Global Journal of Computer Science and Technology
Volume 12 Issue 7 Version 1.0 April 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

OSSM: Ordered Sequence Set Mining for Maximal Length Frequent Sequences A Hybrid Bottom-Up-Down Approach

By Anurag Choubey, Dr. Ravindra Patel & Dr. J.L. Rana
Rajiv Gandhi Technological University, Bhopal, India

Abstract - The process of finding sequential rule is indispensable in frequent sequence mining. Generally, in sequence mining algorithms, suitable methodologies like a bottom–up approach is used for creating large sequences from tiny patterns. This paper proposed on an algorithm that uses a hybrid two-way (bottom-up and top-down) approach for mining maximal length sequences. The model proposed is opting to bottom-up approach called "Concurrent Edge Prevision and Rear Edge Pruning (CEG&REP)" for itemset mining and top-down approach for maximal length sequence mining. It also explains optimality of top-to-bottom approach in deriving maximal length sequences first and lessens the scanning of the dataset.

GJCST Classification: F.3.m

Strictly as per the compliance and regulations of:

OSSM: Ordered Sequence Set Mining for Maximal Length Frequent Sequences

A Hybrid Bottom-Up-Down Approach

Anurag Choubey [α], Dr. Ravindra Patel [σ] & Dr. J.L. Rana [ρ]

Abstract - The process of finding sequential rule is indispensable in frequent sequence mining. Generally, in sequence mining algorithms, suitable methodologies like a bottom–up approach is used for creating large sequences from tiny patterns. This paper proposed on an algorithm that uses a hybrid two-way (bottom-up and top-down) approach for mining maximal length sequences. The model proposed is opting to bottom-up approach called "Concurrent Edge Prevision and Rear Edge Pruning (CEG&REP)" for itemset mining and top-down approach for maximal length sequence mining. It also explains optimality of top-to-bottom approach in deriving maximal length sequences first and lessens the scanning of the dataset.

I. Introduction

Researchers feel enthusiastic on the sequential pattern mining problems and wide range of possibilities of applications regarding the envisaging of the customer buying patterns and scientific discoveries [1, 2, 3, 4, 5] discussed by Agrawal and Srikant [1]. Let us explain with an example like finding the given time stamped sequences of purchase made by a customer. In this example the main objective is to find sequence of same time stamped list of items purchased by the customer. So the algorithm of sequence pattern mining should concentrate on finding the repeated sequences which are called as frequent sequence. Such sequences list out the frequency of common occurrences. several heuristics like GSP [1], SPADE [3], Prefix Span [2] and the SPIRIT [4] attempt to find the frequent patterns in productive method by striving to cut short a series, hence decrease search space. The GSP algorithm [1] utilizes the anti-monotone property (all subsequences of a frequent sequence are also frequent).

The SPADE finds frequent sequences using the lattice search [3] and intersection based approach. In this particular method the sequence database is converted into a vertical format. The candidate sequences will be made into different groups. These frequent sequences will be listed in SPADE utilizing two methods namely breadth first method and depth first method. The base is calculated for the produced sequences. The approach of mentioned three algorithms can be grouped as the candidate-production with a base evaluation. The PrefixSpan [2] algorithm adopts growth method pattern. It utilizes recorded database to accomplish.. Prefix is Projected Sequential Pattern mining which checks prefix subsequences and includes the postfix sub sequences into the databases.

Author α : Dean Academic, Technocrats Institute of Technology, Bhopal M.P., India. E-mail: anuragphd11@gmail.com
Author σ : Associate Professor and Head, Department of Computer Applications at Rajiv Gandhi Technological University, Bhopal, India. Email: ravindra@rgtu.net
Author ρ : Group Director, Radha Raman Group of Institute, Bhopal. Email: jl_rana@yahoo.com

II. Related Work

The sequential item set mining problem was initiated by Agrawal and Srikant, and the same developed a filtered algorithm, GSP [1], based on the Apriori property [1]. Since then, lots of sequential item set mining algorithms are being developed to increase the efficiency. Some are SPADE [3], PrefixSpan[2], and SPAM [11]. SPADE[3] is on principle of vertical id-list format and it uses a lattice-theoretic method to decompose the search space into many tiny spaces, on the other hand PrefixSpan[2] implements a horizontal format dataset representation and mines the sequential item sets with the pattern-growth paradigm: grow a prefix item set to attain longer sequential item sets on building and scanning its database. The SPADE[3] and the PrefixSPan[2] highly perform GSP[1]. SPAM[11] is a recent algorithm used for mining lengthy sequential item sets and implements a vertical bitmap representation. Its observations reveal, SPAM[11] is more efficient in mining long item sets compared to SPADE[3] and PrefixSpan[2] but, it still takes more space than SPADE[3] and PrefixSpan[2]. Since the frequent closed item set mining [12], many capable frequent closed item set mining algorithms are introduced, like A-Close [12], CLOSET [13], CHARM [14], and CLOSET+ [15]. Many such algorithms are to maintain the ready mined frequent closed item sets to attain item set closure checking. To decrease the memory usage and search space for item set closure checking, two algorithms, TFP [17] and CLOSET+2, implement a compact 2-level hash indexed result-tree structure to keep the readily mined frequent closed item set candidates. Some pruning methods and item set closure verifying methods, initiated that can be extended for optimizing the mining

of closed sequential item sets also. CloSpan is a new algorithm used for mining frequent closed sequences [16]. It goes by the candidate maintenance-and-test method: initially create a set of closed sequence candidates stored in a hash indexed result-tree structure and do post-pruning on it. It requires some pruning techniques such as Common Prefix and Backward Sub-Itemset pruning to prune the search space as CloSpan[16] requires maintaining the set of closed sequence candidates, it consumes much memory leading to heavy search space for item set closure checking when there are more frequent closed sequences. Because of which, it does not scale well the number of frequent closed sequences.

III. Sequence, Sub Sequence and Frequent Sequences

We can say a sequence means an ordered set of events [1], set of events $S=\{s_1,s_2,s_3,....s_j,s_{j+1},.....s_n \mid \exists! s_i\, i=\{1,2,3,\ j,\ j+1,....n\}\}$ And every event s_i is considered as an item set, which is a non-empty, unordered, finite set of items, which can be represented as $s_i=\{i_1,i_2,i_3,.....,i_m \mid \exists! i_e\, e=\{1,2,3,4,....,m\}\}$, here $i_e\ for\ e=1..m$ is item. The length of a sequence is the number of items present in the sequence. A sequence can be referred with its length, as an example a sequence of length k is called a k-sequence. A sequence $S1$ is said to be a subsequence of another sequence $S2$ if and only if $s1_i \subseteq s2_{e_i}$ for $\mathrm{i}=\{1,2,3,\ldots.\mathrm{m} \mid e_1<e_2<e_3.....<e_i.....<i_n\}$ and e_i is event of $S2$ .The $S2$ is said to super sequence of $S1$ and $S1$ is said to subsequence of $S2$. The sequence database S is a set of the form (tid, s) where tid is the transaction-id and s is the sequence generated from transaction.

Let the minimum support as a threshold defined by user which indicates the desired minimum occurrences of the sequence to be claimed as frequent.

A sequence s_j is lengthiest if $s_j! \subseteq \{s_i \in S \mid i=\{1..m\}\}$. We explain the maximal support of an event e which consists of items $(i_1,i_2,...,i_k)$ as

$$MS_{e_i}=\max\{\sup(i_1),\sup(i_2),\sup(i_3),.....,\sup(i_m) \mid \exists! i_t\ t=\{1,2,3,....,m\}\}$$

Here $\sup(i_n)$ frequency of item i_n occurrence. And, the maximal support factor MSF of a sequence is represented by sum of maximal support of all events belongs to that sequence. Let s be a sequence of events $\{e_1,e_2,.....,e_m \mid \exists! e_i\ i=\{1,2,3,....,m\}\}$ and then maximal support factor can be measured as

$$MSF(s)=\sum_{i=1}^{m} MS(e_i)$$

The Maximal Support Factor is the threshold used in our proposal to minimize the subsequence search in top to bottom approach.

We apply an ordered search on sequence database, hence the sequence database will be ordered in descending manner by MSF of sequences. Then the search for super sequence of a given sequence is limited to the sequences with greater or equal MSF of the given sequence.

The preprocessed dataset with the sorted transaction list has several properties that can be used to cut short the search space which are hypothesized below:

- **Hypothesis 1:** A super sequence search of given sequence s_c can stop at a sequence s_t if $MSF(s_t)<MSF(s_c)$. Putting differently a candidate sub sequence s_c will not be a subsequence of s_i if $MSF(s_c)>MSF(s_i)$.
- **Hypothesis 2:** A sequence s_c is frequent if count of super sequences for s_c is equal or greater than the given minimum support threshold by user. `minsupport(sc)` can be referred as $\mathrm{maxOccurrence}(\mathrm{s_c})$ and it can be measured as

$$\max Occurrence(s_C)=\sum_{i=1}^{|SM|}\{i \mid if\ (s_C \subseteq s_i)\, i=1 or\ i=0\}$$

- Here $|SM|$ is set of order sequences in descending manner for each sequence MSF is greater or equal to $MSF(s_c)$.
- **Hypothesis 3:** Let s_m be maximal sequence and $\mathrm{maxOccurrence}(s_m) \geq ms$ then avoid all subsequences of s_m from considering to evaluate maximal sequences
- **Hypothesis 4:** Let s_c be sequence such that $\mathrm{maxOccurrence}(s_c)<ms$ then discard all supersets of s_c .Since s_c is infrequent then its super sequences also infrequent.

IV. Ossm Overview

- Initially we apply (CEG & REP)[7] to find closed frequent itemsets, which is bottom-up approach called Concurrent Edge Prevision and Rear Edge Pruning (CEG & REP)[7].
- Eliminate the items from transactions that are not part of any event e such that $e \in I$ and referred as outliers. An item i is an outlier if
  - $i \in T_{tid}$ and $i \notin \{I_s \mid for\ s = \{1.....|\ I\ |\}$
  - Here T_{tid} is a transaction represented by transaction id tid and I_s is an itemset that belongs to set I of frequent itemsets.
- Build sequences by grouping items as events in a given transaction. Here we follow the top-to-bottom approach to build events. First we build events based on maximal length itemsets and continue the process in descending order of the itemset lengths.
- Measure the Maximal support of each event e of the given transaction T_{tid} (refer section 3 for measuring maximal support MS for an event e of transaction T_{tid}).
- Measure the Maximal Support Factor MSF of each sequence S_{tid} (Refer section 3 for measuring Maximal Support Factor MSF of sequence S_{tid})
- Order the sequences in sequence dataset S in descending manner of their MSF.
- Build weighted acyclic directed graph from frequent itemsets of length two.
  - Elements of the itemset considered as vertices
  - Support of that itemset considered as edge weight
- Apply WFI algorithm to find critical path between any two items which represents the maximal sequence between two elements opted.
- Apply all four properties that are hypothesized in section 3 to discard, prune sequences or select maximal length sequences.

V. Ossm: Top-Down Ordered Sequence Set Mining For Maximal Length Sequences

a) *Concurrent Edge Prevision and Rear Edge Pruning (CEG&REP)[7] in OSSM*

i. *Preprocess*

Dataset preprocessing and itemsets Database initialization is performed by us as the first stage of proposal. As we find itemsets with single element, we in parallel prune it with the itemsets of single elements if the support of the selected itemsets is less than the required support.

ii. *Concurrent Edge Prevision*

In this phase, we select all itemsets from given itemset database as input in parallel. Then we start projecting edges from each selected itemset to all possible elements. The first iteration includes the pruning process in parallel, from the second iteration onwards this pruning is not required, which we claimed as an efficient process compared to other similar techniques like BIDE [8]. In first iteration, we project an itemset s_p that spawned from selected itemset s_i from DB_S and an element e_i considered from 'I'. If the $f_{ts}(s_p)$ is greater or equal to rs, then an edge will be defined between s_i and e_i. If $f_{ts}(s_i) \cong f_{ts}(s_p)$ then we prune s_i from DB_S. This pruning process is required and limited to first iteration only. From the second iteration onwards project the itemset S_p that spawned from S_p, to each element e_i of 'I'. An edge can be defined between $S_{p'}$ and e_i if $f_{ts}(s_p)$ is greater or equal to rs. In this description $S_{p'}$ is a projected itemset in previous iteration and eligible as a sequence. Then apply the following validation to find the closed sequence.

iii. *Rear Edge Pruning*

If any of $f_{ts}(s_{p'}) \cong f_{ts}(s_p)$ that edge will be pruned and all disjoint graphs except s_p will be considered as closed sequence and moves it into DB_S and remove all disjoint graphs from memory.

The termination of above process do not take place till the graph becomes empty, i.e. till the elements which are connected through transitive edges and projecting itemsets are available in the memory.

b) *Building Sequence Set from Transaction Dataset*

Here in this section we explore the process of building sequence dataset.

TD is the given transaction dataset

I is the set of closed frequent itemsets of length $1\ to\ m$. Here m is the maximal length of the itemset.

- Initially set of closed frequent itemsets is ordered in descending manner by itemset length.
- For each transaction T_{tid} in the given transaction dataset TD
  - Build events based on the closed frequent itemsets of the set I such that the event lengths will be decided in the descending order of the frequent itemset length.
  - Initially events with length of m determined that is maximal length of the frequent itemsets. Then

events with length of $m-1$ will be determined. This process continues to determine events with length $\{m-i \mid i=\{2,3,....m-1\}\}$.

- Then eliminates the items that are not part of the any event in the transaction T_{tid}, which also referred as outliers.

c) Measuring MSF and Ordering the Sequence Dataset

As a part of the OSSM, we order the sequence dataset in the descending manner of the Maximal support Factor (refer section 3 for details).

Let S be the determined sequence dataset from the given transaction dataset TD and set of frequent itemsets I.

- For each sequence s of the given sequence set S
  - Find Maximal support $MS(e)$ of the each event e of the sequence s (refer section 3 for process of measuring $MS(e)$)
  - Find Maximal Support Factor $MSF(s)$ (refer section 3 for process of measuring $MSF(s)$)
- Order the sequence set in descending order of the MSF

d) Building Weighted Acyclic Directed Graph

In this phase of OSSM, we explore the building of a weighted acyclic directed graph.

Let I_2 be the set of frequent itemsets of length 2.

Let G be the graph initially with vertex count of zero $|V|=0$ and edge count of zero $|E|=0$. Here V is vertex set and E is edge set.

- For each itemset $i_{s\to d}$ of I_2 build an edge ed in graph G
  - Let consider item $s \in i_{s\to d}$ as source vertex, and add item s to vertex set V. Increment $|V|$ by 1.
  - Let consider item $d \in i_{s\to d}$ as destination vertex, and add item d to vertex set V. Increment $|V|$ by 1.
  - Build a directed edge ed between s and d, add directed edge ed to edge set E. Increment $|E|$ by 1.
  - Add support of $i_{s\to d}$ as weight to edge ed.

e) Finding Critical Paths between two Items as Maximal Length Sequences

In this phase we apply WFI algorithm [9, 10]. In the first pass of algorithm we try to identify and evaluate potential long and rich candidates. The rich sequences are the one whose constituent 2-sequences have high support. In the directed graph, the 2-sequenece frequencies are represented by the edge weights; we can easily compute the path with the highest weights between all pairs of nodes. Here we use WFI algorithm [9,10] for the purpose finding critical path(a path with maximal vertex count) with maximal weights.

f) Sequence Evolution

Each critical path generated from the graph G will be considered as candidate sequence and stored in candidate sequence set css

g) Verifying Frequency of Candidate Sequence

- For each candidate sequence cs of candidate sequence set css
  - Find $MSF(cs)$
  - For each sequence s such that $\{s \in S \mid MSF(s) \geq MSF(cs)\}$ (refer hypothesis 1in section 3)
    - If $cs \subseteq s$ then increment $\sup(sc)$ by 1
  - If $\sup(cs) \geq st$ (refer the hypothesis 2 in section 3) then
    - Move cs to frequent sequence set fss
    - For each candidate sequence cs' of candidate sequence set css such that $cs' \neq cs$
    - If $cs' \subseteq cs$ then consider cs' as frequent (refer hypothesis 3 in section 3) and move cs' from css to fss
  - Else
    - For each candidate sequence cs' of candidate sequence set css such that $cs' \neq cs$
    - If $cs \subseteq cs'$ then consider cs' as not frequent and prune it from css (refer hypothesis 4 in section 3)

h) Finding Maximal Length Sequences

Let fss be the frequent sequence set generated in previous phase

- Order the fss in descending manner by Maximal support Factor MSF of the frequent sequences of fss.
- Let a Boolean factor sts as true.
- For each frequent sequence fs of the fss and sts is true
  - For each frequent sequence fs' such that $\{fs' \in fss \mid MSF(fs') \geq MSF(fs)\}$ and $fs \neq fs'$ and sts is true
    - If $fs \subseteq fs'$ then set sts as false
  - If sts is true then move frequent sequence fs to maximal length frequent sequence set $mlfss$

Finally, maximal length frequent sequence set $mlfss$ contains sequences that are not subsequence of any other frequent sequences.

VI. Conclusion

The proposed ordered sequence set mining (OSSM) approach is a scalable and optimal because of its hybrid bottom-up-down approach. OSSM is supported by our earlier Concurrent Edge Prevision and Rear Edge Pruning (CEG&REP)[7] for frequent closed itemset mining, which was proven as efficient in memory usage and scalable on dense datasets. A novel mechanism of sequence dataset generation from transaction dataset is introduced in this paper. The proposed OSSM is capable to generate the longest candidate sequence by weighted acyclic directed graph construction and also efficient and scalable to find frequent sequence set and maximal length frequent sequence set due to top-down approach. To compute the support for a candidate sequence, it uses the maximal support factor of sequences. And the order approach that ordering the sequence dataset in descending order of MSF ensures that whole data set is not scanned. Also, if the data set contains long regular sequences, it is identified early enough and thus all the subsequences of this is also marked regular and need not be evaluated. The longest possible sequence is build up by bottom up algorithms starting from 2-sequence.

References Références Referencias

1. Agrawal, R., and Srikant, R., Mining Sequential Patterns: Generalizations and Performance Improvements Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology p. 3 – 17, (1996).
2. Pei J, Han J. et al: "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix- Projected Pattern Growth" in Int'l Conf Data Engineering,p 215-226 (2001)
3. Zaki, M. J., SPADE: An Efficient Algorithm for Mining Frequent Sequences, Machine Learning, v.42 n.1-2, p.31-60, January-February 2001.
4. Garofalakis M, Rastogi R and Shim, K, "Mining Sequential Patterns with Regular Expression Constraints", in IEEE Transactions on Knowledge and Data Engineering, vol. 14, nr. 3, pp. 530-552, (2002).
5. Antunes, C and Oliveira, A.L: "Generalization of Pattern-Growth Methods for Sequential Pattern Mining with Gap Constraints" in Int'l Conf Machine Learning and Data Mining, (2003) 239- 251
6. Lin, D I, and Kedem, Z M, "Pincer Search: A new algorithm for discovering the maximum frequent set", in International conference on Extending database technology, 1998.
7. Anurag Choubey, Dr. Ravindra Patel and Dr. J.L. Rana, "Concurrent Edge Prevision and Rear Edge Pruning Approach for Frequent Closed Itemset Mining"; (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 11, 2011
8. Jianyong Wang, Jiawei Han: BIDE: Efficient Mining of Frequent Closed Sequences. ICDE 2004: 79-90
9. Floyd, R.: Algorithm 97: Shortest path. Communications of the ACM 5 (1962)
10. Warshall, S.: A theorem on boolean matrices. Journal of the ACM 9 (1962)
11. M. Zaki, SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning, 42:31-60, Kluwer Academic Pulishers, 2001.
12. N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal, Discoving frequent closed itemsets for association rules. In ICDT'99, Jerusalem, Israel, Jan. 1999.
13. M. Zaki, and C. Hsiao, CHARM: An efficient algorithm for closed itemset mining. In SDM'02, Arlington, VA, April 2002.
14. R. Kohavi, C. Brodley, B. Frasca, L.Mason, and Z. Zheng, KDD-cup 2000 organizers' report: Peeling the Onion. SIGKDD Explorations, 2, 2000.
15. J. Han, J. Wang, Y. Lu, and P. Tzvetkov, Mining Top- K Frequent Closed Patterns without Minimum Support. In ICDM'02, Maebashi, Japan, Dec. 2002.
16. P. Aloy, E. Querol, F.X. Aviles and M.J.E. Sternberg, Automated Structure-based Prediction of Functional Sites in Proteins: Applications to Assessing the Validity of Inheriting Protein Function From Homology in Genome Annotation and to Protein Docking. Journal of Molecular Biology, 311, 2002.
17. J. Pei, J. Han, and W. Wang, Constraint-based sequential pattern mining in large databases. In CIKM'02, McLean, VA, Nov. 2002.

This page is intentionally left blank