3

Parallel and Distributed Data Mining

Dr (Mrs). Sujni Paul
Karunya University
Coimbatore,
India

1. Introduction

Data mining is a process of nontrivial extraction of implicit, previously unknown, and potentially useful information (such as knowledge rules, constraints, and regularities) from data in databases. In fact, the term "knowledge discovery" is more general than the term "data mining." Data mining is usually viewed as a step towards the process of knowledge discovery, although these two terms are considered as synonyms in the computer literature. The entire life cycle of knowledge discovery includes steps such as data cleaning, data integration, data selections, data transformation, data mining, pattern evaluation, and knowledge presentation.

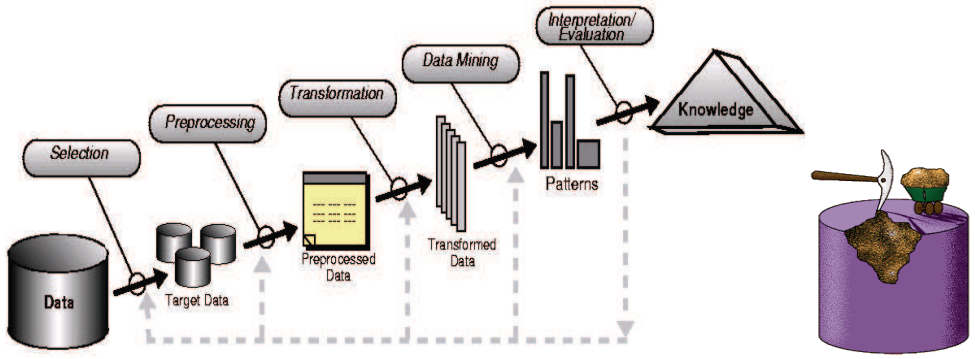


Fig. 1. Life Cycle of knowledge presentation

Data cleaning is to remove noise and inconsistent data. Data integration is to combine data from multiple data sources, such as a database and data warehouse. Data selection is to retrieve data relevant to the task. Data transformation is to transform data into appropriate forms. Data mining is to apply intelligent methods to extract data patterns. Pattern evaluation is to identify the truly interesting patterns based on some interestingness measures. Knowledge evaluation is to visualize and present the mined knowledge to the user. There are many data mining techniques, such as association rule mining, classification, clustering, sequential pattern mining, etc.
Since this chapter focuses on parallel and distributed data mining, let us turn our attention to those concepts.

2. Distributed data mining

Data mining algorithms deal predominantly with simple data formats (typically flat files); there is an increasing amount of focus on mining complex and advanced data types such as object-oriented, spatial and temporal data. Another aspect of this growth and evolution of data mining systems is the move from stand-alone systems using centralized and local computational resources towards supporting increasing levels of distribution. As data mining technology matures and moves from a theoretical domain to the practitioner's arena there is an emerging realization that distribution is very much a factor that needs to be accounted for.

Databases in today's information age are inherently distributed. Organizations that operate in global markets need to perform data mining on distributed data sources (homogeneous / heterogeneous) and require cohesive and integrated knowledge from this data. Such organizational environments are characterized by a geographical separation of users from the data sources. This inherent distribution of data sources and large volumes of data involved inevitably leads to exorbitant communications costs. Therefore, it is evident that traditional data mining model involving the co-location of users, data and computational resources is inadequate when dealing with distributed environments. The development of data mining along this dimension has lead to the emergence of distributed data mining. The need to address specific issues associated with the application of data mining in distributed computing environments is the primary objective of distributed data mining. Broadly, data mining environments consist of users, data, hardware and the mining software (this includes both the mining algorithms and any other associated programs). Distributed data mining addresses the impact of distribution of users, software and computational resources on the data mining process. There is general consensus that distributed data mining is the process of mining data that has been partitioned into one or more physically/geographically distributed subsets.

The significant factors, which have led to the emergence of distributed data mining from centralized mining, are as follows:

- The need to mine distributed subsets of data, the integration of which is non- trivial and expensive.
- The performance and scalability bottle necks of data mining.
- Distributed data mining provides a framework for scalability, which allows the splitting up of larger datasets with high dimensionality into smaller subsets that require computational resources individually.

Distributed Data Mining (DDM) is a branch of the field of data mining that offers a framework to mine distributed data paying careful attention to the distributed data and computing resources. In the DDM literature, one of two assumptions is commonly adopted as to how data is distributed across sites: homogeneously and heterogeneously. Both viewpoints adopt the conceptual viewpoint that the data tables at each site are partitions of a single global table. In the homogeneous case, the global table is horizontally partitioned. The tables at each site are subsets of the global table; they have exactly the same attributes. In the heterogeneous case the table is vertically partitioned, each site contains a collection of columns (sites do not have the same attributes). However, each tuple at each site is assumed to contain a unique identifier to facilitate matching. It is important to stress that the global table viewpoint is strictly conceptual. It is not necessarily assumed that such a table was physically realized and partitioned to form the tables at each site.

3. Parallel and distributed data mining

The enormity and high dimensionality of datasets typically available as input to the problem of association rule discovery, makes it an ideal problem for solving multiple processors in parallel. The primary reasons are the memory and CPU speed limitations faced by single processors. Thus it is critical to design efficient parallel algorithms to do the task. Another reason for parallel algorithm comes from the fact that many transaction databases are already available in parallel databases or they are distributed at multiple sites to begin with. The cost of bringing them all to one site or one computer for serial discovery of association rules can be prohibitively expensive.

For compute-intensive applications, parallelisation is an obvious means for improving performance and achieving scalability. A variety of techniques may be used to distribute the workload involved in data mining over multiple processors. Four major classes of parallel implementations are distinguished. The classification tree in Figure 1 demonstrates this distinction. The first distinction made in this tree is between *task parallel* and *data-parallel* approaches.

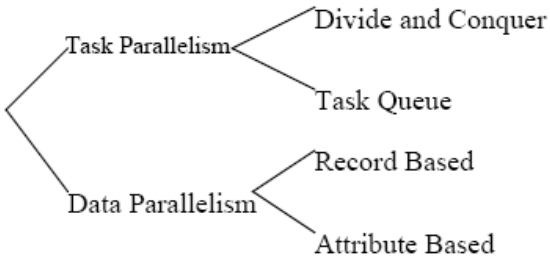


Fig. 2. Methods of Parallelism

Task-parallel algorithms assign portions of the search space to separate processors. The task parallel approaches can again be divided into two groups. The first group is based on a *Divide and Conquer* strategy that divides the search space and assigns each partition to a specific processor. The second group is based on a task queue that dynamically assigns small portions of the search space to a processor whenever it becomes available. A task parallel implementation of decision tree induction will form tasks associated with branches of the tree. A Divide and Conquer approach seems a natural reflection of the recursive nature of decision trees.

However the task of parallel implementation suffers from load balancing problems caused by uneven distributions of records between branches. The success of a task parallel implementation of decision trees seems to be highly dependent on the structure of the data set. The second class of approaches, called data parallel, distributes the data set over the available processors. Data-parallel approaches come in two flavors. A partitioning based on records will assign non-overlapping sets of records to each of the processors. Alternatively a partitioning of attributes will assign sets of attributes to each of the processors. Attribute-based approaches are based on the observation that many algorithms can be expressed in terms of primitives that consider every attribute in turn. If attributes are distributed over multiple processors, these primitives may be executed in parallel. For example, when constructing decision trees, at each node in the tree, all independent attributes are considered, in order to determine the best split at that point.

There are two basic parallel approaches that have come to be used in recent times – *work partitioning* and *data partitioning*.

Work Partitioning - These methods assign different view computations to different processors. Consider, for example, the lattice for a four dimensional data cube. If a name is assigned to the dimensions as "ABCD", 15 views need to be computed. Given a parallel computer with p processors, work partitioning schemes partition the set of views into p groups and assign the computation of the views in each group to a different processor. The main challenges for these methods are load balancing and scalability.

Data Partitioning - These methods work by partitioning the raw data set into p subsets and store each subset locally on one processor. All views are computed on every processor but only with respect to the subset of data available at each processor. A subsequent *merge* procedure is required to agglomerate the data across processors. The advantage of data partitioning methods is that they do not require all processors to have access to the entire raw data set. Each processor only requires a local copy of a portion of the raw data which can, e.g., be stored on its local disk. This makes such methods feasible for shared-nothing parallel machines.

4. Why parallelize data mining?

Data-mining applications fall into two groups based on their intent. In some applications, the goal is to find explanations for the most variable elements of the data set that is, to find and explain the *outliers*. In other applications, the goal is to understand the variations of the majority of the data set elements, with little interest in the outliers. Scientific data mining seems to be mostly of the first kind, whereas commercial applications seem to be of the second kind ("understand the buying habits of most of our customers"). In applications of the first kind, parallel computing seems to be essential. In applications of the second kind, the question is still open because it is not known how effective sampling from a large data set might be at answering broader questions. Parallel computing thus has considerable potential as a tool for data mining, but it is not yet completely clear whether it represents the future of data mining.

5. Technologies

- **Parallel computing**
  Single systems with many processors work on same problem.
- **Distributed computing**
  Many systems loosely coupled by a scheduler to work on related problems.
- **Grid Computing (Meta Computing)**
  Many systems tightly coupled by software, perhaps geographically distributed, are made to work together on single problems or on related problems.

5.1 Properties of algorithms for association discovery

Most algorithms for association discovery follow the same general procedure, based on the sequential *Apriori* algorithm. The basic idea is to make multiple passes over the database, building larger and larger groups of associations on each pass. Thus, the first pass determines the "items" that occur most frequently in all the transactions in the database; each subsequent pass builds a list of possible frequent item tuples based on the results of the

previous pass, and then scans the database, discarding those tuples that do not occur frequently in the database. The intuition is that for any set of items that occurs frequently, all subsets of that set must also occur frequently.
Notice that, for large association sets, this algorithm and its derivatives must make many passes over a potentially enormous database. It is also typically implemented using a hash tree, a complex data structure that exhibits very poor locality (and thus poor cache behavior).
Although there exist workable sequential algorithms for data mining (such as Apriori, above), there is a desperate need for a parallel solution for most realistic-sized problems. The most obvious (and most compelling) argument for parallelism revolves around database size. The databases used for data mining are typically extremely large, often containing the details of the entire history of a company's standard transactional databases. As these databases grow past hundreds of gigabytes towards a terabyte or more, it becomes nearly impossible to process them on a single sequential machine, for both time and space reasons: no more than a fraction of the database can be kept in main memory at any given time, and the amount of local disk storage and bandwidth needed to keep the sequential CPU supplied with data is enormous. Additionally, with an algorithm such as Apriori that requires many complete passes over the database, the actual running time required to complete the algorithm becomes excessive.
The basic approach to parallelizing association-discovery data mining is via database partitioning. Each available node in the networking environment is assigned a subset of the database records, and computes independently on that subset, usually using a variation on the sequential *Apriori* algorithm. All of the parallel data mining algorithms require some amount of global all-all or all-one communication to coordinate the independent nodes.

5.2 Problems in developing parallel algorithms for distributed environment

There are several problems in developing parallel algorithms for a distributed environment with association discovery data mining which is being considered in this research work. These are:

- **Data distribution:** One of the benefits of parallel and distributed data mining is that each node can potentially work with a reduced-size subset of the total database. A parallel algorithm in distributed environment must effectively distribute data to allow each node to make independent progress with its incomplete view of the entire database.
- **I/O minimization:** Even with good data distribution, parallel data mining algorithms must strive to minimize the amount of I/O they perform to the database.
- **Load balancing:** To maximize the effect/efficiency of parallelism, each workstation must have approximately the same amount of work to do. Although a good initial data distribution can help provide load-balancing, with some algorithms, periodic data redistribution is required to obtain good overall load-balancing.
- **Avoiding duplication:** Ideally, no workstation should do redundant work (work already performed by another node).
- **Minimizing communication:** An ideal parallel data mining algorithm allows all workstations to operate asynchronously, without having to stall frequently for global barriers or for communication delays.
- **Maximizing locality:** As in all performance programming, high-performance parallel data mining algorithms must be designed to reap the full performance potential of

hardware. This involves maximizing locality for good cache behavior, utilizing as much of the machine's memory bandwidth as possible, etc.

Achieving all of the above goals in one algorithm is nearly impossible, as there are tradeoffs between several of the above points. Existing algorithms for parallel data mining attempt to achieve an optimal balance between these factors.

5.3 Algorithms in parallel and distributed data mining

The major algorithms used for parallel and distributed data mining are:

- **Count Distribution:** this algorithm achieves parallelism by partitioning data. Each of N workstations gets $1/N^{th}$ of the database, and performs an *Apriori*-like algorithm on the subset. At the end of each iteration however, is a communication phase, in which the frequency of item occurrence in the various data partitions is exchanged between all workstations. Thus, this algorithm trades off I/O and duplication for minimal communication and good load-balance: each workstation must scan its database partition multiple times (causing a huge I/O load) and maintains a full copy of the (poor-locality) data structures used (causing duplicated data structure maintenance), but only requires a small amount of per-iteration communication (an asynchronous broadcast of frequency counts) and has a good distribution of work.
- **Data Distribution:** This algorithm is designed to minimize computational redundancy and maximize use of the memory bandwidth of each workstation. It works by partitioning the current maximal-frequency itemset candidates (like those generated by *Apriori*) amongst work stations. Thus, each workstation examines a disjoint set of possibilities; however, each workstation must scan the entire database to examine its candidates. Thus this algorithm trades off a huge amount of communication (to fetch the database partitions stored on other workstations) for better use of machine resources and to avoid duplicated work.
- **Candidate Distribution:** This algorithm is similar to data distribution in that it partitions the candidates across workstations, but it attempts to minimize communication by selectively partitioning the database such that each workstation has locally the data needed to process its candidate set. It does this after a fixed (small) number of passes of the standard data distribution algorithm. This trades off duplication (the same data may need to be replicated on more than one node) and poor load-balancing (after redistributing the data, the workload of each workstation may not be balanced) in order to minimize communication and synchronization. The effects of poor load balancing are mitigated somewhat, since global barriers at the end of each pass are not required.
- **Eclat:** This sophisticated algorithm avoids most of the tradeoffs above by using an initial clustering step to pre-process the data before partitioning it between workstations. It thus achieves many of the benefits of candidate distribution without the costs. Little synchronization or communication is needed, since each node can process its partitioned dataset independently. A transformation of the data during partitioning allows the use of simple database intersections (rather than hash trees), maximizing cache locality and memory bandwidth usage. The transformation also drastically cuts down the I/O bandwidth requirements by only necessitating three database scans.

6. Role of intelligent agents in distributed data mining

Agents are defined as software or hardware entities that perform some set of tasks on behalf of users with some degree of autonomy. In order to work for somebody as an assistant, an agent has to include a certain amount of *intelligence*, which is the ability to choose among various courses of action, plan, communicate, adapt to changes in the environment, and learn from experience. In general, an intelligent agent can be described as consisting of a *sensing* element that can receive events, a *recognizer* or *classifier* that determines which event occurred, a *set of logic* ranging from hard-coded programs to rule-based inferencing, and a *mechanism* for taking action.

Data mining agents seek data and information based on the profile of the user and the instructions she gives. A group of flexible data-mining agents can co-operate to discover knowledge from distributed sources. They are responsible for accessing data and extracting higher-level useful information from the data. A data mining agent specializes in performing some activity in the domain of interest. Agents can work in parallel and share the information they have gathered so far.

Pericles A. Mitkas et al's work on Software agent technology has matured enough to produce intelligent agents, which can be used for controlling a large number of concurrent engineering tasks. Multi-agent systems are communities of agents that exchange information and data in the form of messages. The agents' intelligence can range from rudimentary sensor monitoring and data reporting, to more advanced forms of decision making and autonomous behavior. The behavior and intelligence of each agent in the community can be obtained by performing data mining on available application data and the respected knowledge domain. An Agent Academy a software platform is designed for the creation, and deployment of multiagent systems, which combines the power of knowledge discovery algorithms with the versatility of agents. Using this platform, agents are equipped with a data-driven inference engine, can be dynamically and continuously trained. Three prototype multi-agent systems are developed with Agent Academy.

Agent-based systems belong to the most vibrant and important areas of research and development to have emerged in information technology. Because of the lively extensive spreading of directions in research no publicly accepted solid definitions of agent-based systems and their elements – agents is provided. Hence, in context of this paper some general definitions are used: Software agent is software that acts as an agent for another as in a relationship of agency. When several agents act they may form a multi-agent system. Intelligent Agent (IA) refers to a software agent that exhibits some form of artificial intelligence. According to Wooldridge intelligent agents are defined as agents, capable of flexible autonomous action to meet their design objectives. They must involve:

- Reactivity: to perceive and respond in a timely fashion to changes occurring in their environment in order to satisfy their design objectives. The agent's goals and/or assumptions that form the basis for a procedure that is currently executed may be affected by a changed environment and a different set of actions may have to be performed.
- Pro–activeness: ability to exhibit goal-directed behavior by taking the initiative, responding to changes in their environment in order to satisfy their design objectives.
- Sociability: capability of interacting with other agents (software and humans) through negotiation and/or cooperation to satisfy their design objectives.

During the mining process the mobile intelligent agents can be used as it keeps monitoring the different workstations in the geographically distributed areas.

7. Architectures

Agent-based distributed data mining systems employ one or more agents to analyze and model local datasets, which generate local models. These local models generated by individual agents can then be composed into one or more new 'global models' based on different learning algorithms, for instance, JAM and BODHI. JAM Java Agents for Meta-learning is a Java-based distributed data mining system that uses a meta-learning technique. The architecture consists of local databases of several financial institutes, learning agents and meta-learning agents. Agents operate on a local database and generate local classifiers. These local classifiers then are imported to a data location where they can be aggregated into a global model using meta-learning.

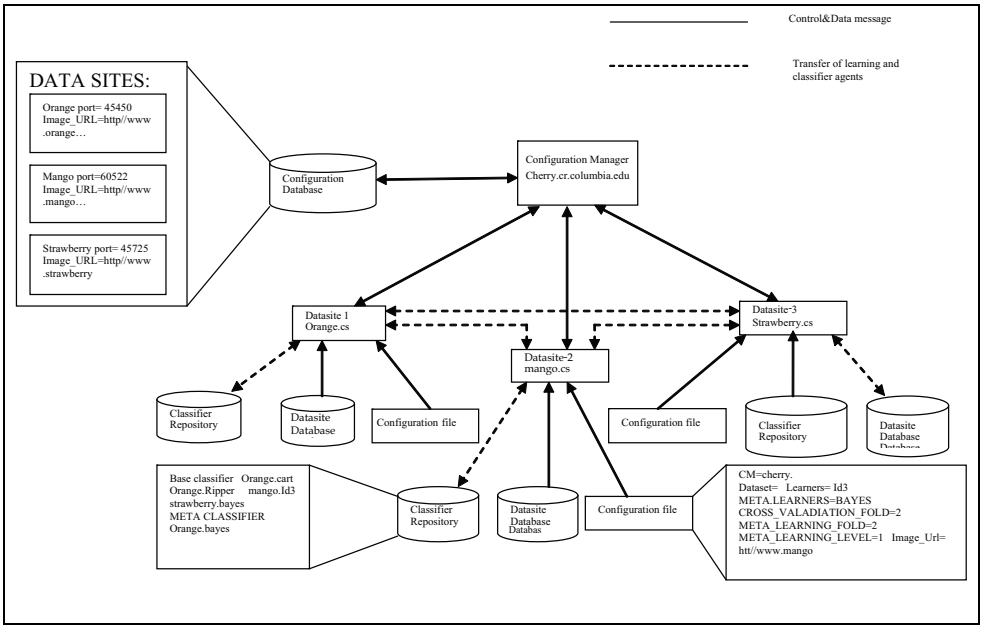


Fig. 3. JAM architecture with 3 datasites

BODHI is a Java and agent based distributed data mining system. BODHI also notes the importance of mobile agent technology. As all of agents are extensions of a basic agent object, BODHI can easily transfer an agent from one site to another site, along with the agent's environment, configuration, current state and learned knowledge. Figure 2.1 shows the BODHI architecture.

PADMA Architecture

The PADMA is an agent based architecture for parallel / distributed data mining. The goal of this effort is to develop a flexible system that will exploit data mining agents in parallel. Its initial implementation used agents specializing in unstructured text document classification.

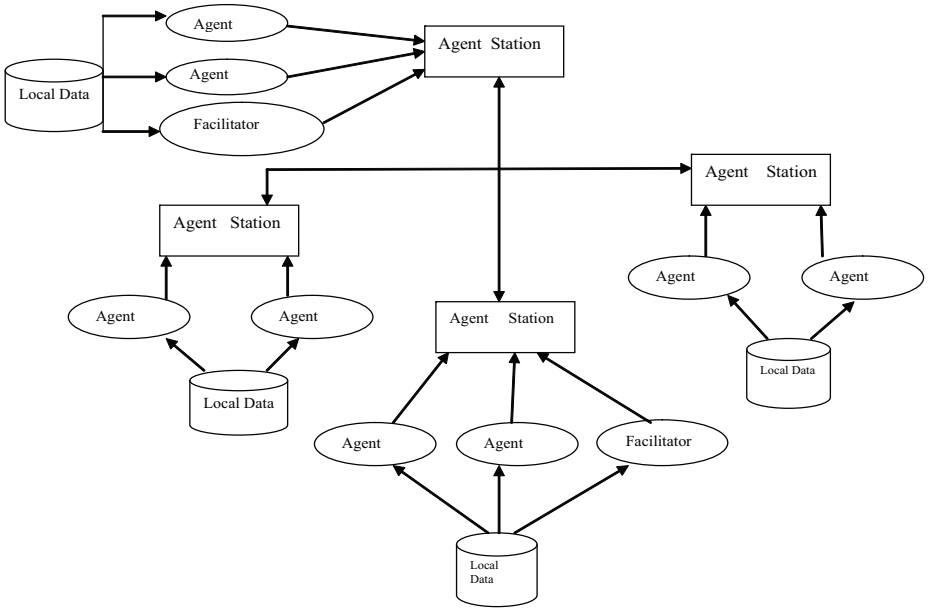


Fig. 4. BODHI: Agent-based distributed data mining system

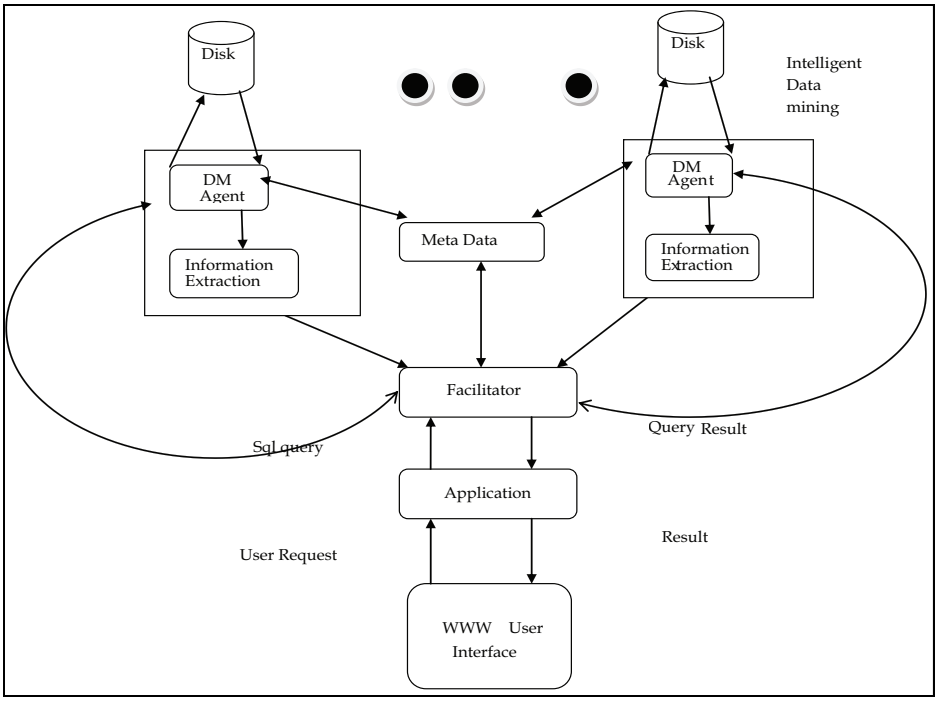


Fig. 5. PADMA architecture

PADMA agents for dealing with numeric data are currently under development. The main structural components of PADMA are 1. Data mining agents 2. Facilitator for coordinating the agents and 3. User interface. Agents work in parallel and share their information through facilitator.

8. Mathematical modeling

Distributed Data Mining (DDM) aims at extraction of useful pattern from distributed heterogeneous databases in order, for example, to compose them within a distributed knowledge base and use for the purposes of decision making. From practical point of view, DDM is of great concern and ultimate urgency.

Rough set theory is a new mathematical approach to imperfect knowledge. The problem of imperfect knowledge has been tackled for a long time by philosophers, logicians and mathematicians. Recently it became also a crucial issue for computer scientists, particularly in the area of artificial intelligence. There are many approaches to the problem of how to understand and manipulate imperfect knowledge. Rough set theory was developed by Zdzislaw Pawlak in the early 1980's. Rough set deals with classification of discrete data table in a supervised learning environment. Although in theory rough set deals with discreet data, rough set is commonly used in conjunction with other technique to do discrimination on the dataset. The main feature of rough set data analysis is non-invasive, and the ability to handle qualitative data. This fits into most real life application nicely. Rough set have seen light in many researches but seldom found its way into real world application.

Knowledge discovery with rough set is a multi-phase process consisted of mainly:

- Discretization
- Reducts and rules generation on training set

The advantage of using mathematical models is beyond increasing performance of the system. It helps knowledge workers in deeper analysis of the business and underlying product/domain. This will increase awareness in the company, knowledge transfer within the company, and higher desire to learn better things. There are many techniques like regression and classification, which are some of the popular mathematical models; however predictive analytics are not limited to these methods.

Regression: Linear Regression, kNN, CART, Neural Net

Classification: Logistic Regression, Bayesian Methods, Discriminant Analysis, Neural Net, kNN, CART.

9. Applications in parallel and distributed data mining

The technology of parallel and distributed data mining can be applied on different real time applications. The major applications are

- Credit card fraudulent detection
- Intrusion detection
- Business analysis – prediction etc.
- Financial applications
- Astrological events
- Anomaly Detection

10. Softwares for result analysis

The RapidMiner (formerly YALE) Distributed Data Mining Plugin allows performing distributed data mining experiments in a simple and flexible way. The experiments are not actually executed on distributed network nodes. The plugins only simulate this. Simulation makes it easy to experiment with diverse network structures and communication patterns. Optimal methods and parameters can be identified efficiently before putting the system into use. The network structure can for example be optimized as part of the general parameter optimization. While this cannot replace testing the system in an actual network, it makes the development stage much more efficient.
The service oriented architecture (SOA) paradigm can be exploited for the implementation of data and knowledge-based applications in distributed environments. The Web Services Resource Framework (WSRF) has recently emerged as the standard for the implementation of Grid services and applications. WSRF can be exploited for developing high-level services for distributed data mining applications. Weka4WS adopts the WSRF technology for running remote data mining algorithms and managing distributed computations. The Weka4WS user interface supports the execution of both local and remote data mining tasks. Other options like Inhambu, Weka Parallel and Grid Weka could also be used.

11. Future research directions

Parallel and Distributed data mining with Neural networks and Fuzzy approach
New Algorithms for performing Association,Clustering and Classification.
Privacy preserving parallel and distributed data mining.
Incremental Mining Algorithms.
Mining heterogeneous dataset in a parallel and distributed environment.
Knowledge Integration in a parallel and distributed environment.
Ant Colony Optimization with parallel and distributed data mining.
Mathematical modeling for a parallel and distributed mining process.

12. References

[Agr, 96] R. Agrawal, J. Shafer: "Parallel mining of association rules". IEEE Transactions on Knowledge and Data Engneering, 8(6) 962-969, 1996.

[AIS, 93] R. Agrawal, T. Imielinski, and A. Swami, "Mining Associations between Sets of Items in Massive Databases," Proceedings of the ACM SIGMOD, Washington, DC, pp. 207-216, May 1993.

[AR, 04] Andrei L. Turinsky, Robert L. Grossman y "A Framework for Finding Distributed Data Mining Strategies That are Intermediate Between Centralized Strategies and In-Place Strategies", 2004.

[ARD, 03] Assaf Schuster, Ran Wolff, and Dan Trock, "A High-Performance Distributed Algorithm for Mining Association Rules". In Third IEEE International Conference on Data Mining, Florida , USA, November 2003.

[AS, 96] R. Agrawal and J. C. Shafer, "Parallel Mining of Association Rules". IEEE Transactions On Knowledge And Data Engineering, 8:962-969, 1996.

[ATO, 99] Albert Y. Zomaya, Tarek El-Ghazawi, Ophir Frieder, "Parallel and Distributed Computing for Data Mining", IEEE Concurrency, 1999.

[ATS, 02] M. Z. Ashra_, D. Taniar, and K. A. Smith, "A Data Mining Architecture for Distributed Environments". IICS 2002, pages 27-38, 2002.

[Ays, 99] Ayse Yasemin SEYDIM "Intelligent Agents: A Data Mining Perspective" Southern Methodist University, Dallas, 1999.

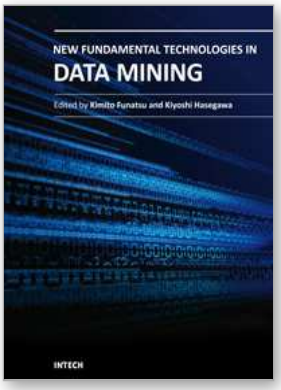
[BH, 02] Byung Hoon Park and Hilloi Karagupta, "Distributed Data Mining: Algorithms, Systems and Applications", University of MaryLand, 2002.

[CF, 04] Cristian Aflori, Florin Leon, "Efficient Distributed Data Mining using Intelligent Agents", in Proceedings of the 8th International Symposium on Automatic Control and Computer Science, 2004.

[FA, 01] Felicity George, Arno Knobbe, "A Parallel Data Mining Architecture for Massive Data Sets", High Performance Research Center, 2001.

[KKC, 99] Kargupta, H., Kamath, C., and Chan, P., "Distributed and Parallel Data Mining: Emergence, Growth and Future Directions, Advances in Distributed Data Mining, (eds) Hillol Kargupta and Philip Chan, AAAI Press, pp. 407-416, 1999.

[SS, 08] Dr. Sujni Paul, Dr.V.Saravanan, "Knowledge integration in a Parallel and distributed environment with association rule mining using XML data", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.5, May 2008.



New Fundamental Technologies in Data Mining

Edited by Prof. Kimito Funatsu

ISBN 978-953-307-547-1
Hard cover, 584 pages
Publisher InTech
Published online 21, January, 2011
Published in print edition January, 2011

The progress of data mining technology and large public popularity establish a need for a comprehensive text on the subject. The series of books entitled by "Data Mining" address the need by presenting in-depth description of novel mining algorithms and many useful applications. In addition to understanding each section deeply, the two books present useful hints and strategies to solving problems in the following chapters. The contributing authors have highlighted many future research directions that will foster multi-disciplinary collaborations and hence will lead to significant development in the field of data mining.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821