patterns using multiple threads on very large databases

Bao Huynh [a,b], Cuong Trinh [c,d], Huy Huynh [c,d], Thien-Trang Van [e], Bay Vo [e,f,*], Vaclav Snasel [d]

[a] Center for Applied Information Technology, Ton Duc Thang University, Ho Chi Minh City, Viet Nam
[b] Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Viet Nam
[c] Department of Computing and Computer Services, Ton Duc Thang University, Ho Chi Minh City, Viet Nam
[d] Faculty of Electrical Engineering and Computer Science, VŠB-Technical University of Ostrava, Ostrava-Poruba, Czech Republic
[e] Faculty of Information Technology, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Viet Nam
[f] College of Electronics and Information Engineering, Sejong University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Sequential patterns
Multi-core processors
Multi-threading
Early pruning

ABSTRACT

Sequential pattern mining (SPM) plays an important role in data mining, with broad applications such as in financial markets, education, medicine, and prediction. Although there are many efficient algorithms for SPM, the mining time is still high, especially for mining sequential patterns from huge databases, which require the use of a parallel technique. In this paper, we propose a parallel approach named MCM-SPADE (Multiple threads CM-SPADE), for use on a multi-core processor system as a multi-threading technique for SPM with very large database, to enhance the performance of the previous methods SPADE and CM-SPADE. The proposed algorithm uses the vertical data format and a data structure named CMAP (Co-occurrence MAP) for storing co-occurrence information. Based on the data structure CMAP, the proposed algorithm performs early pruning of the candidates to reduce the search space and it partitions the related tasks to each processor core by using the divide-and-conquer property. The proposed algorithm also uses dynamic scheduling to avoid task idling and achieve load balancing between processor cores. The experimental results show that MCM-SPADE attains good parallelization efficiency on various input databases.

1. Introduction

Sequential pattern mining (SPM) plays an important role in data mining, which was first introduced by Agrawal and Srikant (1995). Sequential pattern mining has multiple applications, such as market basket analysis (Olson and Delen, 2008; Mostafa, 2015; Olson, 2016), web log analysis (Grace et al., 2011; Wang, 2012; Vemulapalli and Shashi, 2013), DNA sequence analysis (Posada, 2009; Pareek et al., 2011; Wilson et al., 2016) and prediction (Hussein et al., 2015; Chakraborty et al., 2016; Norouzi et al., 2016). The aim of this approach is to find all frequent sequential patterns with supports that satisfy the user-defined minimum support threshold. The main challenge of this is to generate candidate patterns in huge datasets with the minimum computational cost.

There are many efficient algorithms for mining SPM and these can be categorized as those using a horizontal database format, including GSP (Agrawal and Srikant, 1996; Zhang et al. 2002), AprioriAll (Agrawal and Srikant, 1995) and PrefixSpan (Pei et al., 2004) or a vertical database format, including SPADE (Zaki, 2001a), PRISM (Gouda et al., 2010) and CM-SPADE (Fournier-Viger et al., 2014).

However, the mining times of these algorithms are still high because they must explore a huge search space, which is computationally very expensive. In addition, a long sequence includes a combinatorial number of subsequences and sequential pattern mining creates an explosive number of frequent subsequences for long patterns, which is prohibitively costly in both time and memory requirements, especially for very large or dense databases.

Parallel processing has been widely applied to improve processing speed for various problems. While multi-core architectures have been used to speed up processing time, but they have been rarely applied to SPM. Methods based on multi-core architectures include PGP-mc, which uses parallel gradual pattern extraction (Laurent et al., 2012), GapMis-OMP, which is a tool for pairwise short-read alignment (Flouri et al., 2012), SW (Smith–Waterman), which compares sequence lengths

* Corresponding author at: Faculty of Information Technology, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Viet Nam.
E-mail addresses: huynhquocbao@tdt.edu.vn (B. Huynh), trinhphicuong@tdt.edu.vn (C. Trinh), huynhminhhuy@tdt.edu.vn (H. Huynh), vtt.trang@hutech.edu.vn (T.-T. Van), vd.bay@hutech.edu.vn (B. Vo), vaclav.snasel@vsb.cz (V. Snasel).

https://doi.org/10.1016/j.engappai.2018.06.009
Received 16 June 2017; Received in revised form 17 May 2018; Accepted 25 June 2018

(Sánchez et al., 2010), PIB-PRISM (Huynh and Vo, 2015), pDBV-SPM (Huynh et al., 2017) and piISP-IC (Le et al., 2018), which use multi-core processors for SPM.

Multi-core systems have placed increasing pressure on system programmers as well as application developers to make efficient use of the multiple computing cores. The related challenges include determining how to divide applications into separate tasks and minimizing CPU idle time. These must be balanced such that each task carries out an equal amount of work. Just as tasks must be separated, data must also be divided so that it can be accessed by the tasks running on separate cores.

The present study proposes a parallel approach called MCM-SPADE (Multiple threads CM-SPADE) based on a multi-core processor architecture and data structure named CMAP (Fournier-Viger et al., 2014). The proposed algorithm overcomes the drawbacks of CM-SPADE (Fournier-Viger et al., 2014) and further reduces the computational cost of mining. The main contributions of MCM-SPADE are reducing the execution time and memory usage when working with very large databases and this is done as follows:

1. Early pruning of unpromising candidates based on the CMAP data structure, a very compact data structure with a single database scan.
2. Using dynamic scheduling to avoid task idling and achieve load balancing of the workload between processor cores.
3. Fast mining is achieved based on a multi-core architecture system with the multi-threading technique.
4. The experiments carried out in this work show the greater efficiency of MCM-SPADE compared to SPADE, M-SPADE, and CM-SPADE.

The rest of this paper is organized as follows. Section 2 presents the basic concepts, while Section 3 reviews related works. Section 4 summaries the multi-core processor architecture and the MCM-SPADE algorithm is proposed in Section 5. The experimental results are discussed in Section 6. Finally, conclusions and ideas for future work are given in Section 7.

2. Preliminaries

Definition 1 (*Item*). Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of m distinct items. For example, assume $I = \{bread, milk, cheese, butter, cereal\}$, then the items are *bread*, *milk*, *cheese*, *butter*, *cereal*.

Definition 2 (*Itemset*). An itemset $X = (x_1, x_2, \ldots, x_k)$, where $x_j \in I$ $(1 \le j \le k)$, is a non-empty unordered collection of items denoted as a k-itemset. Each itemset is given in brackets, for itemsets that contain only a single item, the brackets are omitted. Without loss of generality, it is assumed that the items in an itemset are sorted in increasing order. For example, the itemset (ABC) represents the sets of items A, B, C and itemset (B) can be written as B.

Definition 3 (*Sequence*). A sequence $\alpha = \langle a_1 a_2 \cdots a_u \rangle$ is a non-empty ordered list of itemsets, where itemset a_1 occurs before a_2, which occurs before a_3 and so on. A sequence element a_i is an itemset and u (the size of a sequence) is the number of itemsets (or elements) in the sequence.

An item can occur at most once in an itemset of a sequence but can occur multiple times in different itemsets of a sequence. For example, sequence $\alpha = \langle CAA(AC) \rangle$ contains four itemsets. It indicates that item C was followed by A, then A and lastly are items A and C occurred at the same time.

Definition 4 (*Length of Sequence*). Given a sequence $\alpha = \langle a_1 a_2 \cdots a_u \rangle$, the length of a sequence is the total number of items in the sequence, denoted as $k = \sum_j |a_j|$. A sequence with length k is called a k-sequence. For example, the sequence $\langle B(AC) \rangle$ is a 3-sequence of size 2.

Table 1
Example of a sequence database.

SID	Sequence
S_1	$\langle (AB)C(FG)GE \rangle$
S_2	$\langle (AD)CB(ABEF) \rangle$
S_3	$\langle ABFG \rangle$
S_4	$\langle B(FG) \rangle$

Definition 5 (*Sequence Database*). A sequence database is a set of sequences, denoted $D = \{S_1, S_2, \ldots, S_n\}$, where n is the number of sequences in D and each sequence S_i $(1 \le i \le n)$ has the form $\langle sid, X \rangle$, where each *sid* is a unique sequence identifier and $X = \langle I_1 I_2 \cdots I_v \rangle$, such that $I_j \subseteq I$ $(1 \le j \le v)$ is an ordered list of itemsets. A pattern is a subsequence of ordered itemsets. Each itemset in a pattern is called an element. For example, Table 1 contains four sequences having the SIDs S_1, S_2, S_3 and S_4. The first sequence $S_1 = \langle (AB)C(FG)GE \rangle$ contains five itemsets. This indicates that items A and B occurred at the same time and were followed by C, then F, G and lastly E.

Definition 6 (*Subsequence and Super Sequence*). A sequence $\beta = (b_1 b_2 \cdots b_m)$ is called a subsequence of sequence $\alpha = (a_1 a_2 \cdots a_n)$ and α is a super sequence of β, denoted as $\beta \subseteq \alpha$, if there exist integers $1 \le j_1 < j_2 < \cdots < j_n \le m$ such that $b_k \subseteq a_{jk}$, $1 \le k \le n$. For example, the sequence $\langle B(AC) \rangle$ is a subsequence of $\langle (AB)E(ACD) \rangle$ but $\langle (AB)E \rangle$ is not a subsequence of $\langle ABE \rangle$.

Definition 7 (*Support*). The absolute support of a sequence p in a database D is defined as the total number of sequences in D that contain p, denoted as $sup(p) = |\{S_i \in D | p \subseteq S_i\}|$. The relative support of p is given as the fraction of sequences that contain p in a dataset. Absolute and relative supports are sometimes used interchangeably.

Definition 8 (*Sequential Pattern Mining*). Given a sequence database D and a user-specified threshold, called the minimum support (denoted *minsup*), a sequence p is said to be a frequent sequential pattern if it occurs more than *minsup* times. That is, $sup\ (p) \ge minsup$. The problem of SPM is to find all sequential patterns in the database D.

Definition 9 (*Maximal Sequential Pattern*). A sequential pattern is *maximal* if it is not a subsequence of any other sequential patterns.

Definition 10 (*Closed Sequential Pattern*). A sequential pattern is *closed* if it is not a subsequence of any other sequential patterns with the same support.

Definition 11 (*Prefix*). A sequence $\alpha = \langle a_1, a_2, \ldots, a_m \rangle$ is called a prefix of a sequence $\beta = \langle b_1, b_2, \ldots, b_n \rangle$ if and only if $m < n$ *and* $a_i = b_i \forall i \in [1, m]$.

Definition 12 (*Extending the Length of a Sequence*). A sequence $\alpha = \langle a_1, a_2, \ldots, a_m \rangle$ and an event e_k. Sequence β is called the extending length of the sequence α with the event e_k if and only if $\beta = \langle a_1, a_2, \ldots, a_m, a_{m+1} \rangle$ and $a_{m+1} = e_k$.

Definition 13 (*Horizontal Database Format*). A sequence database in the horizontal format is a database where each row is a transaction in the form *sid-itemset*, where *sid* is a sequence ID, and itemset is a set of items that appear in that sequence, Table 1 shows a horizontal sequence database.

Definition 14 (*Vertical Database Format*). A sequence database in vertical format is a database where each row has a transaction in the form $\langle item, sid \rangle$, where *sid* is a set of sequence IDs containing *item*. Table 2 shows the vertical representation of the database of Table 1.

Definition 15 (*s-extension*). The s-extension of a sequential pattern $\langle I_1, I_2, \ldots, I_n \rangle$ by item X is defined as $\langle I_1, I_2, \ldots, I_n, X \rangle$.