Sequence Pattern Mining: An Incremental Approach

Nilam Thakkar[1], Ass.Prof. Aakash Shah[2].
SOCET[1 &2].
nilam299@gmail.com[1], aakashshah.ce@socet.edu.in[2]

ABSTRACT

The basic idea of sequential pattern mining was first introduced by Agrawal and Srikant [1]. The sequence mining task is to discover a set of attributes, shared across time among a vast no. of objects in a particular db. For illustration, think through the ‘trades’ db of a bookstore, where the objects represent consumers and the attributes represent authors or books. Let’s say that the database records the books bought by each consumer over a period of time. The discovered patterns are the orders of books most frequently bought by the consumers. An example could be that, “70% of the people who buy Jane Austen’s Pride and Prejudice also buy Emma within a month.” Stores can use these patterns for promotions, shelf placement, etc.

A. Notation: The notation used in this method is defined below.
- D1: unique consumer orders.
- T1: the set of recently combined consumer orders from the recently injected consumer orders.
- U1: the complete restructured consumer orders
- Q1: the no. of recently added consumer orders belonging to old consumers in the unique database.
- S1: the upper support threshold for vast orders.
- S1: the lower support threshold for pre-vast orders, S1 < S2
- L1: set of vast k-orders from D1.
- L1: set of k-orders from T1.
- L1: set of k-orders from U1.
- P1: set of pre-vast k-orders from D1.
- P1: set of pre-vast k-orders from T1.
- P1: set of pre-vast k-orders from U1.
- C1: set of all candidate k-orders from T1.
- I1: a sequence.
- S1(II): no. of occurrences of II in D1.
- S1(II): no. of occurrence increments of II in T1.
- S1(II): no. of occurrences of II in U1.

INTRODUCTION

For the sequence pattern mining sequence database is used..A sequence Database $S$, is set of tuple $<$SID, $s>$ where SID is sequence id and $s$ is a sequence.any item is repeated in sequence database $S$ if support$\geq$min_sup.a repeated Sequence is called a Sequential Pattern[1]

SOME SEQUENCE PATTERN MINING ALGORITHMS

The foremost methods for sequential pattern mining are
- AprioriALL
- GSP
- FreeSpan
- PrefixSpan
- SPADE
- SPAM

A. Aprioriall: Sequential pattern mining was first introduced by Agrawal and Srikant[1]. The authors proposed three Apriori-based algorithms. There are five phases in the whole work flow of the algorithms[1]. Mainly sort phase, L-item set Phase, Transformation Phase, Sequence Phase, Maximal Phase. The kind of drawback of AprioriAll is that there are many passes over the database and many candidates created, which are time overwhelming database[1].

B. Gsp: GSP has two steps in the mining process, candidate generation and test, which is similar to other Apriori-based algorithms. In the candidate generation step, k-sequences candidates are generated based on the large (k-1)-sequences. Given a sequence $s=\langle s_1, s_2,..., s_n \rangle$, and subsequence $c$ is a contiguous subsequence of $s$ if any of the following conditions hold: (1) $c$ is derived from $s$ by dropping an item from either $s_1$ or $s_n$; (2) $c$ is derived from $s$ by dropping an item from an element $s_j$ that has at least 2 items; and (3) $c$ is a contiguous subsequence of $c$, and $c$ is a contiguous subsequence of $s$. The candidate sequences are generated in two phases.

C. Freespan: FreeSpan [9] To create the projected databases, FreeSpan first sorts the large-1 items in descending order of support, say b: 5, c: 4, a: 3, d: 3, e: 3, f: 2. Starting from f, the frequent item with the smallest level of support, the f-projected database includes all and only those sequences that contain item-f with items less frequent than f removed. While the projected sequences will be longest in the f-projected database, this is somewhat compensated for by the fact that the database also contains the fewest number of sequences overall (i.e.[sup(f)]=2) . Also since every f is accounted for, there can never be a pattern containing f that is not present here.

D. Prefixspan: In PrefixSpan [3], it supposes that items within elements are in alphabetical order because the item order within an element does not affect the sequential mining. The first step of PrefixSpan is to scan the sequential database to get the length-1 patterns, which is in fact the
large 1-itemsets. Then the sequential database is divided into different partitions according the number of length-1 sequence. The projected databases only contain the suffix of these sequences, by scanning the projected database all the length-2 sequential patterns that have the parent length-1 sequential patterns as prefix can be generated. Then the projected database is partitioned again by those length-2 sequential patterns. The same process are executed recursively until the projected database is empty or no more frequent length-k sequential patterns can be generated.

**E. Spade:** Use Lattice Search Technique, SPADE[4] decomposes the unique problem into smaller sub-problems using equivalence classes.

**F. Spam:** SPAM [6] To generate and test the candidate sequences, SPAM uses two steps, S-step and I-step, based on the lattice concept. As a depth-first approach, the overall process starts from S-step and then I-step. To extend a sequence, the S-step appends an item to it as the new last element, and the I-step appends the item to its last element if possible. Each bitmap partition of a sequence to be extended is transformed first in the S-step, such that all bits after the first bit with value one are set to one. Then the resultant bitmap of the S-step can be obtained by doing ANDing operation for the transformed bitmap and the bitmap of the appended item.

**INCREMENTAL MINING ALGORITHMS**

Note that all the above Discuss Algorithm can mine set of Frequent Sequential pattern from a static database, but these days all the real world database are dynamic in nature, so Incremental Mining concept is introduced. Incremental mining algorithm utilizes the already mined set of frequent sequential patterns for the unique database to mine the complete set of frequent sequential patterns for the restructured database.

Some Incremental Mining Algorithm are ISM, ISE and INCSPAN etc.

**A. Ism:** Incremental Sequence Mining [6] mining algorithm is based on the Negative Border Concept. Negative border is defined as the set of orders which are not frequent but both of whose generating sub orders are frequent. The generating subsequence of a sequence of length k are two sub orders of length (k-1) obtained by dropping exactly one of its first or second item.

Moreover the orders in undesirable border may have very low support and they may not develop frequent after many subsequent apprises to the database. So there is no opinion in custody those highly uncommon orders.

**B. Ise:** Incremental Sequence Extraction [6] is another Incremental Mining Algorithm. In this algorithm the candidate-generate and test technique is used. The drawbacks of this algorithm include the huge no. of candidate orders to be tested and need of multiple scan of the whole database. So this algorithm turns out to be very costly with respect to time and space requirement.

**C. Incspan:** Incremental mining algorithm called IncSpan based on an existing algorithm called PrefixSpan [4]. To gain efficiency the concept of semi-frequent patterns is introduced. The experimental results in [2] show that IncSpan outperforms the non-incremental algorithm PrefixSpan and an incremental mining algorithm ISM.

Based on Non Incremental PrefixSpan and Incremental ISM, IncSpan has a major drawback of not able to mine the complete set of frequent sequential patterns. for that proposed method is introduced.

**PROJECTED METHOD**

Projected Method emphases on recently inserted consumer orders, which are converted from recently added dealings orders, that is used purely incremental method and that generate complete set of frequent pattern. The Objective of Proposed Method is To observe the effect of various existing algorithms for mining frequent orders on various datasets. To recommend an incremental method for mining the frequent orders for sequence database i.e. for the above problem and to validate the incremental method on different datasets used in various ways existing in the unique database.

**A. Phases of Algorithm:**

1) **Input:** A lesser support threshold $S_{1u}$, an higher support threshold $S_{1w}$, a set of vast and pre-vast orders in the unique database $D$ containing of $(d_1 + c_1)$ consumer orders, the acquisitive amount $b_1$ of new consumer orders belonging to old consumers, and a set of $t_1$ recently inserted consumer orders converted from new dealings.

2) **Output:** A set of absolute vast sequential patterns for the restructured database.

   - **Step 1:** Compute the assessment of the following term as: $fl = ([S_{1u} - S_{1w}]d_1) / (1 - S_{1w})$
   - **Step 2:** Compute the recently inserted consumer orders with the old orders in the unique database and count the value $q$, which is the no. of the recently inserted consumer orders belonging to old consumers.
   - **Step 3:** Now take $b_1 = q_1 + b_1$ and compute the assessment of the term as: $fl = b_1*S_{1u} / (1 - S_{1w})$
   - Where $b$ is the acquisitive expanse of $q$ since the last re-scan.

   - **Step 4:** Take $k_1 = 1$, where $k_1$ is for the total item sets in the orders which are processed currently.
   - **Step 5:** Evaluate all the kind of candidate $k_1$-sequences and their count which are incremented from the recently combined consumer sequences $T$.
   - **Step 6:** Then distribute candidate $k_1$-sequences into two portions rendering to if they are vast, pre-vast or small in the real db.

   - **Step 7:** now apply below sub steps:

     - **Sub step 7-1:** compute the $S_{1u}^{U_1}(II) = S_{1u}^{T_1}(II) + S_{1u}^{D_1}(II)$ as new count.
     - **Sub step 7-2:** If $S_{1u}^{U_1}(II)/(d_1 + c_1 + t_1 - b_1) \geq S_{1w}$, then say that $I$ is a vast sequence, established $S_{1u}^{D_1}(II) = S_{1u}^{U_1}(II)$ and preserve $II$ with $S_{1u}^{D_1}(II)$; else, if $S_{1u}^{U_1}(II)/(d_1 + c_1 + t_1 - b_1) \geq S_{1w}$, then say that $II$ as a pre-vast sequence, established $S_{1u}^{D_1}(II) = S_{1u}^{U_1}(II)$ and preserve $II$ with $S_{1u}^{D_1}(II)$; else, disregard $II$.

   - **Step 8:** For each $k_1$-sequence $II$ in the real pre-vast sequences $P_{1k}^{D_1}$ apply below steps:
Sub step 8-1: Now compute \( S_{1}^{U}(I_1) = S_{1}^{T_{1}}(I_1) + S_{1}^{D_{1}}(I_1) \) as new count.

Sub step 8-2: If \( S_{1}^{U}(I_1)/(d_1 + c_1 + t_1 - b_1) \geq S_{1}^{c_1} \), then allocate \( I_1 \) as a vast sequence, established \( S_{1}^{D_{1}}(I_1) = S_{1}^{U}(I_1) \) and preserve \( I_1 \) with \( S_{1}^{D_{1}}(I_1) \); else, if \( S_{1}^{U}(I_1)/(d_1 + c_1 + t_1 - b_1) \geq S_{1}^{b_1} \), then allocate \( I_1 \) as a pre-vast sequence, set \( S_{1}^{D_{1}}(I_1) = S_{1}^{U}(I_1) \) and preserve \( I_1 \) with \( S_{1}^{D_{1}}(I_1) \); else, disregard \( I_1 \).

- Step 9: Put \( I_1 \), means our sequences in the re-examined set \( R_1 \) sequences \( C_1 \), that is not in the real vast sequences \( L_{1_k}^{D_{1}} \) and not in the pre-vast sequences \( P_{1_k}^{D_{1}} \), to use when re-examining in Step 10 is essential.
- Step 10: If \( c_1 + t_1 \leq f_1 - h_1 \) or \( R_1 \) is valueless, then no need to do anything; else, re-examine the real database to conclude that the sequences in the re-examine-set \( R_1 \) are vast or those are pre-vast.
- Step 11: Procedure candidate \((k_1 + 1)\), sequences \( C_{1_k} \) from lastly vast and pre-vast \( k_1 \)-sequences \((L_{1_k}^{D_{1}} \cup P_{1_k}^{D_{1}})\) that seem in the recently combined dealings.
- Step 12: Set \( k_1 = k_1 + 1 \).
- Step 13: Repeat STEPs 5 to 12 until get all vast or pre-vast orders.
- Step 14: Update the final greatest vast sequence patterns conferring to the recently updated vast sequences.
- Step 15: If \( c_1 + t_1 > f_1 - h_1 \), then set \( d_1 = d_1 + c_1 + t_1, \) \( c_1 = 0 \) and \( b_1 = 0 \); else, set \( c_1 = c_1 + t_1 \).

On the completion of step 15, finally the updated sequences are evaluated.

**EXPERIMENTS RESULT**

INCSPAN and Proposed Method are compared for various attributes on a set of consumer data and the results found are as follows:

![Min_Support vs Execution Time](image)

**CONCLUSION**

The Proposed Method is based on the Incremental method that is improvement over IncSpan Algorithm. The IncSpan Algorithm is used Projected Database that’s the running time of algorithm is very high and According to our observations, the performances of the algorithms are strongly depends on the support levels and the features of the data sets (the nature and the size of the data sets). Therefore we employed it in incremental method to guarantee the time saving in the case of sparse and dense data sets. Thus it saves much time and considered as an efficient method as proved from the results.

**REFERENCES**


[5] Jian Pei Jiawei Han, “Mining Consecutive Patterns Efficiently by Prefix-Predictable PatternGrowth”


[8] Hong Cheng, Xi Feng Yan, Jiawei Han:” IncSpan: Incremental Mining of Consecutive Patterns in Vast Database “by U.S. National Science Foundation NSF

[9] Zhang M., Kao B., Cheung D., Yip C,“Efficient algorithms for incremental update of repeated sequences”, Proc. of Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD’02), 2002.


