Top K Sequential Pattern Mining Algorithm

Karishma B Hathi¹, Jatin R Ambasana²
¹Student, M.E.(CSE), Gardi Vidyapith, Gujarat, India
²Assistant Professor, CSE Department, Gardi Vidyapith, Gujarat, India

ABSTRACT: Sequential pattern mining is a very chief mining technique with wide applications. Still, tune up the minsup parameter of sequential pattern mining algorithms to produce enough patterns is complex and time-consuming. To solve this problem, the assignment of top-k sequential pattern mining has been described, where k is the number of sequential patterns to be discovered, and is set by the user. In this paper, we present proposed approach for improving parameters in TKS Algorithm.

KEYWORDS: Sequential Patterns, Top K, Sequence Database, Pattern mining.

I. INTRODUCTION

The sequential pattern mining is a very important concept of data mining, a further extension to the concept of association rule mining [1]. That has a huge range of real-life application. This mining algorithm solves the problem of discovering the presence of frequent sequences in the given database [2]. Sequential Pattern Mining finds interesting sequential patterns among the huge database. It discovers frequent subsequences as patterns from a given sequence database. It is a well-understood data mining problem with broad applications such as the analysis of web clickstreams, program executions, medical data, biological data and e-learning data [1, 5]. Although many studies have been done on constructing sequential pattern mining algorithms [1, 2, 3, 4], the main problem is how the user should choose the minsup threshold to produce a desired amount of patterns. This problem is important because in practice, users have limited resources (time and storage space) for discovering the results and thus are often only interested in analyzing a certain amount of patterns, and fine-tuning the minsup parameter is very time-consuming. Depending on the choice of the minsup threshold, algorithms can become very slow and produce an extremely huge amount of results or generate none or too few results, getting valuable information. To address this difficulty, it was proposed to redefine the problem of sequential pattern mining as the problem of mining the top-k sequential patterns, where k is the number of sequential patterns to be discovered and is set by the user.

II. RELATED WORK

The problem of sequential pattern mining was proposed by Agrawal and Srikant [2] and is defined as follows. A sequence database SDB is a set of sequences $S = \{s_1, s_2, \ldots, s_l\}$ and a set of items $I = \{i_1, i_2, \ldots, i_m\}$ happening in these sequences. An item is a symbolic value. An itemset $I = \{i_1, i_2, \ldots, i_l\}$ is an unordered set of different items. For example, the itemset $\{a, b, c\}$ shows the sets of items $a$, $b$ and c. A sequence is an ordered list of itemsets $S = < I_1, I_2, I_3, \ldots, I_l >$ such that $Ik \subseteq I_k$ for all $1 \leq k \leq n$. For example having a sequence the sequence database SDB depicted in Figure 1. It contains mainly four sequences having accordingly the sequences ids (SIDs) 1, 2, 3 and 4. In this example, each solo letter represents an item. Items between curly brackets describes an itemset. For instance, the first

This paper is prepared exclusively for International Conference on Information Engineering, Management and Security 2015 [ICIEMS] which is published by ASDF International, Registered in London, United Kingdom. Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honoured. For all other uses, contact the owner/author(s). Copyright Holder can be reached at copy@asdf.international for distribution.

2015 © Reserved by ASDF.international

sequence \( \{a, b, c, f, g, e\} \) shows that items \( a \) and \( b \) happened at the same time, were followed successively by \( c, f, g \) and lastly \( e \). A sequence \( sa = <A1, A2, A3, ..., An> \) is called to be included in another sequence \( sb = <B1, B2, B3, ..., Bm> \) if and only if there exists integers \( 1 \leq I_1 < I_2 < ... < I_n \leq I_m \) such that \( A1 \subseteq B1, A2 \subseteq B2, A3 \subseteq B3, ... , An \subseteq Bm \). The support of a subsequence \( sa \) in a sequence database SDB is described as the number of sequences \( s \in SDB \) such that \( sa \preceq s \) and is denoted by \( \text{sup}(sa) \). The problem of mining sequential patterns in a sequence database SDB is to locate all frequent sequential patterns, i.e. each subsequence \( sa \) such that \( \text{sup}(sa) \geq \text{minsup} \) for a threshold \( \text{minsup} \) set by the user. For example, Figure 2 displays five of the 29 sequential patterns found in the database of table Figure 1 for \( \text{minsup} = 2 \). Many algorithms have been proposed for the problem sequential pattern mining such as PrefixSpan [3], SPAM [4], GSP and SPADE [6].

### III. THE BASIC TKS ALGORITHM

TKS, an algorithm to find the top-\( k \) sequential patterns having the highest support, where \( k \) is set by the user. TKS employs the vertical database representation and basic candidate-generation procedure of SPAM [8]. Furthermore, it also includes various efficient strategies to find top-\( k \) sequential pattern efficient Fine-tuning the minsup parameter of sequential pattern mining algorithms to generate enough patterns is hard and time-consuming. To address this problem, the task of top-\( k \) sequential pattern mining has been defined, where \( k \) is the number of sequential patterns to be found, and is set by the user. So here an efficient algorithm for this problem named TKS (Top-K Sequential pattern mining) is present. TKS utilizes a vertical bitmap database representation, a new data structure named PMAP (Precedence Map) and various efficient strategies to prune the search space. The experimental study on real datasets shows that TKS outperforms TSP, the current state-of-the-art algorithm for top-\( k \) sequential pattern mining by more than an order of magnitude in execution time and memory.

### TKS Algorithm [9]

It takes as parameters a sequence database SDB and \( k \).

1) It first scans SDB once to construct \( V(SDB) \).

2) Let $Sinit$ be the list of items in $V(SDB)$
3) Then, for each item $s \in Sinit$, if $s$ is frequent according to $bv(s)$ it calls the procedure "SAVE".
4) $R = RU \{s, Sinit, \text{items from } Sinit \text{ that are lexically larger than } s\}$
5) WHILE $\exists <r, S1, S2> \in R \text{ AND sup}(r) \geq \minsup$ DO
6) Select the tuple $<r, S1, S2>$ having the pattern $r$ with the highest support in $R$
7) Then calls "SEARCH" find tuple.
8) Finally calls “REMOVE” and delete infrequent patterns from database.

IV. THE PROPOSED ALGORITHM

Limitation of basic TKS algorithm is number of database scan are higher so execution time grows higher due to this limitation. For improving the efficiency of TKS algorithm and overcome the drawbacks of TKS we propose an efficient approach for mining top k sequential patterns. We can improve the efficiency of TKS algorithm by using tree structure in TKS. By using this we can improve the efficiency of TKS algorithm in the terms of execution time.

### Proposed Algorithm

**Input:** SDB, K

**Output:** Top K Sequential Patterns

Let $qtemp$ be the list of items in tree

For each $q \in qtemp$

Save($q, I, k, \minsup$)

---

**Figure 3: Propose Flow**
If $i$-extension $\text{sup}(q) \geq \text{minsup}$
- Save($q$, all the items in $q_{temp}$ that are lexically larger than $q$, $L$, $k$)
- And
- If $s$-extension $\text{sup}(q) \geq \text{minsup}$
- Save($q$, all the items in $q_{temp}$ that are lexically larger than $q$, $L$, $k$)
- End if
- Remove $q_{temp}$ when $\text{sup}(q) < \text{minsup}$

End for
Return $L$

V. TOOL STUDY

Java Technology:
JAVA is an object oriented platform independent and middle level language. It contains JVM (Java Virtual Machine) which is able to execute any program more efficiently. The feature of Platform Independence makes it different from the other Technologies available today.

Eclipse Tool:

Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and thus can be used to develop applications. A vendor-neutral open-source workbench for multi-language development. An extensible platform for tool integration. Plug-in based framework to create, integrate and utilize software tools.

Sequential Pattern Mining Framework:

SPMF is an open-source data mining library written in Java, specialized in pattern mining. The source code of each algorithm can be integrated in other Java software. Moreover, SPMF can be used as a standalone program with a simple user interface or from the command line. The current version is v0.96r16 and was released the 28th April 2015.

VI. EXPERIMENTAL RESULTS

Various parameters are used for sequential pattern mining. Here this proposed approach concludes parameter execution time. The execution time is depend on the number of patterns are found and the number of passes that requires for database scan. We compared the performance of Proposed Algorithm with TKS Algorithm. All algorithms were implemented in Java. Experiments were carried on three real-life datasets having varied characteristics and representing three different types of data (web click stream, text from books, sign language utterances). Those datasets are accordingly FIFA, Bible and Sign. The comparison between TKS and Proposed algorithm is shown in following figures.

Figure 4: Sign Dataset

VII. CONCLUSION AND FUTURE WORK

The Proposed System has improved the performance of TKS algorithm by using tree structure. From the result analysis it is clearly seen that the execution time of proposed algorithm reduced. There is less number of database scans required so the efficiency of the TKS algorithm improves. Thus, we conclude that the proposed system has better performance than TKS algorithm. Extending TKS algorithm for finding Top K Closed Sequential Pattern can be considered as future work.

ACKNOWLEDGEMENT

We are deeply indebted & would like to express gratitude to our thesis guide Prof. Jatin Ambasana, B. H. Gardi College of Engineering & Technology for his great efforts and instructive comments in the dissertation work.

We would also like to extend our gratitude to Prof. Hemal Rajyaguru, Head of the Computer Science & Engineering Department, B. H. Gardi College of Engineering & Technology for his continuous encouragement and motivation.

We would also like to extend our gratitude to Prof. Vaseem Ghada, PG Coordinator, B. H. Gardi College of Engineering & Technology for his continuous support and cooperation.

We should express our thanks to our dear friends & our classmates for their help in this research; for their company during the research, for their help in developing the simulation environment.

We would like to express our special thanks to our family for their endless love and support throughout our life. Without them, life would not be that easy and beautiful.

REFERENCES


