Engineering Applications of Artificial Intelligence 67 (2018) 197–210

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Efficient algorithms for simultaneously mining concise representations of sequential patterns based on extended pruning conditions

Hai Duong [a,b], Tin Truong [a,*], Bac Le [b]

[a] Department of Mathematics and Computer Science, University of Dalat, Dalat, Viet Nam
[b] VNU-HCMC, University of Natural Sciences, Ho Chi Minh, Viet Nam

ARTICLE INFO

Keywords:
Sequential pattern mining
Frequent sequence
Maximal frequent sequence
Frequent closed sequence
Frequent generator sequences
Vertical data format

ABSTRACT

The concise representations of sequential patterns, including maximal sequential patterns, closed sequential patterns and sequential generator patterns, play an important role in data mining since they provide several benefits when compared to sequential patterns. One of the most important benefits is that their cardinalities are generally much less than the cardinality of the set of sequential patterns. Therefore, they can be mined more efficiently, use less storage space, and it is easier for users to analyze the information provided by the concise representations. In addition, the set of all maximal sequential patterns can be utilized to recover the complete set of sequential patterns, while closed sequential patterns and sequential generators can be used together to generate non-redundant sequential rules and to quickly recover all sequential patterns and their frequencies. Several algorithms have been proposed to mine the concise representations separately, i.e., each of them has been designed to discover only a type of the concise representation. However, they remain time-consuming and memory intensive tasks. To address this problem, we propose three novel efficient algorithms named FMaxSM, FGenCloSM and MaxGenCloSM to exploit only maximal sequential patterns, to simultaneously mine both the sets of closed sequential patterns and generators, and to discover all three concise representations during the same process. To our knowledge, MaxGenCloSM is the first algorithm for concurrently mining the three concise representations of sequential patterns. The proposed algorithms are based on two novel local pruning strategies called LPMAX and LPMaxGenClo that are designed to prune non-maximal, non-closed and non-generator patterns earlier and more efficiently at two and three successive levels of the prefix tree without subsequence relation checking. Extensive experiments on real-life and synthetic databases show that FMaxSM, FGenCloSM and MaxGenCloSM are up to two orders of magnitude faster than the state-of-the-art algorithms and that the proposed algorithms consume much less memory, especially for low minimum support thresholds and for dense databases.

1. Introduction

Mining frequent sequences or sequential patterns, since this concept was first proposed by Agrawal (Agrawal and Srikant, 1995) and Srikant (Srikant and Agrawal, 1996), has become an essential data-mining task. It has attracted the interest of numerous researchers, as it has a wide range of applications such as the analysis of Web click streams, customer-transaction sequences, medical data and e-learning data. The threshold to determine whether a sequence is frequent is called *minimum support* (denoted as *minsupp*). The problem of mining the set $\mathcal{FS}$ of frequent sequences is to discover all subsequences appearing frequently in a sequence database, i.e., the frequency of each subsequence is no less than *minsupp*. Until now, several efficient algorithms have been proposed for mining frequent sequences such as PrefixSpan (Pei et al., 2004), SPADE (Zaki, 2001), SPAM (Ayres et al., 2002), CM-SPADE and CM-SPAM (Fournier-Viger et al., 2014b). However, an important drawback of these algorithms is that they can generate a large number of patterns, especially on large, dense databases containing many long sequences or when a low minimum support threshold is used. Consequently, it is often inconvenient and time-consuming for humans to analyze the patterns found. It can also cause the poor performance of the algorithms in terms of time and memory or can make the algorithms fail to finish in a reasonable time and run out of memory.

* Corresponding author.
E-mail addresses: haidv@dlu.edu.vn (H. Duong), tintc@dlu.edu.vn (T. Truong), lhbac@fit.hcmus.edu.vn (B. Le).

https://doi.org/10.1016/j.engappai.2017.09.024
Received 10 March 2017; Received in revised form 26 July 2017; Accepted 25 September 2017
0952-1976/

To address this issue, many studies focus on discovering condensed representations of frequent sequences that summarize $\mathcal{FS}$. Therefore, instead of mining all frequent sequences, only a concise and representative subset is extracted.

Three main condensed representations of frequent sequences that have been proposed are *frequent maximal sequences* (Fournier-Viger et al., 2014a, 2013; Luo and Chung, 2005), *frequent closed sequences* (Bac et al., 2017; Fournier-Viger et al., 2014b; Gomariz et al., 2013; Wang et al., 2007; Yan et al., 2003) and *frequent generator sequences* (Fournier-Viger et al., 2014d; Gao et al., 2008; Lo et al., 2008; Pham et al., 2012; Yi et al., 2011). A frequent maximal sequence is a frequent sequence that is not strictly contained in another frequent sequence. The set $\mathcal{FMS}$ of frequent maximal sequences is generally a very small subset of $\mathcal{FS}$, thus greatly reducing the number of patterns presented to the user. Moreover, $\mathcal{FMS}$ can be used to recover the full set of $\mathcal{FS}$, and the exact frequency of subsequences of each maximal sequence in $\mathcal{FMS}$ can also be computed with an additional database scan. A frequent sequence is said to be closed if there no exists another super-sequence appearing in the same set of input sequences. A frequent sequence is said to be a generator if there no exists a proper subsequence having the same support. The cardinalities of the set $\mathcal{FCS}$ of all frequent closed sequences and the set $\mathcal{FGS}$ of all frequent generator sequences are generally much smaller than $\mathcal{FS}$, but they are usually larger than $\mathcal{FMS}$. Generator sequences and closed sequences are, respectively, minimal and maximal members of equivalence classes of patterns. In this context, an *equivalence class* is a set of sequential patterns appearing in a same set of input sequences, where all patterns have the same support.

In this paper, we are interested in mining the $\mathcal{FMS}$, $\mathcal{FCS}$ and $\mathcal{FGS}$ sets for several reasons. First, these three sets provide the common benefit of having a very small cardinality compared to $\mathcal{FS}$; thus, they can be discovered more efficiently, they use less storage space, and it is easier for humans to analyze them (Fournier-Viger et al., 2014d). Second, each set has its own advantages. For the $\mathcal{FMS}$ set, it is the smallest of three sets and can be used to recover all patterns in $\mathcal{FS}$. Besides, frequent maximal sequences are used to discover the frequent longest common subsequences in texts, analyze DNA sequences, compress data, and mine Web logs (Garcia-Hernandez et al., 2006). For the $\mathcal{FCS}$ set, although it is larger than $\mathcal{FMS}$, $\mathcal{FCS}$ has the advantage of being lossless. Therefore, it keeps all the information regarding the frequency of all frequent sequences, unlike maximal sequences. In addition, frequent closed sequences have numerous applications. For $\mathcal{FGS}$, some researchers have argued that according to the Minimum Description Length principle (Grunwald et al., 2005; Li et al., 2006), frequent generator sequences should be preferred, because they are the minimal sequences of equivalence classes. Third, $\mathcal{FCS}$ and $\mathcal{FGS}$ can be combined to generate non-redundant sequential rules having a generator as the left-hand side and a closed pattern as the right-hand side (Lo et al., 2011; Minh-Thai et al., 2016; Pham et al., 2014) and to quickly recover all sequential patterns in equivalence classes and their supports.

The above reasons illustrate the benefits of discovering $\mathcal{FMS}$, $\mathcal{FCS}$, $\mathcal{FGS}$ and motivate the effort to improve existing algorithms for mining frequent maximal, closed, and generator sequences.

1.1. Related work

Several algorithms that have been recently proposed to discover frequent maximal sequences are AprioriAdjust (Lu and Li, 2004), MFSPAN (Guan et al., 2005), MaxSP (Fournier-Viger et al., 2013) and VMSP (Fournier-Viger et al., 2014a). The AprioriAdjust algorithm often produces a large amount of candidate sequences and needs many original database scans, as it is an apriori-like algorithm. MFSPAN requires maintaining a huge number of intermediate candidates generated during the mining process; thus, it is inefficient in terms of memory. The MaxSP algorithm uses a pattern-growth approach to avoid generating many candidates in main memory. However, it must repeatedly perform database projections, leading to long execution times. VMSP is the most recent algorithm that uses a vertical database format and incorporates three strategies, namely EFN (Efficient Filtering of Non-maximal patterns), FME (Forward-Maximal Extension checking) and CPC (Candidate Pruning by Co-occurrence map), to efficiently prune the search space with a single database scan. Nevertheless, VMSP may still generate many redundant candidates, i.e., the majority of candidates produced are not frequent maximal sequences, especially on *n-SDBs* and 1-*SDBs* with low minimum supports. Thus, VMSP spends a large amount of the time considering candidates that are not maximal sequences.

For closed sequential pattern mining, some algorithms have been proposed, which CloSpan (Yan et al., 2003) was the first algorithm. It uses a candidate maintenance-and-test approach. Since CloSpan needs to maintain many closed sequence candidates, it must check the subsequence relation between many sequences in the pruning step and the post-processing one, often leading to high memory consumption and long execution times. Moreover, in *n-SDBs*, it may provide an incomplete set of frequent closed sequences, as shown in Bac et al. (2017). The next algorithm designed to discover closed sequences is BIDE (Wang et al., 2007). Unlike the CloSpan algorithm, BIDE does not maintain candidates. It explores the search space by using a pattern-growth approach and employs a mechanism called bi-directional extension. However, BIDE can have long execution times, since it must generate numerous projected databases and scan them several times to discover closed sequential patterns. CM-ClaSP (Fournier-Viger et al., 2014b) is a closed sequence mining algorithm that is an improved version of ClaSP (Gomariz et al., 2013). CM-ClaSP uses a vertical database format and the BSPC/BPPC mechanisms presented in Yan et al. (2003). Furthermore, it utilizes co-occurrence information regarding pairs of consecutive items to reduce the search space. However, it remains time-consuming and memory intensive tasks.

Concerning sequential generator pattern discovering, the state-of-the-art algorithms are GenMiner (Lo et al., 2008), FEAT (Gao et al., 2008), FSGP (Yi et al., 2011), MSGPs (Pham et al., 2012) and VGEN (Fournier-Viger et al., 2014d). The first three algorithms use different techniques for storing patterns, pruning the search space, and identifying sequential generator patterns. However, all of them adopt a pattern-growth approach based on PrefixSpan (Pei et al., 2004), utilize a horizontal database, and perform database projections. They thus suffer from the problem of repeatedly scanning projected databases and performing database projections, leading to an enormous time cost. In contrast, the MSGPs algorithm only provides an insignificant performance improvement over previous algorithms, because it is mainly based on the definition of sequential generator patterns. VGEN is a vertical mining algorithm of frequent generators, using a technique named ENG (Efficient filtering of Non-Generator patterns) and co-occurrence information regarding pairs of items in sequences to prune the search space and identify generators. However, for low minimum support values, VGEN may generate many redundant candidates and a large amount of time is spent comparing a new candidate pattern with all smaller (w.r.t. $\sqsubseteq$) candidates previously generated. This check is costly.

Two algorithms for simultaneously mining frequent closed and generator sequences are SCGM (Zang et al., 2010) and CloGen (Thi-Thiet et al., 2013). In the mining process, both the algorithms adopt the candidate generation-and-test approach and only use the simple condition of the support in the definitions of closed and generator sequences to eliminate unpromising candidates. Thus, it also consumes a large amount of time and memory to produce and save many redundant candidates.

To overcome the drawbacks of the above algorithms for mining closed sequential patterns and sequential generator ones, in Bac et al. (2017), Bac et al. proposed a novel measure called *SE* that is computed for each projected database and two Extended Pruning (EP) conditions that are more general than the condition of the mechanism "*early termination by equivalence*" in Yan et al. (2003) for exactly pruning non-closed or non-generator sequences and their descendant branches in the