HIGH PERFORMANCE GRID ENABLED DATA MINING

By

Shehroz Aftab

Saeed Akhtar

Momina Waqar

Omar Mukhtar Farooq

Submitted to the Faculty of Computer Science Department, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment for the requirements of a BE Degree in Computer Software Engineering

March 2008

ABSTRACT

HIGH PERFORMANCE GRID ENBLED DATA MINING

In many application areas data mining algorithms invariably operate on centralized data, in practice related information is often acquired and stored at geographically distributed locations due to organizational or operational constraints. However centralization of such data before analysis may neither be desirable nor feasible for most practical applications due to efficiency and limitations on resources, such as network bandwidth. Moreover, data preprocessing and data mining algorithms are known to be both compute and data intensive. The Grid computing community promises to offer infrastructures that allow on-demand access to distributed resources. [1].

The proposed and implemented solution uses Grid infrastructure to perform mining on the given data sets. In this technique data is mined locally at the sites and suitable representatives are extracted. These representative models are then sent to a global server site where based on these local representatives Global models are formed. This approach increases efficiency by decreasing computational and bandwidth costs required for transmission.

The experimental results further verify this hypothesis by clearly displaying the efficiency difference between centralized data mining and when done in a distributed fashion using the proposed approach and the same data sets.

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of any other award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our families, without whose unflinching support and unstinting cooperation, a work of this magnitude would not have been possible

ACKNOWLEDGEMENTS

We are eternally grateful to Almighty Allah for bestowing us with the strength and resolve to undertake and complete the project.

We gratefully recognize the supervision and motivation provided to us by our Project Supervisor, Head of Computer Science Department (MCS-NUST), Lt. Col. Naveed Sarfraz Khattak. Our gratitude goes to Dr. Ashiq Anjum (CERN-Geneva, UWE-Bristol) who not only was the inspiration behind taking up this endeavor but was also the unflinching support while carrying out efforts associated with it. We are also grateful to Dr. Arshad Ali (NIIT-NUST) for providing us with the requisite facilities for establishing a test bed at his institution for carrying out development work and for constantly overseeing our project.

We are deeply obliged to our families for their never ending patience and support for our mental peace and to our parents for the strength that they gave us through their prayers. We deeply treasure the unparallel support and tolerance that we received from our colleagues, friends and the Faculty of Computer Science Department (MCS-NUST) for their useful suggestions that helped us in the completion of this project.

A word of thanks to the Military College of Signals (MCS) as it has been our foundation and has made us capable to undertake the project.

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

Chapter 1

Introduction

1.1 Preface

With the unprecedented growth rate at which data is being collected today in almost all fields of human endeavor like basic sciences, security, biomedical etc, there is an emerging economic and scientific need to extract useful information from the data. Huge amounts of data are stored in autonomous, geographically distributed sources. The discovery of previously unknown, implicit and valuable knowledge is a key aspect of the exploitation of such sources.

The data warehouse contains the raw material for management's decision support system. The critical factor leading to the use of a data warehouse is that a data analyst can perform complex queries and analysis on the information. The rapid growth and integration of databases provides scientists, engineers, and business people with a vast new resource that can be analyzed to make scientific discoveries, optimize industrial systems, and uncover financially valuable patterns. To undertake these large data analysis projects, researchers and practitioners have adopted established algorithms from statistics, machine learning, neural networks, and databases and have also developed new methods targeted at large data mining problems.

Data mining is one component of the exciting area of machine learning and adaptive computation. The goal of building computer systems that can adapt to

their environments and learn from their experience has attracted researchers from many fields, including computer science, engineering, mathematics, physics, neuroscience, and cognitive science. Out of this research has come a wide variety of learning techniques that have the potential to transform many scientific and industrial fields. Several research communities have converged on a common set of issues surrounding supervised, unsupervised, and reinforcement learning problems.

In recent years several approaches to knowledge discovery and data mining, have been developed, but only few of them are designed for distributed data sources which are not much efficient in terms of computational and communicational cost.

1.2 Grid Computing

Grid computing represents the natural evolution of distributed computing and parallel-processing technologies. Basically, grid computing employs groups of locally or remotely networked machines to work together on specific computational tasks to harness the power of many computers in a network. The primary aim of grid computing is to give IT organizations and application developers the ability to create distributed computing environments that can utilize computing resources on demand. In practice, grid computing can leverage the processing capacity of hundreds, or even thousands, of computers. Thus it can help increase efficiencies and reduce the cost of computing networks by decreasing data-processing time and optimizing resources and distributing

workloads, thereby allowing users to achieve much faster results on large operations and at lower costs.

The development of practical grid computing techniques will have a profound impact on the way data is analyzed. In particular, the possibility of utilizing grid-based data mining applications is very appealing to organizations wanting to analyze data distributed across geographically dispersed heterogeneous platforms.

Benefits provided by a grid enabled solution are, remote job submission, status of jobs can be monitored remotely, status of grid can be monitored remotely, specification of the resources can be viewed, node management of underlying hardware can be done, basic load balancing principle can be applied while allocating jobs (for processing), secure , reliable, scalable, decoupling of GUI from execution engine is possible and in case carrying out data mining it extends the range of data sources that can be queried by data mining engine

1.3 Data Mining

Data mining (DM), also called Knowledge-Discovery in Databases (KDD) or Knowledge-Discovery and Data Mining, is the process of automatically searching large volumes of data for patterns using tools such as classification, association rule mining, clustering, etc.. Data mining is a complex topic and has links with multiple core fields such as computer science and adds value to rich seminal computational techniques from statistics, information retrieval, machine learning and pattern recognition.

Data mining has been defined as "the nontrivial extraction of implicit, previously unknown, and potentially useful information from data" and "the science of extracting useful information from large data sets or databases". It involves sorting through large amounts data and picking out relevant information. It is usually used by businesses and other organizations, but is increasingly used in the sciences to extract information from the enormous data sets generated by modern experimentation.

Although data mining is a relatively new term, the technology is not. Companies for a long time have used powerful computers to sift through volumes of data such as supermarket scanner data, and produce market research reports. Continuous innovations in computer processing power, disk storage, and statistical software are dramatically increasing the accuracy and usefulness of analysis.

1.4 Problem Description

Huge amounts of data are stored in autonomous, geographically distributed sources. The discovery of previously unknown, implicit and valuable knowledge is a key aspect of the exploitation of such sources. Knowledge discovery is a process aiming at the extraction of previously unknown and implicit knowledge out of large databases, which may potentially be of added value for some given application.

Data mining, also known as knowledge discovery, attempts to develop automatic procedures that search these enormous data sets to obtain useful

information that would otherwise remain undiscovered. Such knowledge can take the form of patterns, rules, clusters, or anomalies that exist in the massive datasets. Because of its high computational intensiveness and data intensiveness, data mining serves a good field of application for Grid technology. When large data repositories are coupled with geographic distribution of data, users and systems, it is necessary to combine different technologies for implementing high-performance distributed knowledge discovery systems.

There are many techniques used for data, still all the algorithms based on these techniques use standalone dataset at centralized locations. They are not much feasible for distributed environment.

In recent years several approaches to knowledge discovery and data mining, and in particular to clustering, have been developed, but only a few of them are designed for distributed data sources. The main challenges faced while trying to extract useful information from globally distributed data repositories are as follows:

In many companies data is distributed among several sites, i.e. each site generates its own data and manages its own data repository. Analyzing and mining these distributed sources requires distributed data mining techniques to find global patterns representing the complete information. Traditional data mining algorithms, demanding access to complete data, are not appropriate for distributed applications. Thus, there is a need for distributed data mining

algorithms in order to analyze and discover new knowledge in distributed environments.

One of the most common approaches of business applications to perform distributed data mining is to centralize distributed data into a data warehouse on which to apply the usual data mining techniques. Data warehousing is a popular technology which integrates data from multiple data sources into a single repository in order to efficiently execute complex analysis queries. However, despite its commercial success, this approach may be impractical or even impossible for some business settings like when huge amounts of data are (frequently) produced at different sites and the cost for their centralization cannot scale in terms of communication (bandwidth issues), storage and computation or in some cases, it may not even be possible due to variety of real-life constraints including security, privacy, proprietary nature of data/software and the accompanying ownership and legal issues. For instance whenever data owners cannot do or do not want to release information, which maybe to protect privacy or because disclosing such information may result in a competitive advantage or a considerable commercial added value.

1.5 Concept Evolution

The project has been evolved in a series of steps as a result of the search which started early in the fifth semester about a year and a half back. There were many triggering events and motivating factors that led the team to select this project of great caliber. In search of such a project the project team went to

various organizations like College of Electrical and Mechanical Engineering (E&ME), FAST NU and CARE but eventually found interest in the field of data mining with the help of some professionals at the reputable multinational firm NCR. The main motivation behind taking up this project was the very little amount of research going on in the field of data mining in the South-Asian region as well as the growing demand for its applications world over. Because of its high computational intensiveness and data intensiveness, data mining serves a good field of application for Grid technology, which is the future of the technological world. When large data repositories are coupled with geographic distribution of data, users and systems, it is necessary to combine different technologies for implementing high-performance distributed knowledge discovery systems.

The group was guided by Dr. Ashiq Anjum, who is currently working on this subject in one of the most highly funded projects by the European Union at CERN (Geneva) and UWE (Bristol). He not only helped the team understand the scope of this work but also lead the group in a direction such that the work will be of help in the larger global research on grid based data mining. The work that the group have carried out is just a little effort since this field of study has proved to be an endless sea of knowledge and each day has lead the team to new venues of exploration and learning, hence strengthening grasp of the subject. The concept of the grid and its related technologies is yet so novel for students at the UG level that a lot still needs to be done.

1.6 Intended Solution: The Project

There are many techniques used for data mining like Clustering (using algorithms like K-means, Fuzzy C-means, Hierarchical clustering, Mixture of Gaussians etc), Pattern Recognition (Linear classifier, Quadratic classifier, *k* nearest neighbor classifier (*k*-NN) etc) Association Rule Mining (Partition, Carma, As-Cpa, Viper and Delta etc) etc. But still all the algorithms based on these techniques use standalone dataset at centralized locations. They are not much feasible for distributed environment.

The project aims to address some of the issues relating to the problem statement and has stupendous scope for the future. It is not only a step in this previously unexplored field but it also one of the first of its kind in Pakistan. The project involves a newly devised distributed framework as well as relating improvements in currently existent algorithms in the fields of clustering and classification.

The project scope includes its colossal market value in terms of its usefulness and great research potential. The work cannot only be employed as a part of larger research work in this field but may also be taken up by students at both UG and PG levels for carrying out its future objectives as their research work. This framework and the algorithms however can also be practically deployed for achieving greater efficiency.

1.7 Aim

"The aim of this project was to use the existing Grid infrastructure to mine the data from distributed data sources in response to a user query or request in order to minimize the computational cost in terms of time, access and information delivery as in centralized data mining."

1.8 Research Objective

To find effective ways for Grid enabled distributed data mining in order to reduce computational cost in terms of time and bandwidth required.

Chapter 2

Literature Review

2.1 Introduction

Large-scale data analysis is likely to play a key role in the next generation of data driven collaborative problem solving systems. Advances in computing and communication over wired and wireless networks have resulted in many pervasive distributed computing environments. The Internet, intranets, local area networks, mobile ad hoc wireless networks, peer-to-peer networks, and sensor networks are some examples. These environments often come with different distributed sources of data and computation.

One may classify the distributed data mining literature in various ways with some of the broad categories as Peer-to-Peer Data Mining ,Privacy Preserving Data Mining ,Distributed Data Stream Mining ,Data Mining in Mobile and Embedded Devices , Distributed Data Mining in Sensor Networks , Mining on the Grid , Parallel Data Mining etc . A study of different distributed data mining algorithms is also a vast area of knowledge.

The phase of reviewing the presently available literature based on the grid, data mining and the related frameworks and technologies was a cumbersome task and a brief overview of some important topics has been provided in this section.

2.2 The Grid

The science of the 21st century requires large amounts of computation power, storage capacity and high speed communication. These requirements are increasing at an exponential rate and scientists are demanding much more than are available today. Several astronomy and physical science projects such as CERN's Large Hadron Collider (LHC), Sloan Digital Sky Survey (SDSS), The Two Micron All Sky Survey (2MASS), bioinformatics projects including the Human Genome Project, gene and protein archives (SWISSPROT), meteorological and environmental surveys (NCDC) are already producing Peta and Tera bytes of data which requires to be stored, analyzed, queried and transferred to other sites. To work with collaborators at different geographical locations on Peta scale data sets, researchers require communication of the order of Gigabits / sec. Thus computing resources are failing to keep up with the challenges they face. The concept of the "Grid" has been envisioned to provide a solution to these increasing demands and offer a shared, distributed computing infrastructure.

The *sharing* of distributed computing resources including software, hardware, data, sensors, etc is an important aspect of grid computing. Sharing can be dynamic depending on the current need, may not be limited to client server architectures and the same resources can be used in different ways depending on the objective of sharing.

2.3 Classification of Grid

Classification of grids may be done based on different criteria such as the kind of services provided, the class of problems they address or the community of users. A common method of discrimination depends on whether they offer computational power, *Computational Grids* or data storage, *Data Grids*.

The computational grid has been designed to meet the increased need of computational power by large scale pooling of resources such as compute cycles, idle CPU times between machines, software services etc. The Data Grid is primarily geared towards management of data intensive applications and focuses on the synthesis of knowledge discovered from geographically distributed data repositories, digital libraries and archives.

2.4 Data Mining

The phrase ‘Data Mining’ generally portrays a picture of analyzing huge databases, mostly in the form of large tables, for useful patterns. Also known as knowledge discovery in databases data mining digs out valuable information from large multidimensional apparently unrelated data bases (sets). It’s the integration of business knowledge, statistics, computing technology and algorithms and is used to find hidden patterns and relationships in data.

Data mining identifies trends within data that go beyond simple analysis. Through the use of sophisticated algorithms, users have the ability to identify key attributes of business processes and target opportunities.

The term data mining is often used to apply to the two separate processes of knowledge discovery and prediction. Knowledge discovery provides explicit information that has a readable form and can be understood by a user. Forecasting, or predictive modeling provides predictions of future events and may be transparent and readable in some approaches (e.g. rule based systems) and opaque in others such as neural networks. Moreover, some data mining systems such as neural networks are inherently geared towards prediction rather than knowledge discovery.

A simple example of data mining, often called ‘Market Basket Analysis’, is used for retail sales. If a clothing store records the purchases of customers, a data mining system could identify those customers who favour silk shirts over cotton ones.

Different approaches to data mining exist like classification, estimation, prediction, affinity grouping, clustering and Description.

2.5 Nature of Data Mining Applications

Most of the data mining applications have the common characteristics like data mining applications are highly computation intensive tasks, they are also highly data intensive jobs, in case these jobs are made to execute on normal machines the result is generally much slower response time as compared to execution on distributed frameworks, deals with gigantic data sources and traditional data mining applications and algorithms are usually not able to cope

with the distributed data. This kind of nature of these jobs makes them highly expensive to execute in terms of hardware costs

2.6 Distributed Data Mining

Distributed data mining (DDM) deals with the problem of data analysis in environments with distributed data, computing nodes, and users. Simply put, DDM is data mining where the data and computation are spread over many independent sites. For some applications, the distributed setting is more natural than the centralized one because the data is inherently distributed.

Mining in such environments naturally calls for proper utilization of these distributed resources. Moreover, in many privacy sensitive applications different, possibly multi-party, data sets collected at different sites must be processed in a distributed fashion without collecting everything to a single central site. However, most off-the-shelf data mining systems are designed to work as a monolithic centralized application. They normally down-load the relevant data to a centralized location and then perform the data mining operations. This centralized approach does not work well in many of the emerging distributed, ubiquitous, possibly privacy-sensitive data mining applications.

Distributed Data Mining (DDM) offers an alternate approach to address this problem of mining data using distributed resources. DDM pays careful attention to the distributed resources of data, computing, communication, and human factors in order to use them in a near optimal fashion.

Distributed Data Mining (DDM) applications come in different flavors. When the data can be freely and efficiently transported from one node to another without significant overhead, DDM algorithms may offer better scalability and response time by (1) properly redistributing the data in different partitions or (2) distributing the computation, or (3) a combination of both. These algorithms often rely on fast communication between participating nodes. However, when the data sources are distributed and cannot be transmitted freely over the network due to privacy-constraints or bandwidth limitation or scalability problems, DDM algorithms work by avoiding or minimizing communication of the raw data. In short, DDM offers the technology to analyze data by optimally utilizing the distributed computing, storage, and human resources.

2.7 Mining on the Grid

Data repositories on the grid are inherently distributed and usually heterogeneous in nature. Each of these data repositories have different storage and access mechanisms, schemas and are even owned by different organizations. Furthermore, the data on the grid can be dynamic and streaming in nature. For example, projects concerned with monitoring life processes in frozen lakes in Antarctic send streaming data from sensors in remote locations to the grid, thus forming stream data archives.

In recent years, some architectures have been developed for *distributed* data mining on the grid. These architectures are still evolving and most are work

in progress. Consequently, development of sophisticated data mining algorithms on these architectures still has a long way to go.

Several research projects of mining on the grid include the Knowledge Grid, Grid Miner, Discovery Net,TeraGrid, ADaM on NASA's Information Power Grid, and the DataCutter project have focused on the creation of middleware / systems for data mining and knowledge discovery on top of the data grid. Built on top of a grid environment, various techniques for mining on the grid use basic grid services such as authentication, resource management, communication and information sharing to extract useful patterns, models and trends in large data repositories. Some projects aim to integrate grid services, data mining and On-Line Analytical Processing (OLAP) technologies.

2.8 Analysis of Some Related Work

There are two common approaches to distributed data mining, centralized learning and local learning. Centralized learning moves all the data to a central site for analysis and creation of predictive models while local learning builds predictive models locally at each site and then moves the models to a central location where they are combined.

Ensemble learning is a common technique used to combine the models created at geographically distributed sites. Methods for combining models in an ensemble include meta-learning (Stolfo, et. al., 1997), knowledge selection (Guo,

et. al., 1997), voting schemata (Ditterich, 1997), model selection (Raferty, et. al., 1996), stacking, mixture of experts, and Bayesian model averaging.

Many systems designed for distributed data analysis have been developed, and each of the systems implements different types of learning algorithms. The Kensington system, developed by Guo et al (Guo, et. al., 1997) utilizes knowledge probing that examines learning with a black box perspective and builds a predictive model by inspecting both the input and output of each local model, coupled with the desired output. Kargupta, et. al. developed the BODHI system (Kargupta, et. al., 1999) to employ collective mining methods that rely on techniques from Fourier analysis when combining the individual models into an ensemble. Grossman, et. al. is finalizing the creation of a distributed data mining system known as Papyrus (Grossman, et. al., 2000). Papyrus is intended to support different model and data schemes, including local learning, centralized learning, and various combination strategies, in other words, hybrid learning. Future development planned for the Papyrus system includes work to develop a method to choose an information transfer strategy that can be optimized for a specific data mining task.

An assortment of load balancing methods has been applied to parallel computing techniques over the years. Load balancing is intended to find an optimal strategy for transferring data to the various nodes of a supercomputer or to network workstations. Cheung describes a method of load balancing that optimizes the communication efficiency of parallel computing over a network of clustered compute nodes. Other examples of load balancing have been

discussed, such as load balancing in a heterogeneous computing environment (Grimshaw et. al., 1993) and in distributed object computing systems (Zaki et. al., 1997), but these load balancing techniques do not address issues, such as how to combine predictive models or how to ensure the accuracy level of the subsequent predictive system, that are important in the field of distributed data mining.

There also has been an extensive study on clustering algorithms in the literature. Comprehensive survey on this subject can be obtained from the book *Algorithms for Clustering Data* [12].

2.9 Conclusion

Grid computing promises unprecedented opportunities for unlimited computing and storage resources. The grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. In the case of knowledge discovery the aim is to explore how grid technology is able to provide an infrastructure to access information from different data sources and the computational power to enable high-performing data mining techniques.

Knowledge discovery in large data repositories can find interesting hidden patterns and trends representing them in an understandable way. But data mining is both a data intensive and compute-intensive task. In real world applications sequential algorithms of data mining and data exploration are often unsuitable for datasets with enormous size, high-dimensionality and complex data structure.

Chapter 3

Clustering

3.1 Introduction

Clustering is a division of data into groups of similar objects i.e. a process of partitioning a set of data (or objects) in a set of meaningful sub-classes, called clusters. Where a cluster is a collection of data objects that are “similar” to one another and thus can be treated collectively as one group.

Representing the data by fewer clusters necessarily loses certain fine details, but achieves simplification. It models data by its clusters. Data modeling puts clustering in a historical perspective rooted in mathematics, statistics, and numerical analysis.

From a machine learning perspective clusters correspond to hidden patterns, the search for clusters is unsupervised learning, and the resulting system represents a data concept. From a practical perspective clustering plays an outstanding role in data mining applications such as scientific data exploration, information retrieval and text mining, spatial database applications, Web analysis, CRM, marketing, medical diagnostics, computational biology, and many others.

Clustering is the subject of active research in several fields such as statistics, pattern recognition, and machine learning.

Data mining adds to clustering the complications of very large datasets with very many attributes of different types. This imposes unique computational requirements on relevant clustering algorithms. A variety of algorithms have recently emerged that meet these requirements and were successfully applied to real-life data mining problems.

3.2 Clustering Algorithms

A clustering algorithm attempts to find natural groups of components (or data) based on some similarity. Also, the clustering algorithm finds the centroid of a group of data sets. To determine cluster membership, most algorithms evaluate the distance between a point and the cluster centroids. The output from a clustering algorithm is basically a statistical description of the cluster centroids with the number of components in each cluster.

Typical Algorithms used for clustering are K Means, K Mediods and EM (Expectation Maximization) Clustering.

3.3 K-Means Algorithm

Clustering has been one of the most widely studied topics in data mining and k-means clustering has been one of the popular clustering algorithms.

K-means [7] is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. Although it can be viewed as a greedy algorithm for partitioning the n samples into k clusters so as to minimize the sum

of the squared distances to the cluster centers, it is a procedure that follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori based on attributes/features (K is a positive number/integer). The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.

The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, it is a better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early grouping has been done. At this point the need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After having these k new centroids, a news binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.

Finally, this algorithm aims at minimizing an *objective function*, in this case a squared error function. The objective function which minimizes a chosen distance measure between a data point and the cluster centre is an indicator of the distance of the *n* data points from their respective cluster centers.

3.4 Distributed K-Means

This section describes the new algorithm, Distributed K-means (DK-Means) that have been developed. Initially, describe the main idea behind the algorithm is described and what was the requirement to distribute it over an architectural framework of resources i.e. the GRID followed by the suggested framework itself. The section concludes as the pseudo code is presented, and explains some of the choices that have been made in current implementation.

3.4.1 Main Idea

Distributed K-Means algorithm is a modified form of original K-means algorithm, which allows performing clustering on distributed data sources. Although the original K-means algorithm offers no accuracy guarantees, its simplicity and speed are very appealing in practice and hence becomes choice of algorithm to distribute. The basic idea behind the algorithm is to minimize the time and memory constraints of K Means algorithm by the use of GRID technology. This algorithm is suitable for analyzing data that is distributed across loosely coupled machines.

The need for distributing this algorithm arose from mining excessively large data sets specially those placed globally apart by using techniques for centralized data mining. Centralized data mining is often not convenient as data needed for an analysis may be distributed, costs of centralizing data may be high or simply because of ownership and privacy issues. Sometimes the option of

mining the data in a distributed form or not mining at all may result from the limitation that computational resources needed for analysis are not available to serve the purpose (i.e. costly).

There are three key questions to be addressed. First, how to distribute data sets present at various data sources and send them to computing nodes, what kind of architecture is to be used to implement this design. Second, how to use the computational power and unlimited resources of the grid to effectively and efficiently mine this data. Third, how would it be known that same cluster centers are computed as in the original k-means algorithm and how to form a centralized global model out of the various distributed local models?

Initially a client requests the desired clustering model to be built from data placed at distributed data repositories. DKMeans divides the data into a number of datasets and then each dataset is clustered individually at different locations by running the k-means algorithm, using the same convergence criteria and same initial points as used in case of centralized k-means. In the end the results from each site is collected and then combined together to form global model.

A layered GRID architecture can be shown as in the Figure 3.1 .This figure can describe some of the major implementation level details .The top layer shows the application layer on which various GRID based applications are run and it is that layer which the applications aim. (At the user level middleware layer Java Development kit has been used. Core middleware employed is Globus while Condor is the Local Resource Manager).

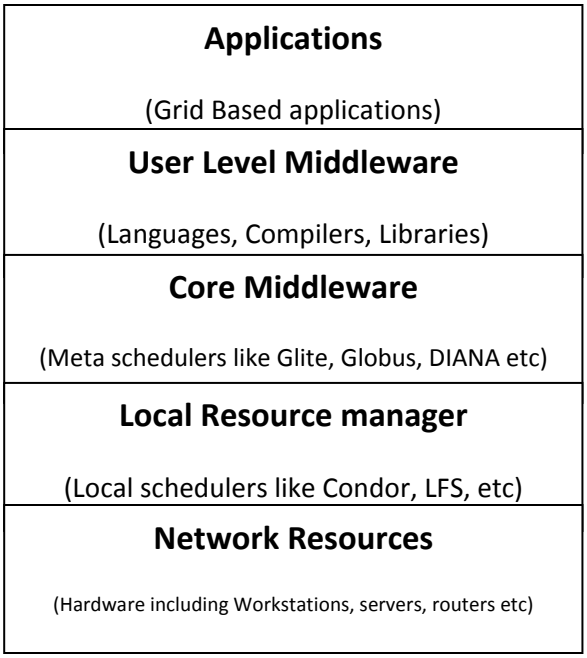

Figure 3.1: Layered Grid Architecture

3.4.2 Framework

The framework for the implementation of this algorithm is depicted in Figure 3.2 while the detailed algorithm is shown in Figure 3.3. DB_i denotes the various datasets present at distributed sources. For each DB, K is initialized random centroids such that each cluster C_{in} is associated with a centroid w_i respectively. This centroid set is then sent to the merger node, which is then sent to a merger node. Here the concept of merger node denotes nothing but simply a node within the GRID having some computational power and which the resource scheduler is aware of. Now the set of centroids i.e. $Centroid_i$ is computed and sent to all DBs.

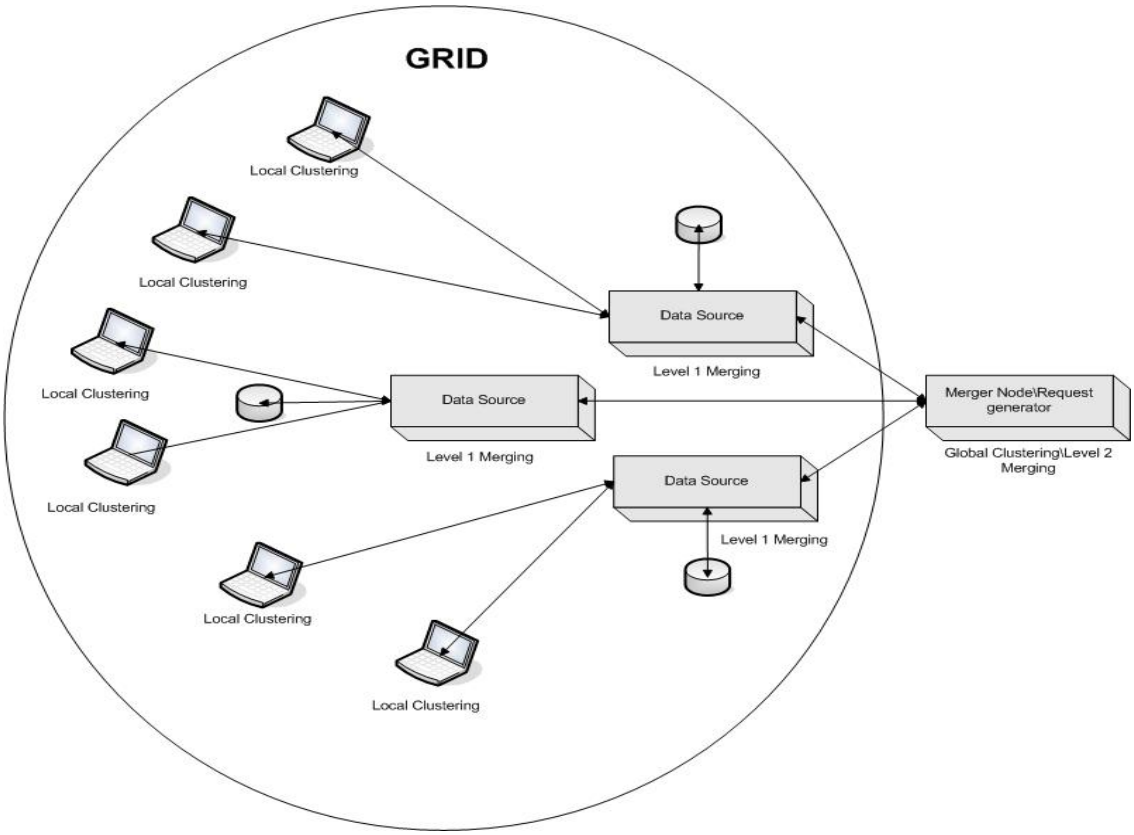

Figure 3.2: Framework of Distributed Grid Based Clustering

This set of centroids will be the starting point for all the available data sources using the DK-Means algorithm.

Let DB_i (I =1…n) be the datasets from distributed sources

For each DB do

Initialize K random centroids {w1,w2…wk}

such that $W_j = a_l$ (j =1…k) (l = 1…m)

Where m is the total instances in the database

{Centroids are selected randomly from the datasets}

Each cluster C_j is associated with the centroid w_j

{Send the centroid set w_j from DB to merger node}

$Global\_Centroid_j = (\sum^{k}_{j=1} w_j)/i$

{Send the centroid set $Global\_Centroid_j$ to all DBs}

{Divide the dataset and send the divided datasets to the execution nodes on the grid}

Do in parallel

Repeat

For each vector input a_l where l=1….z

Where z is the total number of instances given to one node on the grid

Do

Assign a_l to the cluster C_j* with nearest centroid *

(i.e. $|a_l - Global\_Centroid_j^*| <= |a_l – Global\_Centroid_j|$

Where j = {1….k})

For each cluster C_j, do

Update the centroid $Global\_Centroid_j$ to be the centroid of all samples currently in C_j, so that

$$Global\_Centroid_j = \sum_l a_l / |C_j|$$

Compute the error function:

$$E = \sum_j \sum_l |a_l - Centroid_j|^2$$

Until E does not change significantly or cluster membership no longer changes.

{Send $Global\_Centroid_j$ j=1..k to the DB,the data source}

{Combine centroid sets from all execution nodes}

{Send the centroid set from data source to merger node

Combine centroids from all data sources using}

Figure 3.3: The Dk-Means Algorithm

After each data source has received the initial center points, the process of data division takes place. Data on each data source is divided and sent to the execution nodes over the grid. The overlaying Grid scheduler is responsible for scheduling the jobs on the Grid. The following sections explain the important phases of the algorithm.

3.4.2.1 Data Division Process

This is the most important part of the distributed algorithm. Division of data has to be done on the algorithmic level because grid scheduler will not divide the data. If bigger dataset is divided into smaller datasets of same sizes according to

the available nodes, problems faced are the divided dataset can become very small in size which will increase the processing time which will reduce efficiency and the processing power of node may be very low, so it will take much more time as compared to other nodes with more processing power, which will increase the overall processing time.

In order to cater for the first point first of all the numbers of smaller datasets needed to form which will ensure efficiency are determined. From the analysis of testing phase (discussed later in the paper) it was seen that if the size of dataset is smaller than or equal to 500 KB the division and distribution over Grid is not feasible. So any dataset having size less than 500 KB is clustered on the data source only. Keeping this thing in view the number of datasets to be formed can be found as;

$$No\_of\_datasets = Size\ of\ the\ dataset/1024$$

Where 1024 is the size in KB for each dataset. It is intentionally kept higher than the minimum size required because of the reason that dataset can contain different numbers of attributes. If dataset is high dimensional then it would take more space for lesser number of instances.

The second issue was the distribution of data among nodes with different processing power. When data is divided and jobs are created to be sent, the job requirements are specified at that time. The grid scheduler matches the job requirements with the available resources on the GRID, and assigns each job to

a node accordingly. So when the data is divided, minimum resources requirement for the jobs are specified which ensures that job with more requirements will not go to a node with resources lesser than the minimum level which ideally should be available.

3.4.2.2 Submission on Grid

After optimal data division process is completed, the job submission feature of the GRID is used to send divided data to execution nodes. For this a job description file is formed which specifies the requirements of the job, the data to be sent, the code to run on that data, the initial centroids set, and some other parameters. Job requirements are set according to the size of the dataset to be sent. Jobs are prepared and submitted one by one. All the nodes with resources complying with those job requirements will execute the given jobs and they will form the clustering models of the datasets given to them. Then these models are returned to the merger node where all of them are combined to get a global model.

3.4.2.3 Designing a Merger Node

In the Figure 3.2 the merger node is simply a node with some processing power. Merging of models is done at two levels. Firstly, at the data source and, secondly, at the request generator.

3.4.3 Local Cluster Representation

Each cluster in the model is represented by a cluster center and its standard deviation. The cluster center is the mean of all the instances present in that cluster and standard deviation shows the range in which all the instances of a particular cluster lies.

3.4.4 Process

During the process certain notations are used. These are C_j^i represents the j^{th} cluster from i^{th} model. Where j = 1…k (k is the total number of clusters to be formed). $Centroid_j^i$ is the centroid set representing j^{th} cluster of i^{th} model, and n_j^i represents the number of instances in j^{th} cluster of i^{th} model.

The process can be elaborated as first gathering of local models formed at the execution nodes on the GRID at a central location i.e. the data source (First level merger node) then starting from the first cluster from any model formed by any single node, the corresponding cluster from all other models is found this correspondence is formed by finding the clusters with minimum distances between them. Then starting from the 1^{st} model its corresponding clusters C_j^{i*} from other models is the cluster with minimum distance i.e.

$$|C_1^1 - C_j^{i*}| <= |C_1^1 - C_j^i|$$

Now to find the j^{th} cluster of global model, the centroid set of j^{th} cluster from each model are combined as;

$$Global\_center_j = ((\sum_i (Centroid_j^i * n_j^i)) / \sum_i n_i^j$$

Similarly the standard deviations are combined using;

$$Global\_SDev_j = \frac{\sum_i S_i^j - \sum_i Centroid_i^j * n_i^j}{\sum_{i=1}^{N} n_i}$$

Where for every Cluster j and model i:

$$S_i^j = SDev_i^{j^2} n_i^j + Centroid_i^{j^2} n_i^j$$

This completes the global model formation at the data source level. The same process is repeated at 2nd level merger node i.e. the request generator. At that level, the models from all the data sources will be merged together using the above mentioned technique.

This technique ensures reduction in processing time since a virtual Supercomputer (GRID) is used for processing the data. There is also involved a trade-off between accuracy and time. The reduction in accuracy is due to the round-off error caused during the global model formation. But the amount of efficiency achieved in terms of time reduction overcomes that accuracy issue.

3.5 Testing

This section reports on a number of experiments which are conducted to evaluate the DK-Means algorithm. The main goal was to compare the execution time of distributed algorithm with that of the k-means algorithm. Additionally, analyzing the effects that the network issues related to the GRID framework and

the availability of network nodes (execution nodes) had on the execution time. In all the experiments, the basic algorithm that was used to perform the clustering of the data was the classic K-means with the initial center points and the stopping criteria for this algorithm are kept same as those of k-means algorithm when performed on centralized data source. In this case, the initial center points are chosen randomly from the complete data set. The convergence criteria is simple i.e. the algorithm stops when the new centers are not sufficiently different from those generated in the previous iteration.

3.5.1 Testing Dataset

The dataset in the provided experimental results concerned the relative performance of computer processing power on the basis of a number of relevant attributes; each instance represented configurations of single computer. It consisted of 7 attributes (as provided in Figure 3.4). Where

$$PRP = -55.9 + 0.0489\ MYCT + 0.0153\ MMIN + 0.0056\ MMAX + 0.6410\ CACH - 0.2700\ CHMIN + 1.480\ CHMAX.$$

Cycle time (ns) MYCT	Main memory (KB) Min. MMIN	Main memory (KB) Max. MMAX	Cache (KB) CACH	Channels Min. CHMIN	Channels Max. CHMAX	Performance PRP

Figure 3.4: Dataset Attributes

3.5.2 Results and Analysis

Table 3.1 shows the results of tests performed on different number of nodes and dataset with different sets. For better understanding these results are converted to graphical form. As shown in Figure 3.5 to 3.7.

Data set		Clustering Time (sec)			
Size		distributed			
	instances	8 Nodes (4 PC)	6 Nodes (3 PC)	4 Nodes (2 PC)	Centralized
0.5M	17988	32.200	30.500	25.250	16.333
1M	35064	39.000	38.000	42.800	50.500
5M	181463	119.067	121.125	124.000	566.700
10M	362940	254.250	256.833	258.556	616.182
20M	700408	337.000	340.000	491.000	1026.000
30M	1083000	654.875	663.444	887.286	3138.500
50M	1814560	843.167	872.143	1277.750	6038.000

Table 3-1: Clustering Testing Results

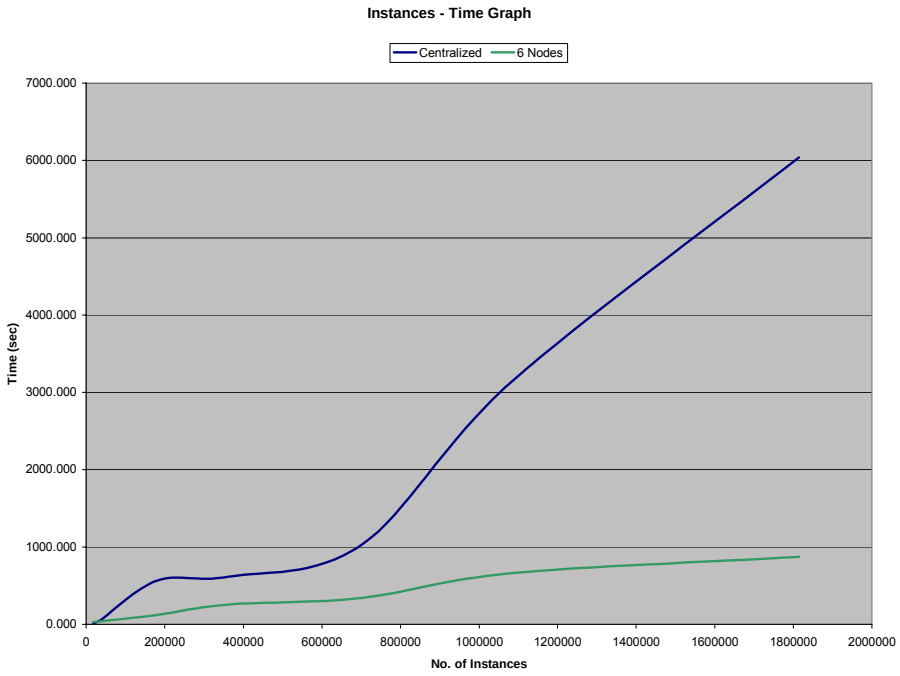

Figure 3.5: Instances – Time Graph with 6 Grid Nodes

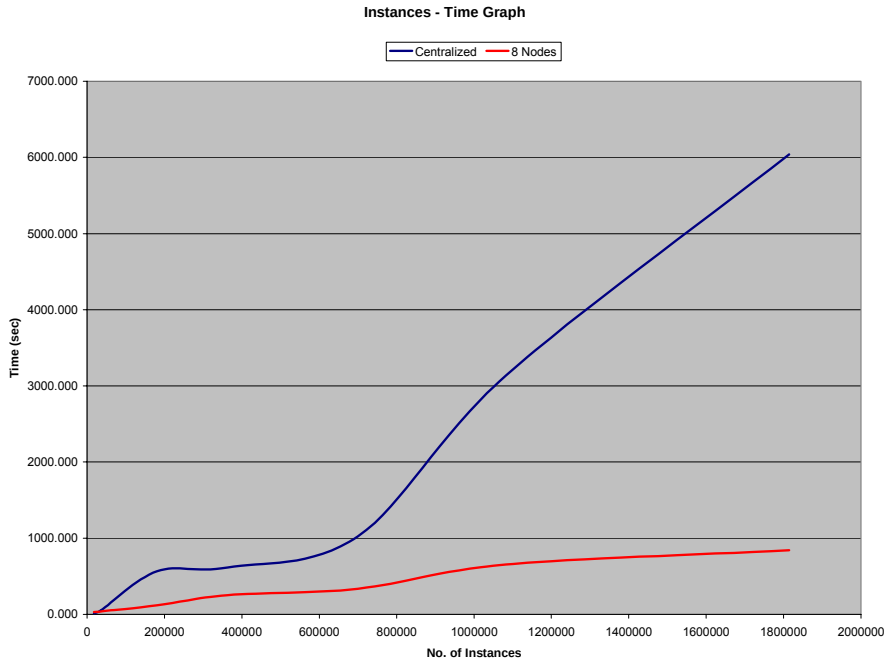

Figure 3.6: Instances – Time Graph with 8 Grid Nodes

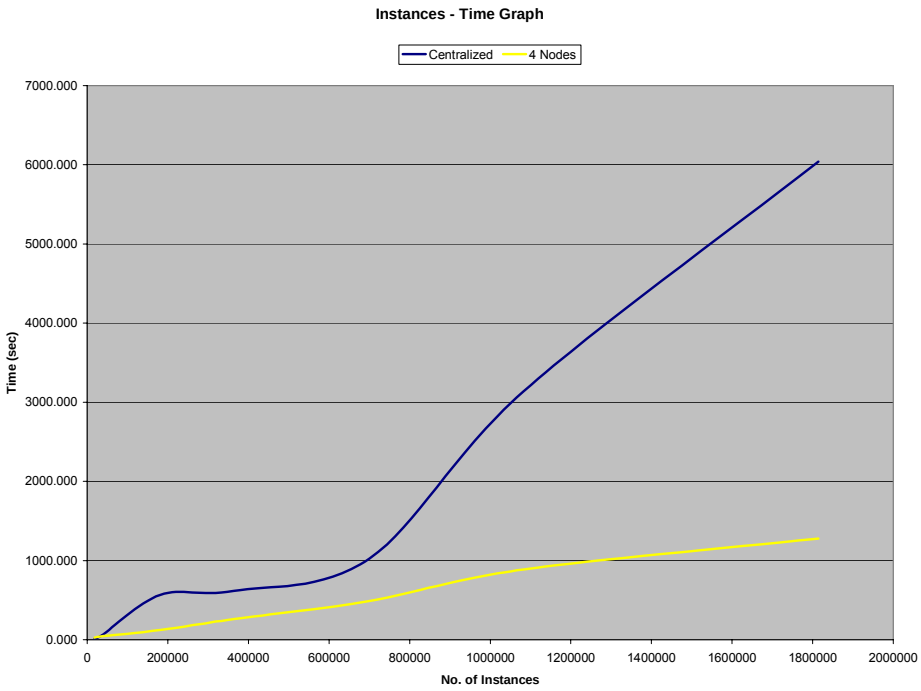

Figure 3.7: Instances – Time Graph with 4 Grid Nodes

The graph shows the trends when execution time was plotted against the number of data instances. The interpolation of the graph points when done for centralized data set can be seen to be increasing drastically as the number of instances grew.

The trends for mining the data across distributed nodes can be seen in the graph, the algorithm DK-Means is suitable for analyzing data across loosely coupled machines.

The interpolation of the graph points when done for centralized data set can be seen to be increasing drastically as the number of instances grew.

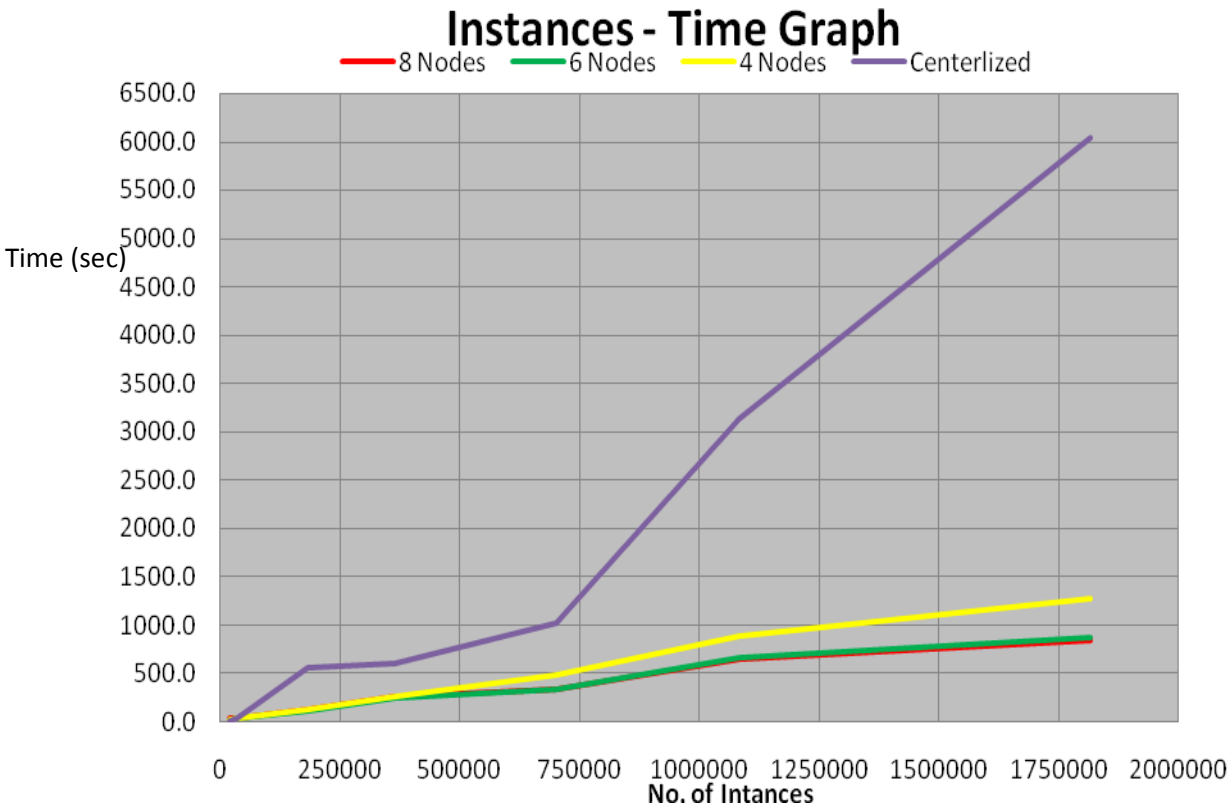

Figure 3.8: Instances – Time Graph of All the Test Results

The trends for mining the data across distributed nodes can be seen in the graph, the algorithm DK-Means is suitable for analyzing data across loosely coupled machines.

The results in Figure 3.5 to 3.7 can be merged to be viewed as in Figure 3.8. It is clear from the shown trends that number of instances of data when plotted against execution time show an exponential increase as the number of instance grow. For centralized data mining this trend, however, is more rapidly

increasing while for a greater number of nodes the performance increases when the data set becomes sufficiently large

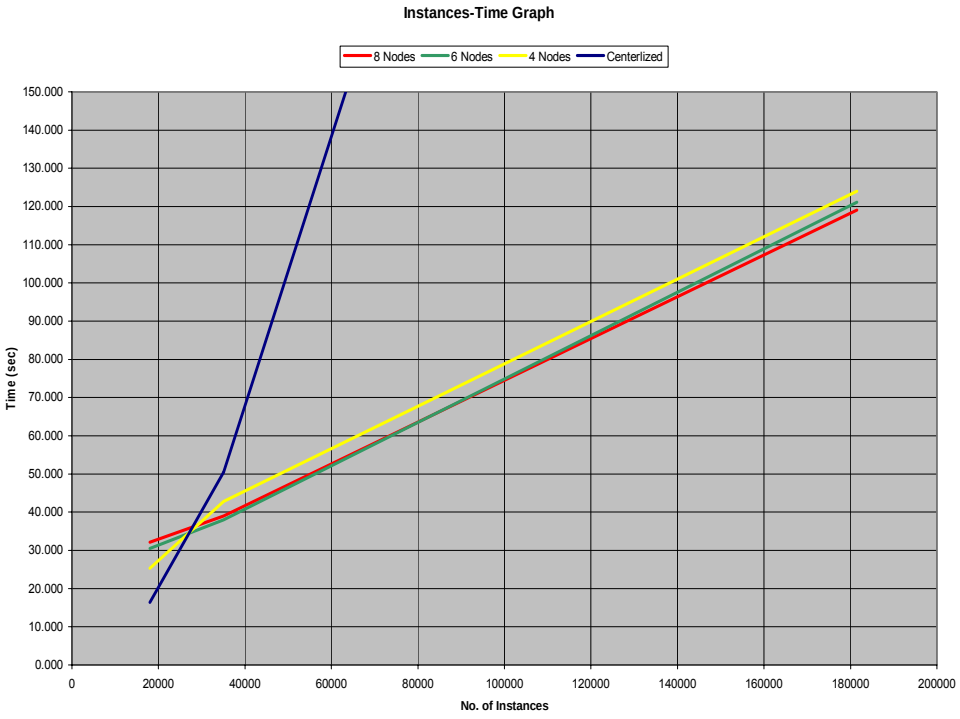

Figure 3.9: Enhanced Instance-Time Graph for Smaller Dataset

However, in Figure 3.9, it can be clearly seen that when the data to be mined has smaller number of instances, centralized data mining is more efficient as the communication and waiting latencies involved in distributing the data for mining tend to overcome the advantages of smaller execution time offered by multiple execution nodes. Performance of 6 nodes clustering is better than 8 nodes when dataset is small. It describes that there must be a minimum dataset size criteria so optimum performance can be achieved. This is something which defines as the optimal performance criteria. For instance, the optimal performance criteria for the figure can be defined as “To achieve the optimal

performance the dataset must contain minimum 4000 instances which approximately counts to 500KB” as shown by the graph .This optimal criteria will vary across different data sets and number of nodes and generalization of this criteria is part of future work of this project.

Chapter 4

Classification

4.1 Introduction

Classification is a predictive modeling task with the specific aim of predicting the value of a single nominal variable based on the known values of other variables. There are many practical situations in which classification is of immense use. Examples include: providing a diagnosis for a medical patient based on a set of test results, estimating the probability of purchase of a given item given the other items purchased, and others [8]. Nowadays all the organizations collect a lot of data from different sources and often these correlated data is stored at many geographically different sites. It is possible that many organizations collect similar data about different peoples. Examples include banks collecting credit card information for their customers or supermarkets collecting transaction information for their clients. On the other hand different organizations may collect different information about the same set of people (also known as vertical partitioning of data). Examples of this include hospitals and insurance companies collecting information or producer / consumer industrial concerns collecting data which can be jointly linked. In all of these cases, mining on the local data is simply not as accurate as mining on the global data. It may lead to inaccurate, even improper results. Thus all corporations would like to leverage their data to get useful knowledge [8].

4.2 Classification Techniques

All the classification techniques use a set of parameters or attributes to classify each object, where these parameters should be relevant to the given task. Classification can be of two types supervised and unsupervised.

In supervised classification, the human expert had determined into, what classes an object may be characterized and also provide a set of object with known classes. This set of objects is called the training set because it is used by the classification program to train the system and form a model. This model will then be used to determine the classes of instances which do not know their classes. In unsupervised learning the human expert doesn't know the training classes beforehand. Although the method requires no user input to create the classified image, the output tends to require a great deal of post classification operations to make the results more meaningful. The four major techniques of supervised classification are discussed below with the arguments to support the selection of Naïve Bayes algorithm for distribution.

4.3 Decision Tree

Decision tree learning s a method of approximating discrete valued functions that is robust to noisy data and capable of learning disjunctive expressions. This technique is useful in domains where data with missing values of some attributes is present. Decision trees are made by finding the gain of the attributes and then finding the attribute with largest gain. The attribute with the

largest gain is made the root node and so on. Gain of any attribute is found out by using the following formula

$$Gain(S,A) = Entropy(S) - \sum|s_v|/|s|Entorpy(S(A))$$

Where, S is the complete data set, A is the attribute for which gain is to be calculated and S_v is the number of instances for which A=v

So in order to build the tree all the data should be placed at each node because, gain of all the attributes is needed to be calculated at each node. If data is distributed then network traffic would be heavily increased. So distributing the decision tree technique is not much feasible.

4.4 Bayesian Learning

Bayesian learning provides a probabilistic approach to inference. It is based on the assumption that the quantities of interest are governed by probability distribution and that optimal decisions can be made by reasoning about these probabilities together with observed data. The Bayesian approach to classifying new instance is to assign the most probable target value, given the attribute values that describe the instance.

$$V_{NB} = argmaxP(v_j)\sum_i P(a_i|v_j)$$

Where V_{NB} denotes the target value output by the naïve bayes classifier.

The learning step of naïve bayes classifier includes calculating the probabilities of different hypothesis based on the observed data. This algorithm can be distributed easily because if data is distributed horizontally on different nodes and calculate probabilities of hypothesis at each node, these probabilities

can be combined together using simple probability formulas to get the global model.

So Bayesian technique can be considered as a candidate technique for distribution.

4.5 Artificial Neural Network (Ann)

ANN provides a general, practical method for learning real-valued, discrete-valued, and vector-valued functions. ANN is robust to errors in the training data. ANNs can be designed using different primitive units like perceptron, linear unit, and sigmoid unit). A number of such primitive units combine together to form a network. The learning part in the neural network is done on these units. Each unit is fined tuned in order to fit the training example, so that they operate correctly on the training data.

If the training data is not present at a central site than in order to correctly tune the network all the data is needed to bring at one place where network is being formed and then train the network. This approach is not feasible because of the fact that data may be in terabytes. Then transferring of data to central location would increase network cost and a lot of time would be lost in transferring of data. Plus it would also result in memory constraints.

So artificial neural networks are not feasible to domain where data is not centralized.

4.6 Algorithm Selection

Survey of classification techniques studied so far reveals that decision tree and ANN cannot be considered as a candidate for distribution, because of the fact that they require data to be present at a central location in order to build the tree or the artificial network. Whereas in Bayesian learning, instances are classified on the basis of probabilities of different hypothesis. These probabilities can be calculated even if the data is distributed. So Bayesian learning algorithms can be used to operate on distributed data.

4.7 Naive Bayes Classifier

The Naive Bayes classifier is a highly practical Bayesian learning method. The following description is based on the discussion of the Naive Bayes classifier in Mitchell [9].

The naive Bayes classifier applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function f(x) can take on any value from some finite set V . A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values < a1, a2...an >. The learner is asked to predict the target value, or classification, for this new instance. The Bayesian approach to classifying the new instance is to assign the most probable target value, vMAP, given the attribute values < a1, a2...an > that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{argmax} \left(P(v_j | a_1, a_2, \ldots, a_n) \right) \quad \text{.....4.1}$$

Using Bayes theorem,

$$v_{MAP} = \underset{v_j \in V}{argmax} \left(\frac{P(a_1, a_2, \ldots, a_n) P(v_j)}{P(a_1, a_2, \ldots, a_n)} \right)$$
$$= \underset{v_j \in V}{argmax} \left(P(a_1, a_2, \ldots, a_n) P(v_j) \right) \quad \text{.....4.2}$$

The Naive Bayes classifier makes the further simplifying assumption that the attribute values are conditionally independent given the target value. Therefore,

$$v_{NB} = \underset{v_j \in V}{argmax} \left(P(v_j) \prod_i P(a_i | v_j) \right) \quad \text{.....4.3}$$

Where, v_{NB} denotes the target value output by the naïve Bayes classifier.

The conditional probabilities $P(a_i|v_j)$ need to be estimated from the training set. The prior probabilities $P(v_j)$ also need to be fixed in some fashion (typically by simply counting the frequencies from the training set). The probabilities for differing hypotheses (classes) can also be computed by normalizing the values received for each hypothesis (class). Probabilities are computed differently for nominal and numeric attributes.

4.8 Training and Classification

The model parameters (with the probabilities) are computed from the training data. The procedure for computing the probabilities is different for nominal and numeric attributes. For a nominal attribute X, with r possible attributes values x1...xr the probability $P(X = x_k|v_j) = n_j/n$

Where n is the total number of training examples for which V = vj , and nj is the number of those training examples which also have X = x_k.

For a numeric attribute, in the simplest case, the attribute is assumed to have a “normal” or “Gaussian” probability distribution, N(μ, Φ^2). The mean μ and variance Φ^2 are calculated for each class and each numeric attribute from the training set. Now the required probability that the instance is of the class v_j, P(X = x_0|vj), can be estimated by substituting x = x_0 in the probability density equation. An instance is classified as per equation 4.3. Thus the conditional probability of a class given the instance is calculated for all classes, and the class with the highest relative probability is chosen as the class of the instance.

4.9 Distributed Naïve Bayesian

Distributed naïve Bayesian algorithm aims at solving the problem of geographic distribution of data. Two issues are addressed, selecting model parameters and to classify a new instance.

In the below mentioned algorithm the basic working is same as that of Naïve Bayes algorithm but the difference is that it caters for the distributed data. At each location standard naïve bayes algorithm is run to create the model using the training data present at that location. Once these models are created they are sent to the centeral server which then combines models form all the sources and form a global model. This global model is then used to classify new instances.

4.9.1 Computing Required Values / Setup

At each data source the procedures for calculating the parameters are different for nominal attributes and numeric attributes. They are described in the subsections below.

4.9.1.1 Nominal Attributes

For a nominal attribute, the conditional probability that an instance belongs to a certain class c given that the instance

$$P(C = c|A = a) \quad = \quad \frac{P(C = c \cap A = a)}{P(A = a)} = \frac{n_{ac}}{n_a} \qquad \text{.....4.4}$$

Where n_{ac} is the number of instances in the training set which have the class value c and an attribute value of a, while n_a is the number of instances which simply have an attribute value of a. Thus, the necessary parameters are simply the counts of instances, n_{ac} and n_a. Each data source locally computes the local count of instances. After each source has the required information for the model formation, it forms a model of its local data. These models are sent to a central node where all are combined to form a global model. The local model of data contains information about the total number of instances, count of instances with specific attribute value and class value. Sample model is shown in Figure 4.1. The sample model shows that 63% instances of the complete dataset has class value yes and 37% has value no. Similarly other statements shows the counts of instances of specific attribute value with a specific class. Assuming that the total number of instances is public, the required probability can simply be

computed by dividing the appropriate sums. For an attribute a with l different attribute values, and a total of r distinct classes, l * r different counts need to be computed for each combination of attribute value and class value. For each attribute value a total instance count also needs to be computed, which gives l additional counts.

Naive Bayes Classifier

Class yes: Prior probability = 0.63

outlook: Discrete Estimator. Counts = 3 5 4 (Total = 12)
temperature: Discrete Estimator. Counts = 3 5 4 (Total = 12)
humidity: Discrete Estimator. Counts = 4 7 (Total = 11)
windy: Discrete Estimator. Counts = 4 7 (Total = 11)

Class no: Prior probability = 0.38

outlook: Discrete Estimator. Counts = 4 1 3 (Total = 8)
temperature: Discrete Estimator. Counts = 3 3 2 (Total = 8)
humidity: Discrete Estimator. Counts = 5 2 (Total = 7)
windy: Discrete Estimator. Counts = 4 3 (Total = 7)

Figure 4.1: Bayes Model of Weather Dataset with 4 Attributes

4.9.1.2 Numeric Attributes

For a numeric attribute, the necessary parameters are the mean μ and variance Φ^2 for all the different classes. Again, the necessary information is split between the parties. In order to compute the mean, each party needs to sum the attribute values of the appropriate instances having the same class value. These

local sums are added together and divided by the total number of instances having that same class to get the mean for that class value. Once all of the means μ_y are known, it is quite easy to compute the variance $\Phi^2{}_y$, for all class values.

4.9.2 Merger Node:

Merger node refers to node where all the models are combined. In case of nominal attributes the mergence is simple. The count of instances of each attribute value having a specific class value from each model are simply added together to get the global count. Whereas for numerical attributes, the means from all the sources are combined together to get a global mean using eq. 4.5

$$\text{Global\_Mean}_j = (\sum_I (\mu_{ji} * N_{ji})) / \sum_I N_{ij} \quad \text{.....4.5}$$

where μ_j^i is the mean of jth attribute in ith model and n_j^i is the count of instances with attribute value j in model i.

Similarly, standard deviation Φ can also be computed using the statistical formula given in eq. 4.6.

$$Global_{SDev_j} = \frac{\sum_i s_i^j - \sum_i \mu_i^j * n_i^j}{\sum_{i=1}^{N} n_i} \quad \text{.....4.6}$$

Where for every numeric attribute j in model i:

$$S_i^j = SDev_i^{j^2} n_i^j + Centroid_i^{j^2} n_i^j \quad \text{.....4.7}$$

Once these parameters are combined, the global model formation is complete and this model can then be used to classify new instances. Classification of new instances is same as is standard Naïve Bayes algorithm.

4.10 Testing

This section reports on experiments that are conducted to evaluate the Distributed Naïve Bayes algorithm. The main goal was to compare the accuracy of new algorithm with that of the traditional Naïve Bayes algorithm.

4.10.1 Testing Dataset

For testing purpose 3 kinds of datasets were used. First dataset was “Contact lens” dataset, having records of patients with their symptoms and prescribed lens type, all its attribute values were nominal. Second dataset was “Iris plant” dataset, having attributes of plants and its corresponding type; all its attributes were numerical other than class attribute. Third dataset was “Weather” dataset, having weather conditions on a particular day and class attribute having decision of playing sports or not, some of its attributes were numerical and some were nominal including class attribute.

4.10.2 Results and Analysis

Table 4.1 shows the results of tests performed on different on different datasets. As it can been seen from the table, error on dataset containing all nominal values is almost zero. But in case of numerical dataset, small error of 1 to 2 percent is there.

	Training Set Instances	Testing Set Instances	Correctly Classified Instances	Percentage
		Contact-lens		
Centralized:	24	134522	44848	33.3388%
Distributed:	24	134522	44848	33.3388%
		Iris		
Centralized:	150	150	144	96.0000%
Distributed:	150	150	144	96.0000%
		Weather		
Centralized:	14	2291403	1145491	49.9908%
Distributed:	14	2291403	1145308	49.9828%
Centralized:	14	4582742	2289742	49.9645%
Distributed:	14	4582742	2288872	49.9455%

Table 4-1: Classification Testing Results

Means and standard deviations of numerical values are calculated locally on each node, and then transmitted to calculate global mean and global standard deviation. These local means and standard deviations are truncated to transmit, which causes small error. But this error is minute.

4.11 Conclusion

In this chapter, it has been shown that distributed Naïve Bayes algorithm provides an efficient and accurate way to perform the predictive modeling task in a highly distributive environment. The results of tests performed on the newly formed algorithm also show that of the model is not compromised during data distribution.

Chapter 5

Testing and Analysis

5.1 Introduction

Software Testing is the process of executing a program or system with the intent of finding errors. Or, it involves any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. [10]

5.2 Unit Testing

Unit testing tests the minimal software component, or module. Each unit (basic component) of the software is tested to verify that the detailed design for the unit has been correctly implemented. In an Object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors [11]. Every unit of Grid Data Miner was tested individually and these tests were successful.

5.3 Integration Testing

Integration testing exposes defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system. While

integrating modules of Grid Data Miner, integration testing was performed after integrating every unit and the integration process was continued only when integration testing was successful.

5.4 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. On the implemented system, system testing was performed in three phases. These are performance testing, reliability testing and security testing.

5.4.1 Performance Testing

Not all software systems have specifications on performance explicitly. But every system will have implicit performance requirements. The software should not take infinite time or infinite resource to execute. "Performance bugs" sometimes are used to refer to those design problems in software that cause the system performance to degrade [10]. Grid Data Miner’s performance was tested. It was seen that on every kind of input value, system checks for exceptions and catches it by itself. And also it returns results in very less time and never take infinite time or infinite resources because resources are managed by the grid middleware.

5.4.2 Reliability Testing

Software reliability refers to the probability of failure-free operation of a system. It is related to many aspects of software, including the testing process. Directly estimating software reliability by quantifying its related factors can be difficult. Testing is an effective sampling method to measure software reliability. Grid Data Miner was tested for reliability. Its reliability depends on network reliability, data source availability and grid nodes reliability. Grid nodes reliability is catered by grid middle ware, so grid nodes are reliable. Data source availability is a troubling matter. But it was catered in the design of software that data source failure did not affect the reliability of the software. If a data source is down, system indicates itself that data source is no longer available. So the system reliability depends on network reliability, network should be reliable within the grid cluster.

5.4.3 Security Testing

Software quality, reliability and security are tightly coupled. Flaws in software can be exploited by intruders to open security holes. With the development of the Internet, software security problems are becoming even more severe. Grid Data Miner is completely secure. Because every time when a user wants to mine data through Grid Data Miner, he must have to log in using he own id and password. So there are no security holes.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Many experts in IT, science, finance and commerce are recognizing the importance of scalable data mining solutions in their business. *So now there is need to take the techniques that have been developed for things like business intelligence and data mining that goes on around that and think how those can be applied in these realms as well, how to take every step of the process and have it be very visual and only require as much software understanding as is absolutely necessary.* It can be concluded that the importance of high-performance data mining is going to be considered a real added value. In this scenario, the Grid can offer an effective infrastructure for deploying data mining and knowledge discovery applications. It can represent in a near future an effective infrastructure for managing very large data sources and providing high-level mechanisms for extracting valuable knowledge from them. To solve this class of tasks, advanced tools and services for knowledge discovery are vital. Here it's presented solution to problem of compute intensive nature of clustering and classification techniques of data mining. This solution provides professionals and scientists to have a quick access to the analysis of their stored data. The future use of the Grid is mainly related to its ability embody many of those properties and to manage world-wide complex distributed applications. Among those, knowledge-based applications are a major goal. Data warehouse solution providers can include

grid based data mining solution to incorporate large execution time problem of data mining applications.

6.2 Future Work

Future work in this direction includes modifying algorithms of other data mining techniques like association rule mining, pattern recognition etc to make them grid enabled. The solution provided in this project is only for distributed homogeneous dataset .It can be extended to incorporate heterogeneous datasets.

A plug and play data mining tool can be developed which contain grid enabled algorithms and will use the underlying gird infrastructure to perform mining.

Bibliography:

[1] I. Foster, C. Kesselman, S. Tuecke. “The Anatomy of the Grid: Enabling Scalable Virtual Organizations”. *International J. Supercomputer Applications*, 2001.

[2] F. Berman, A. Hey and G. Fox, “Grid Computing – Making the Global Infrastructure a Reality” John Wiley & Sons, Ltd 2003 ISBN: 0-470-85319-0

[3] I. Foster, “What is the Grid? A Three Point Checklist. GRIDToday”, July 20, 2002.

[4] Condor, http://www.cs.wisc.edu/condor/.

[5] “*Application Experiences with the Globus Toolkit”*, Brunett, S. *et al*. (1998) *Proc. 7th IEEE Symp. on High Performance Distributed Computing*, IEEE Press, 1998,

[6] “Mark Ellisman’s Telescience application” http://www.npaci.edu/Alpha/telescience.html

[7] J. B. MacQueen “Some methods for Classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium, on Mathematical Statistics and Probability”, Berkeley, University of California Press, 1967

[8] “Privacy Preserving Naive Bayes Classifier for Horizontally Partitioned Data” by Murat Kantarcıo˘glu

[9] T. Mitchell "*Machine Learning"*. McGraw-Hill Science/Engineering/Math, 1st edition, 1997.

[10] Sofware Testing "http://www.ece.cmu.edu/~koopman/des_s99/sw_testing/", Spring 1999

[11] Binder, Robert V. "*Testing Object-Oriented Systems: Objects, Patterns, and Tools"*. Addison-Wesley Professional, 1999. ISBN 0-201-80938-9.

[12] "*Algorithms for Clustering Data*." Prentice-Hall International, 1988

[13] Pavel Berkhin. "Survey of clustering data mining techniques". Technical report, Accrue Software, 2002.

[14] Joydeep Ghosh. "Scalable clustering methods for data mining". Lawrence Ealbaum Assoc, 2003.