Asian Journal of Information Technology 10 (2): 84-95, 2011
ISSN: 1682-3915

A Review of Some Issues and Challenges in Current Agent-Based Distributed Association Rule Mining

[1]A.O. Ogunde, [2]O. Folorunso, [2]A.S. Sodiya and [1]G.O. Ogunleye
[1]Redeemer's University (RUN), Redemption City, Mowe, Ogun State, Nigeria
[2]University of Agriculture, P.M.B. 2240, Abeokuta, Ogun State, Nigeria

Abstract: Association rule mining is today one of the most important aspects of data mining tasks. Data mining itself has shifted from stand-alone systems to distributed databases since most real life data are now distributed. Current studies in these areas are also considering the adoption of mobile agents in mining association rules from these distributed sites. Some of the algorithms presented in the literature are also distributed while others are not. All of them with their own peculiar issues and challenges which are of diversified forms. There is a great need to harness these diversified issues and challenges in one study to serve as reference point for researchers in this area of study. This study therefore, intends to review current distributed association rule mining issues and the challenges associated with them.

Key words: Data mining, distributed association rule mining, distributed frequent itemset mining, knowledge integration, algorithms, mobile agents

INTRODUCTION

The use of distributed systems is continuously spreading in several applications domains. Extracting valuable knowledge from raw data produced by distributed parties, in order to produce a unified global model may present various challenges related to either the huge amount of managed data or their physical location and ownership. In case data are continuously produced (stream) and their analysis is required to be performed in real time, communication costs and resource usage are issues that require careful attention in order to run computation in the optimal location.

Distributed Data Mining (DDM) is the semi-automatic pattern extraction of distributed data sources. The next generation of the data mining studies will be distributed data mining for many reasons. First of all, most of the current used data mining techniques require all data to be resident in memory i.e., the mining process must be done at the data source site. This is not feasible for the exponential growth of the data stored in organization (s) databases.

Another important reason is that data is inherently distributed for fault tolerance purposes. DDM requires two main decisions about the DDM implementations: A distributed computation paradigm (message passing, Remote Procedure Calls (RPC), mobile agents) and the used integration techniques (Knowledge probing) in order to aggregate and integrate the results of the various distributed data miners. Recently, the new distributed computation paradigm which has been evolved as software agents are widely used. Mobile agent is a thread of control that can trigger the transfer of arbitrary code to a remote computer. Mobile agents paradigm has several advantages: Conserving bandwidth and reducing latencies. Also, complex, efficient and robust behaviors can be realized with surprisingly little code. Mobile agents can be used to support weak clients, allow robust remote interaction and provide scalability.

With the development of hardware especially the development of large space storage, lots of organization build large storage database and collect large volume of data. Those organizations have the desire to extract useful information from the ultra large amount of data. So some traditional way will be not enough to handle the information.

Association Rule Mining (ARM), fist proposed by Agrawal *et al.* (1993), tries to find frequent patterns, associations, correlations or casual structures sets of items or objects in transaction database, relational database, etc. The idea is to find out the relation or dependency of occurrence of one item based on occurrence of other items. Lots of algorithms in this area, both sequentially and parallel had been proposed. Since association rule mining is dedicated to handle the ultra large amount of data so the time complexity and resource complexity have to be carefully considered.

Corresponding Author: A.O. Ogunde, Redeemer's University (RUN), Redemption City, Mowe, Ogun State, Nigeria

In this research, up to date review of related researches in the field of distributed data mining with particular focus on association rule mining will be done. Problems related to Association Rule Mining and agent-based association rule mining from distributed databases will be examined in detail. The research will mainly focus on distributed association rule mining algorithms as the building block of the approximate distributed algorithms. Existing systems will be analysed and issues and challenges pertaining to this domain will be raised.

LITERATURE REVIEW

This research work intends to review existing works in Distributed Data Mining (DDM), Distributed Frequent Itemset mining (DFIM), Distributed Association Rule mining (DARM) and knowledge integration in distributed databases.

Data Mining (DM): Data mining is a powerful new technology with great potential to help companies focus on the most important information in the data they have collected about the behavior of their customers and potential customers (Rao and Vidyavathi, 2010). Data mining involves the use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data set. These tools can include statistical models, mathematical algorithm and machine learning methods. It discovers information within the data that queries and reports can not effectively reveal.

The explosive growth in data stored in databases and data warehouses has generated an urgent need for new techniques that can intelligently transform this huge amount of data into useful knowledge. Consequently, data mining has become an important research area (Chen *et al*., 1996). Data mining differs from other data analysis techniques in that the system takes the initiative to generate patterns by itself. Data mining is concerned with the algorithmic means by which patterns, changes, anomalies, rules and statistically significant structures and events in data are extracted from large data sets (Grossman *et al*., 1999). Data mining studies can be classified into two generations. Studies in the first generation have focused on which kinds of patterns to mine. Studies in the second generation have focused on how mining can interact with other components in the framework like DBMS (Johnson *et al*., 2000).

Distributed Data Mining (DDM): Distributed data mining refers to the mining of distributed data sets. The data sets are stored in local databases hosted by local computers which are connected through a computer network (Webb, 2000; Ashrafi *et al*., 2004). When data mining is undertaken in an environment where users, data, hardware and the mining software are geographically dispersed, it is called distributed data mining. Typically such environments are also characterised by heterogeneity of data, multiple users and large data volumes (Chia and Kannapan, 1997). Data mining takes place at a local level and at a global level where local data mining results are combined to gain global findings. Distributed data mining is often mentioned with parallel data mining in literature (Paul, 2010).

While both attempt to improve the performance of traditional data mining systems they assume different system architectures and take different approaches. In distributed data mining computers are distributed and communicate through message passing. In parallel data mining a parallel computer is assumed with processors sharing memory and or disk.

Computers in a distributed data mining system may be viewed as processors sharing nothing. This difference in architecture greatly influences algorithm design, cost model and performance measure in distributed and parallel data mining (Paul, 2010).

Two requirements dictate the need for distributed data mining: data may be inherently distributed for a variety of practical reasons including security and fault tolerant distribution of data and services or mobile platform. Also, the cost of transporting data to a single site is usually high and sometimes unacceptable (Prodromidis, 1999).

The second requirement is that many of the mining algorithms require all data to be resident in memory. This might be unfeasible for large data sets because these learning algorithms do not have the capability to process this huge amount of data. Data partitioning is one of the popular solutions for this problem (Provost and Kolluri, 1997).

Consequently, data in this case is artificially distributed (Malhi, 1998). DDM offers techniques to discover knowledge in distributed data through distributed data analysis using minimal communication of data. Typical DDM algorithms involve local data analysis from which a global knowledge can be extracted using knowledge integration techniques. A typical DDM framework is shown in Fig. 1.

Categorization of DDM models: There are predominantly two architectural models used in the development of DDM systems namely Client Server (CS) and software agents. The agents category can be further divided on the basis of whether the agents have the ability to migrate in a self-directed manner or not (i.e., whether the agents exhibit the characteristic of mobility or not). This is shown in Fig. 2.

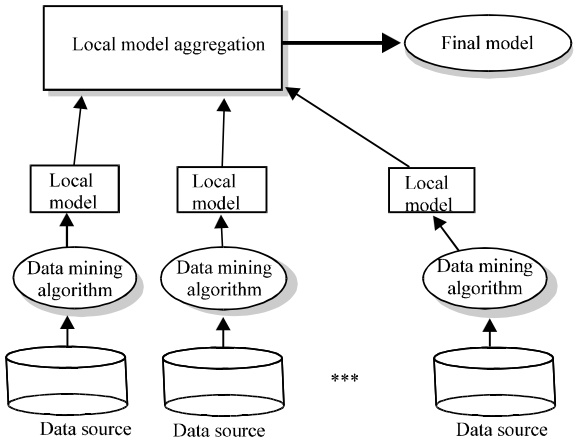

Fig. 1: A typical DDM framework (Rao and Vidyavathi, 2010)

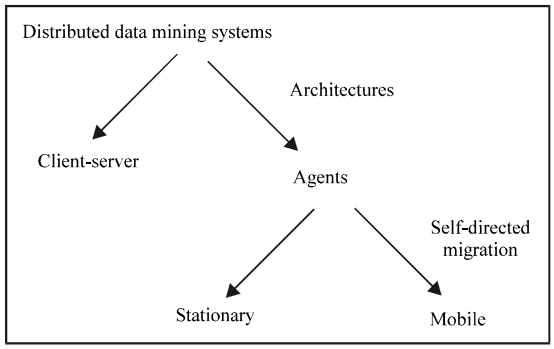

Fig. 2: Taxonomy of distributed data mining architectures

Client/server based DDM model: The Client/Server model uses the Remote Procedure Call (RPC) mechanism in the communication between the clients and the server. The RPC allows a program on the client to invoke a procedure on the server using stubs on each side. The client-side stub acts as a proxy for the real procedure. It accepts calls for the procedure and arranges for them to be forwarded to the server.

The server-side stub receives the call for a procedure and returns the results to the client-side stub. Finally, the client-side stub returns the result to the original RPC call (Crowley, 1997; Gray, 1995). The CS-based DDM uses one or more DM servers.

The client requests are sent to DM server that determines the required data sources and collects data from different locations and brings all the required data for the specified mining process to the DM sever. The DM server in turn houses the data mining algorithms. The mining process is accomplished on the DM server and the results are returned to the requested client. Figure 3 shows typical Client/Server based DDM process.

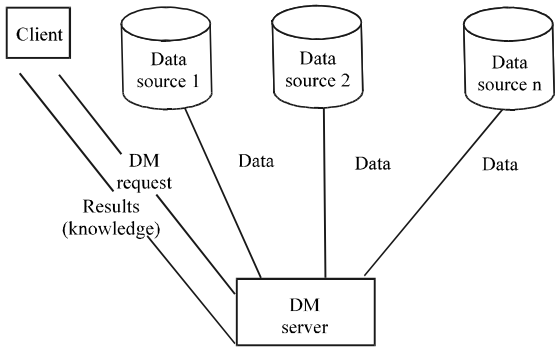

Fig. 3: Client-server based DDM process (Ariwa *et al.*, 2003)

A fundamental problem exists with Client/Server architectures is that if the server does not provide the exact service that the client requires then the client must take a series of RPCs to obtain the end service. This might result in an overall latency increase and in intermediate information between the client and the server during the service processing. Consequently, CS architecture may waste the network bandwidth (Dale, 1997; Gray *et al.*, 2000).

Association Rule Mining (ARM): Association rule mining is the discovery of associations or connections among objects. Since its inception, association rule mining has become one of the core data-mining tasks and has attracted tremendous interest among researchers and practitioners. ARM is undirected or unsupervised data mining over variable-length data and it produces clear, understandable results. It has an elegantly simple problem statement. Association rule mining has a wide range of applicability such market basket analysis, medical diagnosis/research, Website navigation analysis, homeland security and so on. An association rule is in the form of A1 ^ .....^ Ai → B1 ^ ..... ^ Bj which means objects B1; .....;Bj tend to appear with objects A1; ……..; Ai in the target data. Association rules at multiple conceptual levels will reveal such kind of association in the relevant set (s) of data in a database. For example one may discover that a set of symptoms often occur together with another set of symptoms and then further study the reasons behind this association.

The conventional algorithm of association rules discovery proceeds in two steps. All frequent itemsets are found in the first step. The frequent itemset is the itemset that is included in at least minsup transactions. The association rules with the confidence at least minconf are generated in the second step. According to Hipp *et al.*

(2000), there is a border that separates the frequent itemsets from the infrequent ones thus the problem is restricted on finding that border.

Modern organizations are geographically distributed. Typically, each site locally stores its ever increasing amount of day to day data. Using centralized data mining to discover useful patterns in such organizations data is not always feasible because merging data sets from different sites into a centralized site incurs huge network communication costs. Data from these organizations are not only distributed over various locations but also vertically fragmented making it difficult if not impossible to combine them in a central location. Distributed data mining has thus emerged as an active sub-area of data mining research. Therefore, this study proposes an agent-based architecture for a distributed Association Rule Mining in performing the mining process.

The field of distributed data mining has therefore gained increasing importance in the last decade. The Apriori algorithm by Agrawal and Srikant (1994) has emerged as one of the best Association Rule mining algorithms. It also serves as the base algorithm for most parallel algorithms. The enormity and high dimensionality of datasets typically available as input to problem of association rule discovery, makes it an ideal problem for solving on multiple processors in parallel. The primary reasons are the memory and CPU speed limitations faced by single processors.

The formal definition of association rule mining is: Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals called items and D be a set of transactions where each transaction T is a set of items such that $T \subseteq I$. Associated with each transaction is a unique identifier, called its TID. We say that a transaction T contains X, a set of some items in I, if $X \subseteq T$.

Association rule mining process could be decomposed into two main phases to enhance the implementation of the algorithm. The phases are:

Frequent item generation: This is to find all the itemsets that satisfy the minimum support threshold. The itemsets are called frequent itemsets.

Rule generation: This is to extract all the high confidence rules from the frequent itemsets found in the first step. These rules are called strong rules.

Association rules: An association rule is an implication expression of the form $X \Rightarrow Y$ where $X \subset I$, $Y \subset I$ and X and Y are disjoint itemsets, i.e., $X \cap Y = \phi$. The strength of an association rule can be measured in terms of its support and confidence. The rule $X \Rightarrow Y$ holds in the transaction set D with confidence c and support s, if c% of the transactions in D that contains X also contains Y and s% of transactions in D contains $X \cup Y$. Both the antecedent and the consequent of the rule could have more than one Item. The formal definitions of these two metrics are:

$$\text{Supports, } s\ (X \Rightarrow y) = \frac{\Sigma(x \cup y)}{n}$$

$$\text{Confidence, } c\ (X \Rightarrow y) = \frac{\Sigma(x \cup y)}{\Sigma x}$$

There have been many algorithms developed for mining frequent patterns which can be classified into two categories:

- Candidate-generation-and-test
- Pattern-growth methods

The first category, the candidate-generation and test approach such as the Apriori algorithm (Agrawal and Srikant, 1994) is directly based on an important property of frequent itemsets: if a pattern (set) with k items is not frequent, none of its super-patterns (supersets) with (k+1) or more items can be frequent. This is known as the downward closure property. Since its introduction in 1994, the Apriori algorithm, developed by Agrawal and Srikant (1994) has been the basis of many subsequent ARM and/or ARM-related algorithms. In their research, it was observed that ARs can be straightforwardly generated from a set of Frequent Itemsets (FIs). Thus, efficiently and effectively mining FIs from data is the key to ARM. The Apriori algorithm iteratively identifies FIs in data by employing the downward closure property of itemsets in the generation of candidate itemsets where a candidate (possibly frequent) itemset is confirmed as frequent only when all its subsets are identified as frequent in the previous pass.

The Apriori algorithm performs repeated passes of the database, successively computing support-counts for sets of single items, pairs, triplets and so on. At the end of each pass, sets that fail to reach the required support threshold are eliminated and candidates for the next pass are constructed as supersets of the remaining (frequent) sets. Since no set can be frequent which has an infrequent subset, this procedure guarantees that all frequent sets will be found. A candidate-generation-and-test approach iteratively generates the set of candidate patterns of length (k+1) from the set of frequent patterns of length k and checks their corresponding occurrence frequencies in the database. The Apriori algorithm achieves good

reduction on the size of candidate sets. However, when there exist a large number of frequent patterns and/or long patterns, candidate generation-and-test methods tend to produce very large numbers of candidates and require many scans of the database for frequency checking. Since, the number of database passes of the Apriori algorithm equals the size of the maximal frequent itemset, it scans the database k times even when only one k-frequent itemset exists. The drawback of this method is that if the dataset is very large, the required multiple database scans can be one of the limiting factors of the Apriori algorithm. Many algorithms have been proposed, directed at improving the performance of the Apriori algorithm using different types of approaches. An analysis of the best known algorithms can be found in Ivancsy *et al*. (2004).

A second category of methods, pattern-growth methods such as FP-growth (Han *et al*., 2004) and Apriori-TFP (Coenen *et al*., 2004) have been proposed. These algorithms typically operate by recursively processing a tree structure into which the input data has been encoded. FP-growth uses two data structures, the FP-tree and a header table in which to store the input data. The FP-tree is a set enumeration tree structure that has links going back up as well as down, the tree and links between nodes representing the same item. The header table contains links to a first item in the tree for each item. The multiple links allow for fast processing. The input data is initially translated into a start FP-tree and header table. FP-growth then recursively processes this start FP-tree structure and header table to generate frequent itemsets staring with 1 item sets and continuing until no more frequent itemsets can be discovered. At each recursion further FP-trees and header tables are generated. The major strength of FP-growth is that it is a very fast algorithm, certainly with respect to Apriori but has some disadvantages. Its first principal drawback is that because many FP-trees are repeatedly generated, FP-growth can have significant storage requirements. Secondly, the large number of links makes it difficult to distribute the tree. These disadvantages are particularly significant with respect to dense datasets.

In the study shown by Han *et al*. (2000), a novel frequent itemset tree structure, FP-tree was proposed. FP-tree is an extended prefix-tree structure for storing compressed information about frequent itemsets. It consists of one root labeled as NULL and a set of item prefix sub trees as the descendants of the root. A frequent-item header table is also kept to link all transactions containing that item. Each node in the item prefix sub tree consists of three fields: item-name, count and node-link where item-name registers which item this node represents, count registers the number of transactions represented by the path reaching this node and node-link links to the next node in the FP-tree carrying the same item-name or NULL if there is none.

Based on the FP-tree (Han *et al*., 2000), an itemset fragment growth method, FP-growth was designed to avoid the costly generation of a large number of candidate sets. The Fp-growth algorithm is a partition-based, divide and conquer method used to decompose the mining task into a set of smaller tasks for mining confined itemsets in conditional databases, thereby dramatically reducing the search space. The size of the FP-tree is usually small and will not grow exponentially. According to Han *et al*. (2000), the FP-growth method is efficient and scalable for mining both long and short frequent itemsets and is about an order of magnitude faster than the apriori algorithm.

In the overall, no single ARM algorithm has been identified to fit all types of data (Albashiri *et al*., 2009). Real data sets can be sparse and/or dense according to their applications. For example for telecommunication data analysis, calling patterns for home users vs. business users can be very different: some are frequent and dense (e.g., to family members and close friends) while others are large and sparse. Similar situations arise for market basket analysis, census data analysis, etc. It is hard to select an appropriate ARM method anyhow when no algorithm fits all. Large applications need more scalability. Many existing methods are efficient when the data set is not very large. Otherwise, their core data structures (such as FP-tree) or the intermediate results (e.g., the set of candidates in Apriori or the recursively generated sub-trees in FP-growth) may not fit in main memory and may cause thrashing, hence based on the strength and weaknesses of the two state-of-the-art algorithms which are Candidate-generation-and-test and Pattern-growth methods, the architecture proposed in this research work intends to incorporate these two algorithms (specifically, Apriori and FP-Growth) into different mobile mining agents in distributed environments. The coordinating agent in the proposed system determines which of the algorithm to use for mining, depending on the size of the data at each data site/source (either sparse or dense) in the distributed system.

Existing DDM models: Several DDM systems have been proposed in the literature. In Kargupta PADMA system was presented. The PADMA system employed the approach of a central-learning strategy. They described a parallel DM system (PADMA) that uses software agents for local data accessing and analysis and a web

based interface for interactive data visualization. Partial data cluster models are first computed by stationary agents locally at different sites. All local models are collected to a central site that performs a second-level clustering algorithm to generate the global cluster model. PADMA has been used in medical applications. In Botia *et al*. (1998), the basic design and implementation guidelines for a generic data mining system was presented. In Martin *et al*. (1999), an agent infrastructure for data mining systems was also proposed. In Stolfo *et al*. (1997) Java Agents for Meta-learning (JAM) over distributed databases was proposed. JAM provided a set of learning programs implemented either as JAVA applets or applications that computed models over data stored locally at a site. JAM supported the launching of learning and meta-learning agents to distributed database sites. In Kargupta Collective Data Mining (CDM) theory and implementation have been studied. In Chattratichat *et al*. (1999), architecture for distributed enterprise data mining was presented. In Guo and Sutiwaraphun (1999), a knowledge integration technique using knowledge probing was presented.

Papyrus (Bailey *et al*., 1999) is a Java-based system addressing wide-area DDM over clusters of heterogeneous data sites and metaclusters. It supports different task and predictive model strategies including C4.5. Mobile DM agents move data, intermediate results and models between clusters to perform all computation locally and reduce network load or from local sites to a central root which produces the final result. Each cluster has one distinguished node which acts as its cluster access and control point for the agents. Coordination of the overall clustering task is either done by a central root site or distributed to the (peer to peer) network of cluster access points. Papyrus supports various methods for combining and exchanging the locally mined predictive models and metadata required to describe them by using a special markup language.

Theoretical cost models for Client/Server, mobile agents and hybrid DDM models have been proposed by Krishnaswany. They proposed theories that combined client-server and distributed models. Not much work was done in the study as the researchers concluded that the work still needs improvement and experimental validation. Ariwa *et al*. (2003) also proposed an OIKI model: e-business model for DDM using mobile agents. OIKI model is a mobile agent based DDM model that overcome the drawbacks of the traditional mobile agent based DDM model. Instead of transferring the results from each data server to the client, the client controls migration of the results among data servers to be integrated locally and finally, the final results are transferred to the client. Only few conceptual views are presented in the study while the researchers concluded that the implementation issues of their proposed model still have a lot of work to be done.

Gyorodi *et al*. (2004) carried out a comparative study of distributed algorithms in mining association rules. She compared two representative distributed ARM algorithms CDA (Count Distribution Algorithm) which uses data parallelism and FDM (Fast Distributed Algorithm for Data Mining) which was based on task parallelism. It was discovered that both scaled well relative to different support factors and size of data set. Silvestri (2006) studied Distributed and Stream Data Mining Algorithms for frequent pattern discovery in his PhD thesis. They offered a perspective on DDM algorithms in the context of multiagent systems, discussing broadly the connection between DDM and MAS and also providing a high-level survey of DDM with the main focus of their study on distributed clustering algorithms and some potential applications in multi-agent-based problem solving scenarios. Byrd and Franke (2007) gave the state of DDM in their study focusing mainly on the state of the art in distributed clustering and frequent itemset mining.

Albashiri *et al*. (2009) worked on EMADS, an ExtendibleMulti-Agent Data mining System. The EMADS vision is that of a community of data mining agents, contributed by many individuals, interacting under decentralised control to address data mining requests. EMADS is seen both as an end user application and a research tool. The study detailed the EMADS vision, the associated conceptual framework while concentrating mainly on agent based data classification.

Badal and Tripathi (2010) proposed the VS_Apriori which is an extension of the classical Apriori algorithm. The researchers claimed that VS_Apriori algorithm works faster than the classical and also scales well when the support threshold decreases. Rao and Vidyavathi (2010) in their study combined data mining wit game theory as a way of striking a balance and reaching a steady state between the data miner and the adversary in adversary data mining. Paul and Saravanan (2008) proposed a knowledge integration method in a parallel and distributed environment with association rule mining using XML data. A scanty write-up on the knowledge integration aspect was done and how the real knowledge integration of the XML data would be done was not presented in the study. Paul (2010) also proposed an optimized algorithm for distributed and parallel data mining but with XML data because of the complexity of XML data. In their approach,

multiple nesting problems in XMl data was handled appropriately to assure the correctness of their result. The Umarani and Punithavalli (2010) also carried out a recent overview on Sampling-based ARM, positing that sampling can speed-up the mining of association rules.

Common to all approaches is that they aim at integrating the knowledge which is discovered out of data at different geographically distributed network sites with a minimum amount of network communication and maximum of local computation. All these studies have presented a different data mining scenarios involving various architectural models which are still very open for further research. EMADS (Albashiri *et al*., 2009) is the closest to this research but it uses a distributed algorithm and is mainly based on agent-based data classification while the architecture proposed in this system will be purely based on agent-based association rule mining. A drawback noted here is that after mining, all the individual data results have to migrate to back to the requesting server which may incur serious communication costs and also bringing large set of results together on one the requesting system may pose serious challenges in term of memory constraints. Also, since the algorithm is distributed, any fault at any of the data sites may hinder the successful completion of the mining process.

Agents: Agents are defined by Wooldridge (2009) as computer software that are situated in some environment and are capable of autonomous action in this environment in order to meet their design objectives. Intelligent agents (Rudowsky, 2004; Wooldridge, 2009) are defined as agents that can react to changes in their environment have social ability (communication) and the ability to use computational intelligence to reach their goals by being proactive. Agents are active, task-oriented, modelled to perform specific tasks and capable of autonomous action and decision making.

Multi-agent systems: By combining multiple agents, in one system, to solve a problem, the resultant system is a Multi-Agent System (MAS) (Wooldridge, 2009). These systems are comprised of agents that individually solve problems that are simpler than the overall problem. They can communicate with each other and assist each other in achieving larger and more complex goals. Thus problems that software developers had previously thought of as being too complex (Martin *et al*., 1999) can now be solved by localizing the problem solving. In general, MAS adhere to the following three characteristics. First, MAS must specify appropriate communication and interaction protocols. Secondly, MAS must be open and decentralised with no prior knowledge of for example, the number of participants or behaviours. In a running MAS, new agents may join at any time having only to conform to the communication protocol being able to act in the way they choose, often in an unpredictable manner. Finally, MAS may consist of possibly heterogeneous agents that are scattered around the environment and act autonomously or in collaboration. Well documented advantages of MAS according to Wooldridge (2009) include: Decentralised control, Robustness, Simple extendability, Sharing of expertise and Sharing of resources.

Mobile agent based DDM model: The agent-based model is a popular approach to constructing distributed data mining systems and is characterized by a variety of agents co-ordinating and communicating with each other to perform the various tasks of the data mining process. Agent technology is seen as being able to address the specific concern of increasing scalability and enhancing performance by moving code instead of data and thereby reducing the communication overhead incurred in the CS model. However, the absence of dedicated data mining servers and the lack of control over available computational resources at remote sites are limitations.

A mobile agent does not waste the bandwidth because the agent migrates to the server. The agent performs the necessary sequence of operations locally and returns just the final result to the client, Gray *et al*., (2000). The major drawback in the CS-based DDM model is that huge amount of data sets migrate form the data sources locations to the DM sever to accomplish the required DM process. This results into a considerable waste in the network bandwidth and consequently a big increase in latency. A typical mobile agent-based DDM process begins with a client request for a DM process. The client determines the required data severs for the DM process and multicasts a set of mobile agents data miners MADMs. The MADMs migrate to the data servers and perform the data mining operations locally and return the final results (knowledge) to the client. Finally, the client uses a Knowledge Integration (KI) program to integrate the DM results from the different MADMs. Figure 4 shows the described mobile agent-based DDM process.

Multi-Agent Data Mining (MADM): There are two themes of agent and DM interaction and integration in the literature (Cao *et al*., 2007): DM for agents, referred to as mining-driven agents (Symeonidis and Mitkas, 2006) and

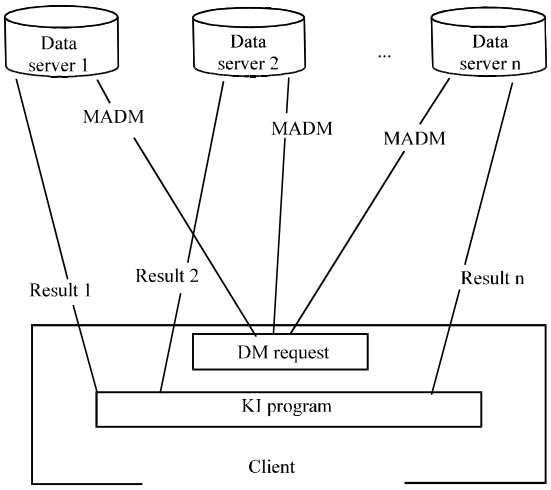

Fig. 4: Typical mobile-agent based DDM process (Ariwa *et al*., 2003)

agents for DM, referred to as agent-driven DM commonly known as Multi-Agent Data Mining (MADM). The former concerns issues of transforming the discovered knowledge, extracted by DM into the inference mechanisms or simply the behaviours of agents and multi-agent systems as well as the arguable challenge of generating intelligence from data while transferring it to a separate, possibly autonomous, software entity. A FIPA-compliant multi-agent platform based on mining-driven agents (Agent Academy) that offers facilities for design, implementation and deployment of multi-agent systems is proposed by Symeonidis and Mitkas (2006). The researchers describe the Agent academy as an attempt to develop a framework through which users can create an agent community having the ability to train and retrain its own agents using DM techniques. Furthermore, several systems have been developed for agent-based distributed data mining. These systems can be classified according to their strategy to three types; central learning, meta-learning and hybrid learning.

Central learning strategy is when all the data can be gathered at a central site and a single model can be build. The only requirement is to be able to move the data to a central location in order to merge them and then apply sequential DM algorithms. This strategy is used when the geographically distributed data is small. The strategy is generally very expansive but also more accurate. However, as pointed in Jensen *et al*. (2003), this strategy in general is unfeasible. Agent technology is not very preferred in such strategy. Meta-learning strategy offers a way to mine classifiers from homogeneously distributed data. Meta learning follows three main steps. The first is to generate base classifiers at each site using a classifier learning algorithms. The second step is to collect the base classifiers at a central site and produce meta-level data from a separate validation set and predictions generated by the base classifier on it. The third step is to generate the final classifier (meta-classifier) from meta-level data via a combiner or an arbiter. Copies of classifier agent will exist or deployed on nodes in the network being used. Perhaps the most mature systems of agent-based meat-learning systems are: JAM system (Stolfo *et al*., 1997) and BODHI (Krebs, 2003) and recently EMADS (Albashiri *et al*., 2009).

Hybrid learning strategy is a technique that combines local and centralized learning for model building (Lloyd, 2003) for example, Papyrus is designed to support both learning strategies. In contrast to JAM and BODHI, Papyrus can not only move models from site to site but can also move data when that strategy is desired. Papyrus is a specialized system which is designed for clusters while JAM and BODHI are designed for data classification. The major criticism of such systems is that it is not always possible to obtain an exact final result, i.e., the global knowledge model obtained may be different from the one obtained by applying the one model approach (if possible) to the same data.

OTHER ISSUES AND CHALLENGES

Many current data mining tasks can be accomplished successfully only in a distributed setting (Paul, 2010). New methods for mining vast amounts of heterogeneous data from several data sources are emerging all the time (Byrd and Franke, 2007). The PADMA system is a document analysis tool working on a distributed environment, based on cooperative agents. It works without any relational database underneath. Instead, there are PADMA agents that perform several relational operations with the information extracted from the documents.

Distributed frequent itemset mining is currently not very actively researched. Thus, despite their years of publication, the presented overview papers like (Zaki, 1999) give an up-to-date view on the existing algorithms. Recent study are focusing on minimizing the communication cost of the prior algorithms and are taking dynamic datasets into account. According to Klusch *et al*. (2003), another problem arises with the need to scale up to massive data sets which are distributed over a large number of sites. For example, the NASA Earth

Observing System (EOS) is a data collector for satellites producing 1450 data sets of about 350 GB day^{-1} and pair of satellites at a very high rate which are stored and managed by different systems geographically located all over the USA. Any online mining of such huge and distributed data sets in a central data warehouses may be prohibitively expensive in terms of costs of both communication and computation.

Important open problems in the area of distributed frequent itemset mining are two fold. First, in a setting where the dataset is initially in one big database, the crucial question of how to distribute the data to best facilitate the specific data mining task of frequent itemset mining is not solved. Second, equivalents to most centralized frequent itemset mining approaches on data streams are not existing. Especially sliding window approaches and approximate mining algorithms using the common error threshold from the centralized case could be useful contributions.

With the ever-growing database sizes, we have enormous quantities of data but unfortunately we cannot use raw data in the day today reasoning/decisions. We desperately need knowledge. This knowledge is in most cases in the gathered data but the extraction of it is a very time and resources consuming operation. Association rule mining finds interesting association or correlation relationships among a large set of data items (Gyorodi *et al*., 2004).

The implementation issues of the proposed OIKI model are research areas that have a lot of research to be done (Ariwa *et al*., 2003). The study of the efficient knowledge integration techniques is an essential research area to the OIKI DDM model implementation. EMADS, a related research by Albashiri *et al*. (2009) only concentrated on agent based data classification but not agent based association rule mining.

Till date, existing literature in MADM has revealed that most MADM tasks involves a global knowledge integration after the local data mining at each data sources. This obviously results in serious communication overheads as all local mining results must be transported to a central server for knowledge integration to occur. Apart from high communication costs incurred by these methods for most real-life applications with really large-sized datasets and widely distributed dense data sources involved, getting all local mining results into one data server for knowledge integration poses enormous memory challenges as the whole set of local results coupled with the knowledge integration algorithm may not fir into the system memory. Data transfer costs which are estimate of the time needed for data to be transferred from the data site to the DARM server is enormous (Albashiri, 2010). There could be different models of distributed data mining here but one could involve a NOC that collects data from the distributed sites and another in which all sites are treated equally. The goal here obviously would be to minimize the amount of data shipped between the various sites-essentially to reduce the communication overhead. In distributed mining, one problem is how to mine across multiple heterogeneous data sources: multi-database and multirelational mining (Rao and Vidyavathi, 2010).

End users of association rule mining tools encounter several well known problems in practice. First, the algorithms do not always return the results in a reasonable time. It is widely recognized that the set of association rules can rapidly grow to be unwieldy, especially as we lower the frequency requirements. The larger the set of frequent itemsets the more the number of rules presented to the user, many of which are redundant. This is true even for sparse datasets but for dense datasets it is simply not feasible to mine all possible frequent itemsets et alone to generate rules since they typically produce an exponential number of frequent itemsets; finding long itemsets of length 20-30 is not uncommon. Although, several different strategies have been proposed to tackle efficiency issues, they are not always successful (Kotsiantis and Kanellopoulos, 2006).

Incremental Knowledge Integration proposed by Ariwa *et al*. (2003) does not take into consideration the size of the agents, size of results, bandwidth and other computational resources at the data servers. Formal methods for achieving this are major considerations in this research. Mining for association rules and frequent patterns is a central activity in data mining. However, most existing algorithms are only moderately suitable for real-world scenarios (Legler *et al*., 2009).

According to Albashiri (2010), it is hard to select an appropriate ARM method when no algorithm fits all. Large applications need more scalability. Many existing methods are efficient when the data set is not very large. Otherwise, their core data structures (such as FP-tree) or the intermediate results (e.g., the set of candidates in Apriori or the recursively generated sub-trees in FP-growth) may not fit in main memory and may cause thrashing (Albashiri, 2010).

It should also be noted that using a distributed algorithm such as the ones discussed in this study may also be faced with several other challenges that affects

distributed systems. When the mining algorithm is distributed and there is a fault in one or more of the data sites then the possibility of having the distributed association rule mining completed in reasonable time is hampered. Also, we can then not be sure that the final mining results represents is a true representation of the entire distributed database.

There is therefore, the need for more research in the area of getting more fault tolerant algorithms in the distributed association rule mining setting. This may be possible through the distribution of the data but not the algorithm or through certain procedures that will integrate the knowledge discovered at each of the data sites in an efficient manner as opposed to the global knowledge integration already in vogue.

CONCLUSION

In this research, a review of some of the related research in the field of distributed data mining with particular focus on distributed association rule mining was carried out. Special attention was also given to the review of existing agent-based systems in the area of study. The strengths, weaknesses and challenges of various methods were also presented. It should be noted that appreciable study has been done in this area yet there is still much grounds to be covered in terms of technicalities and efficiency of the systems.

Researchers in this area should also focus more on developing algorithms and architectures that will reduce massive data movement in global knowledge mining and integration. This has the prospect of greatly reducing the response time of distributed association rule mining. Future algorithms and methods should also consider the development of adaptive, fault-tolerant and easily extendable systems in the area of agent-based distributed association rule mining. Such systems will greatly reduce communication and interpretation costs, improve autonomy, efficiency and scalability, collaboration, security and trustworthiness of the DARM system, all of which are common issues with existing systems.

REFERENCES

Agrawal, R. and R. Srikant, 1994. Fast algorithm for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, Sept. 12-15, San Francisco, CA, USA., pp: 487-499.

Agrawal, R., T. Imielinski and A. Swami, 1993. Mining association rules between sets of items in large databases. Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, May 25-28, ACM, New York, USA., pp: 207-216.

Albashiri, K.A., 2010. EMADS: An investigation into the isssues of multi-agent data mining. Ph.D. Thesis, The University of Liverpool, Ashton Building, Ashton Street, Liverpool L69 3BX, United Kingdom.

Albashiri, K.A., F. Coenen and P. Leng, 2009. EMADS: An extendible multi-agent data miner. Knowledge Based Syst. J., 22: 523-528.

Ariwa, E.I., M.B. Senousy and M.M. Medhat, 2003. Informatization and E-business model application for distributed data mining using mobile agents. Proceedings of the International Conference WWW/Internet, (WWWI'03), USA., pp: 85-92.

Ashrafi, M.Z., D. Taniar and K. Smith, 2004. ODAM: An optimized distributed association rule mining algorithm. IEEE Distributed Syst. Online, Vol. 5, No. 3. 10.1109/MDSO.2004.1285877

Badal, N. and S. Tripathi, 2010. Frequent data itemset mining using VS_apriori algorithms. Int. J. Comput. Sci. Eng., 2: 1111-1118.

Bailey, S., R. Grossman, H. Sivakumar and A. Turinsky, 1999. Papyrus: A system for data mining over local and wide area clusters and super-clusters. Proceedings of the Conference on Supercomputing, Nov. 14-19, Portland, USA., pp: 63-63.

Botia, A., R. Garijo and F. Skarmeta, 1998. A generic data mining system: Basic design and implementation guidelines. Proceedings of the Workshop on Distributed Data Mining at the 4th International Conference on Data Mining and Knowledge Discovery (KDD-98).

Byrd, M. and C. Franke, 2007. The state of distributed data mining. ECS265 Project Report, UC Davis, Davis CA., USA.

Cao, L., C. Luo and C. Zhang, 2007. Agent-mining interaction: An emerging area. Autonomous Intell. Syst.: Multi-Agents Data Min., 4476: 60-63.

Chattratichat, J., J. Darlington, Y. Guo, S. Hedvall, M. Kohler and J. Syed, 1999. An Architecture for distributed enterprise data mining. Proceedings of the 7th International Conference on High-Performance Computing and Networking, April 1999, Springer-Verlag, London, UK., pp: 573-582.

Chen, M.S., J. Han and P.S. Yu, 1996. Data mining: An overview from a database perspective. IEEE Trans. Knowledge Data Eng., 8: 866-883.

Chia, T.H. and S. Kannapan, 1997. Strategically mobile agents. Proceedings of the 1st International Workshop on Mobile Agents, April 7-8, Springer-Verlag London, UK., pp: 149-161.

Coenen, F., G. Goulbourne and P. Leng, 2004. Tree structures for mining association rules. Data Min. Knowledge Discovery, 8: 25-51.

Crowley, C.P., 1997. Operating Systems: A Design-Oriented Approach. Irwin Publications, Boston, pp: 883.

Dale, J., 1997. A mobile agent architecture to support distributed resource information management. Ph.D. Thesis, Department of Electronics and Computer Science, Faculty of Engineering, University of Southampton.

Gray, R., 1995. Proposal: Transportable agents. Ph.D. Thesis, Department of Computer Science, Dartmouth College.

Gray, R.S., D. Kotz, G. Cybenko and D. Rus, 2000. Mobile agents: Motivations and state of the art systems. Technical Report.

Grossman, R., S. Kasif, R. Moore, D. Rocke and J. Ullman, 1999. Data mining research: Opportunities and challenges. A Report of Three Workshops on Mining Large, Massive and Distributed Data.

Guo, Y. and J. Sutiwaraphun, 1999. Integrating knowledge in distributed data mining. Department of Computing, Imperial College.

Gyorodi, C., R. Gyorodi and S. Holban, 2004. A comparative study of association rules mining algorithms. Proceedings of the 1st Romanian-Hungarian Joint Symposium on Applied Computational Intelligence, May 25-26, Timisoara, Romania, pp: 213-222.

Han, J., J. Pei and Y. Yin, 2000. Mining frequent patterns without candidate generation. Proceedings of ACM SIGMOD International Conference on Management of Data, May 15-18, Dallas, TX., pp: 1-12.

Han, J., J. Pei, Y. Yin and R. Mao, 2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining Knowledge Discovery, 8: 53-87.

Hipp, J., U. Guntzer and G. Nakhaeizadeh, 2000. Algorithms for association rule mining-a general survey and comparison. SIGKDD Explorations, 2: 58-64.

Ivancsy, R., F. Kovacs and I. Vajk, 2004. An analysis of association rule mining algorithms. Proceedings of the 4th International ICSC Symposium on Engineering of Intelligent Systems, Feb. 29-March 2, Island of Madeira, Portugal, pp: 774-778.

Jensen, D., M. Rattigan and H. Blau, 2003. Information awareness: A prospective technical assessment. Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 24-27, Washington, DC. USA., pp: 378-387.

Johnson, T., L.V.S. Lakshmanan and R.T. Ng, 2000. The 3W model and algebra for unified data mining. Proceedings of the 26th International Conference on Very Large Data Bases, Sept. 10-14, Cairo, Egypt, pp: 21-32.

Klusch, M., S. Lodi and G. Moro, 2003. Agent-Based Distributed Data Mining: The KDEC Scheme. In: Intelligent Information Agents: The Agent Link Perspective, Klusch, M., S. Bergamaschi, P. Edwards and P. Petta (Eds.). Springer, New York.

Kotsiantis, S. and D. Kanellopoulos, 2006. Association rules mining: A recent overview. GESTS Int. Trans. Comput. Sci. Eng., 32: 71-82.

Krebs, B., 2003. Online piracy spurs high-tech arms race. The Washington Post, http://www.washingtonpost.com/ac2/wp-dyn/A34439-2003Jun26.

Legler, T., W. Lehner, J. Schaffner and J. Kruger, 2009. Robust and distributed top-n frequent-pattern mining with SAP BW accelerator. J. Proc. VLDB Endowment, 2: 1438-1449.

Lloyd, B., 2003. Been gazumped by Google: Trying to make sense of the Florida update. Search Engine Guide, http://www.searchengineguide.com/lloyd/2003/1125_bl1.html.

Malhi, B., 1998. Providing support for resource management tools in a wide area high performance distributed data mining system. Master Thesis, Laboratory for Advanced Computing, University of Illinois at Chicago.

Martin, G., A. Unruh and S. Urban, 1999. An agent infrastructure for knowledge discovery and event detection. Technical Report MCC-INSL-003-99, Microelectronics and Computer Technology Corporation (MCC).

Paul, S. and P. Saravanan, 2008. Knowledge integration in a parallel and distributed environment with association rule mining using XML data. Int. J. Comput. Sci. Network Security, 8: 334-339.

Paul, S., 2010. An optimized distributed association rule mining algorithm in parallel and distributed data mining with xml data for improved response time. Int. J. Comput. Sci. Inform. Technol., 2: 88-101.

Prodromidis, A., 1999. Management of intelligent learning agents in distributed data mining systems. Ph.D. Thesis, School of Arts and Science, Columbia University.

Provost, F. and V. Kolluri, 1997. Scaling up inductive algorithms: An overview. Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Aug. 14-17, Newport Beach, California, USA., pp: 239-242.

Rao, V.S. and S. Vidyavathi, 2010. Distributed data mining and mining multi-agent data. Int. J. Comput. Sci. Eng., 2: 1237-1244.

Rudowsky, I., 2004. Intelligent agents. Commun. Assoc. Inform. Syst., 14: 275-290.

Silvestri, C., 2006. Distributed and stream data mining algorithms for frequent pattern discovery. Ph.D. Thesis, Universita Ca Foscari di Venezia.

Stolfo, S., A.L. Prodromidisz, S. Tselepis, W. Lee, D.W. Fan and 1997. JAM: Java agents for meta-learning over distributed databases. Proceedings of the 3rd International Conference on Data Mining and Knowledge Discovery, Aug. 14-17, AAAI Press, Newport Beach, California, pp: 74-81.

Symeonidis, A.L. and P.A. Mitkas, 2006. Agent Intelligence Through Data Mining (Multiagent Systems, Artificial Societies and Simulated Organizations). Vol. 26, Springer-Verlag, New York, pp: 1-206.

Umarani, V. and M. Punithavalli, 2010. Sampling based association rules mining-a recent overview. Int. J. Comput. Sci. Eng., 2: 314-318.

Webb, G.W., 2000. Efficient search for association rules. Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 20-23, Boston, MA. USA., pp: 99-107.

Wooldridge, M., 2009. An Introduction to MultiAgent Systems. 2nd Edn., John Wiley and Sons, New York, pp: 461.

Zaki, M.J., 1999. Parallel and distributed association mining: A survey. IEEE Concurrency Special Issue Parallel Mechan. Data Mining, 7: 14-25.