

## Learning Resource Center

BOOK NUM.:

**This book was issued is overdue due on the date stamped below. if the book is kept over due, a fine will be charged as per the library rules.**

[illegible]



# PARALLEL DATA MINING ALGORITHMS

YASHWANT SINGH THAKUR 081326

POOJA NARAIN 081338

JASKARAN PAL SINGH 081357

Under the supervision of  
Mr. PARDEEP KUMAR



Submitted in partial fulfillment of the Degree of

Bachelor of Technology

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,**  
**WAKNAGHAT, SOLAN, HIMACHAL PRADESH**



## TABLE OF CONTENTS

Chapter No.	Topics	Page No.
	Certificate from the Supervisor	4
	Acknowledgement	5
	Summary	6
Chapter-1	Introduction	7
	1.1 Data mining	7
	1.2 Data mining uses	7
	1.3 Characteristics of data mining system	8
	1.4 Data mining algorithms	8
Chapter-2	Literature Survey	10
	2.1 Kinds of information collection	10
	2.2 Knowledge discovery in databases	12
	2.3 Kind of data to be mined	14
	2.4 Classification of data mining system	16
	2.5 Issues in data mining	17
	2.6 Scope of data mining	19
	2.7 A survey of association rules	21
	2.8 Association rule mining	23
	2.9 Useful concepts about ARM	23
	2.10 ARM algorithms	24
	2.11 Increasing the efficiency of ARM algorithms	25
Chapter-3	Related work/ Background Material	26
	3.1 Apriori based algorithms	26
	3.2 Pseudo code(apriori algorithm)	26
	3.3 Deficiencies in apriori algorithm	27
	3.4 Various researches on apriori based algorithms	27
	3.5 Parallel apriori methods	28

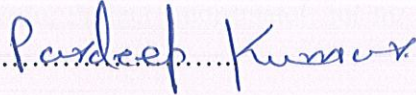


Chapter-4	Contribution	29
	4.1 Pseudo code	29
	4.2 Pseudo code explanation	30
	4.3 Program code	37
Chapter-5	Results	45
Chapter-6	Conclusion	47
	References	48
	Brief bio-data of group members	49



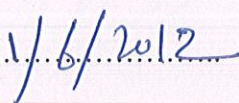
## CERTIFICATE

This is to certify that the work titled "**PARALLEL DATA MINING ALGORITHMS**" submitted by **Yashwant Singh Thakur, Pooja Narain and Jaskaran Pal Singh** in partial fulfillment for the award of degree of **BACHELOR OF TECHNOLOGY** of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor .....  .....

Name of Supervisor      Mr. PARDEEP KUMAR

Designation              SENIOR LECTURER, Dept of CSE, JUIT

Date .....  .....



## ACKNOWLEDGEMENT

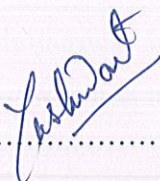
Through this acknowledgment, I express my sincere gratitude to all those people who have been associated with this project and have helped me with it and made it a worthwhile experience.

Firstly I extend my thanks to the various people who have shared their opinions and experiences through which I received the required information crucial for my report.

I express my sincere gratitude to my project guide, Mr.Pardeep Kumar (Senior Lecturer CSE), JUIT Solan who gave me this opportunity to learn the subject in a practical approach and guided me and gave me valuable suggestions regarding the project report.

My special thanks to Dr. Nitin, Project Coordinator, for his kind cooperation in the completion of my project work.

Signature of the student

  
.....

Name of Student

Yashwant Singh Thakur

Date

.....1/06/2012.....

Signature of the student

  
.....

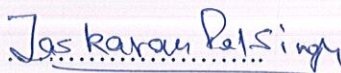
Name of Student

Pooja Narain

Date

.....1/6/2012.....

Signature of the student

  
.....

Name of Student

Jaskaran Pal Singh

Date

.....1/6/2012.....



## SUMMARY

Our project titled “**parallel data mining algorithms**” is basically a research based project. Volumes of data are exploding in both scientific and commercial domains. Data mining techniques that extract information from huge amount of data have become popular in many applications. Many algorithms are designed to analyse those volumes of data automatically in an efficient way so that the users don't have to look through that massive amount of data manually for generating various association rules among them. Apriori algorithm, which is the most famous and frequently used data mining algorithm, is our main focus of concern. The basic apriori algorithm uses a very naive way of finding associations among various data objects, by using frequent itemsets at each iteration and then finding the items having the count lower than the minimum count and then later removing them from the frequent itemset so that the next generation of the itemset does not contain any infrequent items relating to the database. The work done by us is to parallelize the Apriori algorithm along with finding a new way to implement it so that when it is implemented on a database , it leads to lower read through along with fast executions of the code for the frequent itemset generation. For this sole purpose we have gathered the database in a file and created a numerical image of the database which is in the form of a file in our algorithm because the read operations are faster in the files, moreover threads have been implemented in the code for the pair generations so that if a large number of pairs are present, then the itemset generation will be faster and they will be removed if the minimum count is greater than their count in database. The threads and the numerical image provide an easy and effective way of increasing the speed of the code along with only one read operation of the database.

Signature of Supervisor :

Name : Dr Nitin

Date :



# CHAPTER 1

## INTRODUCTION

### 1.1 DATA MINING

Data mining which is a relatively young field of computer science, is the process by which new patterns are generated in a large data set. The main goal of the data mining process is to extract information or knowledge from an existing data set and transform it into a human-understandable structure for further use. That information can be used in various forms such as cost cutting or increasing revenue. Although the term data mining is relatively new, but the technology has been their for years. Many companies have used powerful computers to look through massive amount of supermarket data and analyse market research reports. It helps companies make better business decisions by discovering patterns and relationships in the data.

### 1.2 DATA MINING USES

- Business
  - 1) Market segmentation – It helps in segmenting or differentiating the market based on customers who buy similar products from the company.
  - 2) Fraud detection – It helps in predicting which customers are likely to be fraudulent.
  - 3) Market basket analysis – It helps in detecting what goods or products are commonly purchased together.
- Science and engineering

Data mining is widely used in the areas of science and engineering, such as **genetics**(to find out how the changes in an individual's DNA sequence affects the risks of developing common diseases such as cancer), **electrical power engineering** (for condition monitoring of high voltage electrical equipment) and **education**(to study the factors which leads the students engage in activities which reduce their learning).



## 1.3 CHARACTERISTICS OF THE DATA MINING SYSTEM

- Large quantity of data

Volume of data is so great that it has to be analysed by automated techniques such as satellite information, credit card transactions etc.

- Noisy and incomplete data

1. Imprecise data is a common characteristic of all data collections.
2. Databases are usually contaminated by errors and they cannot assume that the data they contain is correct e.g. some attributes depends on the subjective or measurement judgments.

- Complex data structure, hence conventional statistical analysis is not possible.
- Data is stored in the legacy system.

## 1.4 DATA MINING ALGORITHMS

Volumes of data are exploding in both scientific and commercial domains. Data mining techniques are used to extract information from huge amount of data. Algorithms are basically designed to analyse those volumes of data automatically in efficient ways, so that users can absorb or grasp the hidden knowledge in that data without the need to look over the data manually. As the size of data is increasing rapidly from megabytes to gigabytes and now to terabytes, sequential data mining algorithms may not be able to deliver the results in permissible amount of time. Hence there is an increasing interest in the research for parallel data mining algorithms. There are many challenges associated with parallel data mining algorithms like minimizing I/O, reducing communication and increasing processing speed.

Improving the algorithm performance remains to be the major concern. As such there are no parallel data mining algorithms. Research is done to parallelize the existing sequential data mining algorithms. Sequential data mining algorithms include Apriori, Eclat, D-CLUB and FP-growth. All these algorithms are discussed in detail in upcoming chapters. Our main focus is on the apriori based algorithms.

We will try to parallelize the existing apriori algorithm by implementing it in JAVA and using multiple threads to facilitate parallelism. Moreover we will also try to remove the database scan overhead by using a file because read operations are faster in files.



In the report chapter 3 presents all the previous related work done in the area of apriori algorithm to make it more efficient and optimum. It also presents the deficiencies in the existing approaches to apriori algorithm and various ways of making it efficient.

The work done by us is presented in chapter 4 which includes the pseudo code of our proposed algorithm explained through an example dataset and the program code of the algorithm implemented in JAVA.



## CHAPTER 2

### LITERATURE SURVEY

The current period of time is often referred to as the information age. In this information age, we have been collecting tremendous amounts of information because we believe that information, from the technologies such as computers, satellites etc leads to power and success. Initially, with the invention of computers and means for mass digital storage, we started to collect and store all sort of data, counting on the power of computers to help sort through this mixture of information. Unfortunately, this massive amount of data stored on disparate structures became overwhelming very rapidly. The initial chaos led to the creation of database management systems (DBMS) and structured database. The database management systems (DBMS) have been very crucial assets for management of a large amount of data and especially for effective and efficient retrieval of particular information from a large collection of data. The generation of database management systems has also contributed to massive accumulation of all sorts of information. Today, we have a lot more information than we can handle, from scientific data and business transactions, to satellite images, text reports and military intelligence. Information retrieval or data mining is not enough for decision-making. With huge amalgamation of data, we have now created new desires to help us make better managerial choices. These needs are automatic summarization of data, extraction of the “essence” stored information, and the patterns discovery in raw data.

#### 2.1 KINDS OF INFORMATION COLLECTION

We have been collecting massive amounts of data, from simple numerical measurements and text documents, to more complex information such as spatial data, multimedia channels, and hypertext documents. Following is a non-exclusive list of a variety of information collected in digital form in databases and in flat files.

- **Business transactions:**

Each transaction in the business industry is (often) “memorized” for perpetuity. Such transactions consume time and can be intra-business such as exchanges, purchases, banking,



stock, etc., or intra- business operations such as management of in-house wares and assets. The efficient utilization of the data in a reasonable amount of time for competitive decision-making is definitely the most crucial problem to solve for businesses that struggle to survive in a highly competitive world.

- **Scientific data:**

There is huge amounts of scientific data that need to be analysed. We can capture and store more new data at a faster rate than we can analyse the old data which is already accumulated.

- **Personal and medical data:**

Right from governmental census to personnel and customer files, huge collection of information is continuously collected about groups and individuals. Companies, governments and organizations are gathering very important quantities of personal data to help them manage human resources, better understand a market.

- **Surveillance video and pictures:**

Videotapes from surveillance footages are usually recycled and hence the content is permanently lost. However, today there is a huge tendency to store those tapes and digitize them for future use, reference and analysis.

- **Satellite sensing:**

There are a countless number of satellites around the globe, but all of them sends a never ending stream of data to the surface. Many satellite images and crucial data is made available to the public as soon as they are received in the hope of that other researchers can retrieve and analyse them.

- **Digital media:**

Many radio stations; television channels and film studios are making their video and audio collections digital to improve the management of their multimedia assets.

- **CAD and Software engineering data:**

There are a multitude of Computer Assisted Design (CAD) systems for architects to design structures and engineers to build system components or circuits. These systems generate



huge amount of data. Moreover, software engineering is a source of similar data with code, function libraries, objects, etc., which need very powerful tools for maintenance and management.

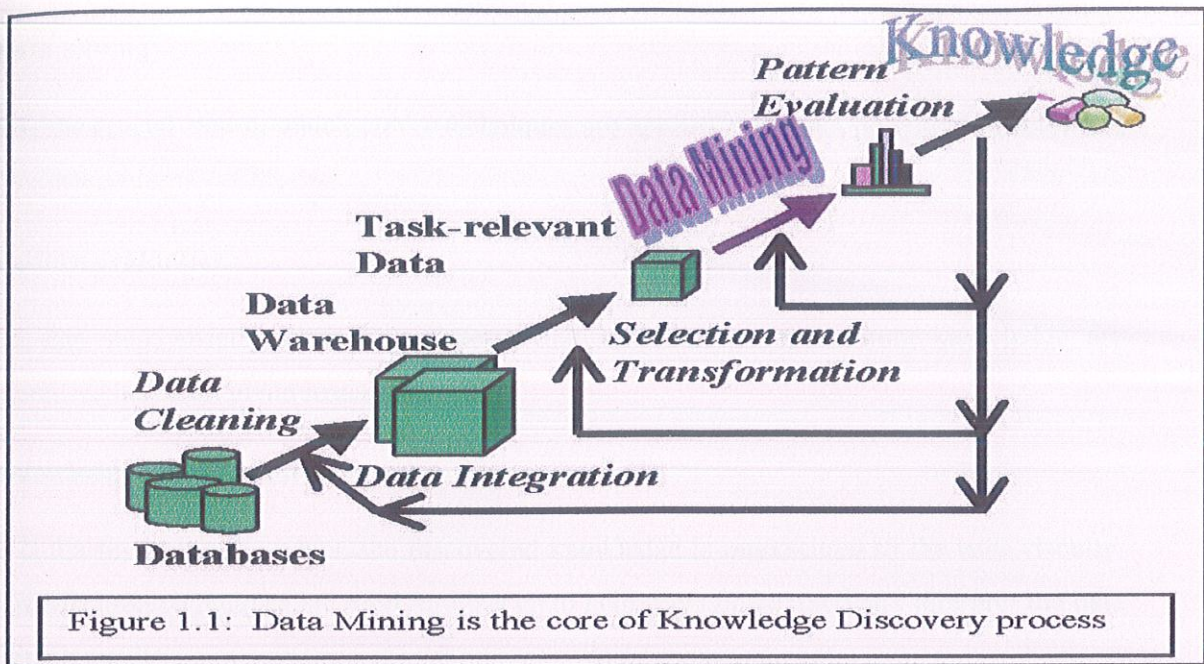
- **The World Wide Web repositories:**

Since the inception of the WWW or World Wide Web in 1993, documents and reports of all sorts of formats, description and content have been collected and inter-connected with hyperlinks which make it the largest database of data ever built. Despite its dynamic and unstructured nature, its heterogeneous characteristic, and its very often redundancy and inconsistency, the WWW is the most crucial data collection which is regularly used for reference because of the broad variety of topics covered and the never ending contributions of publishers and resources.

## **2.2 KNOWLEDGE DISCOVERY IN DATABASES (KDD)**

With the loads of data stored in databases, files and other repositories, it is increasingly important, if not necessary, to develop strong tool for analysing and interpretation of such data and for the extraction of interesting and relevant information and knowledge that could help in decision-making. *Data Mining*, which is a popularly known as *Knowledge Discovery in Databases* (KDD), refers to the nontrivial extraction or mining of information which was previously unknown and potentially useful. The figure (Figure 1.1) showing data mining as a step in an iterative knowledge discovery process is as follows:





The Knowledge Discovery in Databases comprises of a few steps starting from data in the raw form collections to some form of new knowledge. The KDD process consists of the following steps:

- **Data cleaning:**

It is also known as data cleansing, it is a phase in which noisy data or outliers and irrelevant data are removed from the collection.

- **Data integration:**

At this stage, multiple sources of data, which are heterogeneous, may be combined in a common source to produce the integrated database.

- **Data selection:**

In this step, the data which is relevant to the analysis is first selected and then retrieved from the data collection.

- **Data transformation:**

It is known as data consolidation, it is a phase in which the selected or chosen data is transformed into forms appropriate for the mining procedure.



- **Data mining:**

It is the crucial step in which clever techniques are applied to extract potentially useful and valuable patterns.

- **Pattern evaluation:**

In this step, strictly interesting patterns and associations representing knowledge are identified based on given measures.

- **Knowledge representation:**

It is the final phase in which the discovered knowledge is represented to the user visually. Visualization techniques are used in this step to help users understand and interpret the data mining results.

## **2.3 KIND OF DATA TO BE MINED**

Data mining is not specific to one type of media or data. Data mining is applicable to any kind of information repository. Data mining is being used a lot and is being studied for databases, including relational databases, object-relational databases and object-oriented databases, data warehouses, transactional databases, repositories such as the World Wide Web, multimedia databases, time-series databases and textual databases, and even flat files. Here are some examples in more detail:

- **Flat files:**

Flat files are actually the most often used data source for data mining algorithms, especially at the research level. Flat files are simple data files in text or binary format with a structure known by the data-mining algorithm to be applied. The data in these files can be transactions, time-series data, scientific measurements, etc.

- **Relational Databases:**

In short, a relational database consists of a set of tables containing either values of entity attributes, or values of attributes from entity relationships. Tables have columns and rows, where columns represent attributes and rows represent tuples. A tuple in a relational table corresponds to either an object or a relationship between objects and is identified by a set of



attribute values representing a unique key.

The most commonly used query language for relational database is SQL, which allows retrieval and manipulation of the data stored in the tables, as well as the calculation of aggregate functions such as average, sum, min, max and count.

Data mining algorithms using relational databases can be more versatile than data mining algorithms specifically written for flat files, since they can take advantage of the structure inherent to relational databases. While data mining can benefit from SQL for data selection, transformation and consolidation, it goes beyond what SQL could provide, such as predicting, comparing, detecting deviations, etc.

- **Data Warehouses:**

A data warehouse (basically like a storehouse), is a repository of data collected from multiple data sources (often heterogeneous) and is intended to be used as a whole under the same unified schema. A data warehouse gives the option to analyze data from different sources under the same roof. To help in decision-making and multi-dimensional views, data warehouses are usually modelled by a multi- dimensional data structure.

- **Transaction Databases:**

A transaction database is a set of records representing transactions, each with a time stamp, an identifier and a set of items. Associated with the transaction files could also be descriptive data for the items. Since relational databases do not allow nested tables (i.e. a set as attribute value), transactions are usually stored in flat files or stored in two normalized transaction tables, one for the transactions and one for the transaction items. One typical data mining analysis on such data is the so-called market basket analysis.

- **Multimedia Databases:**

Multimedia databases include video, images, audio and text media. They can be stored on extended object-relational or object-oriented databases, or simply on a file system. Multimedia is characterized by its high dimensionality, which makes data mining even more challenging. Data mining from multimedia repositories may require computer vision, computer graphics, image interpretation, and natural language processing methodologies.

- **Spatial Databases:**



Spatial databases are databases that, in addition to usual data, store geographical information like maps, and global or regional positioning. Such spatial databases present new challenges to data mining algorithms.

- **Time-Series Databases:**

Time-series databases contain time related data such stock market data or logged activities. These databases usually have a continuous flow of new data coming in, which sometimes causes the need for a challenging real time analysis. Data mining in such databases commonly includes the study of trends and correlations between evolutions of different variables, as well as the prediction of trends and movements of the variables in time.

- **World Wide Web:**

The World Wide Web is the most heterogeneous and dynamic repository available. A very large number of authors and publishers are continuously contributing to its growth and metamorphosis, and a massive number of users are accessing its resources daily. Data in the World Wide Web is organized in inter- connected documents. These documents can be text, audio, video, raw data, and even applications. Conceptually, the World Wide Web is comprised of three major components: The content of the Web, which encompasses documents available; the structure of the Web, which covers the hyperlinks and the relationships between documents; and the usage of the web, describing how and when the resources are accessed. A fourth dimension can be added relating the dynamic nature or evolution of the documents. Data mining in the World Wide Web, or web mining, tries to address all these issues and is often divided into web content mining, web structure mining and web usage mining.

## **2.4 CLASSIFICATION OF DATA MINING SYSTEMS**

There are many data mining systems available and being developed. Some are specialized systems dedicated to a given data source or are confined to limited data mining functionalities, other are more versatile and comprehensive. Data mining systems can be categorized according to various criteria.



- **Classification according to the type of data source mined:**

This classification categorizes data mining systems according to the type of data handled such as spatial data, multimedia data, time-series data, text data, World Wide Web, etc.

- **Classification according to the data model drawn on:**

This classification categorizes data mining systems based on the data model involved such as relational database, object-oriented database, data warehouse, transactional, etc.

- **Classification according to the king of knowledge discovered:**

This classification categorizes data mining systems based on the kind of knowledge discovered or data mining functionalities, such as characterization, discrimination, association, classification, clustering etc. Some systems tend to be comprehensive systems offering several data mining functionalities together.

- **Classification according to mining techniques used:**

Data mining systems employ and provide different techniques. This classification categorizes data mining systems according to the data analysis approach used such as machine learning, neural networks, genetic algorithms, statistics, visualization, database-oriented or data warehouse-oriented, etc. The classification can also take into account the degree of user interaction involved in the data mining process such as query-driven systems, interactive exploratory systems, or autonomous systems. A comprehensive system would provide a wide variety of data mining techniques to fit different situations and options, and offer different degrees of user interaction.

## **2.5 ISSUES IN DATA MINING**

Data mining algorithms represent techniques that have sometimes existed for many years, but have only lately been applied as reliable and scalable tools that time and again outperform older classical statistical methods. While data mining is still in its infancy stage, it is becoming a trend and ubiquitous. Before data mining develops into a conventional, mature and trusted discipline, many pending issues have to be addressed. Some of these issues are addressed below.



- **Security and social issues:**

Security is an important issue with any data collection that is intended to be used for strategic decision-making and is shared with other users. In addition, when data is collected for customer profiling, user behaviour understanding, correlating personal data with other information, etc., large amounts of sensitive and private information about individuals or companies is gathered and stored. This becomes controversial given the confidential nature of some of this data and the potential illegal access to the information. Moreover, data mining could disclose new implicit knowledge about individuals or groups and this could be against privacy policies, especially if there is potential dissemination of discovered information. Another issue that arises from this concern is the appropriate use of data mining. Due to the value of data, databases of all sorts of content are regularly sold, and because of the competitive advantage that can be attained from implicit knowledge discovered, some important information could be withheld, while other information could be widely distributed and used without control, hence this becomes an issue of concern.

- **User interface issues:**

The knowledge discovered by data mining tools is useful as long as it is interesting, and above all is such that it can be understood by the user. Good data visualization eases the interpretation of data mining results, as well as helps users understand their needs better. The major issues related to user interfaces and visualization are 'screen real-estate', information rendering, and interaction. Interaction with the data and data mining results is crucial since it provides means for the user to focus and refine the mining tasks, as well as to picture the discovered knowledge from different angles and at different conceptual levels.

- **Mining methodology issues:**

These issues are concerned with the data mining approaches applied and their limitations. Topics such as versatility of the mining approaches, the diversity of data available, the dimensionality of the domain, the broad analysis needs (when known), the assessment of the knowledge discovered, the exploitation of background knowledge and metadata, the control and handling of noise in data, etc. are all examples that can dictate mining methodology choices. For instance, it is often desirable and likely to have different data mining methods available since different approaches may perform differently depending upon the data at hand. Moreover, different approaches may suit and solve user's needs in a different manner.



- **Performance issues:**

Many artificial intelligence and statistical methods are used for data analysis and interpretation. However, these methods were often not designed for the very large datasets data mining deals with today. Terabyte sizes are common. This raises the issues of scalability and efficiency of the data mining methods when processing and dealing with considerably large data. Algorithms with exponential and even medium- order polynomial complexity cannot be of practical use for data mining. Linear algorithms are usually the norm. In same theme, sampling can be used for mining instead of the whole dataset. However, concerns such as completeness and which sample to choose may arise. Other topics in the issue of performance are *incremental updating*, and parallel programming. There is no doubt that parallelism can help solve the size problem if the dataset can be subdivided and the results can be merged later. Incremental updating is important for merging results from parallel mining, or updating data mining results when new data becomes available without having to re- analyze the complete dataset.

## 2.6 SCOPE OF DATA MINING

Data mining, as the name suggests is the searching of valuable and important business information in a large database e.g. finding liked products in gigabytes. Data mining technology can be used for these purposes:

- **Automated discovery of previously unknown patterns:**

Data mining sweeps through the database and identifies previously hidden patterns in one step. For example, consider the case of retail sales, where we have to identify patterns between seemingly unrelated products that are often purchased together. This could be similar to detecting fraud credit cards and anomalous data that could represent entry while entering data.

- **Automated prediction of trends and behaviour:**

Data mining automates the process of finding predictive information in large databases. Problems that require a lot of human analysis earlier can be easily answered using data mining methods. Data mining can be used in target marketing, forecasting bankruptcy and identifying segments of a population likely to respond similarly in given events.



Data mining techniques can yield the benefits of automation on existing software and hardware platforms and can be implemented on the new systems, they can analyze massive database in less time. High speed makes the use of large database practical, which in turn point to improved predictions. Data mining has many fields of application. Some of which are:

- **Marketing:**

1. Identifying purchase patterns of customers.
2. Finding associations among customer's demographic characteristics.
3. Predicting response to mailing campaigns.
4. Market basket analysis.

- **Banking:**

1. Detecting patterns of fraud credit card usage.
2. Identifying loyal customers who give maximum business.
3. Identifying stock market rules and strategies from historical market data.
4. Finding the hidden correlations between different financial indicators.
5. Determining credit card spending by customer group.
6. Predicting customers who are likely to change their credit card company

- **Medicine:**

1. Characterize patient behaviour patterns to predict clinical visit.
2. Identifying successful medical therapies and treatments for different illnesses.

- **Insurance and Health Care:**

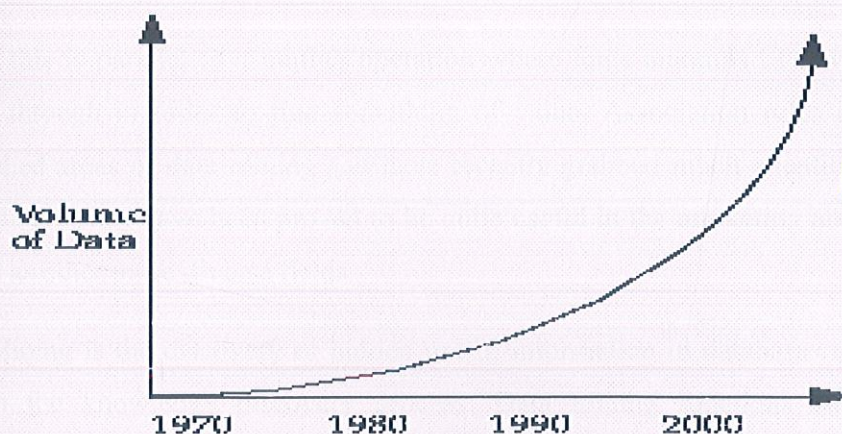
1. Claiming analysis that means which medical procedures are claimed together.
2. Predicting which customers are likely buy new policies.
3. Identifying behaviour patterns of risky customers.



#### 4. Identifying fraud behaviour.

### 2.7 A SURVEY ON ASSOCIATION RULES

In the past two decades, there has been a dramatic increase in the amount of information or data being stored in electronic format. It has been figured out that the amount of information in the world doubles every 20 months and the number and size of databases are increasing even more rapidly. The increase in use of electronic data gathering devices such as point-of-sale or remote sensing devices has added to this explosion of available data. Below graph illustrates the data explosion.



#### The Growing Base of Data

The storage of data became easier and cheaper as the cost of computing power and electronic data storage devices decreased rapidly. The organisations had concentrated so much attention on the accumulation of data; the problem was now what to do with this valuable resource. It was realised that information is at the core of business operations and that decision-makers could make use of the data stored to gain valuable insight into the business. Traditional on-line transaction processing systems are good at feeding and saving data into databases quickly, safely and efficiently but are not good at delivering meaningful analysis in return. Data analysis can provide more knowledge about a business by going beyond the data explicitly stored, to derive important knowledge about the business. This is where Data Mining or Knowledge Discovery in Databases (KDD) has obvious gains for any enterprise.



Data Mining, or Knowledge Discovery in Databases (KDD) as it is also known, is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. This includes a number of different technical approaches, such as clustering, data reduction, data summarization, learning classification rules, finding dependency networks, analyzing changes, and detecting anomalies.

In data mining analysis, the best techniques are those which are developed with an orientation towards large volumes of data, making use of as much of the collected data as possible to arrive at reliable conclusions and decisions. The analysis process commences with a set of data, utilises a methodology to develop an optimal representation of the structure of the data during which time knowledge is acquired. Once knowledge has been gained, this can be applied to larger sets of data working on the assumption that the larger data set has a structure similar to the sample data.

Again, this is parallel to a mining operation where huge amounts of low-grade materials are sieved through in order to find something of value. Association rules are one of the most researched areas of data mining and have recently grabbed much attention from the database community. They have been proved to be quite useful in the marketing and retail communities as well as other more diverse fields.

Data Mining is the discovery of hidden useful information in databases and can be seen as a step in the knowledge discovery process. Data mining functions encompass clustering, classification, prediction, and link analysis (associations). Mining association rules is one of the most important data mining applications. Association rules are used to identify patterns among a set of items in a database. These relationships or patterns are not based on inherent properties of the data themselves (as with functional dependencies), but rather based on co-occurrence of the data items. Association rules are usually used by retail stores to analyze market basket transactions. The identified association rules can be used by the management to increase the effectiveness and gains (and reduce the cost) associated with advertising, marketing, inventory, and stock location on the floor. Association rules are also helpful in other applications such as prediction of failure in telecommunications networks by identifying what events occur before a failure



## 2.8 ASSOCIATION RULE MINING

Association rule mining (ARM) is an important core data mining technique which discovers patterns or rules among items in a large database of variable-length transactions. The goal of ARM is to identify groups of items that most often occur together. It is widely used in market-basket transaction data analysis, graph mining applications like substructure discovery in chemical compounds, pattern finding in web browsing, word occurrence analysis in text documents, and so on.

One of the most important and well researched techniques of data mining, ARM was first introduced in [1]. Association rules are widely used in various areas such as telecommunication networks, market and risk management, inventory control etc.

## 2.9 USEFUL CONCEPTS ABOUT ARM

Association rule mining is to find out association rules that satisfy the predefined minimum **support** and **confidence** from a given database. The problem is usually decomposed into two subproblems. One is to find those itemsets whose occurrences exceed a predefined threshold in the database; those itemsets are called frequent or large itemsets. The second problem is to generate association rules from those large itemsets with the constraints of minimal confidence. Support and confidence are there to confirm the validity of the rule.

Example database:

Transaction ID	A	B	C	D
1	1	1	0	0
2	0	0	1	1
3	1	0	0	0
4	1	1	1	0
5	0	1	0	0

There are some constraints in ARM like minimum thresholds on support and confidence:

**SUPPORT** - The support(X) of any itemset X is defined as the proportion of transactions in the data set which contain the itemset.



In the above database, the itemset {A,B} has a support of  $2/5=0.4$ . It occurs in 40% of all transactions.

**CONFIDENCE** – The confidence of a rule is defined as

$$\text{Confidence}(X \Rightarrow Y) = \text{support}(X \cup Y) / \text{support}(X)$$

In the above database, the rule {A, B}  $\Rightarrow$  {C} has a confidence of  $0.2/0.4=0.5$  which means that in 50% of the transactions in which A and B occurs, C also occur.

## 2.10 ARM ALGORITHMS

### 1) APRIORI ALGORITHM

It is one of the most popular data mining algorithms. The Apriori-based algorithms find frequent itemsets based upon an iterative bottom-up approach to generate candidate itemsets. An Apriori algorithm generally works on databases containing transactions. The algorithm works until no more frequent itemsets are found. Once frequent itemsets are obtained, it is straightforward to generate association rules with confidence larger than or equal to a user specified minimum confidence.

### 2) ECLAT ALGORITHM

It is also used to perform mining on itemsets. It is a depth first search algorithm using intersection of sets. The basic idea is to perform intersection on transaction id's to find the candidate itemset support avoiding the generation of those subsets that are not in the prefix tree.

### 3) FP GROWTH ALGORITHM

FP growth or frequent pattern growth algorithm uses a data structure (FP tree or prefix tree) to store the entire database in a compressed form. By storing the database in a tree like structure, the costly step of database scan is avoided. It uses a divide and conquer approach (tree based frequent pattern mining method ) to avoid the costly process of candidate generation.

### 4) OPUS SEARCH

Opus is an efficient association rule mining that does not require constraints such as support or confidence. Initially it was used to find rules for a fixed items but gradually it has been extended to find rules for any items.



## **2.11 INCREASING THE EFFICIENCY OF ARM ALGORITHMS**

### **1) Reducing the number of passes over the database**

The number of passes over the database can be removed by using a FP tree or frequent pattern mining in which the frequent itemsets are generated with only two passes over the database and without any candidate generation process.

### **2) Sampling**

The sampling approach can be divided into two phases. During phase 1 a sample of the database is obtained and all associations in the sample are found. These results are then validated against the entire database. To maximize the effectiveness of the overall approach, lowered minimum support on the sample is used. Since the approach is probabilistic (i.e. dependent on the sample containing all the relevant associations) not all the rules may be found in this first pass. Those associations that were deemed not frequent in the sample but were actually frequent in the entire dataset are used to construct the complete set of associations in phase 2.

### **3) Parallelization**

Association rule discovery techniques have gradually been adapted to parallel systems in order to take advantage of the higher speed and greater storage capacity that they offer. The transition to a distributed memory system requires the partitioning of the database among the processors, a procedure that is generally carried out indiscriminately.

### **4) Adding extra constraints on the structure of patterns**

Many data mining techniques consist in discovering patterns frequently occurring in the source dataset. Typically, the goal is to discover all the patterns whose frequency in the dataset exceeds a user-specified threshold. However, very often users want to restrict the set of patterns to be discovered by adding extra constraints on the structure of patterns. Data mining systems should be able to exploit such constraints to speedup the mining process.



## CHAPTER 3

### RELATED WORK

#### 3.1 APRIORI BASED ALGORITHM

Our main area of focus is Apriori based algorithms. It is one of the most popular data mining algorithms. The Apriori-based algorithms find frequent itemsets based upon an iterative bottom-up approach to generate candidate itemsets. An Apriori algorithm generally works on databases containing transactions. The algorithm works until no more frequent itemsets are found. Once frequent itemsets are obtained, it is straightforward to generate association rules with confidence larger than or equal to a user specified minimum confidence.

#### 3.2 PSEUDO CODE (APRIORI ALGORITHM)[2]

Join step :  $C(k)$  is generated by joining  $L(k-1)$  with itself

Prune step : Any  $(k-1)$ -itemset that is not frequent cannot be a subset of a frequent  $k$ -itemset

$C(k)$ : Candidate itemset of size  $k$

$L(k)$ : Frequent itemset of size  $k$

$L(1) = \{\text{frequent items}\};$

**for** ( $k = 1; L(k) \neq \emptyset; k++$ ) **do**

**begin**

$C(k+1) = \text{candidates generated from } L(k);$

**for each** transaction  $t$  in database **do**

        increment the count of all candidates in  $C(k+1)$  that are contained in  $t$

$L(k+1) = \text{candidates in } C(k+1) \text{ with min\_support}$

**end**

**return**  $\cup(k) L(k);$



### 3.3 DEFICIENCIES IN APRIORI BASED ALGORITHM

The apriori like algorithms suffer from various deficiencies like too many scans of the transaction database when seeking frequent itemsets (after every iteration, the algorithm scans the whole database to find frequent itemsets), too large amount of candidate itemsets generated unnecessarily (large number of candidate itemsets are generated even though their count is less than minimum count and are then pruned after generation), the redundant generation of identical sub-itemsets and the repeated search for them in the database (itemsets like ab and ba are considered to be same but still they are generated), and so on.

### 3.4 VARIOUS RESEARCHES ON APIORI BASED ALGORITHMS

Various Researches have been done in the area of Apriori based algorithms:-

- 1) A paper by **Lei Ji, Baowen Zhang, Jianhua Li** on **A New Improvement on Apriori Algorithm**. In this paper, the High-Dimension Oriented Apriori algorithm, the HDO Apriori for short, is proposed for mining the association rules in the high-dimensional data. Based on the classical Apriori, the new algorithm can cut down the redundant generation of identical sub-itemsets from candidate itemsets, by means of pruning the candidate itemsets with the infrequent itemsets with lower dimension. So it can obtain a higher efficiency than that of the original algorithm when the dimension of data is high[3].
- 2) A paper by **Huan Wu, Zhigang Lu, Lin Pan, Rongsheng Xu** on **An Improved Apriori-based Algorithm for Association Rules Mining**. In this paper, an improved Apriori-based algorithm (IAA) for ARM is proposed. Experimental results demonstrate that this algorithm will significantly improve the performance of the original algorithm by decreasing the time of prune operation and candidate itemsets verification operation[4].
- 3) A paper by **Yanbin Ye and Chia-Chu Chiang** on **A Parallel Apriori Algorithm for Frequent Itemsets Mining**. In this paper, a parallel Apriori algorithm based on Bodon's work is implemented and its performance on a parallel computer is analyzed. The reason for adoption of Bodon's implementation for parallel computing is because Bodon's implementation using the tree data structure outperforms the other implementations using hash tree[5].
- 4) A paper by **Zachary K. Baker and Viktor K. Prasanna** on **Efficient Parallel Data Mining with the Apriori Algorithm on FPGAs**. In this paper Apriori algorithm is investigated and a popular strategy is designed for progressively grouping together frequent itemsets in large databases given a particular frequency cutoff. The paper also presents several strategies developed for adapting the Apriori algorithm to use in a systolic array. A strategy called systolic injection is also presented in this paper[6].



### 3.5 PARALLEL APRIORI METHODS

To improve the performance of apriori based algorithms, many ways also have been proposed. Many ways are there to parallelize apriori based algorithms. They can be categorized into Count Distribution, Data Distribution and Candidate Distribution methods[7].

- 1) **Count Distribution** – This method adapts a data parallel strategy. It basically divides the database into horizontal partitions which are then scanned independently for obtaining the local counts of all candidate itemsets on each process. The local counts are summed up after every iteration to obtain global count to find frequent itemsets.

The steps for the Count Distribution method are generalized as follows for distributed-memory multiprocessors.

- (a) Divide the database evenly into horizontal partitions among all processes.
  - (b) Each process scans its local database partition to collect the local count of each item.
  - (c) All processes exchange and sum up the local counts to get the global counts of all items and find frequent 1-itemsets.
  - (d) Set level  $k = 2$ ;
  - (e) All processes generate candidate  $k$ -itemsets from the mined frequent  $(k-1)$ -Itemsets.
  - (f) Each process scans its local database partition to collect the local count of each candidate  $k$ -itemset.
  - (g) All processes exchange and sum up the local counts into the global counts of all candidate  $k$ -itemsets and find frequent  $k$ -itemsets among them;
  - (h) Repeat (5) - (8) with  $k = k + 1$  until no more frequent itemsets are found.
- 2) **Data Distribution** – This method attempts to utilize the aggregate main memory of parallel machines by partitioning both the database and the candidate itemsets. Since each candidate itemset is counted by only one process, all processes have to exchange database partitions during each iteration in order for each process to get the global counts of the assigned candidate itemsets.
  - 3) **Candidate Distribution** – This method also partitions candidate itemsets but selectively replicates instead of partition-and-exchanging the database transactions, so that each process can proceed independently.



## CHAPTER 4

### CONTRIBUTION

The basic Apriori algorithm discussed above in the report used a very naive way of finding association among various data objects, by using frequent itemsets at each iteration and then finding the items having the count lower than the minimum count and then later removing them from the frequent itemset so that the next generation of the itemset does not contain any infrequent items relating to the database.

The work done by us is to parallelize the Apriori along with finding a new way to implement it so that when it is implemented on a database, it leads to lower read through along with fast executions of the code for the frequent itemset generation. For this sole purpose we have gathered the database in a file and created a numerical image of the database which is in form of a file in our algorithm because the read operations are faster in the files, moreover threads have been implemented in the code for the pair generations so that if a large number of pairs are present, then the itemset generation will be faster and they will be removed if the minimum count is greater than their count in database. The threads and the numerical image provide an easy and effective way of increasing the speed of the code along with only one read operation of the database. The number of reads of file depends on the number of transactions, but since read operations in files are way faster than those in the database so it does not really affect the code execution time. Pseudo code for this purpose is as follows.

#### 4.1 Pseudo Code:

- 1) Read the file for creation of numerical image.
  - (i) Read the first transaction and fill the corresponding cell of the item with their count in the transaction.
  - (ii) Do it for all the transactions.
- 2) Find the minimum count.
  - (i) Create an array and fill it with the corresponding global counts of the items.
  - (ii) Global count calculated with the help of the numerical image.



(iii) (Only the first time) Through a comparison find the minimum value which is equal to minimum count.

3) Pair generation

(i) Creation of  ${}^nC_r$  number of threads where  $n$  is the number of items and  $r=2$ .

(ii) Threads create the pairs and store them in array which then calculates their count and removes the items having count lower than the minimum count.

4) The process iterates until no more pairs can be generated and thus we get the most frequent itemsets or set.

## 4.2 PSEUDO CODE EXPLANATION

1) Read the file for creation of numerical image.

(i) Read the first transaction and fill the corresponding cell of the item with their count in the transaction.

(ii) Do it for all the transactions.

To show this lets take an example of six transactions as:

fdbe

feb

adb

ae fc

ade

acfe



Now when the code runs, the numerical image so formed will be of this sort:

0	1	0	1	1	1
0	1	0	0	1	1
1	1	0	1	0	0
1	0	1	0	1	1
1	0	0	1	1	0
1	0	1	0	1	1

For every occurrence of the item in the transaction there count is mentioned in that transaction in the numerical image. The first position always shows the **a** item and second position always shows the **b** item and so on.

Lets take another example:

fdbef

feb

adbb

aefc

ade

acfe



The numerical image we get is now:

0	1	0	1	1	<b>2</b>
0	1	0	0	1	1
1	<b>2</b>	0	1	0	0
1	0	1	0	1	1
1	0	0	1	1	0
1	0	1	0	1	1

Here 2 is the count of **b** and **f** in their respective transactions.

And hence for every increase in the number of item their count increases. And this is how we get the numerical image. The screen shot of the code for this purpose is below:

```

    }
    public int read() throws IOException{
        Integer[] temp;
        Scanner cin=new Scanner(file);
        String s;
        int[] count=new int[iCount];
        int x,y;
        while(cin.hasNext()){
            s=cin.nextLine();
            temp=new Integer[iCount];
            for(x=0;x<iCount;x++) temp[x]=0;
            for(x=0;x<iCount;x++)
                for(y=0;y<s.length();y++)
                    if(s.charAt(y)=='a'+x) temp[x]++;
            in.add(temp);
        }
    }

```

This particular code provides us with the numerical image of the file. By printing the **in** array we can get the output as mentioned above.

## 2) Find the minimum count.

- (i) Create an array and fill it with the corresponding global counts of the items.
- (ii) Global count calculated with the help of the numerical image.
- (iii) (Only the first time) Through a comparison find the minimum value which is equal to minimum count.



```

29         in.add(temp);
30     }
31     for(Integer[] i:in)
32         for(x=0;x<iCount;x++)
33             count[x]+=i[x];
34     int min=count[0];
35     for(x=1;x<iCount;x++) if(min>count[x]) min=count[x];
36     for(Integer[] p:in){
37         for(y=0;y<iCount;y++) System.out.print(p[y]+" ");
38         System.out.println();
39     }
40     System.out.print("min"+min);
41     return min;
42 }
43 public ArrayList<String> findFreqItemSet(int min) throws Exception{
44

```

Here the **in** array list stores the numerical image which row by row is passed into the **i** integer list which stores it in **count** which adds each time a new row comes in into it. Taking the example of the first set of transactions:

0	1	0	1	1	1
0	1	0	0	1	1
1	1	0	1	0	0
1	0	1	0	1	1
1	0	0	1	1	0
1	0	1	0	1	1

Now in first iteration **i** gets:

0	1	0	1	1	1
---	---	---	---	---	---

And **count** gets:

0	1	0	1	1	1
---	---	---	---	---	---

Second time **i** gets :

0	1	0	0	1	1
---	---	---	---	---	---

And **count** gets the addition of the old values and the new **i** values:



0            2            0            1            2            2

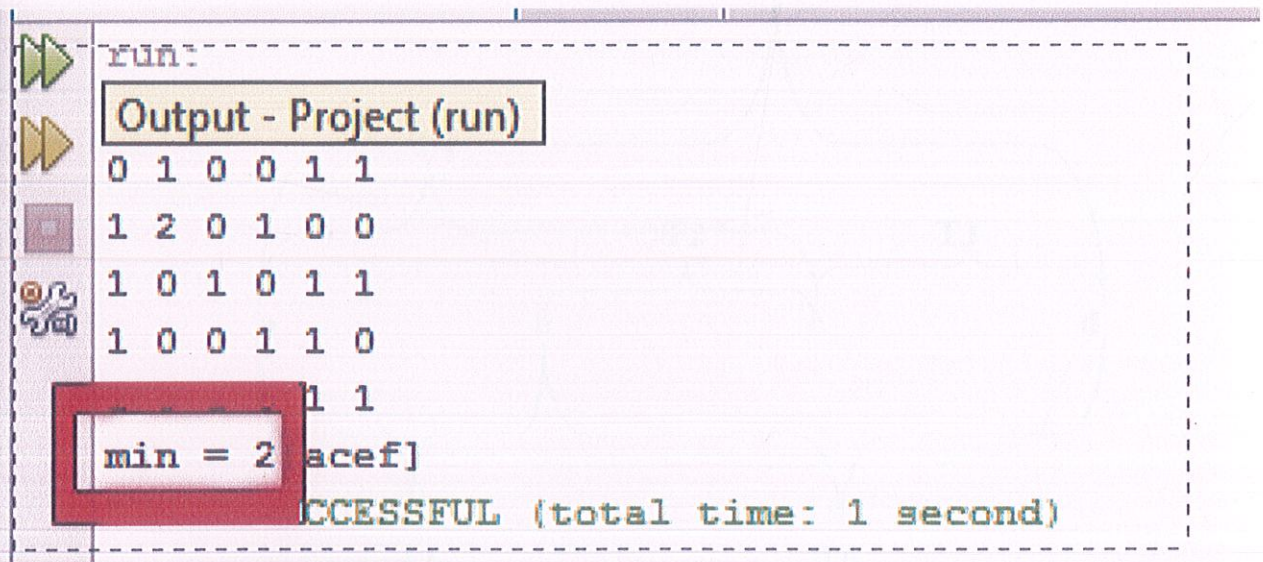
And so for the entire operation we get the final **count** values as:

4            3            2            3            5            4

And thus we then calculate the min count from this count using the statement:

```
for(Integer[] i:in)
    for(x=0;x<iCount;x++)
        count[x]+=i[x];
int min=count[0];
for(x=1;x<iCount;x++) if(min>count[x]) min=count[x];
for(Integer[] p:in){
    for(y=0;y<iCount;y++) System.out.print(p[y]+" ");
    System.out.println();
}
```

And hence we get the **minimum count** using this code above, thus for the above example the **minimum count** will be 2.



### 3) Pair generation

- (i) Creation of  $n_r C$  number of threads where **n** is the number of items and **r= 2**.
- (ii) Threads create the pairs and store them in array which then calculates their count and removes the items having count lower than the minimum count.



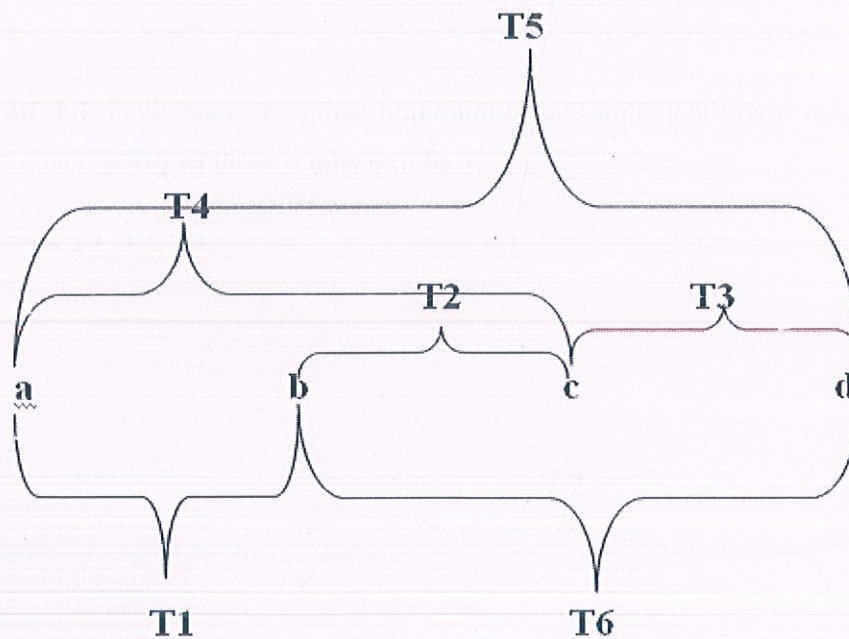
The combination formula is used here for the creation of threads because we have been taking two items at a time and creating new pairs, thus total number of combinations can be given by the above mentioned formula.

For example let us take four items:

**a                      b                      c                      d**

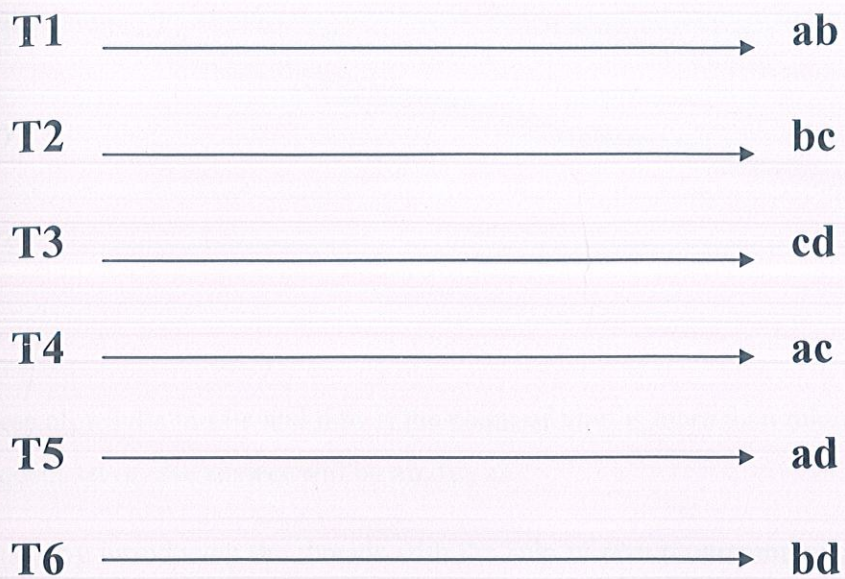
Now according to the formula we have to take 6 threads.

Namely **T1,T2,T3,T4,T5,T6.**



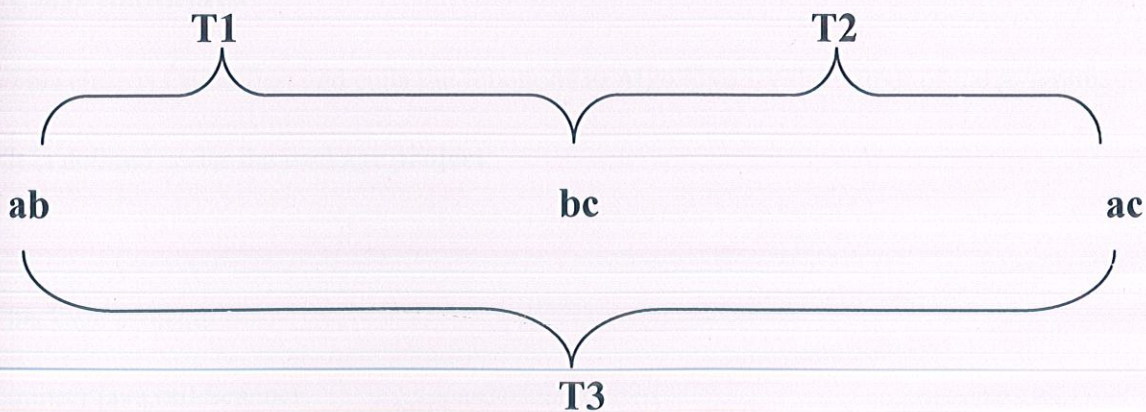


These threads now act simultaneously and provide us the pairs in form of



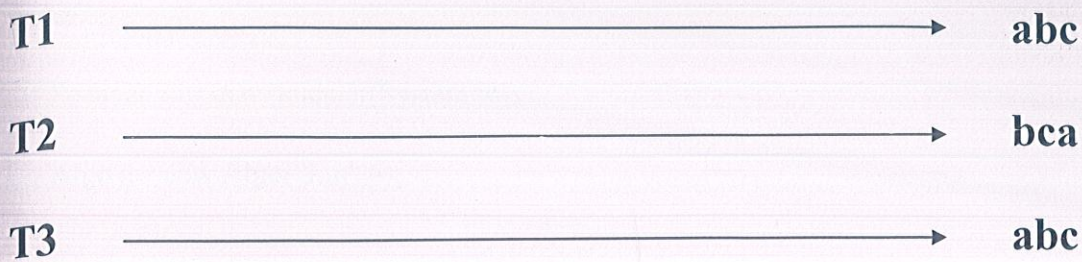
Now suppose **cd**, **ad**, **bd** have counts less than minimum counts so they are removed, now **ab**, **bc**, **ac** remain. The new number of threads now will be 3.

Let the threads now be **T1**, **T2**, **T3**.





And the new threads will create the pairs:



These all results in **abc** and thus if the count of **abc** is more than minimum count we get our most frequent set or else answer will be **ab, bc, ac**.

Hence, by introducing the threads with the help of java programming we have made the algorithm parallel thus making the creation of pairs parallel. Now a very less time will be taken if there are a large number of items which need to make pairs in the algorithm. Making pairs simultaneously has greatly affected the algorithms performance thus making it faster as during the process many thousands of pairs have to be made which require a lot of time and computation, but using threads have greatly reduced the time and moreover reduced the complexity.

#### 4.3 Program code including classes:

##### Class main.java

This class is called first and calls the functions in **Algo.java** for the output of the program.

It is defined under the package **project**.

```
package project;

import java.util.Scanner;

public class Main

{
```



```

public static void main(String[] args)
{
    Scanner cin=new Scanner(System.in);

    Algo gs=new Algo("I.in",6);

    try
    {
        System.out.println(gs.findFreqItemSet(gs.read()));
    }catch(Exception e){}
}
}

```

### **Class algo.java**

It defines the entire algorithm i.e. from numerical image making to pair construction to final result.  
It calls the function in **Thread.java** for the creation of threads.

```

package project;

import java.io.*;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Scanner;

public class Algo {

    File file;

    int iCount;

```



```

ArrayList<Integer[]> in;

public Algo(String filename, int itemCount){

    file=new File(filename);

    iCount=itemCount;

    in=new ArrayList<Integer[]>();

}

public int read()throws IOException{

    Integer[] temp;

    Scanner cin=new Scanner(file);

    String s;

    int[] count=new int[iCount];

    int x,y;

    while(cin.hasNext()){

        s=cin.nextLine();

        temp=new Integer[iCount];

        for(x=0;x<iCount;x++) temp[x]=0;

        for(x=0;x<iCount;x++)

            for(y=0;y<s.length();y++)

                if(s.charAt(y)=='a'+x) temp[x]++;

        in.add(temp);

    }

    for(Integer[] i:in)

        for(x=0;x<iCount;x++)

```



```

        count[x]+=i[x];

int min=count[0];

for(x=1;x<iCount;x++) if(min>count[x]) min=count[x];

for(Integer[] p:in){

    for(y=0;y<iCount;y++) System.out.print(p[y]+" ");

    System.out.println();

}

System.out.print("min = "+min);

return min;

}

public ArrayList<String> findFreqItemSet(int min) throws Exception{

    PrintWriter out;

    out = new PrintWriter(new BufferedWriter(new FileWriter("check.txt")));

    char[] a;char ch;

    int x,y,z,l,size=2,ct,zl,mn,jp;

    ArrayList<String> ar=new ArrayList<String>(),ar1;

    ArrayList<String> pre=new ArrayList<String>();

    ArrayList<String> pre1=new ArrayList<String>();

    HashMap<String,Integer> temp;

    String[] tp;

```



```

String t,t1;

for(x=0;x<iCount;x++)

    for(y=x+1;y<iCount;y++)

        ar.add(Character.toString((char)('a'+x))+

            Character.toString((char)('a'+y)));

while(true){

    temp=new HashMap<String, Integer>();

    l=ar.size();

    for(x=0;x<l;x++){

        t=ar.get(x);

        temp.put(t,0);

        for(Integer[] i:in){

            mn=i[t.charAt(0)-'a'];

            for(y=1;y<size;y++)

                if(mn>i[t.charAt(y)-'a']) mn=i[t.charAt(y)-'a'];

            jp=temp.get(t)+mn;

            temp.put(t, jp);

        }

        if(temp.get(t)<min) temp.remove(t);

    }

    //System.out.println(ar);

    pre1=new ArrayList<String>(pre);

    pre=new ArrayList<String>(temp.keySet());

```



```

if(temp.size()<=1) break;

ar=new ArrayList<String>();

size++;

tp=temp.keySet().toArray(new String[0]);

for(x=0;x<tp.length;x++){

    t=tp[x];

    for(y=x+1;y<tp.length;y++){

        t1=tp[y];

        t1=new Thread(t,t1).compute();

        if(t1.length()==size && !ar.contains((t1)))

            ar.add(t1);

        /*ct=0;

        for(z=0;z<t.length();z++){

            if(t1.contains(Character.toString(t.charAt(z)))){ ct++;continue; }

            a=new char[t1.length()+1];

            ch=t.charAt(z);

            t1+=ch;

            a=t1.toCharArray();

            for(z1=a.length-2;z1>=0;z1--){

                if(a[z1]<ch)break;

                a[z1+1]=a[z1];

            }

            a[z1+1]=ch;

```



```

        t1="";

        for(z1=0;z1<a.length;z1++)

            t1+=a[z1];

    }

    if(t1.length()==size && !ar.contains(t1))

        ar.add(t1);*/

    }

}

}

out.println(pre);

out.println(pre1);

out.flush();

if(pre.isEmpty()) return pre1;

return pre;

}

}

```

### **Class Thread.java**

This is used for the creation of threads and assigning them the elements for pair construction. They give back the result to **Algo.java** that then calculates the minimum count for them and then later send the result to **Main.java** where it is printed.

```

package project;

public class Thread implements Runnable{

    String t1,t2;

```



```

public Thread(String tp1,String tp2){

    t1=tp1;t2=tp2;

}

public String compute(){

    run();

    return t2;

}

public void run() {

    int ct=0,z1;char[] a;char ch;

    for(int z=0;z<t1.length();z++){

        if(t2.contains(Character.toString(t1.charAt(z)))){ ct++;continue; }

        a=new char[t2.length()+1];

        ch=t1.charAt(z);

        t2+=ch;

        a=t2.toCharArray();

        for(z1=a.length-2;z1>=0;z1--){

            if(a[z1]<ch)break;

            a[z1+1]=a[z1];

        }

        a[z1+1]=ch;

        t2="";

        for(z1=0;z1<a.length;z1++)

            t2+=a[z1];}}

```



## CHAPTER 5

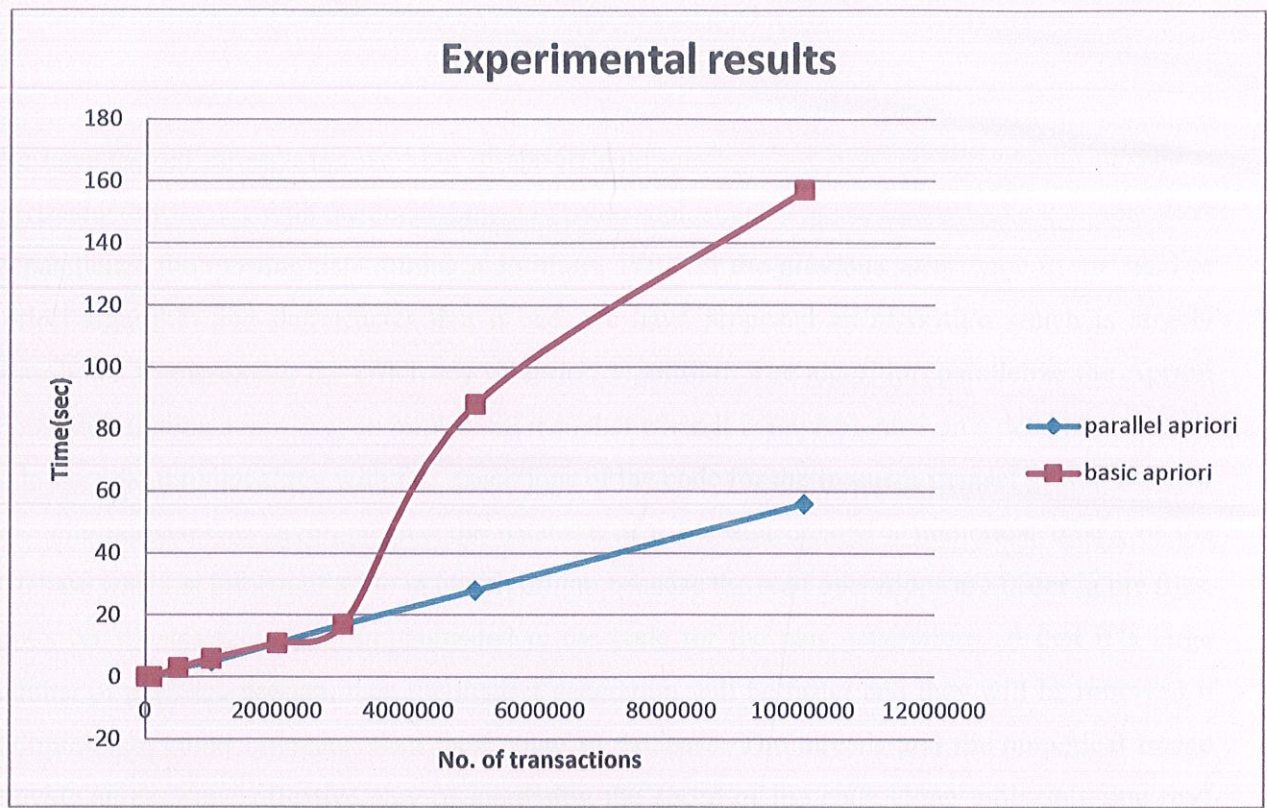
### RESULT

All the experiments are performed on a 3.01 GHz Intel core i5 pc running on windows 7 home premium 64 bit. The programs are coded in JAVA on the Integrated development environment(IDE) of Net beans 7.1. In this experiment, we compared the run time of original sequential apriori with the parallelized algorithm. The itemsets have been generated with the help of Microsoft visual studio 2010. The execution time of the algorithms corresponding to the number of transactions is as shown below. From the results we concluded that the algorithms performed in almost similar fashion (time) that is for lower number of transactions because the number of itemsets created were low due to which the thread execution time was almost similar to the sequential execution time. But as we moved to higher number of transactions, the difference between execution time became more prominent. Thos is because in the exponential increase in the number of itemsets created in various dimensions due to which the thread execution became more effective because of it's simultaneous executions. The algorithm has been compared only with the original apriori algorithm and not with other association rule mining algorithms.

No. of Transactions	Parallel apriori execution time(seconds)	Basic apriori execution time(seconds)
10000	0	0
70000	0	0
100000	0	0
500000	3	3
1000000	5	6
2000000	11	11
3000000	17	17
5000000	28	88
10000000	56	157



### Graph for transactions – execution time





## CHAPTER 6

### CONCLUSION

We have looked through the working of apriori data mining algorithm. As the amount of data is increasing day by day with the development of more sophisticated applications, there is a great need to parallelize the existing data mining algorithms. With all the previous work done in the field of apriori algorithm and deficiencies that it had, we have proposed an algorithm which is largely contributed to increasing the efficiency of apriori algorithm. The algorithm parallelize the Apriori along with finding a new way to implement it so that when it is implemented on a database , it leads to lower read through along with fast executions of the code for the frequent itemset generation. For this sole purpose we have gathered the database in a file and created a numerical image of the database which is inform of a file in our algorithm because the read operations are faster in the files, moreover threads have been implemented in the code for the pair generations so that if a large number of pairs are present, then the itemset generation will be faster and they will be removed if the minimum count is greater than their count in database. The threads and the numerical image provide an easy and effective way of increasing the speed of the code along with only one read operation of the database. The number of reads of file depends on the number of transactions, but since read operations in files are way faster than those in the database so it does not really affect the code execution time.



## REFERENCES

- 1) Agrawal, R., Imielinski, T., and Swami, A. N. 1993. "Mining association rules between sets of items in large databases". In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, 207-216.
- 2) [http://en.wikipedia.org/wiki/Apriori\\_algorithm](http://en.wikipedia.org/wiki/Apriori_algorithm)
- 3) Ji, Baowen Zhang and Jianhua Li. "A New Improvement on Apriori Algorithm."
- 4) Huan Wu, Zhigang Lu, Lin Pan, Rongsheng Xu." An Improved Apriori-based Algorithm for Association Rules Mining".
- 5) Yanbin Ye and Chia-Chu Chiang. " A Parallel Apriori Algorithm for Frequent Itemsets Mining".
- 6) Zachary K. Baker and Viktor K. Prasanna. "Efficient Parallel Data Mining with the Apriori Algorithm on FPGAs".
- 7) Jianwei Li, Ying Liu, Wei-keng Liao and Alok Choudhary. "Parallel Data Mining Algorithms for Association Rules and Clustering".