Frequent Itemsets Mining: 
An Efficient Graphical Approach

Senthil Kumar A V 
Department of MCA 
Hindusthan College of Science and Commerce
 Coimbatore – 641 028, India
 avsenthilkumar@gmail.com

Adnan Al-Rabea 
Albalqa’ Applied University 
Salt 19117 
Jordan 
adnan_alrabea@yahoo.com

Ibrahiem M.M. El Emary 
King Abdulaziz University 
Jeddah 
omary57@hotmail.com

Abstract: Recent advances in computer technology in terms of speed, cost, tremendous amount of computing power and decrease data processing time has spurred increased interest in data mining applications to extract useful knowledge from data. Over the last couple of years, data mining technology has been successfully employed to various business domains and scientific areas. Various data mining techniques are now available and data mining software has become more matured in recent years. Discovering association rules that identify relationships among sets of items is an important problem in data mining. Finding frequent itemsets is computationally the most expensive step in association rule discovery and therefore it has attracted significant research attention. The approach used in this paper uses a hashing technique to generate a candidate set of large 2-itemsets, directed graphs are formed using the support of 2-itemsets as a result generating all possible frequent k-itemsets in the database.

Key word: Association rules . Data mining . Directed graphs . Frequent itemsets . Minimum support

I. INTRODUCTION

Data mining has attracted much attention in database communities because of its wide applicability. One major application area of data mining is to discover potentially useful information from transaction databases. Mining association rules within a large database is representative problem in data mining. The basic idea in association rule mining is to discover important and useful associations among data items such that the presence of some items in a transaction will imply the presence of other items in the same transaction.

Studies on mining association rules have evolved from techniques for discovery of functional dependencies [10], strong rules [4], classification rules [7],[13], causal rules [11], clustering [6], etc to disk-based, efficient methods for mining association rules in large set of transaction data [1],[3],[4],[12]. Discovery of association rules is an important class of data mining and aims at deciphering interesting relationships among attributes in the data [11],[4],[5],[13]. To achieve this, efficient algorithms are to be implemented to conduct mining on these data. As a base, for a given database of sales transactions, one could like to decipher all transactions among items such that the presence of some items in a transaction will imply the presence of some items in the same transaction. The problem of mining association rules on the basis of database was first explored in [1]. Various algorithms have been proposed to discover the large itemsets [2],[4],[5],[8]. Generally, these algorithms construct a candidate set of large itemsets based on some heuristics, and then discover the subset that indeed contains large itemsets. The same process is continued iteratively in the sense that large items discovered in one iteration are used as the basis to generate the candidate set for the next subsequent iterations. To find the possible frequent itemsets, a pass over the database is made at each level.

The main limitation of almost all proposed algorithms [1],[3],[4],[9],[12] is that they make repeated passes over the disk-resident database partition, incurring high I/O overheads. Moreover, these algorithms use complicated hash structures which entails additional overhead in maintaining and searching them, and they typically suffer from poor cache locality [14]. The problem with Partition, even though it makes only two scans, is that, as the number of partitions increase, the number of locally possible frequent itemsets increases.

While this can be reduced by randomizing the partition selection, but the results from sampling experiment [15],[16],[18],[19],[20] indicate that the randomized partitions will have a large number of possible frequent itemsets in common. Partition can thus spend a lot of time in performing redundant computation [2]. An efficient approach for finding frequent itemsets by using graph approach shall be found in [21].
The work reported in this paper could be viewed as a step towards the generation of all possible frequent k-itemsets in the database by forming directed graphs using the support of 2-itemsets. The planning of the rest of the paper is as follows. Section II gives an insight into the detailed problem description. Section III describes the method for forming directed graphs which in turn used for generating possible frequent itemsets. Experimental results were given in Section IV and Conclusion is reported in Section V.

II. PROBLEM DESCRIPTION

The association rule mining problem can be described as follows [1]. Let I={i_1,i_2,....,i_m} be a set of items. Let D, the task relevant data, be a set of database transactions where each transaction T is a set of items such that T⊆I. Each transaction is associated with an identifier, called TID. Let A be a set of items. A transaction T is said to contain A if and only if A⊆T. An association rule is an implication of the form A⇒B, where A⊆I, B⊆I, and A∩B=∅. A⇒B holds in the transaction set D with support s, where s is the percentage of transactions in D that contain A∪B (i.e. both A and B). This is taken to be the probability, P(A∪B). A⇒B has confidence c in the transaction set D if c is the percentage of transactions in D containing A that also contain B which is taken to be the conditional probability P(B|A). That is, support (A⇒B)=P(A∪B) and confidence (A⇒B)=(P(A∪B|A).

An item is referred as an itemset. An itemset that contains k items is a k-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset which is also known as the frequency, support count, or count of the itemset. An itemset satisfies minimum support min_supp if the occurrence frequency of the itemset is greater than or equal to the product of min_supp and the total number of transactions in D. An itemset satisfying minimum support is a frequent itemset.

The problem of mining association rules that have support and confidence greater than the user-specified minimum support and minimum confidence respectively. Conventionally, the problem of discovering all association rules is composed of the following two steps: (1) Find the large itemsets that have transaction support above a minimum support and (2) From the discovered large itemsets, generate the desired association rules. In this paper, we focus exclusively on the first step: generating frequent itemsets

III. DIRECTED GRAPHS

Various algorithms used for discovering large itemsets make multiple passes over the data [4] [8]. During the first pass, a count is made to find the support of individual items. As a result, the support of individual items are used to determine which of them are large, ie. have minimum support. In each subsequent iteration, the set of itemsets found to be large in the previous pass is used as the base for generating new potentially large itemsets, called candidate itemsets. A count is made to find the actual support for these candidate itemsets during the pass over the data. The candidate itemsets which are actually large is identified at the end of the pass, which forms a base for the next pass. The same process is repeated until no other new large itemsets are found.

In this section, an efficient method for finding frequent itemsets is described. The proposed method works as follows. During the first database scan, the number of occurrences of each item is determined and the infrequent ones are discarded. During the second database scan, the transactions are read and the frequent items of them are used for generating directed graphs. In this way, the database is pruned and is compressed into the memory. The aim of using directed graphs is to store the transactions in such a way that discovering the frequent itemsets can be achieved efficiently. Part 1 of this paper gets a set of large 1-itemsets and finds counts of support of 2-itemsets. In Part 2, the counts of support of 2-itemsets are used to generate directed graphs for the possible efficient 2-itemsets and by using these directed graphs, directed graphs for the possible frequent k-itemsets are generated from which possible frequent k-itemsets in the database can be identified. To illustrate, consider a transaction database D, with items {a,b,c,d,e} and a set of 4 records {acdef, abcef, bcef, be} as shown in TABLE 1. (Not necessarily in this order). For convenience, we will use the notation be, for example, to denote the set of items {b,e}.

<table>
<thead>
<tr>
<th>T.No</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>a c d e f</td>
</tr>
<tr>
<td>T2</td>
<td>a b c e f</td>
</tr>
<tr>
<td>T3</td>
<td>b c e f</td>
</tr>
<tr>
<td>T4</td>
<td>b e</td>
</tr>
</tbody>
</table>
In Part 1, all the transactions of the database are scanned to count the support of the 1-itemsets to form the candidate set of large 1-itemsets i.e., $C_1 = \{ \{a\}, \{b\}, \{c\}, \{e\}, \{f\} \}$. For this purpose, a hash tree for $C_1$ is built on the fly, for the purpose of efficient counting. For each item in the database, it is checked with the item in the hash table. If the item is already present in the hash table, then the corresponding count of this item is incremented by one. If the item is not present in the hash table, then the new item is inserted into the hash table and the count is initialized as one. For each transaction, after occurrences of all the 1-subsets are counted, all the efficient 2-itemsets of the transaction are generated as shown in Fig. 1.

![Database](image)

### Generation of 2-itemsets

<table>
<thead>
<tr>
<th>T.No</th>
<th>Items</th>
<th>2-itemset</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>acdef</td>
<td>{ac, ad, ae, af, cd, ce, cf, de, df, ef}</td>
<td>{ab} 1</td>
</tr>
<tr>
<td>T2</td>
<td>abcef</td>
<td>{ab, ac, ae, af, be, bc, bf, ce, cf, ef}</td>
<td>{ac} 2</td>
</tr>
<tr>
<td>T3</td>
<td>bcef</td>
<td>{bc, be, bf, ce, cf, ef}</td>
<td>{af} 2</td>
</tr>
<tr>
<td>T4</td>
<td>be</td>
<td>{be}</td>
<td>{bc} 2</td>
</tr>
</tbody>
</table>

### Generating $C_2$

<table>
<thead>
<tr>
<th>2-itemset</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ab}</td>
<td>1</td>
</tr>
<tr>
<td>{ac}</td>
<td>2</td>
</tr>
<tr>
<td>{ae}</td>
<td>2</td>
</tr>
<tr>
<td>{af}</td>
<td>2</td>
</tr>
<tr>
<td>{bc}</td>
<td>2</td>
</tr>
<tr>
<td>{be}</td>
<td>2</td>
</tr>
<tr>
<td>{bf}</td>
<td>2</td>
</tr>
<tr>
<td>{ce}</td>
<td>2</td>
</tr>
<tr>
<td>{cf}</td>
<td>2</td>
</tr>
<tr>
<td>{ef}</td>
<td>2</td>
</tr>
</tbody>
</table>

Fig. 1 Generation of efficient 2-itemsets

In Part 2, to discover the possible k-itemsets, first directed graphs for the possible efficient 2-itemsets are generated. By using these directed graphs, directed graphs for the possible frequent k-itemsets are generated. A directed graph is one of the prevailing techniques to depict associations. A directed graph $G = (V, E)$ consists of a finite set $V$,
together with a subset $E \subseteq V \times V$. The elements of $V$ are the vertices of the graph, and the elements of $E$ are the edges of the graph. An edge of a directed graph is an ordered pair $[u,v]$ where $u$ and $v$ are the vertices of the graph. We say that the vertex $v$ is adjacent to the vertex $u$, and the vertex $u$ is adjacent to the vertex $v$. Moreover, we say that the edge is incident from the vertex $u$ and incident to the vertex $v$. An association graph can quickly turn into a tangled display with as few as a dozen rules. 4 directed graphs are constructed for the efficient 2-itemsets as shown in Fig. 2.

No of counts

$2(i)$

$2(ii)$

$2(iii)$

$2(iv)$
Fig. 2 Directed Graphs for the efficient 2-itemsets

The nodes of the trees in Fig 2(i), 2(ii), 2(iii) and 2(iv) represents the respective pairs of the root nodes, i.e., efficient 2-itemsets, which are identified from the given database. Each root node specifies the various items presented in L1. The values of the edges represent the number of counts of each pair from the efficient 2-itemsets. The edges which as the minimum support s, whose value is 2, are highlighted.

After directed graphs for efficient frequent 2-itemsets have been constructed, n directed graphs are constructed for generating possible frequent k-itemsets by using the directed graphs generated using efficient 2-itemsets as shown in Fig.3. The steps involved in constructing the directed graphs for the possible frequent k-itemsets are as follows:

1. **Fig. 3(i)**. Directed graph 2(i) is used as the base and the other parts of the tree is constructed by using the directed graphs 2(ii), 2(iii) and 2(iv) in such a manner that a node of the parent tree acts as the parent node for the subtree.

2. **Fig. 3(ii)**. Directed graph 2(ii) is used as the base and the other parts of the tree is constructed by using the directed graphs 2(iii) and 2(iv) in such a manner that a node of the parent tree acts as the parent node for the subtree.

3. **Fig. 3(iii)**. Directed graph 2(iii) is used as the base and the other parts of the tree is constructed by using the directed graph 2(iv) in such a manner that a node of the parent tree acts as the parent node for the subtree.

**Fig. 3(iv)**. Since the directed graph 2(iv) is formed only by using a single 2-itemset there are no other subtrees to connect with this.
The edges which has the minimum support 2 are highlighted and is used for the identification of the large itemsets. Fig 3(i) and 3(ii) represents the possible frequent k-itemsets \{bcef\} for the given database.
IV. EXPERIMENTAL RESULTS

The experiments are performed on a 1.88 GHz P-IV Core 2DUO with 1 GB DDR2 800 MHz main memory, 80 GB HDD running on Microsoft Windows XP. All programs are written in Microsoft Visual Basic 6.0. Synthetic transactional databases generated using IBM Quest synthetic data generator [17] is used. All the experiments were conducted using T20I7D200k dataset. The naming conventions of the datasets are shown in TABLE 2. The number of the items that can occur in the transactions is 1000.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>Average length of the transactions</td>
</tr>
<tr>
<td>I</td>
<td>Average size of maximal frequent itemsets</td>
</tr>
<tr>
<td>D</td>
<td>Number of transactions</td>
</tr>
<tr>
<td>K</td>
<td>Thousands</td>
</tr>
</tbody>
</table>

To test the behavior of the approach vis-à-vis different support thresholds, a set of experiments was conducted. The mining process tested different support levels, which are 0.0025% that revealed almost 78k frequent patterns, 0.005% that revealed nearly 64k frequent patterns, 0.0075% that generated 51k frequent patterns and 0.01% that returned 37k frequent patterns. Fig. 4 presents the time needed in seconds for each one of these runs. Fig.5 shows the execution times in seconds for the various datasets with transaction sizes 50, 100, 150, 200 and 250.

In Fig. 6, the peak memory sizes in megabytes are illustrated as a function of the number of transactions when the average size of the maximal frequent items is 7 and the average size of the transactions is 20. The minimum support threshold is set to 0.7%. The memory requirement of the approach depends only on the number of efficient frequent 2-itemsets in the given transactions. Since the approach stores only the items needed for finding efficient frequent 2-itemsets which are then used to form directed graphs in the main memory, the memory requirement of the approach does not depend on the number of transactions.
V. CONCLUSION

In this paper the issue of mining frequent itemsets among items in a large database of sales transactions is concerned. The problem of discovering large itemsets was solved by constructing directed graphs for the efficient frequent 2-itemsets first and these efficient frequent 2-itemsets are used in generating the possible frequent k-itemsets.

REFERENCES


[18] Bart Goethals, Mohammed J. Zaki, Advances in frequent itemset mining implementations: report on FIMI'03, ACM SIGKDD Explorations Newsletter, v.6 n.1, June 2004

