AN APPROACH OF DATA MINING ON COMPRESSED TRANSACTION

Indu Maurya (M.tech, I.T)  Ashish Maurya (B.tech, Civil Engg.)
Galgotia college of Engg.& Technology, Gr. Noida  Axis College of Engg. & Technology

Abstract- Data mining has many useful applications in recent years because it can help user discover interesting knowledge in large databases. However, existing compression algorithms are not appropriate for data mining. The various data mining approaches which is used to compress the database and then perform the mining operation such as, M²TQT PINCER-SEARCH algorithm, APRIORI algorithm and ID3 algorithm, TM algorithm etc. in this paper we are using the M²TQT approach to improve the performance of mining. Thus, it is observed that, M²TQT performs better than existing approach.

Index terms- Association rule, Data mining, Merged transaction

I. INTRODUCTION

Data mining is used to help users discover interesting and useful knowledge more easily. The spread of computing has led to an explosion in the volume of data to be stored on hard disks and sent over the internet. This growth has led to a need for data compression i.e. the ability to reduce the amount of storage or internet bandwidth requires handling the data. It is more and more popular to apply the association rule mining in recent years because of its wide application in many fields such as stock analysis, web log mining, medical diagnosis, customer market analysis and bioinformatics. The main focus is on the association mining and data pre-process with data compression. It consists of two steps-

1) Data pre-process transforms the original database into a new data representation where several transactions are merged to become a new transaction. It generates a new transaction database at the end of the data pre-process step.

2) Data mining step, it uses Apriori – like algorithm of association rule mining to find useful information. For example association rules were defined for transaction database. An association rule R is an implication of the form X=>Y, where X and Y are set of items and X∩Y=Φ. The support of a rule X=>Y is the fraction of transaction in the database which contains XUY. The confidence of a rule X=>Y is the fraction of transaction containing X which also contains Y. An association rule can be considered interesting if it satisfies the minimum support threshold and minimum confidence threshold, which are satisfied by domain experts.

The most common approach to mining association rules consisting of two separate task –

1st phase) all frequent items sets that satisfy the user minimum support are generated.

2nd phase) it uses these frequent items sets in order to discover all the association rules that meet a confidence threshold.

A transaction database is a set of records representing transactions, where each record consists of a no. of items that occur together in a transaction. The most famous example of transaction is market basket data, in which each transaction corresponds to the set of items brought by a customer during a single visit to store. Transaction database have important role in data mining. Association rules identify relationship among set of items in a transaction database.

Characteristics of compression approach-
1) Reduce the process time of association rule mining by using a quantification table.
2) Compressed database can be decompressed to the original form.

II. MINING TECHNIQUES

a) Apriori algorithm - It is a classic algorithm for learning association rules. It is designed to operate on databases containing transactions (ex- collection of items bought by customers etc). The algorithm attempts to find subsets which are common to at least a minimum no. C (the cutoff or confidence threshold) of the intervals. It uses a “bottom up” approach, where frequent subset is extended one item at a time (a slip known as candidate generation and group of candidates are tested against the data).

Algorithm 1: Apriori algorithm

b) Pincer-search algorithm - This is an efficient algorithm for discovering maximum frequent set. Pincer-search approach combines both top-down searches and bottom – up searches to prune candidates. Bottom – up approach is used in the case, where all maximal frequent item sets are short whereas top-down approach is used when all maximal frequent item sets are long. It reduces number of candidate and number of passes in frequent set discovery process.

Algorithm 2: Pincer algorithm

III. REPRESENTATION OF DATABASE AND MINING

M²TQT approach - M²TQT Approach is a more efficient approach, called Mining Merged Transactions with the Quantification Table (M²TQT) is proposed, which can compress the original database into a smaller one and perform the data mining process without the problems such as: The paper focuses on compressed transaction, a technology that both reduces the effective price of logical data storage capacity, and improves query performance. Multiple times because of the high cost of re-checking the
frequent item sets. \( M^2 \)TQT uses the transaction relation distance to merge the relevant transactions.

IV. MERGE MINING ALGORITHM

There are two sub-processes in the data preprocess. One sub-process transforms the original database into a new data representation. It uses lexical symbols to represent raw data. Here, it’s assumed that items in a transaction are sorted in lexicographic order. Another sub-process is sorting all the transactions to various groups of transactions and then merges each group into a new transaction. For example, \( T_1 = \{A, B, C, E\} \) and \( T_2 = \{A, B, C, D\} \) are two transactions. \( T_1 \) and \( T_2 \) are merged into a new transaction \( T_3 = \{A, B, C, D1, E1\} \). The process called merge-mining algorithm is used to find frequent item sets from the new transaction DM. There are two phases in this algorithm. The first phase is finding frequent item sets. The second phase is to prune redundancy. It is possible that frequent item sets generated in the first phase might not exist in the DM. For this reason, it needs to verify those frequent item sets by scanning DM again.

![Fig1: An overview of the merged transaction algorithm](image)

Algorithm 3: The pseudo code of merge-mining algorithm

```
// Phase one
D^M = compressed database
L^M = (large1- ieitemsets in compressed database);
for(k = 2; L^M_k = O; k++) do begin
    C^M_k = merging-gen(L^M_k);
    for all transactions \( T^M_k \in D^M \) do begin
        C^M_k = subset(C^M_k, t^k); // Candidate contained in t^k
        for all candidates \( c^k \in C^M_k \) do begin
            c^k.count = c^k.count + min – frequency(c^k); // the smallest item frequency in Candidate
        end
        L^M_k = \{ c^k \in C^M_k | c^k.count \geq minsup \};
    end
    Answer = \cup_k L^M_k;
end

// Phase two
D = original database
for all transactions \( T_k \in D \) do begin
    L^M_k = subset(L^M_k, t_k); // Large itemset contained in t
    for all large itemsets \( I^k \in L^M_k \) do begin
        I^k.count ++;
    end
end
L_k = \{ I^k \in L_k | I^k.count \geq minsup \};
Answer = \cup_k L_k;
```

V. PROPOSED METHOD

We used \( M^2 \)TQT approach for the increment of data mining compressed transaction. We present a novel method for \( M^2 \)TQT approach, which shows-----

1. Support local transaction variation
2. Recover the transaction database to its original state
3. Make the compressed database much smaller than the original one
4. Reduce data mining time

We called our approach the Mining Merged Transactions with the Quantification Table (\( M^2 \)TQT) which has three phases:
(1) Merge related transactions to generate a compressed database
(2) Build a quantification table
(3) Discover frequent item sets

**M²TQT APPROACH-** M²TQT uses transaction relation distance to merge the relevant transactions. Based on the distance relation between transactions. It can be merged with closer relationship to generate a better compressed database. Then we create a quantification table to reduce number of candidate’s item sets to be generated.

It helps to prune non-frequent item sets.

**Transaction Relation Distance-** Based on the relation distance between transactions one can merge transactions with closer relationship to generate a better compressed database. Here the transaction relation and transaction relation distance are defined as follows:

**Definition 1:**

1. **Transaction Relation:** The relation between two different transactions T1 and T2 is that T1 is either a subset or a superset of T2.
2. **Transaction Relation Distance:** Distance is the number of different items between two transactions.

**Two transactions**

Example 1: T1= {ABCE} and T2= {ABC}, DT1-T2= 1
Example 2: T3= {A} and T4={C}, DT3-T4= 2

**A QUANTIFICATION TABLE-** To reduce the number of candidate item sets to be generated, additional information is required to help prune non-frequent item sets. A simple quantification table is used to record this information when each transaction is processed. Assuming the items in a transaction appear in a lexicographical order, our approach starts working from the left-most item and calls it a prefix-item. After finding the length of the input transaction as n, it records the count of the item sets appearing in the transaction under the respective entries of length Ln, Ln-1,.., L1. A quantification table is composed of these entries where each Li contains a prefix-item and its support count.

<table>
<thead>
<tr>
<th>TID</th>
<th>TRANSACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>ABCDE</td>
</tr>
<tr>
<td>200</td>
<td>CDE</td>
</tr>
<tr>
<td>300</td>
<td>AD</td>
</tr>
</tbody>
</table>

---

**COMPRESSION OF DATABASE-** Let d be a relation distance and it is initialized to 1 at the beginning. Transactions will be merged into their relevant transaction groups in the merged blocks based on the transaction relation distance.

M²TQT consists of the following steps:

Step 1: Read a transaction at a time from the original database.
Step 2: Record the information of the input transaction to build a quantification table.
Step 3: Compute the length n of the transaction.
Step 4: If the merged block is not empty, read the relevant transaction groups from the merged block.
Step 5: Compute relation distance between the transaction and relevant transaction groups. If the transaction is a superset of the longest transaction of a relevant transaction group, a subset of the smallest transaction of a relevant transaction group, or equal to one transaction of a relevant transaction group, it can be merged into the relevant transaction group. For example, we assume d=1. Two transactions {BCG} and {BG} are merged into a relation transaction group {BCG=2.1.2}. A “=” symbol is used to separate items and their respective support counts. We read another transaction {BC} and compute the relation distance between {BCG=2.1.2} and {BC}. Since the relation distance is 1, {BC} is merged into the relation
transaction group. Finally, the relevant transaction group becomes \( \{ \text{BCG}=3.2.2 \} \).

Step 6: Compute the relation distance between the transaction and those transactions coming from \((n+d)\) block, \(n\) block, and \((n-d)\) block where \(n > d\). If it finds the satisfied relevant transactions, merge the transactions to become a relevant transaction group and then classify it as \((n+d)\) merged block, \(n\) merged block or \((n-d)\) merged block. If no relevant transaction can be found, the transaction is classified as \(n\) merged block.

Step 7: Repeat the above six steps until the last transaction is read.

Step 8: Read a transaction from the merged blocks.

Step 9: Compute the relation distance between the transaction and all other transactions in the relevant transaction groups. If the transaction is a sub-transaction of the maximum length transaction of a relation transaction group and its distance is satisfied, it can merge the transaction into the relation transaction group to generate a new count. The process continued until the last transaction is read.

Step 10: Set \(d\) to \(d+1\).

Step 11: Repeat the above steps 8 - 10 until no more relation distance is found between transactions.

**VI. RESULT ANALYSIS**

\( \text{M}^{2}\text{TQT} \) and Merged Transactions Approach were implemented in java programming language and all experiments run on a PC of Intel(R) core(TM) i3 M380 processor with 2.53GHz 4GB main memory. We analyzed practical on the basis of some datasets produced by generator that \( \text{M}^{2}\text{TQT} \) gives better result than previous studied algorithm and improve compressed transaction performance.

**Table III: Parameters used in the dataset generator**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>D</strong></td>
<td>Number of transactions</td>
<td></td>
</tr>
<tr>
<td><strong>T</strong></td>
<td>Average size of transactions</td>
<td></td>
</tr>
<tr>
<td><strong>I</strong></td>
<td>Average size of maximal potentially–large item sets</td>
<td></td>
</tr>
<tr>
<td><strong>L</strong></td>
<td>Number of potentially-large item sets</td>
<td></td>
</tr>
<tr>
<td><strong>N</strong></td>
<td>Number of items</td>
<td></td>
</tr>
</tbody>
</table>

**Table IV: Database parameter settings**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>T</th>
<th>I</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>T415D10K</td>
<td>4</td>
<td>5</td>
<td>228K</td>
</tr>
<tr>
<td>T10I4D10K</td>
<td>10</td>
<td>4</td>
<td>407K</td>
</tr>
<tr>
<td>T415D1K</td>
<td>4</td>
<td>5</td>
<td>21K</td>
</tr>
</tbody>
</table>

We are using T415D1K for increment mining that is used to run the experiment of merged transaction approach and our algorithm after taking the average size of the potentially large item set be 5 for the minimum supports are 15\%, 20\%, 25\%, and 30\%. Here, 20\% of the dataset T415D1K are used for the updates and 80\% as the original data. Algorithm.1 shows the incremental data mining.

**Fig 2: The experiment of T415D1K**
VII. CONCLUSION AND FUTURE WORK

In this paper, we have studied the various data mining techniques and algorithms used to compress data such as M^2TQT, PINCER-SEARCH algorithm, APRIORI & ID3 algorithm. These techniques were analyzed. Among them algorithm M^2TQT performs in a better way by reducing the processing time and I/O time and by decompression of compressed database to original database and by scanning the transaction database only once. M^2TQT approach utilizes the compressed transactions to mining association rule efficiently with a quantification table and compression of database.

As a future work, it is possible to improve the data mining approach on compressed transaction and to improve the compression rate by incorporating FP Tree in M^2TQT.

VIII. REFERENCES


3. J.ArockiaRenjit,Dr.K.L.Shunmuganathan ,“Mining The Data From Distributed Database Using An Improved Mining Algorithm” (IJCSIS) International Journal of Computer Science and Information, March 2010.


7. D. Xin, J. Han, X. Yan, and H. Cheng, "Mining Compressed Frequent-Pattern Sets,” in Proceedings of the 31st international conference on Very Large Data Bases, pp. 709-720, 2005. World Academy of Science, Engineering and Technology 40 2008 529