IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 15, Issue 4 (Nov. - Dec. 2013), PP 20-23
www.iosrjournals.org

Mining Top-k Closed Sequential Patterns in Sequential Databases

K.Sohini[1], Mr.V.Purushothama Raju [2]
[1] *Dept of CSE, Shri Vishnu Engineering College for Women, Bhimavaram, A.P, India*
[2] *Associate Professor, Dept of CSE, Shri Vishnu Engineering College for Women, Bhimavaram, A.P, India*

***Abstract**: In data mining community, sequential pattern mining has been studied extensively. Most studies require the specification of minimum support threshold to mine the sequential patterns. However, it is difficult for users to provide an appropriate threshold in practice. To overcome this, we propose mining top-k closed sequential patterns of length no less than min_l, where k is the number of closed sequential patterns to be mined, and min_l is the minimum length of each pattern. We mine closed patterns since they are solid representations of frequent patterns.*
***Keywords:** closed pattern, data mining, sequential pattern, scalability*

I. Introduction

Sequential pattern mining is an important data mining task that has been studied extensively. Given a set of sequences, which consists of a list of itemsets, and given a user-specified minimum support threshold (min_support), sequential pattern mining is to find all frequent subsequences whose frequency is no less than min support. This mining task leads to the following two problems that may hinder its popular use.

First, sequential pattern mining often generates an exponential number of patterns, which is unavoidable when the database consists of long frequent sequences. The similar phenomena also exist in itemset and graph patterns when the patterns are large. For example, assume the database contains a frequent sequence $<(a_1)\,(a_2)\ldots(a_{100})>$, it will generate 2^{100}-1 frequent subsequences. It is very possible some subsequences share the exact same support with this long sequence, which are essentially redundant patterns.

Second, setting *min_support* is a subtle task: A too small value may lead to the generation of thousands of patterns, whereas a too big one may lead to no answer found. To come up with an appropriate *min_support*, one needs to have prior knowledge about the mining query and the task specific data, and be able to guess beforehand how many patterns will be generated with a particular threshold.

A solution to the first problem was proposed recently by Yan, et al. [1]. Their algorithm, called CloSpan, can mine closed sequential patterns. A sequential pattern s is closed, if there exists, no super pattern of s with the same support in the database. When the patterns were mined for closed sequences and information lossless then the Patterns will be reduced, because it can be used to derive the complete set of sequential patterns.

As to the second problem, a similar situation occurs in frequent itemset mining. As proposed in [3], a good solution is to change the task of mining frequent patterns to mining top-k frequent closed patterns of minimum length *min_l*, where k is the number of closed patterns to be mined, top-k refers to the k most frequent patterns, and *min_l* is the minimum length of the closed patterns. This setting is also desirable in the context of sequential pattern mining. Unfortunately, most of the techniques developed in [3] cannot be directly applied in sequence mining, because this subsequence testing requires order matching which is more difficult than subset testing. Moreover, the search space of sequences is much larger than that of itemsets. Nevertheless, some ideas developed in [3] are still influential in our algorithm design.

II. Related Work

Efficient algorithms like PrefixSpan [5] SPADE[6] GSP[7] were developed for mining the sequential patterns. As sequential pattern mining produces many patterns so closed sequential pattern mining algorithms like CloSpan[1] , BIDE[8] were developed. All these algorithms can deliver less patterns than sequential pattern mining, but do not lose any information. Top-k closed pattern mining will reduce the number of patterns further by only mining the most frequent ones. As to mining the top-k patterns, CloSpan[1] and TFP [3] are the most related. CloSpan mines the frequent closed sequential patterns while TFP discovers top-k closed itemsets.

III. Method Development

The sequential pattern mining based on the concept of projection based, is introduced (PrefixSpan[5]) it gives some background idea. Then a new Top-k mining algorithm was introduced with background of CloSpan and top-k.

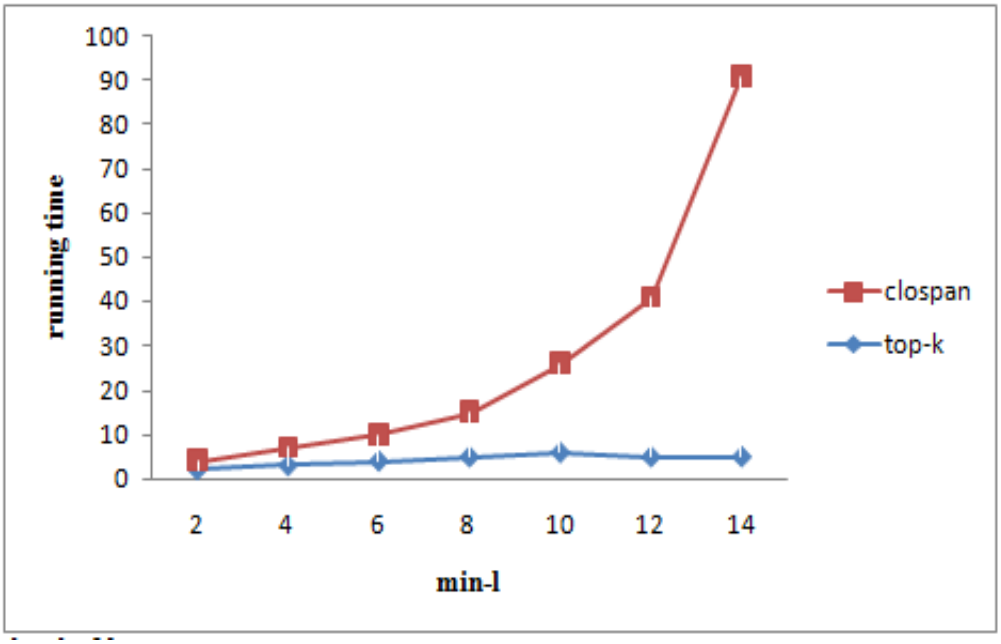
For each discovered sequence s and its projected database D_s, it performs itemset extension and sequence-extension recursively until all the frequent sequences with prefix s are discovered.
Given a sequence $s=<t_1,\ldots..,t_m>$ and an item a, $s\varphi a$ means s concatenates with a
It can be I-Step extension ,$s\varphi_i a =<t_1,\ldots..,t_m U\{a\}>$
Or S-Step extension, $s\varphi_s a =<t_1,\ldots..,t_m,\{a\}>$
Example:<(ae)> is an I-Step extension of <(a)>
<(a)(c)> is an S-Step extension of <(a)>

Algorithm: Top-k mining
Input: A sequence s, a projected database D_s, minimum length *min_l*, histograms H and constant factor f
Output: The top-k closed sequence set k
1. if support of s is less than min_support then return
2. if length of s is equal to minimum length then
3. call Prefix Span With Support Raising (s, D_s, min_support, k) ;
4. return;
5. scan the database D_s once and find every frequent item a such that s can be extended to $s\varphi a$;
6. insert a in histogram at length (l+1);
7. next_level_top_support ← Get Top Support From Histogram (f, H[l(s)+1])
8. for each a, support(a)>=next_level_top_support do
9. call Top-k mining ($s\varphi a$, $D_{s\varphi a}$, min_1);
10. return;

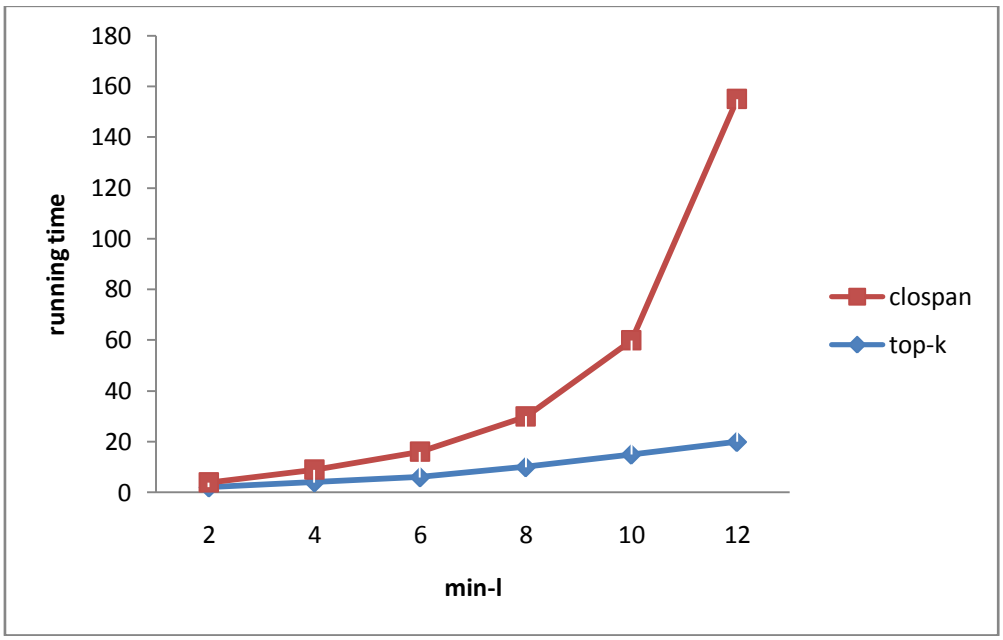
In the prefix span it finds all the frequent sequences, they include both closed and non-closed sequences. Since our task is to mine top-k closed sequential patterns without min_support threshold, the mining process should start with min_support = 1, raise it progressively during the process, and then use the raised min_support to prune the search space. As soon as at least k closed sequential patterns with length no less than *min_l* are found, min_support can be set to the support of the least frequent pattern, and this min_support raising process continues throughout the mining process. This min_support raising technique is simple and can lead to efficient mining. There is only one algorithm CloSpan[1], that guarantees the slightest k closed sequential patterns are found as a result min_support can be raised during the mining.
In Top-k mining, it includes if the support of s is less than min_support then return, if the length of s is equal to *min_l* then call Prefix Span With Support Raising. Then scan the database D_s once and find the closed sequence so that s can be extended to $s\varphi a$. Then insert a into histogram, finally if the support of a is greater or equal to next level top support then call top-k mining.

IV. Performance evaluation

This section reports the performance testing of Top-k closed sequential patterns in large data sets. The performance of Top-k with CloSpan is compared. The comparison is by assigning the optimal min-support to CloSpan so that it generates the same set of top-k closed sequential patterns for specified values of k.
These were performed on a 2.10GHz Intel core duo PC with 2GB main memory, Windows XP/7 Professional.
The performance of algorithms was compared by varying k. When k is fixed, its value is set to either 50 or 500 which covers the range of typical values for this parameter. Fig.1, show the performance of sample dataset. This dataset consists of relatively short sequences, each sequence contains 6 itemsets on average and the itemsets have 3 items on average. This experiment shows that top-k mines the data efficiently and with minimum running time than CloSpan. Mainly there are two reasons for better performance of Top-k in this dataset, first it uses min_l condition to reduce short sequences during the mining process that will reduce the search space and improves performance. Second, Top_k have efficient pattern verification and stores the result set that contains only a small number of patterns.



a. when k=50



b. when k=500

Fig. 1: performance of clospan and top-k when k is fixed.

These were the graphs obtained when compared with sample database on clospan and top-k with constant min-l and running time.

V. Conclusions

In this paper, we have studied the problem of mining top-k closed sequential patterns with length no less than *min_l* and we proposed an efficient algorithm with the following distinct features: it implement a new Top-k algorithm that mine the sequential patterns quickly and raise support dynamically, and it perform efficient verification during the mining process, and it extend optimization technique together with applying the minimum length constraint (min_l).

References

[1] X. Yan, R. Afshar, and J. Han. CloSpan: Mining closed sequential patterns in large datasets. In May 2003, *SDM*, in California.
[2] C. J. Hsiao and M. J. Zaki. CHARM: An efficient algorithm for closed itemset mining. In April 2002, *SDM*, in Arlington,VA.
[3] J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining topk frequent closed patterns without minimum support. In Dec. 2002, ICDM, in Maebashi, Japan.
[4] R. Agrawal and R. Srikant. Mining sequential patterns. In Mar. 1995, ICDE, in Taipei, Taiwan.
[5] J. Pei, J. Han, B. Mortazavi-Asl, M.C. Hsu , Q. Chen,U. Dayal, and H. Pinto. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In April 2001, ICDE, in Germany.
[6] M. Zaki. In 2001, SPADE: An efficient algorithm for mining frequent sequences.
[7] R.Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In Avignon, France, Mar. 1996.
[8] J.Wang and J. Han. BIDE, Efficient Mining of Frequent Closed Sequences, in 2004, ICDE.