International Journal of Research Studies in Science, Engineering and Technology
Volume 3, Issue 3, March 2016, PP 12-17
ISSN 2349-4751 (Print) & ISSN 2349-476X (Online)

Intelligent Agents in Distributed Data Mining

D. Charishma
ECM Department
Sreenidhi Institute of Science & Technology
JNTUH, Hyderabad, India
charishmachoudary321@gmail.com

Dhathrika Sai Ganesh
ECM Department
Sreenidhi Institute of Science & Technology
JNTUH, Hyderabad, India
saiganesh.dhatrika@gmail.com

Abstract: *Data mining (DM) is a procedure of non-trivial mining of inherent, formerly anonymous, and potentially practical information from data in relational databases. In detail, the term knowledge discovery (KD) is more common than the term DM. It is usually examined as a step towards the procedure of KD, while these two terms are measured as synonyms in the computer literature. Data cleaning is to remove sound and conflicting data. Data integration is to join data from numerous data sources, such as a database and data warehouse (DW). Data selection is to improve data related to the job. Data transformation is to convert data into suitable forms. DM is to apply quick techniques to mine data patterns. Pattern assessment is to recognize the truly attractive patterns based on some interestingness evaluations. Knowledge assessment is to imagine and present the extracted knowledge to the end user.*

Keywords: *Data mining (DM), Knowledge discovery (KR), Parallelism, Data integration, Distribution*

1. Distribution of Data Mining

DM algorithms transact with easy data formats; there is growing amount of focus on removal of complexity and higher data types such as object-oriented (OO) spatial and temporal data (TD). Another characteristic of this expansion and development of DM systems is the shift from stand-alone structures using federal and local computational assets towards supporting growing levels of distribution. As DM technology matures and moves from a hypothetical domain to the practitioner's field there is rising recognition that distribution is very much an issue that needs to be accounted for. Databases in today's information age are essentially distributed **[3, 4]**. Institutions that function in universal markets need to perform DM on distributed data sources and require consistent and integrated knowledge from this data. Such as institutional environments are categorized by an environmental separation of users from the data sources. This intrinsic distribution of data sources and huge volumes of data involved predictably leads to steep communications costs.

Therefore, it is obvious that conventional DM model linking the co-location of users, data and computational resources is insufficient when dealing with distributed surroundings. The growth of DM along this dimension has directed to the emergence of distributed DM **[6, 7 and 8]**. Generally; DM surroundings consist of users, data, hardware and mining software. Distributed DM tackles the impact of sharing of users, s/w and computational sources on the DM procedure. There is general agreement that distributed DM is the procedure of mining data that has been partitioned into one or more physically/geographically distributed subsets. DDM is a division of the field of DM that offers a structure to mine distributed data carefully. In the DDM fiction, one of two hypotheses is normally employed as to how information is distributed across sites: consistently and heterogeneously.

The size and high dimensionality of datasets classically available as input to the difficulty of AR discovery, makes it an ideal difficulty for solving numerous CPUs in parallel **[1, 5]**. The main causes are the main memory and processor speed restrictions addressed by single CPUs. Thus it is critical to design well-organized parallel algorithms to do the job. Another cause for parallel algorithm comes from the truth that many business databases are distributed at multiple sites to start with. The cost of bringing them all to one site or one computer for serial finding of ARs can be prohibitively costly. For compute-intensive applications, parallelisation is an evident means for improving performance and achieving scalability. A variety of methods may be used to distribute the workload involved in DM over numerous CPUs **[2]**. Four key classes of parallel operations are familiar. The organization of tree in the Figure 1 exhibits their irregularity. The first difference made in this tree is between job parallel and data-parallel procedures.

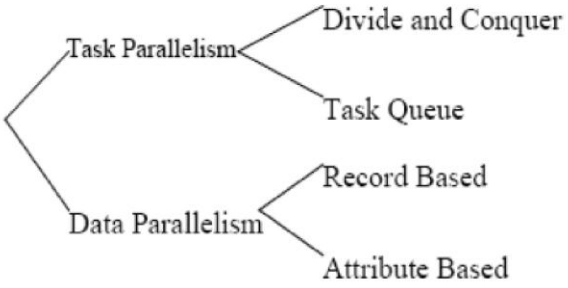

Figure1. *Schemes of Parallelism*

Task-parallel algorithms assign portions of the explore space to separate CPUs. The assignment of parallel techniques can again be splited into two clusters. The first cluster is based on a divide and conquer (DAC) approach that partitions the explore space and assigns each separation to a specific CPU. The second group is based on a task queue that vigorously allocates small segments of the explore space to a CPU at whatever time it is available. A task parallel operation of decision tree training will form tasks linked with branches of the tree. A DAC move towards a natural indication of the recursive scenery of decision trees. Yet the task of parallel execution lacks from load balancing difficulties caused by irregular allocation of records between branches. The success of a task parallel operation of decision trees seems to be highly reliant on the organization of the data set. The second class of advances, called parallel data, distributes the data set over the available CPUs. Parallel data advances in two flavours. A division is based on records will allocate non-overlapping sets of records to each of the CPUs. Instead a partitioning of attributes will allocate sets of attributes to each of the CPUs. Attribute-based procedures are based on the inspection that many algorithms can be articulated in terms of primitives that believe every attribute in turn. If attributes are distributed over numerous CPUs, these primitives may be implemented in parallel.

2. TERMINOLOGIES

• Parallel computation

Single systems with many CPUS work on same difficulty.

• Distributed computation

Many systems loosely coupled by a scheduler to work on related difficulties.

2.1. Algorithms for Association Rule Discovery

Most algorithms for AR finding follow the same general process, based on the chronological Apriori algorithm (AA). The basic idea is to make numerous passes over the database, building larger and larger cluster of associations on each pass. Thus, the first pass decides the items that occur most recurrently in all the transactions in the database; each consequent pass builds a list of possible recurrent item tuples based on the results of the preceding pass, and then examines the database, neglecting those tuples that do not occur normally in the database. The feeling is that for any set of items that occurs regularly, all subsets of that set must also occur recurrently. Observe that, for huge association sets, this algorithm and its derivatives must make many passes over a potentially huge database. It is also typically executed using a hash tree, a difficult data structure that displays very poor area. Although there are workable chronological algorithms for DM, there is a frantic need for a parallel key for most realistic-sized difficulties. The most evident argument for parallelism spins around database dimension. The databases used for DM are typically tremendously huge, often containing the particulars of the entire history of conventional databases. As these databases grows past hundreds of GBs towards a TB or additional, it becomes nearly impractical to practice them on a single sequential computer, for both time and space causes: no more than a part of the database can be kept in memory at any given point of time, and the amount of local disk storage and bandwidth needed to keep the sequential CPU supplied with data is huge. In addition, with an algorithm such as AA that requires many absolute passes over the database, the actual operating time required to complete the algorithm becomes extreme.

The basic advance to parallelizing AR discovery DM is via database separation. 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 AA. All of the parallel DM algorithms need some amount of global communication to organize the self-governing nodes.

2.2. Problems in Maturing Parallel Algorithms in Distributed Surroundings

There are several difficulties in budding parallel algorithms in distributed surrounding with association discovery DM which is being measured in this paper. They are:

• D**istribution of data:** One of the advantages of PAD DM is that each node can potentially work with a condensed size subset of the total database. A parallel algorithm in distributed surrounding must efficiently distribute data to allow each node to make self-governing development with its deficient view of the entire database.

• **I/O Minimization**: Even with good data distribution, parallel DM algorithms must strive to minimize the quantity of I/O they execute to the database.

• **Load balance**: To exploit the efficiency of parallelism, each terminal must have roughly the same amount of work to do. Even though a good initial data sharing can help provide load-balancing, with some algorithms, episodic data reorganization is required to obtain good load-balancing.

• R**educing communication:** An ideal parallel DM algorithm allows all terminals to function asynchronously, without having to stall commonly for global obstacles.

2.3. Parallel and Distributed (PAD) Data Mining Algorithms

The major algorithms used for parallel and distributed DM are:

• **Count Distribution:** this algorithm achieves parallelism by partitioning data. Each of N workstations gets 1/Nth 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 data structures used, but only requires a small amount of per-iteration communication 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 item sets candidates 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 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.

3. 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 independence. In order to work for somebody as an associate and mediator has to comprise a certain quantity of intelligence, which is the skill to choose between diverse courses of act, graph and converse adjust to changes in the surroundings, and study from practice. Broadly, a clever agent can be illustrated as a sensing element that can receive events, a recognizer that determines which incident occurred, a set of logic series from hard-coded applications 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. Software agent technology has matured enough to produce intelligent agents, which can be used for controlling a large number of concurrent business jobs. Multi agent (MA) systems are societies of agents that exchange information and data in the variety of communications. The agents' cleverness can vary from elementary sensor monitoring and information reporting, to more superior varieties of choice making and independent performance. The performance and cleverness of every 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 multi agent systems, which combines the power of knowledge discovery algorithms with the versatility of agents. Using this platform, agents are equipped with a data-driven assumption engine, can be vigorously and endlessly skilled. 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.

4. ARCHITECTURES

Agent-based distributed data mining systems use one or more agents to analyze and model confined data sets, which produce limited models. These restricted 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 meta-learning technique. The architecture consists of local databases of several financial institutes, learning agents and meta learning agents. Agents function on a local database and produce neighbouring classifiers. These local classifiers then are imported to a data location where they can be aggregated into a global model using meta-learning which is shown in the Figure 2.

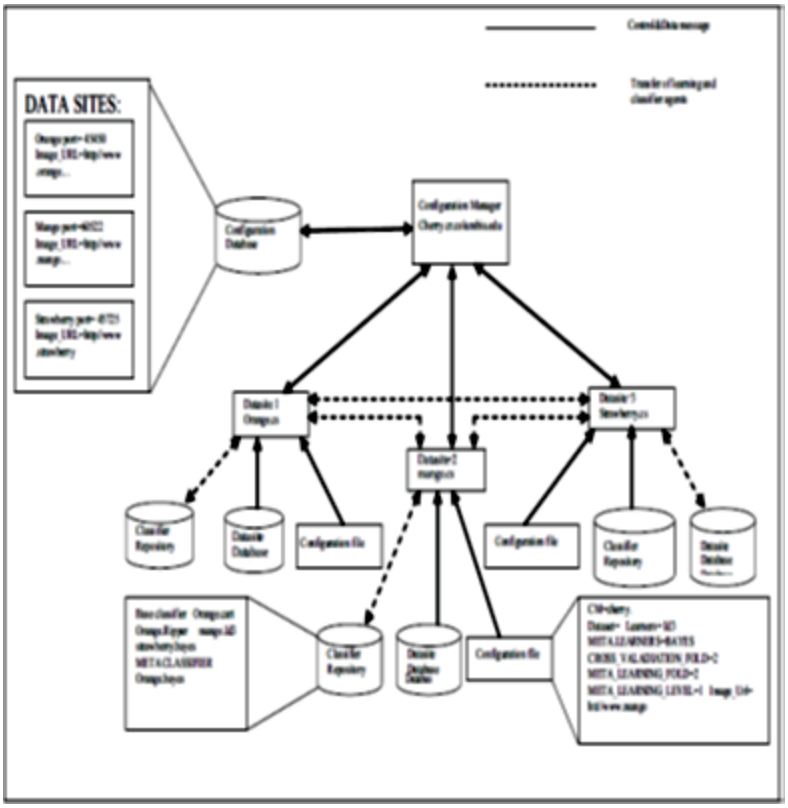

Figure2. *JAM architecture with 3 data locations*

BODHI is a Java and agent based distributed data mining method. It also notes the significance of portable agent technology. As all of agents are extensions of a basic agent object, BODHI can easily transfer an agent from one location to another location, along with the agent's environment, configuration, current state and learned knowledge. Figure 3 shows the BODHI architecture.

4.1. 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.

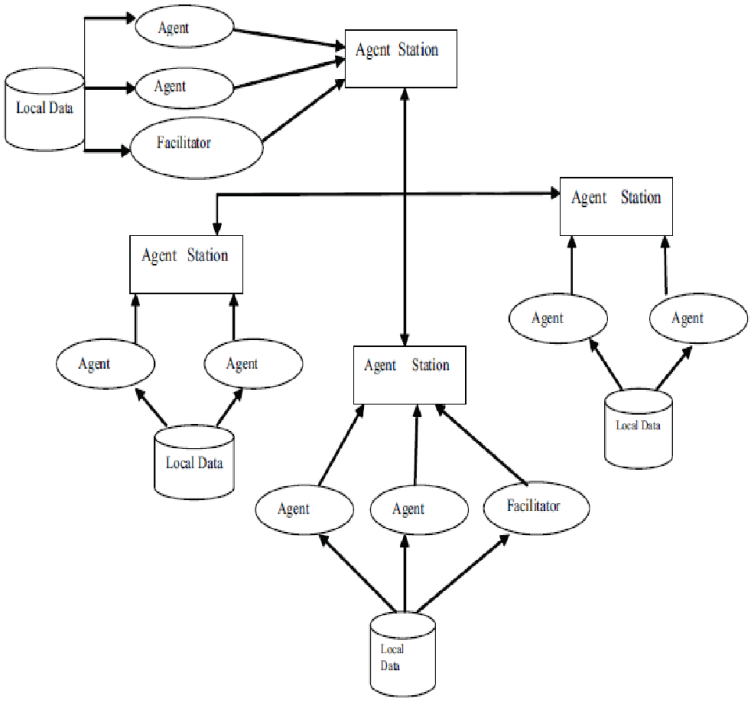

Figure3. *BODHI: Agent-based distributed data mining*

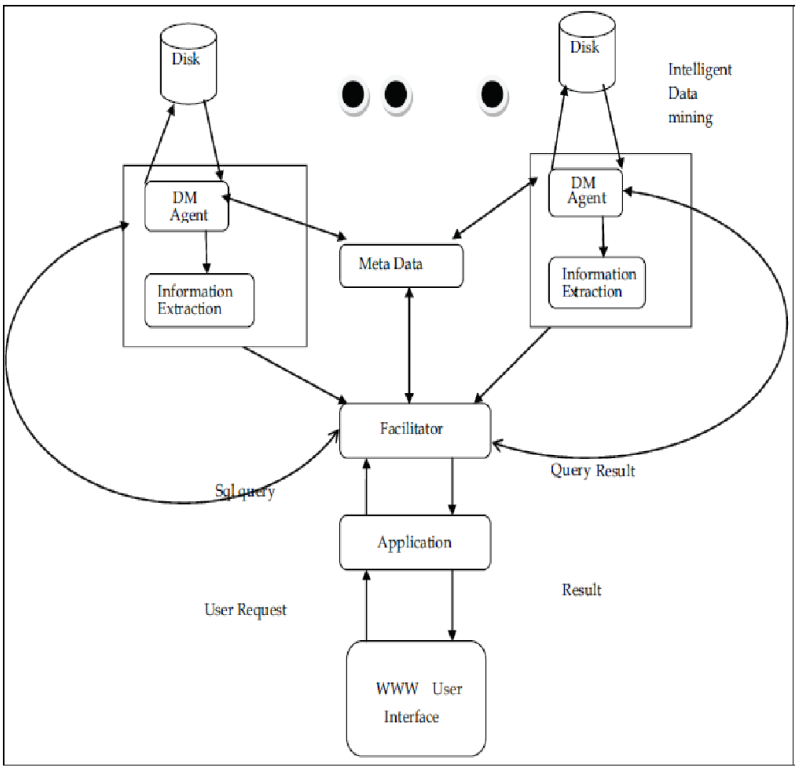

Figure4. *PADMA architecture*

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.

5. CONCLUSION

The Rapid Miner Distributed DM Plug-in allows performing distributed DM experiments in a simple and simple way. The operations are not actually performed on distributed network nodes. The plug-ins only simulates this. Simulation makes it easy to experiment with dissimilar network structures and message patterns. Most encouraging procedures and constraints can be recognized efficiently before putting the method into utilize. The network organization can be optimized as part of the general parameter minimization. The service oriented architecture (SOA) concept can be exploited for the implementation data and knowledge-based applications in distributed environments.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Management of Sreenidhi Institute of Science & Technology, Hyderabad for their constant encouragement and co-operation

REFERENCES

[1] R. Agrawal, J. Shafer: “Parallel mining of association rules”. IEEE Transactions on Knowledge and Data Engineering, 8(6) 962-969, 1996.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

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

[7] Ayse Yasemin SEYDIM “Intelligent Agents: A Data Mining Perspective” Southern Methodist University, Dallas, 1999.

[8] Byung Hoon Park and Hilloi Karagupta, “Distributed Data Mining: Algorithms, Systems and Applications”, University of Maryland, 2002.