Static Data Mining Algorithm with Progressive Approach for Mining Knowledge

Shilpa#1 and Sunita Parashar*2

#Student, Department of Computer Science & Engineering
Haryana College of Technology & Management, Kaithal, Haryana, India
E-mail: 1shilpa.goel12@gmail.com

*Associate Professor, Department of Information Technology,
Haryana College of Technology & Management, Kaithal, Haryana, India
E-mail: 2sunita.tu@gmail.com

Abstract

Frequent itemsets generation is an important area of data mining. This paper is concerned with applying progressive approach to extract interesting information from a static database using dynamic approach. This provides an intelligent environment to discover frequent itemsets while reading a particular set of transaction from static database. We performed extensive experiments and calculate the execution time to generate frequent itemsets on the basis of support and number of transaction read at a time.

Keywords: Static data mining, Dynamic data mining, Support, Number of transactions read at a time, Execution time.

Introduction

With the rapid growth in size and number of available databases in commercial, industrial, administrative and other applications, it is necessary and interesting to examine how to extract knowledge automatically from huge amount of data [1]. Knowledge discovery in databases (KDD), or Data Mining, is the effort to understand, analyze, and eventually make use of huge volume of data available. Data mining is the discovery of hidden information found in databases and can be viewed as a step in overall process of Knowledge Discovery in databases (KDD) [2][3]. It is the integration of various techniques from multiple disciplines such as statistics, machine learning, pattern recognition, neural networks, image processing, and database management system and so on[4]. It makes use of various algorithms to perform a variety of tasks. These algorithms examine the sample data of a problem
and determine a model that fits close to solving the problem. The models that we
determine to solve a problem are classified as predictive and descriptive [5][6].
Predictive mining tasks perform inference on current data in order to make
predictions. The data mining task that forms the part of predictive model are
Classification, Regression, and Time series analysis. Descriptive mining tasks
characterize the general properties of the data in the database. This enables us to
determine the patterns and relationships in a sample data. A data mining task that
forms the part of descriptive model are Clustering, Summarization, Association rules,
Sequence discovery. Classification derives a function or model that describes and
distinguishes data classes or concepts, which determines the class of objects whose
class label is unknown. The derived model is based on the analysis of a set of training
data. The training data includes data objects whose class label is known. Regression is
to forecast future data values based on present and past data values by means of
mathematical formula. Time series analysis is to predict future values for current set
of values that are time dependent. Clustering identifies the classes also called clusters
or groups for the set of objects whose classes are unknown. The objects are so
clustered that the intraclass similarities are maximized and the interclass similarities
are minimized. This is done based on the criteria defined on the attributes of the
objects [1][6]. Summarization is the abstraction or generalization of data. This results
in a smaller set, which gives a general overview of data, usually with aggregated
information. Summarization is used to summarize huge amount of data containing in
a web page or document. The summarization can go to different abstraction levels and
can be viewed from different angles. It is also known as characterization or
generalization. Association rule mining is to generate correlation between large
unclassified data items based on certain attributes and characteristics and association
rule. Association rules are used to identify relationship among a set of items in
database of transactions on the basis of large itemsets [7]. Sequence discovery is to
determine the sequential patterns that exist in data by using the time factor.

In data mining, with the increasing amount of data stored in real application
system, the discovery of association relationship (Association Rule mining) attracts
more and more attention. Mining for association rules can help in business, decision
making, and the development of customized marketing programs and strategies. Thus
goal of data mining is to turn “data into knowledge” [8].Therefore, mining association
rules from large database has been a focused topic in recent research into knowledge
discovery in databases [9].

Database can be static and dynamic. Static databases are those databases that do
not change with time while in dynamic databases, new transactions append as time
advances. This may introduce new frequent itemsets and some existing frequent
itemsets may become invalid. Thus, the maintenance of large itemsets for dynamic
databases is very costly if re-run of previous mining algorithm on updated database is
applied because it repeats much of work done in previous computations. Furthermore,
there is not enough space to store all the data for its processing. So instead of finding
large itemsets again some heuristics are used for mining dynamic databases [10].

This paper is organized as follows. . In Section 2, related work to the new
algorithm is discussed. In Section 3, Static Data Mining algorithms are discussed. In
Section 4, Dynamic Data Mining algorithms are discussed. In Section 5, progressive approach for mining is discussed. In Section 6, results related to current work are discussed. In Section 6, the paper is concluded.

**Related Work**
Static data mining algorithms like Apriori, Fp-Growth, Fast Algorithm, Partition Based Algorithms apply only on original database. If there is a need to modify or delete some or all the existing set of data during the process of data mining then repetition of whole procedure is required, which is time-consuming in addition to its lack of efficiency. So incremental update methods like Fast Update, Probability based & Promising based algorithms are used to extract interesting information from dynamic databases. On the basis of this, new approach (PAPRIORI) can be used that takes original database progressively i.e. read a particular set of transactions at a time while we know the size of original database. PAPRIORI is static data mining algorithm that uses dynamic approach. Since execution time to generate frequent itemsets remains a great challenge, so the goal is to calculate the execution time of proposed approach at varying value of number of transactions read at a time (K).

**Static Data Mining**
Data Mining that uses static database for mining is known as static data mining. There are different static data mining algorithms like Apriori, Fp-Tree, Fast algorithm, Partition based algorithm etc.

**Apriori Algorithm**
Apriori is the most widely accepted static data mining algorithm [7][9]. This is described as a “fast algorithm for mining association rules”. Apriori algorithm is driven by market-basket data. It efficiently generates large itemsets along with generation of candidate itemsets by repeatedly scanning the database. Apriori algorithm is based upon candidate set generation and test method. The problem that always appears during mining frequent relations is multiple scans of original database, huge number of candidate generation and tedious workload of support counting for candidates. So there is need to reduce passes of transaction database scans, to shrink number of candidates and to facilitate support counting of candidates.

**FP-Growth Algorithm**
FP-Tree is an order of magnitude faster than the Apriori algorithm. This is used for mining static databases. In this, the frequent patterns generation process includes two sub processes: constructing the Fp-Tree, and generating frequent patterns from the FP tree. This uses divide-and-conquer method and takes 2 scans of database [11]. Candidate itemsets generation does not occur in this.
Fast Algorithm
Most time consuming operation in the discovery of association rules from the database is the computation of the frequency of the occurrences of interesting subset of items called candidates. So there is need to develop a method that avoids or reduces candidate generation and test and utilizes some novel data structures to reduce the cost in frequent pattern mining. Fast algorithm uses TreeMap which is a structure in java that store key / value pair[12]. Moreover Arraylist technique that greatly reduces the need to traverse the database is also used. This reduces usage of memory.

Partition Based Algorithm
Partition based algorithm divides the database into partitions that reduces the number of database scans to two. This algorithm reduces both CPU and I/O overheads [13]. This algorithm is especially suitable for very large size databases. During first scan, divide database into partitions and generate frequent itemsets in different partitions separately by scanning the database once in each partition. During second scan, counters for each of these itemsets are set up and their actual support is measured to determine if they are large across entire database. If the items are uniformly distributed across partitions then a large fraction of itemsets will be large.

Dynamic Data Mining
Data Mining that uses dynamic databases that take into considerations all updates (insert, update, and delete problems) into account is known as dynamic data mining. There are different dynamic data mining algorithms like Fast Update (FUp), incremental method like promising based algorithm and probability based algorithm.

Fast Update Algorithm
An incremental updating technique FUp (Fast Update) algorithm is used for efficient maintenance of discovered association rules when new transactional data are added to a transaction database [14]. In this, we separate winners (those that remain large in updated database) from losers (that are not large in updated database) among large items in original database and find new winners that are large in original database (DB) and incremental database (db) i.e. (DB U db). This algorithm is 2 to 16 times faster than Apriori.

Promising Based Incremental Approach
Promising frequent itemset algorithm, an incremental method, is proposed for dynamic data mining [15]. This algorithm uses maximum support count of 1-itemsets obtained from previous mining to estimate infrequent itemsets, called promising itemsets, of an original database. These itemsets are capable of being frequent itemsets when new transactions are inserted into the original database. Thus, the
algorithm reduces a number of times to scan the original database. As a result, the algorithm has execution time faster than that of previous methods like FUP (Fast Update).

**Probability Based Incremental Approach**
Probability-based incremental association rule discovery algorithm is used to extract interesting information from dynamic databases [16]. This uses principle of Bernoulli trial to find expected frequent itemsets that reduces number of scans to original database. This proposes a new updating and pruning algorithm that guarantee to find all frequent itemsets of an updated database efficiently. The results show that this algorithm has better performance than that of FUp (Fast Update).

**New Static Data Mining Algorithm(PAPRIORI)**
PAPRIORI algorithm generates frequent itemsets progressively in static database by means of reading K transactions at a time. It is based upon basic data mining algorithm(Apriori). For first K transactions m large itemsets will be generated then for next K transactions m, m+1 large itemsets will be generated progressively and so on. This is based on the following considerations.

- The itemsets that are counted initially or does not satisfy minimum support are Estimated Infrequent (EI) itemsets.
- The itemsets that satisfy minimum support threshold are Estimated Frequent (EF) itemsets.
- CF (Confirmed Frequent) itemsets are those that have been counted throughout whole database once and satisfy minimum support.
- CI (Confirmed Infrequent) itemsets are those that have been counted throughout whole database once and do not satisfy minimum support.
- Following are the algorithmic steps:

**Step 1**: Set all 1-itemsets as Estimated Infrequent (EI) itemsets.

**Step 2**: Read database with K transactions at a time (until transactions read is less than total number of transactions in database).

- For each transaction, increase counter for the itemset.
- For each itemset that belongs to EI if value of counter satisfies minimum support then set itemset as EF.
- If itemsets belong to EF or CF then their immediate superset is set as EI.
- For each itemsets that belongs to EF if it is read throughout the whole database once move that into CF.
- On the other hand if itemsets belongs to EI, if it is read throughout the whole database once move it into CI.
This is repeated until Estimated Frequent (EF) and Estimated Infrequent (EI) itemsets are present.

**Experimental Setup**
To evaluate the performance of PAPRIORI algorithm, the algorithm is implemented and tested on a workstation with Pentium(R) Dual-Core CPU, 2.19 GHz and 2.93GB main memory. The experiments are conducted on a Synthetic dataset and Zoo dataset. The Synthetic dataset comprises 1,000 transactions over 10 items. The Zoo dataset comprises 101 transactions over 15 items. Proposed algorithm is used to find frequent itemsets from static database consisting of transactions. Set fixed value of support for both datasets and vary number of transactions read at a time (K) to calculate execution time.

**Results for Synthetic Dataset**
On the basis of K and execution time the following graphs with fixed value of support (50%, 45%) can be drawn for analysing the results.

![Figure 1 Execution Time with Support = 50%](image-url)
Results for Zoo dataset
On the basis of K and execution time the following graphs with fixed value of support (50%, 55%) can be drawn for analysing the results.
It is obtained from the Figure 1, Figure 2, Figure 3 and Figure 4 that at intermediate value of K, execution time of PAPRIORI algorithm is less. So selection of right value of K is required. If value of K is very less, no frequent itemsets can be obtained easily and execution time will increase. On the other hand, if value of K is very large then again execution time increases and it behaves like Apriori Algorithm.

**Conclusion**

Mining knowledge from database is both practical and desirable. We have proposed static data mining algorithm that generates itemsets progressively with less execution time at intermediate number of transactions read. In the future, further researches and experiments on the proposed algorithm will be presented.

**References**


[6] Jiawei Han and Micheline Kamber, “Data Mining: Concept and Techniques,”


[14] David W. Cheung, Jiawei Han, Vincent T. Ngt C.Y. Wongj,” Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique”, in proceedings of the 12th ICDE, New Orleans, Louisiana (IEEE), pp 106-114, February 1996.

