ENHANCING ASSOCIATION RULES ALGORITHMS FOR MINING DISTRIBUTED DATABASES

WALID ADLY ATTEYA ABDO

PHD

UNIVERSITY OF BRADFORD

2012

ENHANCING ASSOCIATION RULES ALGORITHMS FOR MINING DISTRIBUTED DATABASES

Integration of Fast BitTable and Multi-agent Association Rules mining in distributed medical databases for Decision Support.

Walid Adly Atteya Abdo

Submitted for the degree of
Doctor of Philosophy

Department of Computing
School of Computing, Informatics and Media
University of Bradford

2012

Keywords: Association Rules, Data Mining, Multi-Agent Systems, Distributed Systems, Decision Support Systems.

Abstract

Over the past few years, mining data located in heterogeneous and geographically distributed sites have been designated as one of the key important issues. Loading distributed data into centralized location for mining interesting rules is not a good approach. This is because it violates common issues such as data privacy and it imposes network overheads. The situation becomes worse when the network has limited bandwidth which is the case in most of the real time systems. This has prompted the need for intelligent data analysis to discover the hidden information in these huge amounts of distributed databases.

In this research, we present an incremental approach for building an efficient Multi-Agent based algorithm for mining real world databases in geographically distributed sites. First, we propose the Distributed Multi-Agent Association Rules algorithm (DMAAR) to minimize the all-to-all broadcasting between distributed sites. Analytical calculations show that DMAAR reduces the algorithm complexity and minimizes the message communication cost. The proposed Multi-Agent based algorithm complies with the Foundation for Intelligent Physical Agents (FIPA), which is considered as the global standards in communication between agents, thus, enabling the proposed algorithm agents to cooperate with other standard agents.

Second, the BitTable Multi-Agent Association Rules algorithm (BMAAR) is proposed. BMAAR includes an efficient BitTable data structure which helps in compressing the database thus can easily fit into the memory of the local sites. It also includes two BitWise AND/OR operations for quick candidate itemsets generation and support counting. Moreover, the algorithm includes three transaction trimming techniques to reduce the size of the mined data.

Third, we propose the Pruning Multi-Agent Association Rules algorithm (PMAAR) which includes three candidate itemsets pruning techniques for reducing the large number of generated candidate itemsets, consequently, reducing the total time for the mining process.

The proposed PMAAR algorithm has been compared with existing Association Rules algorithms against different benchmark datasets and has proved to have better performance and execution time. Moreover, PMAAR has been implemented on real world distributed medical databases obtained from more than one hospital in Egypt to discover the hidden Association Rules in patients' records to demonstrate the merits and capabilities of the proposed model further. Medical data was anonymously obtained without the patients' personal details. The analysis helped to identify the existence or the absence of the disease based on minimum number of effective examinations and tests. Thus, the proposed algorithm can help in providing accurate medical decisions based on cost effective treatments, improving the medical service for the patients, reducing the real time response for the health system and improving the quality of clinical decision making.

Declaration

I hereby declare that this thesis has been genuinely carried out by myself and has not been used in any previous application for any degree. The invaluable participation of others in this thesis has been acknowledged where appropriate.

I also declare that this thesis has not been submitted, either in the same or different form, to this or any other university for any degree.

Dedication

In the Name of Allah, the Most Gracious, the Most Merciful

This thesis is dedicated to my parents, wife and children whose encouragement served as a source of inspiration. I would like to thank them for their patience, support, understanding and love.

Acknowledge

In the Name of Allah, the Most Gracious, the Most Merciful

All Praise to Allah for His Glorious Ability and Great Power for giving me the patience and knowledge to complete this doctoral thesis.

I would like to express my gratitude to all those who supported and encouraged me to complete this thesis.

First of all, I would like to express my sincere appreciation to my supervisor, Doctor Keshav Dahal for his unlimited guidance, advice and invaluable comments over years and for the interesting experience in writing and publishing papers. I would not be here without his help.

Last but not least, my deep appreciation to all my family: my mother, my father, my lovely wife and kids.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
BitTable	Bit Table data structure
BitTableFI	Bit Table Frequent Itemset
BMAAR	BitTable Multi-Agent Association Rules Algorithm
BT	Bit Table Algorithm
CD	Count Distribution algorithm
CM	Confusion Matrix
CV	Cross Validation
DHP	Direct Hashing and Pruning
DM	Data Mining
DMA	Distributed Mining of Association Rules
DMAAR	Distributed Multi-Agent Association Rules Algorithm
DSS	Decision Support Systems
DTFIM	Distributed Tire Frequent Itemset Mining algorithm
FAMDFS	Fast Algorithm for Mining Global Frequent Sub-Tree
FIPA	Foundations for Intelligent Physical Agents
FN	False Negative
FP	False Positive
FPM	Fast Parallel Mining algorithm

HDFS	Hadoop Distributed File System
KB	Knowledge Base
KD	Knowledge Discovery
KDD	Knowledge Discovery in Databases
MAE	Mean Absolute Error
MAS	Multi-agent systems
MID	Middle variable
Minconf	Minimum Confidence
Minsup	Minimum Support
MPI	Message Passing Interface
MSE	Mean Squared Error
NPV	Negative Predictive Value
PDM	Parallel Data Mining algorithm
PMAAR	Pruned Multi-Agent Association Rules Algorithm
PPV	Positive Predictive Value
RMSE	Root Mean Squared Error
TID	Transactional identifier
TN	True Negative
TP	True Positive
UCI	University of California Irvine

Chapter One

1. Introduction

1.1. Introduction

During the past decades, various techniques have been proposed to collect and store data in database systems. Some systems require by nature physically separated computer machines such as point-of-sale system, banking system, supply chain management system, etc. Other systems require decentralized environments for the sake of reliability, security, fault tolerance, incremental growth, offered data and services, flexibility and performance. The low costs of computer hardware have supported the increasing demand for these distributed systems. However, despite the need of these systems for intelligent data analysis, yet, data collection and storage have outpaced the analysis and the extraction of useful knowledge inside data (Suh, 2011).

Data Mining is the discovery of hidden patterns inside huge amounts of transactional database in order to extract useful information (Maimon and Rokach, 2010). Data Mining identifies the dependencies and the relationships between different attributes inside the database (Cios et al., 2010). Consequently, it can predict the future trends, forecast the data behavior and

identify future missing values. This helps decision makers to take knowledge driven decisions while solving business problems that are traditionally time consuming when solved by the domain experts (Dunham, 2006).

When the data is distributed among different sites, it is not feasible to move the data from all sites to one location in order to apply the Data Mining techniques. Loading distributed data into centralized location for mining interesting rules is not a good approach. This is because it violates common issues such as data privacy and it imposes network overheads. The situation becomes worse when the network has limited bandwidth which is the case in most of the real time systems (Liu et al., 2011). This has prompted the need for advanced Data Mining techniques to discover the useful information in distributed databases.

One of the main challenges of the distributed Data Mining is the data communication which is considered as the bottleneck of these systems (Da Silva et al., 2005). Other challenges include the distance between the distributed sites, the network bandwidth, the cost of moving data through the network, the ability of the implemented algorithms to exploit parallelism in order to enhance their performance.

For these reasons, centralized Data Mining techniques cannot be applied to distributed databases and must be modified to work in decentralized environments. The goal is to present new techniques that can allow each site to mine and extract useful information from its own local data. At the same time, the site should benefit from the data available at other sites without moving or having direct access to this remote data.

1.2. Motivation

Most of the databases nowadays are geographically distributed. Association rules algorithms seek to increase the performance of the mining process by proposing additional features. Two different paradigms were proposed (Cheung et al., 2002). The first is Count Distribution and the second is Data Distribution.

In the Count Distribution paradigm, every site has to keep the count supports of all local candidate itemsets for all other sites in each iteration. This requires much space especially if the number of candidate itemsets is large. Algorithms like Count Distribution (CD) (Agrawal and Shafer, 1996) and Parallel Data Mining (PDM) (Park et al., 1997) are categorized under the Count Distribution paradigm.

On the other side, algorithms like Data Distribution (DD) (Agrawal and Shafer, 1996), Intelligent Data Distribution (IDD) (Han et al., 1997) and Hash Based Parallel (HPA) (Shintani and Kitsuregawa, 1996) are categorized under the Data Distribution paradigm.

In this paradigm, every site is responsible for keeping the support count of a subset of candidate itemsets. Transactional database records from all other sites that are related to this subset must be shipped to this site for the counting process. Transferring transactional database between sites is a redundant computational process since all records must be processed the same number as the number of sites. Transferring transactional database is also a time consuming process especially if the network bandwidth is limited and the size of the distributed databases is huge. Moreover, performance of the Data

Distribution paradigm is worse than the Count Distribution paradigm (Cheung et al., 2002). For these reasons, we have considered only the Count Distribution paradigm in our research.

Count Distribution algorithm (CD) (Agrawal and Shafer, 1996) is an extension for Apriori sequential Association Rules algorithm (Agrawal and Srikant, 1994). In this algorithm, every site calculates the candidate itemsets based on the frequent itemsets generated from the previous iteration. Local support counts for these candidates are counted at every site and broadcasted to all other sites. Subsequently, all sites calculate the globally large itemsets by summing up the support counts received from all other sites. Finally, every site proceeds to the next iteration. CD algorithm has a simple communication scheme for count exchange. This makes it very convenient for parallel mining especially when considering the response time (Cheung et al., 2002). However, the algorithm generates lots of candidates and incurs a large amount of communication due to all to all broadcasting (Yang et al., 2010, Cheung et al., 2002). The number of candidates (C_k) generated by Apriori is calculated as (Park et al., 1997):

$$\left|C_k\right| = \binom{\left|L_{k-1}\right|}{2} = \frac{\left|L_{k-1}\right|!}{(\left|L_{k-1}\right| - 2)! \times 2!}$$

Where (L_{k-1}) is the set of frequent itemsets generated at iteration (k-1). For instance, if we have 100 large itemsets at iteration (k-1), 4950 candidate itemset are generated for the next iteration (k) which is a very high number.

Parallel Data Mining (PDM) (Park et al., 1997) which is considered as an adaptation for the Direct Hashing and Pruning sequential algorithm (DHP) (Park

et al., 1995) has been proposed to work in parallel environment. Similar to Apriori, each site broadcasts its candidate itemsets support counts to all other sites. Globally large itemsets are calculated by summing up support counts received from other sites. For quick candidate generation, all sites have to broadcast its hashing result to all other sites. However, PDM as well as CD algorithms suffer from two major problems. First, the number of candidate itemsets generated at every iteration is large (XuePing et al., 2010). Second, both algorithms require $O(n^2)$ messages for support count exchange for every candidate itemset to find the large frequent itemsets at every iteration, where n is the number of sites (Cheung et al., 2002).

Distributed Mining of Association Rules (DMA) was proposed in (Cheung et al., 1996). Similarly, the algorithm requires $O(n^2)$ complexity. However, due to the generation of less number of candidate itemsets than CD and PDM, DMA claims that the complexity is even less than $O(n^2)$. DMA has two major problems. First, DMA sends not only the support counts of the candidate itemsets like CD and PDM but also the candidate itemsets themselves. This increases the communication messages between sites especially when the number of candidate itemsets is large. Second, the performance of DMA is very sensitive to two data characteristics which are the data skewness property and the workload balance. Moreover, DMA needs at least two rounds of message exchanges each iteration. This significantly increases the algorithm response time and makes DMA not efficient for distributed mining (Cheung et al., 2002).

In another extension for DMA, Fast Parallel Mining algorithm (FPM) (Cheung et al., 2002) proposed some partitioning algorithm to partition the database so that the resulting partitions have high balance and skewness

before applying DMA. The algorithm is suitable for parallel mining. However, FPM is not practical to be applied to distributed environments as it repartitions the data among the sites. Repartitioning the data from a site to another is not acceptable as it violates data privacy and is not suitable for limited bandwidth environments.

In Summary, Count Distribution algorithm (CD) has proved to be well-established and effective algorithm. Lots of research adopted its paradigm as a basic infrastructure and improved some of its limitations. Recent extensions of CD applied the Message Passing Interface (MPI) technique to minimize the communication cost between the distributed sites due to all-to-all broadcasting (Chia-Chu and Shen, 2010, Kaosar et al., 2009).

The proposed Message Passing Interface algorithm (MPI) in (Kaosar et al., 2009) reduced the communication overhead by avoiding transmitting unnecessary count values of all itemsets. However, the algorithm did not consider the large number of generated candidate itemsets (Kaosar et al., 2009). The proposed algorithm in (Chia-Chu and Shen, 2010) reduced the size of candidate itemsets, however, the approach required many synchronization points, consequently, more network overhead (Chia-Chu and Shen, 2010).

From the above discussion we conclude that there are many factors that affect the performance of the distributed Association Rules algorithms. Each algorithm claimed to be better in specific criteria, while others claimed to be better in other criteria.

This research presents the development of a generic model and a basic infrastructure for mining databases in geographically distributed sites. The

model is based on the proposed algorithm which enhances the performance criteria that influence the overall efficiency of the distributed Association Rules algorithms.

This model combines different types of technologies, mainly the Association Rules as a Data Mining technique, the Multi-Agent systems to build a model that can operate on distributed databases rather than working on centralized databases only and the Foundations for Intelligent Physical Agents (FIPA) as a global standard in communication between agents thus enabling them to cooperate with other standard agents and allowing the future extension for the proposed model.

Since huge amounts of healthcare data have been collected in Egypt hospitals, yet, not mined over the past few years, there has been an urgent need for intelligent data analysis for the hidden information inside these amounts of distributed data. For this reason, one of the goals of the model is to work in real time environments and to mine the distributed databases without loading the data into centralized location so as not to violate data privacy nor impose network overheads. Moreover, the model should take into account the limited network bandwidth and the limited computer memory.

1.3. Research aim and objectives

The overall aim of the research is to design, develop and implement an efficient algorithm for mining real world databases in geographically distributed sites, taking into consideration the moderate network bandwidth and the limited computer configurations of these sites.

In summary the main objectives of the research are:

- To investigate the literature and the state-of-the-art techniques (Data Mining, Association Rules, Distributed Databases and Distributed Data Mining) and to discover the imposed challenges when applying Data Mining techniques to distributed databases.
- To identify the strength and weakness of the existing distributed Data Mining techniques and to use this to build a list of the performance criteria that can evaluate the future proposed algorithms.
- To identify the existing Data Mining and Knowledge Discovery techniques that can be used to build a distributed Data Mining algorithm.
- To develop a fast and effective agent based algorithm for mining Association Rules in real world distributed databases. The proposed algorithm should enhance the performance criteria mentioned in the literature.
- To compare the proposed algorithm with the existing Data Mining algorithms against benchmark datasets in various domains and at different support values to demonstrate better performance and execution time.

- To illustrate the capability and the effectiveness of the proposed algorithm by applying it to a real world case study for distributed medical databases related to geographically distributed hospitals in Egypt.

1.4. Research Contributions

In summary the main contributions of the research are as follows:

- A list of the performance criteria that affects the overall performance for Association Rules algorithms has been proposed from the strength and weakness of the existing distributed Data Mining techniques. This list provides lots of benefits. First, it standardizes the performance criteria for the distributed Association Rules algorithms. Second, it can be used to measure the efficiency of existing or future Association Rules algorithms. This can help in comparing the efficiency of an association rule algorithm with respect to others. Third, it provides a roadmap for the features that are to be enhanced in order to validate the proposed algorithm.
- An efficient multi-agent based algorithm called the Distributed Multi-Agent Association Rules (DMAAR) for mining real world databases in geographically distributed sites has been proposed. Analytical calculations show that the proposed Multi-Agent based algorithm reduces the complexity and minimizes the message communication cost. The distribution of the mining tasks across different kinds of agents maximizes the algorithm parallelism and minimizes the

waiting time of the algorithm processes. The proposed algorithm complies with the Foundation for Intelligent Physical Agents (FIPA), which is considered as the global standards in communication between agents, thus enabling the ability for cooperating with other standard agents.

- An enhanced version of the algorithm called the BitTable Multi-Agent Association Rules (BMAAR) has been proposed to enhance the performance of DMAAR and to address its memory fitting problems. BMAAR includes an efficient Bit data structure which helps in compressing the database thus can easily fit in memory at local sites. In addition, BMAAR includes two BitWise AND/OR operations for quick candidate itemsets generation and support counting. This helps in increasing the algorithm performance and reducing the overall mining time. Moreover, BMAAR included three transaction trimming techniques to reduce the size of the mined data, consequently reducing the total time needed to extract the rules.
- One of the main problems of Association Rules mining is the large number of candidate itemsets which is time consuming. In order to tackle this problem, BMAAR has been extended to the Pruning Multi-Agent Association Rules algorithm (PMAAR) which includes three candidate itemsets pruning techniques to reduce the number of candidate itemsets. PMAAR has been tested against existing algorithms on five benchmark datasets from UCI machine learning repository that are related to different application domains.

Experimentations have been conducted at different supports. The results showed that PMAAR outperformed other algorithms.

- To illustrate the capability and the effectiveness of the proposed algorithm, PMAAR has been applied to a real world case study for distributed medical databases related to geographically distributed hospitals in Egypt. The goal was to discover the hidden Association Rules in the distributed medical databases efficiently and in less time when compared to existing algorithms. The model has been compared with the traditional Association Rules algorithms and has proved to be more efficient and more scalable. The implementation provided lots of benefits. First, Association Rules related to patients' tests and examinations that are useful for prediction have been extracted. Second, the generated knowledge base can help to improve the response time for the health system by reducing time to identify the patients' disease and predicting the existence or the absence of the disease based on the minimum number of effective tests. Third, the results obtained can help in providing accurate medical decisions based on cost effective treatments and improving the medical service for the patients.

1.5. Methodology and Research Phases

Association Rules algorithms presented in the literature seek to enhance the performance of the mining process by enhancing the data structure, the candidate generation process, the candidate counting process, etc.

Although each algorithm presents different features, yet, there is one common similarity between these algorithms. Each algorithm claims that it outperforms other algorithms in terms of specific criteria but unfortunately it ignores other criteria that may affect the overall performance and efficiency of the mining process.

For this reason, the list of the performance criteria that affects the overall performance for Association Rules algorithms has been collected from literature. The list standardizes the performance criteria for the Association Rules algorithms. In addition, it can be used as to evaluate the efficiency of an algorithm and to compare it with other algorithms. Moreover, it provides a roadmap for the features that are to be enhanced in order to validate the proposed algorithm. The proposed performance criteria for the Association Rules algorithms are as follows:

1. Reducing the time needed for the candidate itemsets generation process.
2. Reducing the time needed for the candidate itemsets counting process.
3. Pruning the large number of generated candidate itemsets.
4. Reducing the number of database transactions.
5. Minimizing the waiting time of the Association Rules tasks.

6. Maximizing the parallelism between the Association Rules processes.
7. Avoiding the all-to-all broadcasting between sites.
8. Minimizing the algorithm complexity cost.
9. Minimizing the algorithm communication and message negotiation costs.
10. Providing an efficient data structure.
11. Compliance with global standards.
12. Suitability for real world environments with limited computer memory.
13. Suitability for real world environments with limited network bandwidth.
14. Algorithm flexibility, extendibility and scalability.

In order to achieve an efficient algorithm, the above performance criteria have been investigated. The implementation steps for the whole model have been divided into four phases as shown in Figure 1.1.

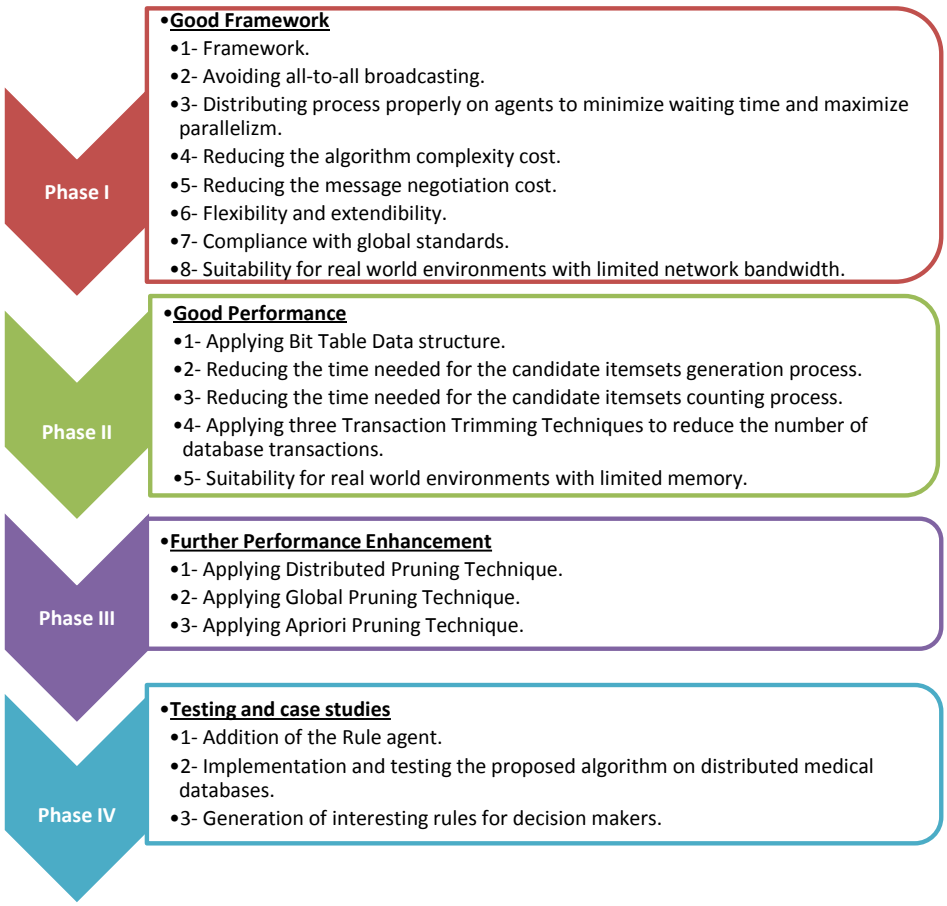

Figure 1.1 Implementation phases of the proposed algorithm

The goals of the first phase were to design and implement a distributed Association Rules algorithm that avoids the all-to-all broadcasting between sites, minimizes the waiting time of the algorithm processes, maximizes the tasks parallelism, minimizes the algorithm complexity and the message negotiation costs, provides flexibility, scalability and is compliant with the global communication standards to cooperate with external entities.

The first phase included the design and the implementation of a distributed Association Rules algorithm based on a good framework but with some modification to avoid the all-to-all broadcasting between sites. In addition, the order and the distribution of the mining tasks were modified to minimize the waiting time and to maximize the tasks parallelism. One of the main goals of the new framework algorithm was to minimize the algorithm complexity and the message negotiation costs. The proposed algorithm was designed to be flexible, compliant with the global standards to communicate with external entities and scalable with the increase in the number of sites.

In this phase, the Distributed Multi-Agent Association Rules algorithm (DMAAR) was presented. DMAAR modified the existing paradigms that were based on the all-to-all broadcasting between distributed sites. DMAAR included a Main Agent that had a global view of all other agents and was considered as the main controller. The proposed algorithm framework together with the Association Rules mining tasks distributed on different agents, decreased the algorithm complexity from $O(n^2)$ to $O(n)$ with less message negotiation cost. Moreover, the order and the distribution of the mining tasks were modified to minimize the waiting time and to maximize the tasks parallelism. Messages between agents are compliant to the global communication standard, the Foundation for Intelligent Physical Agents (FIPA). DMAAR agents generate global and local frequent itemsets unlike CD, PDM, DMA and FPM which generate global frequent itemsets only.

Two kinds of evaluations have been conducted. First, the analytical evaluations for the algorithm complexity and the message negotiation between agents have been conducted and compared with the existing algorithms.

Second, the performance evaluation for the proposed DMAAR algorithm have been tested against existing algorithms on five different benchmark datasets related to different domains. Results have showed better performance and execution time.

The second and the third phases include the performance enhancement for the proposed algorithm by incorporating the best features of the state of the art algorithms. This is not an easy process as some features may contradict with others.

In the second phase, the BitTable Distributed Multi-Agent Association Rules algorithm (BMAAR) is presented. BMAAR is an enhanced version of DMAAR in terms of performance. BMAAR used the efficient BitTable data structure proposed in (Dong and Han, 2007) and adopted in (Thi et al., 2010, Nawapornanan and Boonjing, 2011, Chen and Xiao, 2010, Song et al., 2008, Yang and Yang, 2010). The BitTable data structure has proved better performance and less memory than hash trees data structure used by CD, PDM, DMA and FPM. In addition, BMAAR algorithm applied two BitWise AND/OR operations proposed in BitTableFI algorithm (Dong and Han, 2007) for quick candidate itemsets generation and quick support counting. Moreover, BMAAR applied three transaction trimming techniques, namely Direct Hashing and Pruning proposed in (Park et al., 1997) and adopted in (Najadat et al., 2011, Özel and Güvenir, 2001), the proposed Grouping Identical Transactions technique and the Non Frequent Itemset Removal Technique proposed in (Park et al., 1997) and adopted in (Najadat et al., 2011, Özel and Güvenir, 2001).Three kinds of evaluations were conducted. First, analytical evaluations for the algorithm complexity and the message negotiation costs were conducted

and compared with existing algorithms. Second, performance evaluation for the applied transaction trimming techniques was conducted. In this evaluation, the total number of transactions and the number of transactions after applying the trimming techniques were calculated and compared with existing algorithms. Third, the performance evaluation of BMAAR was tested against DMAAR and existing algorithms on five different benchmark datasets related to different domains. BMAAR showed better performance and execution time.

In the third phase, the Pruned Multi-Agent Association Rules algorithm (PMAAR) which is considered as further improvement for BMAAR was presented. The algorithm applied three pruning techniques to dramatically reduce the number of generated candidate itemsets, namely, Global Pruning Technique proposed in (Cheung et al., 2002) in adopted in (Rui and Zhiyi, 2011), Distributed Pruning Technique proposed in (Cheung et al., 2002) and adopted in (Rui and Zhiyi, 2011) and Apriori Pruning Technique proposed in (Agrawal and Shafer, 1996) and adopted in (Li and Li, 2010, Chong and Yanqing, 2011). PMAAR allocated the candidates generation process to the Local Agents instead of the Main Agent as proposed by BMAAR to apply the pruning techniques efficiently. Two kinds of evaluations were conducted. First, analytical evaluation was conducted to compare the complexity of PMAAR with the existing algorithms. Second, performance evaluation of PMAAR was conducted with existing algorithms against five benchmark datasets. PMAAR showed better performance and execution time.

In the fourth phase, PMAAR was implemented on distributed medical databases for the Inflammation of urinary bladder disease and the Nephritis of renal pelvis origin disease in order to demonstrate the capability of applying the

proposed algorithm to real world environment. To discover the hidden Association Rules in the medical database, the proposed Rule Agent was added to PMAAR. The Rule Agent is based on Apriori Algorithm presented in (Agrawal and Srikant, 1994) and adopted in (Hanguang and Yu, 2012). However, some parts of this algorithm were modified to work in distributed and Multi-Agent environment. Moreover, the Rule Agent algorithm was also modified to deal with the itemsets discovered by the Main Agent. The extracted medical rules helped in identifying the minimum effective number of tests and the prediction of the existence or the absence of the diseases for patients. The proposed Multi-Agent based algorithm also improved the diagnostic knowledge of the doctors. Performance evaluation of PMAAR showed better performance when compared with the existing algorithms.

1.6. Thesis Structure

The research consists of three proposed algorithms and a case study. For each algorithm, the sequence diagram, the activity diagram, the algorithm description, the message negotiation cost, the algorithm complexity cost and the performance evaluation of the algorithm are presented. The thesis consists of seven chapters organized as follows:

Chapter 1: Introduction

This chapter presents the introduction, the motivations, the research questions, the methodology and the research objectives. The chapter summarizes the research contributions from the technical and the medical points of view.

Chapter 2: Literature Review

This chapter contains a literature review about relevant fields of literature. This includes basic terminologies for the different technologies, namely the distributed Data Mining, Distributed Artificial Intelligence and the Multi-Agent systems.

Chapter 3: The proposed Distributed Multi-Agent Association Rules Algorithm (DMAAR)

This chapter presents the proposed DMAAR algorithm. This includes the solution architecture, the sequence of the FIPA messages between agents and the UML Sequence and Activity diagrams. The chapter also presents the analytical evaluation of the proposed algorithm including the algorithm complexity and the message negotiation costs in addition to the performance evaluation of the algorithm against five UCI repository benchmark datasets.

Chapter 4: The proposed BitTable Multi-Agent Association Rules Algorithm (BMAAR)

This chapter presents the proposed extended version of DMAAR named as BMAAR. This includes the solution architecture, the sequence of the FIPA messages between agents and the UML Sequence and Activity diagrams. The chapter also presents the analytical evaluation of BMAAR in terms of algorithm complexity and the message negotiation costs in addition to the performance evaluation of BMAAR against five UCI repository benchmark datasets. Another evaluation has been conducted to test the effect of applying the transaction trimming techniques to the proposed algorithm.

Chapter 5: The proposed Pruned Multi-Agent Association Rules Algorithm (PMAAR)

This chapter presents the proposed extended version of BMAAR named as PMAAR. Similarly the chapter includes the solution architecture, the sequence of the FIPA messages between agents and the UML Sequence and Activity diagrams. The chapter also presents the analytical evaluation of PMAAR in terms of algorithm complexity and message negotiation costs in addition to the performance evaluation of PMAAR against five UCI repository benchmark datasets. Another evaluation has been conducted to test the effect of applying the pruning techniques to the proposed algorithm.

Chapter 6: A case study for early prediction of urinary bladder inflammation disease

This chapter presents the addition of the Rule Agent to PMAAR and the application of the whole model to medical databases for mining Association Rules in the geographically distributed hospitals.

Chapter 7: Conclusion and Future Work

This chapter presents the conclusion and the future work for the research.

1.7. Publications

1.7.1. Journals

1. **2012:** W. Atteya, K. Dahal, and M. Hossain, "BitTable Structured Multi-Agent Association Rules Mining Algorithm for Distributed Databases." In Agent-based Modeling & Simulation of Complex Adaptive Communication Networks & Environments (CACOONS). http://mc.manuscriptcentral.com/simulation (submitted).

1.7.2. Conferences

1. **2011:** W. Atteya, K. Dahal, and M. Hossain, "Multi-Agent Association Rules Mining in Distributed Databases," in Soft Computing in Industrial Applications, 15th World Conference on Soft Computing in Industrial Applications (WSC15). vol. 96, A. Gaspar-Cunha, R. Takahashi, G. Schaefer, and L. Costa, Eds., ed: Springer Berlin / Heidelberg, 2011, pp. 305-314.

2. **2011:** W. Atteya, K. Dahal, and M. Hossain, "Distributed BitTable Multi-Agent Association Rules Mining Algorithm. Knowledge-Based and Intelligent Information and Engineering Systems." In Proceedings of the 15th international conference on Knowledge-based and intelligent information and engineering systems. vol. 6881, A. König, A. Dengel, K. Hinkelmann, K. Kise, R. Howlett, and L. Jain, Eds., ed: Springer Berlin / Heidelberg, 2011, pp. 151-160.

3. **2011:** W. A. Atteya, K. Dahal, and M. A. Hossain, "An efficient agent based framework for distributed medical databases," in 5th International Conference on Software, Knowledge Information, Industrial Management and Applications (SKIMA),IEEE, 2011,pp.1-6.

4. **2010:** W. A. Atteya, K. Dahal, and M. A. Hossain, "Multi-agent system for early prediction of urinary bladder inflammation disease," in 10th International Conference on Intelligent Systems Design and Applications (ISDA), IEEE, 2010, pp. 539-544.

Chapter Two

2. Literature Review

2.1. Chapter Overview

The purpose of this chapter is to present a literature review which focuses on several relevant themes. (i) A brief background on Knowledge Discovery in databases (KDD), Data Mining (DM), Association Rules as a Data Mining technique and Multi-agent systems. (ii) A more extensive literature review regarding the related exiting work.

2.2. Knowledge Discovery in databases (KDD)

Traditionally, human experts have acquired their knowledge from their own personal experience and observation. With the advances in computer automation and due to the enormous volumes of data being collected and stored in databases, Knowledge Discovery has become an important Artificial Intelligence (AI) research topic in a large number of organizations. Knowledge Discovery is defined as the extraction of implicit and previously unknown knowledge from data or observations (Fayyad et al., 1996a).

KDD extracts the hidden information that can be turned into knowledge for strategic decision-making and problem solving (Mitra et al., 2002). KDD is an iterative process with six stages (Fayyad et al., 1996b):

1) Developing a good understanding for the problem and its application domain and for the main purpose of the application.
2) Creating a target dataset using by selecting a sample of the required data.
3) Removing and correcting corrupted data. This can be achieved by data preprocessing, data cleaning, fixing missing values and removing noisy data.
4) Applying data reduction by selecting the data whose features can represent the whole data.
5) Applying a data-mining algorithm such as Association Rules, neural network, decision trees, etc.
6) Interpreting the mined patterns and understanding the results. The results obtained should be used in decision making process.

Although all stages are equally important, however, some stages may be skipped. Moreover, the stages are not necessarily sequential. Results obtained at certain stage can cause the process to return back to a previous stage. Stages of the Knowledge Discovery process are presented in Figure 2.1.

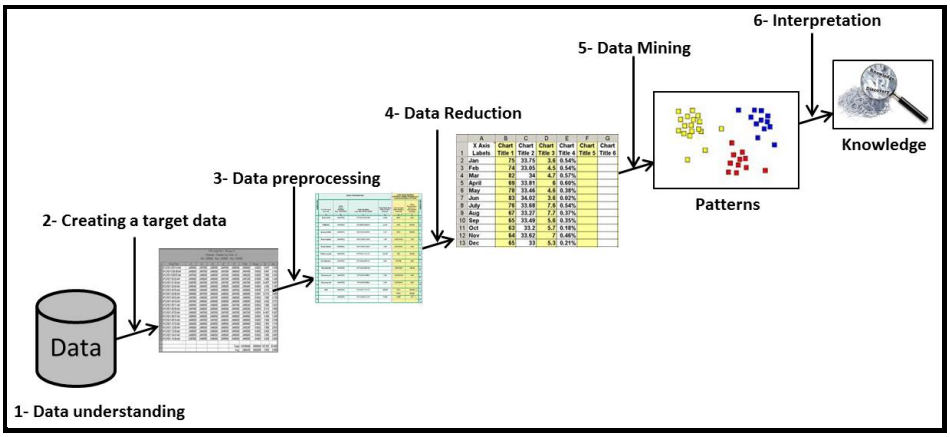

Figure 2.1 Knowledge Discovery in Databases

2.3. Data Mining (DM)

While KDD refers to the overall process of discovering useful knowledge from data, Data Mining refers to a particular step in this process. Data Mining is application of specific algorithms for extracting hidden patterns from datastages (Fayyad et al., 1996b).

Due to the increasing amounts of large data in many fields of information technology, Data Mining techniques have become very popular. However, these techniques often require high performance approaches in order to cope with the overwhelming amount of data and the complexity of algorithms (Di Fatta and Fortino, 2007). Data Mining process may fall into one of these three general categories (Kantardzic, 2011):

- ***Discovery:*** The process of discovering unpredicted and unknown patterns in data without any predetermined idea about these patterns.
- ***Predictive Modeling***: The process of using these discovered patterns to predict the future.

- ***Forensic Analysis:*** The process of using these patterns to find anomalous or unusual data elements.

Two kinds of learning are involved in Data Mining. In supervised learning, sometimes named as directed analysis, the data has a specific column that is used as the goal for discovery or prediction. While in the unsupervised learning, the system has no teacher, but simply tries to find interesting clusters of patterns within the dataset. Unsupervised discovery can sometimes be used for data segmentation or clustering (e.g., finding classes of customers that group together).

In order to extract the implicit knowledge and relationships, which are not explicitly stored in databases, a wide variety of Data Mining techniques have been proposed such as Classification, Association Rules, Sequential pattern, Clustering, Outlier Analysis, Regression, Neural Networks, Decision Trees, genetic Algorithms, Bayesian network and Nearest Neighbor (Tan, 2007, Dunham, 2006). However, a single data-mining technique has not been proven appropriate for every domain and for every dataset (Olson and Delen, 2008).

Data Mining is widely used in various application domains such as e-learning (Scheuer and McLaren, 2012), traffic data (Marukatat, 2007), education (Romero and Ventura, 2007), textual databases (Menon et al., 2005), credit cards (Imberman, 2002), medical domain (Lei and Ren-hou, 2007), etc.

A system of Data Mining implies different user categories, which mean that the user's behavior must be a component of the system. The problem at this level is to know which algorithm of which method to employ for an exploratory end, which one for a decisional end, and how can they collaborate

and communicate. Agent paradigm presents a new way of conception and realizing of Data Mining system. The purpose is to combine different algorithms of Data Mining to prepare elements for decision-makers, benefiting from the possibilities offered by the Multi-Agent systems (Zghal et al., 2005).

2.4. Association Rules

Association rules technique is one of the most important techniques in Data Mining. It is applied to lots of applications such as market basket analysis in order to discover the dependency between items in a transactional database (Hahsler et al., 2005).

Discovering Association Rules between items over market basket data was introduced by Agrawal in Apriori Algorithm (Agrawal and Srikant, 1994). Market basket data, also called scanner panel data, is collected by most retail business organizations such as super markets, mail order companies, and so on. A record in the basket data typically consists of items bought by a customer along with other data such as the date of transaction, quantity and price of the items, etc. Association rules identify the sets of items that are most often purchased with another set of items. For example, an association rule may state "95% of customers who bought items A and B also bought C and D." This type of information may be used for cross marketing, store layout, product placement, customer segmentation, etc.

Definition 2.1: Formally, the problem can be stated as follows: Let the itemset $I = \{i_1, i_2, .. i_m\}$ be a set of m distinct literals called items. D = is a set of variable length transactions over I. T = is a set of items $i_1, \ldots, i_k \subset I$ that has a unique

identifier TID. Thus *D* can be considered a relation over $i_1, i_2, \ldots, i_m$ and TID. The length of an itemset is the number of items in this itemset. An itemset *C* is called k-itemset if it consists of k items ($c.item_1$, $c.item_2$, , , , $c.item_k$) or (c[l]. c[2] , … , c[k]). The items in an itemset are always in the lexicographic order, i.e., for an itemset c: $c.item_1 < c.item_2 <, \ldots, < c.item_k$. This will ensure no duplicate itemsets are generated and considered in counting the support. Similarly, the items in a transaction are stored in the lexicographic order. This requirement does not affect the generality of the discussion below. However, it simplifies the set inclusion operation to test if an itemset is contained in the transaction.

Definition 2.2: An association rule is in the form: $X \rightarrow Y$ where $X \subset I$, $Y \subset I$, and $X \cap Y = \Phi$. For an association rule $X \rightarrow Y$, *X* is called the antecedent and *Y* is called the consequent of the rule.

Definition 2.3: The support for the rule LHS => RHS is the percentage of transactions that hold all of the items in the LHS and RHS (Agrawal and Srikant, 1994). If the support is low, it implies that there is no overwhelming evidence that items in LHS and RHS occur together, because they both happens in only a small fraction of transactions. The support of the rule $X \rightarrow Y$ is computed as the ratio: *X U Y / total_transactions.*

Definition 2.4: The confidence for the association rule LHS =>RHS is the percentage of transactions that include LHS and RHS to those who include LHS (Agrawal and Srikant, 1994). The confidence of the rule $X \rightarrow Y$ is computed as the ratio: *Support (X U Y) /Support(X).*

2.4.1. **Apriori algorithm in a nutshell**

Apriori algorithm is one of the most well-known Association Rules mining algorithms. The algorithm is used to extract rules and implicit knowledge from database attributes in order to find the association and the correlation between these attributes (Ye and Chiang, 2006).

The problem of mining Association Rules can be stated as follows: Given a transactional database *D*, the association rule problem is to find all rules that have supports and confidences greater than certain user-specified thresholds, denoted by *minsupp* and *minconf*, respectively. The association rule problem can be decomposed into the following sub-problems:

1. Discovering large itemsets: All itemsets that have support above the user specified minsupp are generated. These itemsets are called the (frequent/large) itemsets. All others are said to be small.
2. Generating Association Rules: Use these frequent itemsets to calculate the confidence (*support(X U Y) / support(X)*). If the confidence is greater than *minconf* then generate the rule.

(i) Discovering Large Itemsets:

Discovering large itemsets involves counting the support of the potentially frequent itemsets called candidate itemsets for the given database and identifying which of them actually have the necessary support. Discovering all large itemsets is a nontrivial problem if the cardinality of the set of items (|I|) and the database*(D)* are large (Agrawal and Srikant, 1994). For example, if |I| =

m, the number of possible distinct itemsets is $2^m - 1$. The problem is to identify which of the large number of itemsets have the minimum support for the given set of transactions. For very small values of m, it is possible to setup $2^m - 1$ counters to count the support of every itemset by scanning the database once. However, for many applications m can be more than 1,000. Clearly, this approach is impractical. It should be noted that a very small fraction of this exponentially large number of itemsets would have minimum support. Hence, it is not necessary to test the support for every itemset. Even if practically feasible, testing support for every possible itemset results in much wasted effort. To reduce the combinatorial search space, all algorithms exploit the following property:

Any subset of a large/frequent itemset must also be large/frequent. For instance, if a transaction contains frequent itemset ABCD, then A, AB, BC, ABC, etc. are also frequent. Conversely, all extensions of a small itemset are also small. Therefore, if the itemset ADE is small, then none of the itemsets which are extensions of *ADE*, i.e., *ADEF, ADEFG*, etc., need be tested for minimum support. This is termed as ***Apriori Pruning technique.*** This helps in pruning the number of generated candidate itemsets that are to be tested in the next iteration. For this reason, support for smaller length itemsets can be counted first and only the extensions of frequent itemsets need be counted subsequently.

All existing algorithms for mining Association Rules are variants of the following general approach: initially, support for all itemsets of length 1 (1-itemsets) are tested by scanning the entire database. The itemsets that are found to be small are discarded. A set of 2-itemsets called candidate itemsets

are generated by extending the large 1-itemsets generated in the previous pass by one (1-extension) and their support is tested by scanning the entire database. Many of these itemsets may turn out to be small, and hence discarded. The remaining itemsets are extended by 1 and tested for support. This process is repeated until no more large itemsets are found. In general, the k^{th} iteration contains the following steps:

1. The set of candidate k-itemsets is generated by 1-extensions of the large (k - 1)-itemsets generated in the previous iteration.
2. Supports for the candidate k-itemsets are generated by a pass over the database.
3. The itemsets that do not have the minimum support are discarded and the remaining itemsets are designated large k-itemsets.

Therefore, only extensions for large itemsets are considered in subsequent passes. This process is stopped when in some iteration n, no large itemsets are generated. The algorithm, in this case, makes n database scans. For very large databases with millions of customer transactions, scanning the data repeatedly may be expensive.

(ii) Generating Association Rules

Once the large itemsets are discovered, Association Rules can be generated in a straightforward manner by considering the support of an itemset and the supports of its subsets. Since the subset of a large itemset is also large, the support for all subsets is also available during this step. Hence no additional

itemsets or the transaction data need to be reconsidered. Therefore, generating rules is relatively inexpensive compared to discovering the large itemsets.

Assume the example shown in Table 2.1 for a supermarket database. The example illustrates how the support and the confidence are calculated.

Table 2.1 Transactional database

TID	Items	Time
T1	Milk, bread, juice	08:05
T2	Milk, juice	08:45
T3	Milk, eggs	09:10
T4	Bread, cookies, coffee	09:12

Support of the rule (X =>Y) = number of transactions containing X and Y divided by the number of transactions. Thus, for the rule Milk => Juice has 50% support, while Bread => Juice has only 25% support.

The confidence of the rule (X => Y) = number of transactions containing X and Y divided by the number of transactions containing X. thus, the confidence for the rule Milk => Juice is 66.7% (meaning that, of three transactions in which milk occurs, two contain juice) and bread => juice has 50% confidence (meaning that of two transactions containing bread one contains juice).

In summary, we can say that 66.7% of the customers that buy milk tend to buy juice and 50% of all customers buy both of these items.

For small sets of data involving few items, generating all Association Rules as defined above is straightforward. However, the problem becomes challenging when applied to a large number of transactions involving thousands of items. The emphasis here is to efficiently generate the Association Rules. It

should also be noted that the goal is to find all Association Rules regardless which particular items are contained in the rules. The Association Rules algorithm is shown in Figure 2.2.

The minimum support and minimum confidence constraints directly affect the complexity of generating the Association Rules. For example, if the minimum support is set very high, only few sets of items will qualify and hence the generation of all the Association Rules may be simple even for very large transaction sets. On the other hand, setting the value too low will cause an extremely large number of sets of items to be considered. The choice of suitable values for minimum support and confidence is not discussed here. However, it should be noted that if only a few Association Rules are generated, strong but mostly obvious associations may be discovered and many interesting associations may be lost. On the other hand, generating too many Association Rules may confuse the users.

The Apriori Algorithm

•**Pseudo-code:**
C_k: Candidate itemset of size k
L_k: frequent itemset of size k

1. $L1$ = {Frequent 1-itemsets};
2. **for** ($k=2$; $L_{k-1}!=\emptyset$; k++) **do begin**
3. C_k= Apriori-gen(L_{k-1});
4. **for all** transactions $t \in D$ **do begin**
5. C_t = subset(C_k, t);
6. **for all** candidates $c \in C_t$ **do**
7. $c.count$++;
8. **end;**
9. **end;**
10. $L_k = \{c \in C_k | c.count \geq minsup\}$
11. **end;**
12. Answer = $\cup_k L_k$;

• **Apriori-gen:**
insert into C_k
select $p.item_1$, $p.item_2$,, ... , $p.item_{k-1}$, $q.item_{k-1}$
from L_{k-1} p, L_{k-1} q
where $p.item_1 = q.item_1$, . . ., $p.item_{k-2} = q.item_{k-2}$, $p.item_{k-1} < q.item_{k-1}$;

•**Apriori-prune:**
for all itemsets $c \in C_k$ **do**
for all (k-1) subsets s of c **do**
if ($s \notin L_{k-1}$) then
delete c from C_k;
end if;
end;
end;

Figure 2.2 Apriori Algorithm

One of the limitations of Apriori is that it generates large number of candidate itemsets (Park et al., 1997, Hong et al., 2004). Many variations of Apriori algorithm have been proposed to improve the efficiency of the algorithm. Some of these variations focus on pruning the number of generated candidate itemsets (Park et al., 1997). Others focus on using different data structure such

as the Trie data structure proposed in (Bodon, 2005), the tree-like data structure in FP-Growth (Han et al., 2007, Han and Pei, 2000) and the BitTable data structure proposed in (Dong and Han, 2007, Song et al., 2008) which has proved better performance over the hash tree proposed by Apriori. Others focus on reducing the number of transactional database records such as (Park et al., 1997).

2.5. Multi-Agent Systems (MAS)

An agent is a program that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors (Russell and Norvig, 2003). A human agent has eyes, ears, and other organs for sensors, and hands, legs, mouth, and other body parts for effectors. A robotic agent substitutes cameras and infrared range finders for the sensors and various motors for the effectors. A generic agent is diagrammed in Figure 2.3.

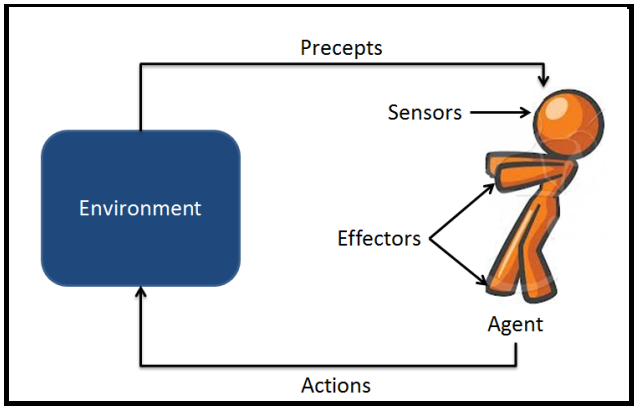

Figure 2.3 Agent-environment interactions through sensors and effectors

Software agents are computational processes-instantiated programs that exist within an environment that they sense and affect (Regli et al., 2009). They exhibit high degree of autonomy, perform actions in their environment based on information received from the environment (Panait and Luke, 2005).

Multi-Agent Systems (MAS) are ones in which several agents attempt, through their interaction to jointly solve tasks or to maximize utility. Agents can be automatic agents that run continuously or semi-automatic agents which only run when triggered by other agents (Mohan et al., 2005).

Multi-agent systems related research is an emerging subfield of Distributed Artificial Intelligence, which aims at providing both: principles for the construction of complex systems involving multiple agents and mechanisms for the coordination of independent agents' behavior. The most important reason to use Multi-Agent systems is to have a more natural modeling for real-life domains that require the cooperation of different parties (Alhajj and Kaya, 2005). In particular, if there are different people with different perspectives or organizations with different goals and proprietary information, then a Multi-Agent system is needed to handle their interaction. Multiple agents are recognized as crucial for many real-world problems, such as engineering design, intelligent search, medical diagnosis, robotics, etc.

2.5.1. **Multi-agents characteristics**

Intelligent agents are an emerging technology that is making computer systems easier to use by allowing people to delegate work back to the computer. In our lives, we seek help in the form of assistants (people who take

care of things that we could do ourselves, but prefer not to do them). In computer world, intelligent agents play the role of assistants. To achieve Artificial Intelligence of the agents, it can enumerate certain characteristics like autonomous, goals-driven, reactive, proactive, collaborative, social and adaptive (Ouali et al., 2003, González et al., 2006).

Several researchers have attempted to provide various classification for the agents' attributes. A list of common agent attributes is shown below:

1. **Autonomy:** An agent is a computational mechanism that performs actions in its environment based on information (sensors, feedback) received from the environment (Panait and Luke, 2005). Agents should be able to do most of their tasks without any direct assistance from an outside source, while controlling their own actions and states (Regli et al., 2009). Autonomous agent is capable of making decisions about what actions to take without constantly referring back to its user (Minghua et al., 2003).
2. **Collaborative / Cooperative / Sociable ability:** The ability to work with other agents or with human to achieve a common goal, resolve common conflicts and inconsistence in information (Xie and Tachibana, 2007).
3. **Continuous:** Agents persist their identity and state over long periods of time (Flores-Mendez, 1999). They are not spawned and terminated for each individual task (Regli et al., 2009).
4. **Reactivity (Responsiveness):** Agents should have the ability to sense and respond to external signals due to changes in the environment. This can be in the form of received and reply

messages (Vlassis, 2007). Agents should be able to respond appropriately to the prevailing circumstances in dynamic and unpredictable environments (Minghua et al., 2003).

5. **Proactiveness:** In some systems, agents may take the attempt to achieve their goals and initiate actions of their own and not only react to the environment nor wait for requests (Marík et al., 2002).
6. **Adaptive:** An agent should have the ability to learn and improve his experience (Flores-Mendez, 1999).
7. **Trust worthy:** An agent should serve users' needs in a reliable way so that users will develop trust in its performance model (Wei et al., 2007).

The proposed agents in the Multi-Agent based framework are autonomous, collaborative, responsive, adaptive and trust worthy.

2.5.2. Agent Classification

From the application point of view, agents are grouped into five classes based on their main purpose within the system (Russell and Norvig, 2003) :

1. **Watcher Agents:** looks for specific information
2. **Learning Agents:** tailors to an individual's preferences by learning from the user's past behavior.
3. **Shopping Agents:** compares "the best price for an item"
4. **Information Retrieval Agents:** helps the user to search for information in an intelligent fashion. The proposed agents in the

Multi-Agent based framework are classified under this class of agents.

5. **Helper Agents:** performs tasks autonomously without human interaction.

Agents can be grouped also based on their degree of perceived intelligence and capability (Russell and Norvig, 2003) :

1. **Simple reflex agents:** Simple reflex agents act only on the basis of the current percept. The agent function is based on the condition-action rule: **if condition then action**. For example: **if** car-in-front-is-braking **then** initiate-braking. Some reflex agents can also contain information on their current state which allows them to disregard conditions already triggered by actuators. The proposed agents in the Multi-Agent based framework are categorized under this group of agents.
2. **Model-based reflex agents:** Model-based agent can handle partially observable environments. Its current state describes the unseen part of the world. This behavior requires information on how the world behaves and works. This additional information completes the “World View” model.
3. **Goal-based agents:** Goal-based agents store information regarding desirable situations. Among multiple possibilities, agents choose the one which reaches the goal state.
4. **Utility-based agents:** Goal-based agents only distinguish between goal states and non-goal states. It is possible to define a measure of how desirable a particular state is. This measure can

be obtained through the use of a utility function which maps a state to a measure of the utility of the state.

5. **Learning agents:** Learning agents have an advantage that they can initially operate in unknown environments and due to their adaptive behavior they become more knowledgeable.

2.6. Related Work

Data Mining plays an important role in the Knowledge Discovery process. Most of the database systems nowadays are distributed among several sites, making the centralized processing of the data very inefficient and vulnerable to security risks. Distributed Data Mining explores techniques to apply Data Mining in a non-centralized way.

One of the limitations of the distributed data mining is the communication due to message negotiation between sites (Da Silva et al., 2005). Algorithms have proposed different algorithms to minimize this cost. Two different paradigms have been proposed (Cheung et al., 2002). The first is Count Distribution and the second is Data Distribution.

In the Data Distribution paradigm, transactional database records must be shipped between sites for the counting process. This is a redundant computational and a time consuming process, especially, if the network bandwidth is limited and the size of the distributed databases is huge. Moreover, performance of the Data Distribution paradigm is worse than the Count Distribution paradigm (Cheung et al., 2002). For these reasons, we have considered only the Count Distribution paradigm in our research. Algorithms like

Data Distribution (DD) (Agrawal and Shafer, 1996), Intelligent Data Distribution (IDD) (Han et al., 1997) and Hash Based Parallel (HPA) (Shintani and Kitsuregawa, 1996) are categorized under the Data Distribution paradigm.

In the Count Distribution paradigm, every site keeps the count supports of all local candidate itemsets for all other sites. This requires much space especially if the number of candidate itemsets is large. Algorithms like Count Distribution (CD) (Agrawal and Shafer, 1996) and Parallel Data Mining (PDM) (Park et al., 1997) are categorized under the Count Distribution paradigm.

Count Distribution algorithm (CD) (Agrawal and Shafer, 1996) is an extension for Apriori sequential Association Rules algorithm (Agrawal and Srikant, 1994). In this algorithm, every site broadcasts local support counts only to other sites. This is a very simple communication scheme for count exchange (Cheung et al., 2002). However, the algorithm generates lots of candidates and incurs a large amount of communication due to all to all broadcasting (Cheung et al., 2002, Yang et al., 2010). The number of candidates (C_k) generated by Apriori is calculated as (Park et al., 1997):

$$|C_k| = \binom{|L_{k-1}|}{2} = \frac{|L_{k-1}|!}{(|L_{k-1}|-2)!\times 2!}$$

Parallel Data Mining (PDM) (Park et al., 1997) adopted the same paradigm of CD, however, for quick candidate generation, all sites have to broadcast their hashing results to all other sites. PDM is considered as an adaptation for the Direct Hashing and Pruning sequential algorithm (DHP) (Park et al., 1995) to be used in the parallel environments. However, both PDM and CD algorithms suffer from the large number of generated candidate

itemsets (XuePing et al., 2010) and high algorithm complexity which is $O(n^2)$ messages (Cheung et al., 2002), where n is the number of sites.

Distributed Mining of Association Rules (DMA) was proposed in (Cheung et al., 1996). Similarly, the algorithm requires $O(n^2)$ complexity. However, due to the generation of less number of candidate itemsets than CD and PDM, DMA claimed that the complexity is even less than $O(n^2)$. However, the message negotiation cost of DMA is higher than CD and PDM. Moreover, the performance of DMA is very sensitive to data skewness property and workload balance. This significantly increases the algorithm response time and makes DMA not efficient for distributedmining (Cheung et al., 2002).

In another extension for DMA, Fast Parallel Mining algorithm (FPM) (Cheung et al., 2002) proposed a partitioning algorithm to partition the database so that the resulting partitions have high balance and skewness before applying DMA. The algorithm is suitable for parallel mining. However, FPM is not practical to distributed environments since repartitioning data from a site to another is not acceptable as it violates data privacy and is not suitable for limited bandwidth environments. FPM algorithm has been adopted in (Jiayi et al., 2010, Renjit and Shunmuganathan, 2010, Wang et al., 2010, Hua-jin et al., 2010).

Most of the Apriori like algorithms such as CD, PDM, DHP, DMA and FPM algorithms are based on the hash trees data structure. Although hash trees data structure is fast, yet, it is complex and memory consuming (Bodon and Rónyai, 2003). In order to overcome these limitations, various algorithms proposed simpler and less memory consuming data structures than hash trees.

In (Bodon and Rónyai, 2003), Bodon proposed the Trie data structure as an alternative to the hash-trees data structure. Experiments conducted on real-life datasets showed that Tries data structure is simpler and faster than hash trees. In his second paper, the Trie data structure has been applied to Apriori Association Rules mining algorithm rather than Hash trees. Experiments proved that the Trie data structure is much faster than Hash trees (Bodon, 2003, Ansari et al., 2008). In another extension, Bodon applied the Trie data structure to mine the frequent itemset sequences (Bodon, 2005).

Although Trie data structure has been proved to be much faster than hash trees, yet, no techniques have been proposed in the algorithm to reduce the time needed for candidate generation nor support counting processes (Yin, 2009).For this reason, Dong proposed the Bit Table Frequent Itemset algorithm (BitTableFI) to address the problem of candidate itemsets generation and support counting (Dong and Han, 2007).

In this algorithm, a special BitTable data structure has been proposed to compress the database in order to fit the transactions into the memory. Moreover, two BitWise AND/OR operations have been proposed to generate the candidate itemsets and to count their supports. Experiments conducted in (Dong and Han, 2007) showed that BitTable data structure is much faster than the hash trees implemented by Apriori. They also showed that it used less memory than hash tree due to the database compression. Moreover, experiments showed that when the BitWise AND/OR operations were applied to the BitTable data structure, the candidate itemsets generation and counting processes were much faster than applying the hash trees.

The BitTable data structure has been used in (Song et al., 2008, Thi et al., 2010, Yang and Yang, 2010, Chen and Xiao, 2010, Nawapornanan and Boonjing, 2011).

In spite of the efficiency of the BitTable data structure in reducing the time needed to generate the candidate itemsets and the support counting, yet, the algorithm did not address the problem of large candidate itemsets generation which increases the time needed for the candidate generation and support counting processes. Different algorithms have presented techniques to reduce the number of generated candidate itemsets.

Apriori Pruning technique proposed in (Agrawal and Shafer, 1996) was based on the fact that *"Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset".* Thus, every generated candidate itemset is checked against the frequent itemsets calculated in the previous pass. If any subset of this candidate itemset is not frequent, the candidate is removed from the list of candidate itemsets for the next iteration. The technique was proved to be successful in pruning large number of generated candidate itemsets and has been well established and was applied to many algorithms and application domains (Li and Li, 2010, Chong and Yanqing, 2011).

Two pruning techniques have been proposed in (Cheung et al., 2002), namely, the Global Pruning and the Distributed Pruning techniques.

The global Pruning techniques based on calculating the expected maximum support for the k itemset based on the support counts of the k-1 subsets for the itemset. If the sum of the maximum supports from all sites is

less than the global minimum support, this itemset is pruned for the next iteration.

In the Distributed Pruning technique, local candidate itemsets are generated in each site and then broadcasted into all other sites rather than broadcasting the frequent itemsets to all other sites then generating the candidate itemsets at each site. The algorithm proved that the proposed technique reduced the number of generated candidate itemsets (Cheung et al., 2002).

Both techniques were successful in pruning large number of generated candidate itemsets and were adopted in (Rui and Zhiyi, 2011). However, another factor that affects the support counting process is the number of transactions in the database. When the number of transactions in the databases is large, the support counting process becomes very time consuming.

For this reason, various algorithms have proposed different techniques to trim part of the transactions that is not useful for the next iteration.

In the Direct Hashing and Pruning technique (DHP), an effective algorithm for trimming the database transactions has been proposed (Park et al., 1997). The algorithm reduces the number of database transactions in order to reduce the search space at early stages. The algorithm also removes the non-frequent 1-itemsets after the first iteration in order to reduce the size of the transactions. Moreover, the algorithm applies a hash table to reduce the size of the candidate k+1 itemsets generated at each step. The transaction trimming technique proposed by DHP has been applied in many algorithms (Najadat et al., 2011, Özel and Güvenir, 2001).

The basic idea proposed in (Özel and Güvenir, 2001) was inspired from the Direct Hashing and Pruning (DHP) algorithm. The algorithm uses the same transaction trimming techniques presented in DHP. However, it improves the proposed candidate itemsets counting process. The algorithm has been tested with real datasets obtained from a large retailing company. Results showed that the algorithm performs better than Apriori algorithm.

In Summary, the Count Distribution algorithm (CD) has proved to be well-established and effective algorithm. Lots of research adopted its paradigm as a basic infrastructure and improved some of its limitations.

Recent research that was based on the Count Distribution paradigm (CD) included the Fast Algorithm for Mining Global Frequent Sub-Tree (FAMDFS)which used the tree data structure (Chuanshen et al., 2010) and the Distributed Trie based Frequent Itemset Mining algorithm (DTFIM) (Ansari et al., 2008) which used the Trie data structure. However, these techniques suffered the limitations of huge memory and high network traffic (Chuanshen et al., 2010).

Count Distributed paradigm has been also adopted in the cloud computing (Grid) model. Cloud Computing is a new model that distributes the computing tasks to the pool of large number of computer resources. This enables a variety of application systems to obtain computing power and storage space (Yang et al., 2010, Wu et al., 2012, Li and Zhang, 2011, Sumithra and Paul, 2010). Cloud computing is based on Google MapReduce which is the emerging parallel programming model and Google file system called Hadoop Distributed File System (HDFS) (White, 2012, Dean and Ghemawat, 2008).

Recent extensions of CD applied the Message Passing Interface (MPI) technique to minimize the communication cost between the distributed sites due to all-to-all broadcasting (Chia-Chu and Shen, 2010, Kaosar et al., 2009).

In (Kaosar et al., 2009), each site constructed a bit vector representing its local candidate itemsets. For each candidate itemset, the corresponding bit vector element was set to 1 if it is locally frequent and to 0 otherwise. The bit vector was sent by each site to a pre-assigned site which performed an OR operation between its constructed bit vector and the received vector. The result of the OR operation was sent to the next pre-assigned site. The last site broadcasted the final bit vector to all sites. When the final bit vector was received, the support count of the candidate itemset whose corresponding bit element =1 was broadcasted to all other sites. This is due to the fact that “if an itemset is globally frequent, there exists at least one site where this itemset is locally frequent”

Unlike the Count Distribution algorithm (CD) which broadcasted the support counts of all candidate itemsets, the algorithm broadcasted only those support counts whose candidates were locally frequent in at least one site. This reduced a significant amount of communication overhead by avoiding transmitting unnecessary count values of all itemsets. However, the algorithm did not consider optimization in the number of generated itemsets (Kaosar et al., 2009).

The Message Passing Interface (MPI) approach has been adopted in (Chia-Chu and Shen, 2010) in order to reduce the size of candidate itemsets,

however, the approach required many synchronization points, consequently, more network overhead (Chia-Chu and Shen, 2010).

2.7. Summary and Conclusion

In this chapter, a general overview of the main related themes was presented. A brief overview and background of the most thesis related topics such as Knowledge Discovery, Data Mining, Association Rules and Multi Agents systems were highlighted. A range of existing work in the field of Association Rules was presented.

In summary, the main research issues discussed in the literature which have been motivation factors for the research reported in the thesis are:

1. The absence of the list of performance factors that can measure the efficiency of the Association Rules algorithms.
2. The challenges imposed due to the sites communication which is considered as the bottleneck of the distributed Data Mining.
3. The challenges imposed due to the time needed to generate the candidate itemsets and to count their supports.
4. The challenges imposed due to the large number of generated candidate itemsets.
5. The challenges imposed due to the existence of some database transactions that are not useful for the next iterations, yet, are increasing the time needed for candidate itemsets support count.

6. The limitations of the existing association rules approaches to propose a model that can combine and integrate efficient techniques to overcome the previously mentioned challenges.
7. The necessity of an efficient distributed model that can be applied to different data sets related to different application domains in addition to real world distributed databases.
8. The extraction of useful association rules in distributed databases.

In this thesis, the main focus is to explore the research issues highlighted above by investigating the existing approaches and proposing a distributed association rules model for building a knowledge base that can help decision makers.

In summary the research issues that are addressed through this research in order to build a reliable and efficient model are as follows:

1. The absence of the list of performance factors that can measure the efficiency of the Association Rules algorithms has been addressed in the research by investigating the literature, gathering the list of strong and weak factors for each of the existing association rules algorithms and building the list of performance criteria from these factors.
2. The challenges imposed due to the sites communication which is the bottleneck of distributed data mining have been tackled by proposing an agent based framework that avoids the all-to-all broadcasts between sites.

3. The challenges imposed due to the time needed to generate the candidate itemsets and to count their supports have been tackled by including the BiWise AND/OR operations which are applied to the BitTable data structure to reduce the time needed for candidate itemsets generation and support counting.
4. The challenges imposed due to the large number of generated candidate itemsets have been tackled by including three pruning techniques to reduce the number of generated candidate itemsets.
5. The challenges imposed due to the existence of some database transactions that are not useful for the next iterations have been tackled by including three transaction trimming techniques in order to dramatically reduce the number of database transactions, consequently, the time needed for candidate itemsets support counting.
6. The limitations of the existing association rules approaches to propose a model that can combine and integrate efficient techniques to overcome the previously mentioned challenges have been tackled by proposing a model that includes various techniques that can enhance and improve the distributed mining process.
7. The necessity of an efficient distributed model that can be applied to different datasets and real world databases has been addressed by applying the proposed model to various benchmark datasets related to different application domains from UCI

machine learning repository in addition to real world distributed medical databases.

8. The extraction of useful association rules in distributed databases has been addressed by applying the proposed model to real world distributed medical databases related to three hospitals in Egypt in order to generate the useful association rules and discover the hidden medical knowledge.

Chapter Three

3. The Distributed Multi-Agent Association Rules Algorithm (DMAAR)

3.1. Chapter Overview

In this chapter, we present the proposed Distributed Multi-Agent Association Rules Algorithm (DMAAR) for mining Association Rules in distributed databases. Different types of technologies are combined, namely the Association Rules as a Data Mining technique and the Multi-Agent systems to build a model that can operate on distributed databases rather than working on centralized databases only.

DMAAR is a Multi-Agent based algorithm whose autonomous and social agents provide the ability to operate cooperatively with each other and with other different external agents. This offers a generic platform and a basic infrastructure that can deal with other Data Mining techniques. Moreover, DMAAR is compliant to the Foundation for Intelligent Physical Agents communication standard.

The proposed solution architecture avoids the all-to-all broadcasting implemented by CD, DMA and FPM.

The proposed algorithm tasks are designed to be functionally distributed among the Main and the Local Agents. This helps in reducing the total time required for the whole mining process. Moreover, the order of the algorithm processes is designed to maximize the parallelism and to reduce the waiting time of the Main and the Local Agents.

Analytical evaluation showed that the algorithm complexity cost of DMAAR is better than the existing algorithms. It also showed that the message negotiation cost between agents is less than that of DMA algorithm. When compared to the count distribution algorithm, the message negotiation cost of DMAAR is better when the number of sites is greater than or equal to three.

DMAAR has been compared with the Count Distribution algorithm on UCI benchmark datasets that are related to different domains and has proved to be more efficient and more scalable.

3.2. DMAAR algorithm solution architecture

This section presents the proposed DMAAR algorithm framework and the message negotiation process between the agents.

Apriori Paradigm algorithms including CD, PDM, DMA and FPM are categorized under shared memory architectures (parallel processing). One of the main goals of the distributed memory architectures rather than the shared memory architectures is to minimize the communication cost (Kumar and Zaki,

2005). For this reason, the proposed algorithm is based on the adaptation of Apriori paradigm and its main tasks to work as distributed algorithm rather than parallel algorithm. This enhances the all-to-all broadcasting limitation of Apriori paradigm.

The framework of the proposed algorithm is based on the existence of a main site that has a global view of all distributed sites. The main site coordinates with the local sites using negotiation messages. The algorithm avoids the all-to-all broadcasts between distributed sites in the existing Association Rules algorithms. Analysis proves that the new system architecture together with the proposed algorithm distributed tasks reduce the algorithm complexity to O(n) with less number of message negotiations when compared to existing algorithms.

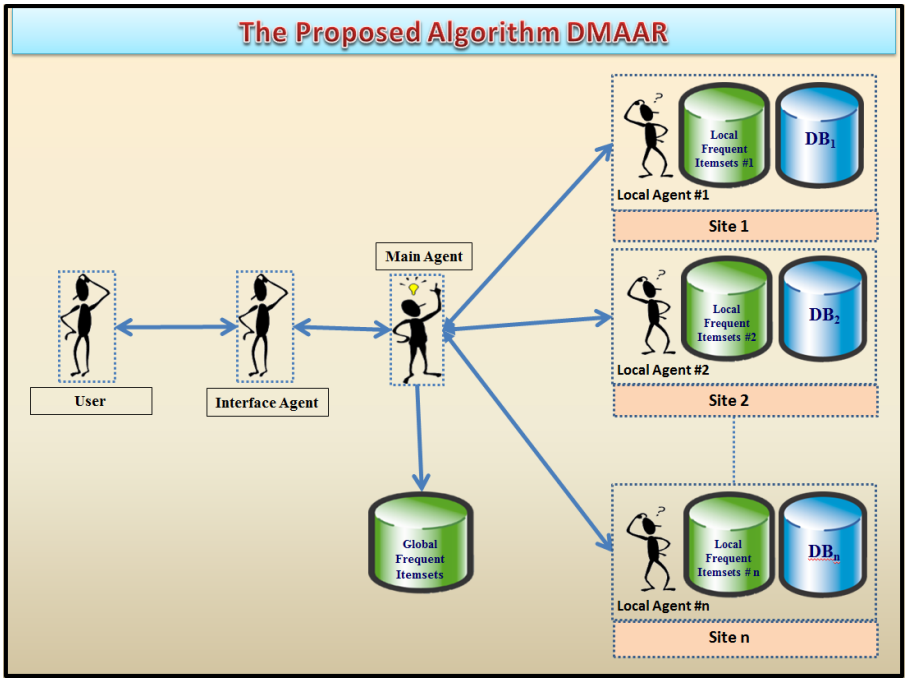

Figure 3.1 The Proposed DMAAR Algorithm Framework

Figure 3.1 shows the proposed algorithm framework. The framework consists of three types of cooperative agents that work together to achieve the required goals. The first kind of agents is the Interface Agent which accepts the user required support. The Interface Agent sends a message containing the value of the support to the Main Agent, which in turn sends it to the third type of agents namely the Local Agent. Local agents are responsible for counting the support counts of candidate itemsets and sending them to the Main Agent. The Main Agent sums the support counts received from all Local Agents then sends the results back to all Local Agents. The Main Agent is also responsible for generating new candidate itemsets for the next iteration. Candidate itemsets are sent to all Local Agents. This process is done until no more candidate itemsets are generated. The proposed framework proves to achieve better complexity and message communication costs.

The algorithm implemented by the Local Agents at the first iteration is presented in Algorithm 3.1.

Algorithm 3.1 The Proposed Algorithm at Local Agents (at the first iteration)

```
1:  begin
2:      Calculate local support counts for items;
3:      Send local support counts only to the main site;
4:      for each item I_i do
5:          If total support (I_i) >= s^i then //s^i is the minsup at site i
6:              Add I_i to set of local large itemsets L^i_1;
7:          end;
8:      end;
12:     Save local large 1-itemsets L^i_1 in the local knowledge database;
13:     Receive global 2-candidate itemsets C_2 from main site;
14: end;
```

The algorithm implemented by the Local Agents at k-iteration is presented in Algorithm 3.2.

Algorithm 3.2 The Proposed Algorithm at Local Agents (at k iteration)

```
1:  Begin
2:      Calculate support count for global candidate itemsets C_k;
3:      Sends local support counts only of C^i_k to the main site;
4:      for each itemset C^i_k in the global candidate itemsets C_k do
5:          If total support (C^i_k) >= s^i then
6:              Add C^i_k to set of large itemsets L^i_k;
7:          end;
8:      end;
9:      Save L^i_{k+1} in local knowledge database;
10:     Receive global candidate itemsets C^i_{k+1} from main site ;
11:     If global k+1 candidate itemsets C^i_{k+1} exists then
12:             go to step 1;
13:     end;
14: end;
```

The algorithm implemented by the Main Agent at k-iteration is presented in Algorithm 3.3.

Algorithm 3.3 The Proposed Algorithm at main site (at k iteration)

```
1:  Begin
2:      Sends the required support and confidence to Local Agents;
3:      Receives k-itemsets support counts from all Local Agents;
4:      Calculates total counts for k-itemsets;
5:      for each itemset X in the k-itemsets do
6:          If X.supp>= MinimumSupport then
7:              Mark X as Frequent itemset;
8:          end;
9:      end;
10:     Generates candidate itemsets C_{k+1} from L_k;
11:     If k+1 candidate itemsets C_{k+1} exists then
12:         Sends global candidate itemsets C_{k+1} to Local Agents;
13:         Saves Frequent itemsets in global knowledge database L_k;
14:         go to step 3;
15:     end;
16: end;
```

The communication method between the Interface Agent, the Main Agent and the Local Agents complies with the FIPA standards. An example for a FIPA message sent from the Main Agent to the Local Agents with an "Inform performative" is presented as follows:

(**Inform**

:**sender** (agent-identifier :name main_agent)

:**receiver** (set (agent-identifier :name local_agent))

:**content** "frequent itemsets at k=1 are successfully generated")

Detailed description for the message negotiation process of the proposed DMAAR algorithm is described in section 3.3.

3.3. FIPA negotiation messages between agents

The FIPA negotiation messages between agents are described as follows:

1. The Interface Agent accepts the required support from the user.

2. The Interface Agent sends a "propose performative" FIPA message to the Main Agent:

(**Propose**

:**sender** (agent-identifier :name Interface Agent)

:**receiver** (set (agent-identifier :name Main Agent))

:**content** "Start Mining, sending the support")

3. The Interface Agent sends the support to the Main Agent.

4. The Main Agent sends a "propose performative" FIPA message to all Local Agents:

(**Propose**

:**sender** (agent-identifier :name Main Agent)

:**receiver** (set (agent-identifier :name Local Agent))

:**content** "Start mining with support = minsupp"

:**reply-with** start mining proposal)

5. The Local Agents reply with an "agree performative" to the Main Agent as follows:

(**Agree**

:**sender** (agent-identifier :name Local Agent)

:**receiver** (set (agent-identifier :name Main Agent))

:**content** "proposal approved and mining started at k=1"

:**in-reply-to** start mining proposal)

6. Each Local Agent starts counting the local supports for all 1- candidate itemsets in its local database according to its local number of records.

7. The Local Agent replies with "inform performative" to the Main Agent as follows:

(**Inform**

:**sender** (agent-identifier :name Local Agent)

:**receiver** (set (agent-identifier :name Main Agent))

:**content** "finished counting candidate 1-itemsets")

8. The Main Agent compares the summation of the local supports received from all agents for the 1-candidate itemsets with the minimum support supplied by the user.

9. The Main Agent finds the 1-large itemsets and save it in the database in the list of frequent itemsets.

10. The Main Agent sends an "Inform performative" FIPA message to all Local Agents:

(**Inform**

:**sender** (agent-identifier :name Main Agent)

:**receiver** (set (agent-identifier :name Local Agent))

:**content** "frequent itemsets at k=1 are successfully generated")

11. The Main Agent generates the k-candidate itemsets from the list of frequent itemsets.

12. The Main Agent sends a "Request performative" FIPA message to all Local Agents:

(**Request**

:**sender** (agent-identifier :name Main Agent)

:**receiver** (set (agent-identifier :name Local Agent))

:**content** "candidates are generated at iteration =*k*"

:**reply**-with iteration *k*)

13. The Main Agent sends the generated k-candidate itemsets to all Local Agents.

14. Each Local Agent receives the generated k-candidate itemsets and counts their local support counts in the local databases

15. The Local Agents send an "Inform performative" message to Main Agent:

(**Inform**

:**sender** (agent-identifier :name Local Agent)

:**receiver** (set (agent-identifier :name Main Agent))

:**content** "Finished counting candidate itemsets for iteration =*k*"

:**in-reply-to** iteration *k*)

16. The Main Agent considers any k-candidate itemset as frequent if the summation of all local supports received from all Local Agents for this itemset is greater than the minimum global support.

17. Frequent itemsets are saved in the list of k-frequent itemsets while non-frequent itemsets are not considered for the next iteration.

18. Steps (11) to (17) are iterative and finish when there are no more k+1 candidate itemsets.

18. The Main Agent sends “Inform performative” message to all Local Agents, then sends the frequent itemsets to the Interface Agent for representation.

(**Inform**

:**sender** (agent-identifier :name Main Agent)

:**receiver** (set (agent-identifier :name Local Agent))

:**content** "Finished mining of frequent itemsets")

3.4. The Sequence Diagram of DMAAR algorithm

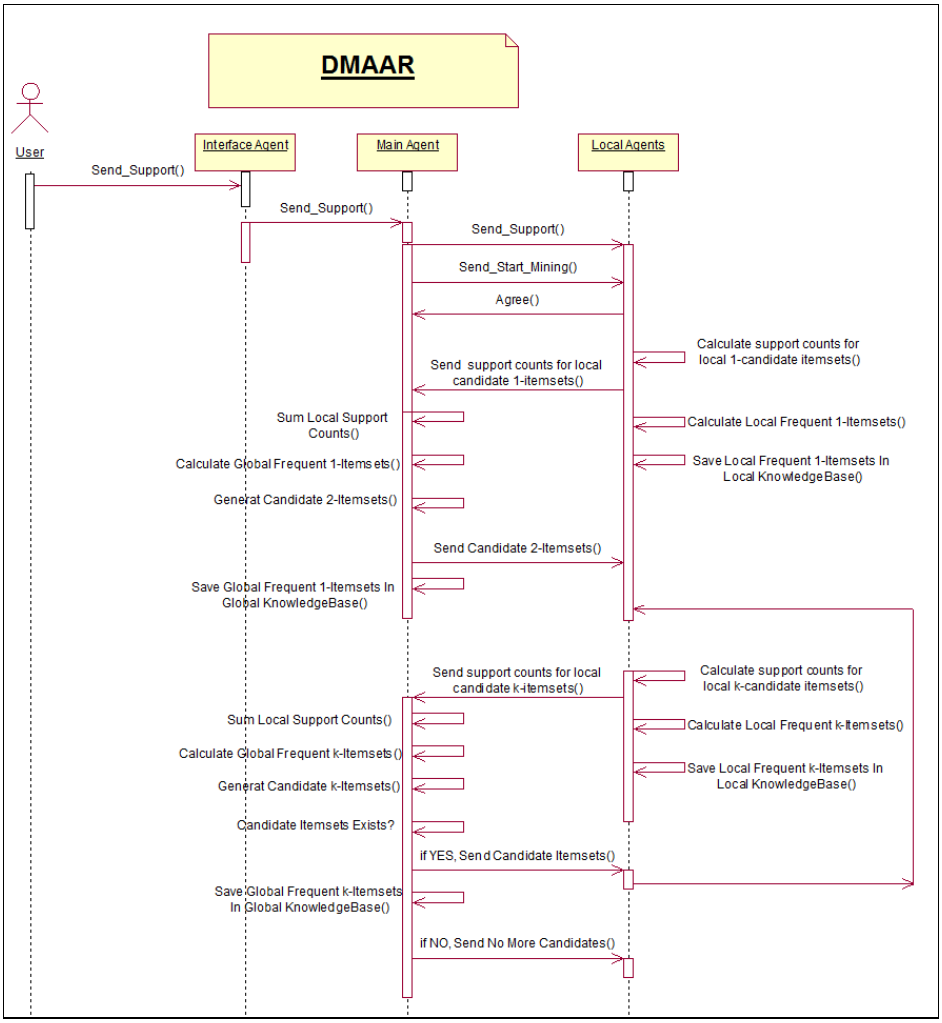

Figure 3.2 The Sequence Diagram of the proposed DMAAR algorithm

DMAAR Sequence diagram is presented in Figure 3.2 in order to show the interactions between agents in addition to the sequence of operations during the execution of the algorithm.

The existing distributed Association Rules algorithms exhibit an iterative paradigm. In this paradigm, each site applies the mining tasks to its local database. These tasks include the candidate itemsets generation, the generated candidate itemsets support counting, the calculation of the frequent itemsets, etc. When the mining process is finished, the extracted information is synchronized with all other sites before moving to the next iteration. However, the different factors affecting each site such as the processing power and speed, the available computer memory and the size of data in its local databases can force some sites to wait for others. This increases the waiting time for the whole mining process.

In the proposed algorithm, the algorithm mining tasks are distributed across the Main and the Local Agents. These mining tasks are executed in parallel in order to minimize the total mining time. For instance, as shown in Figure 3.2, while the Local Agents are calculating the local frequent k-itemsets and saving them into the local knowledge base, the Main Agent is computing the sum of the local support counts, calculating the global frequent k-itemsets and generating the candidate k-itemsets.

3.5. The Activity Diagram of DMAAR algorithm

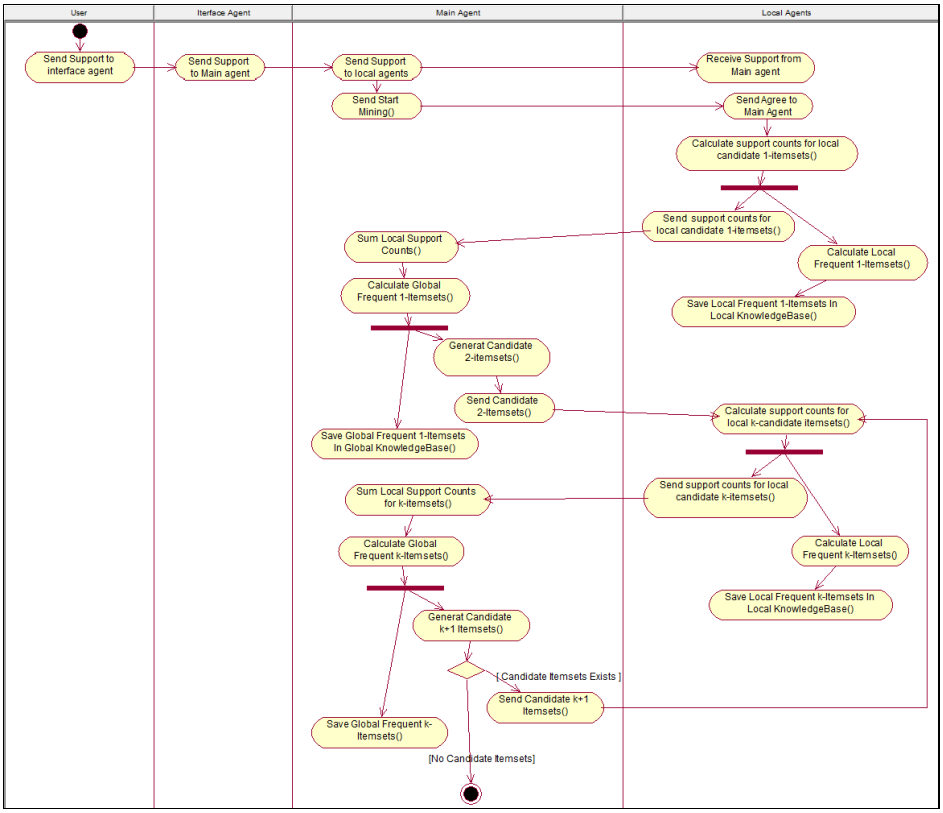

Figure 3.3 The Activity Diagram of the proposed DMAAR algorithm

DMAAR activity diagram is presented in Figure 3.3 to show the order of the parallel activities for the proposed algorithm. The diagram shows how the proposed algorithm has been designed to minimize the waiting time for the mining process.

In order to minimize the waiting time, the proposed algorithm breaks the mining process into smaller subtasks. The order of these subtasks within the same agent is modified to give higher priority to those subtasks that other agents are waiting for. This helps to minimize the waiting time for the algorithm.

For instance, as shown in Figure 3.3, three tasks depend on the calculation of the support count for the local candidate k-itemsets at the local agents. These tasks are calculating the local frequent k-itemsets, sending support counts of the local candidate k-itemsets and saving the local frequent k-itemsets.

However, calculating the local frequent k-itemsets and saving the local frequent k-itemsets before sending support counts of local candidate k-itemsets task blocks the main agent and forces him to wait until these tasks are finished. Consequently, the total waiting time increases.

For this reason, the proposed algorithm sends the support counts of the local candidate k-itemsets first to the Main agent to enable the agent to start executing its mining tasks in parallel with the local agent. This order helps in reducing the total waiting time of the mining process.

3.6. Analytical Evaluation

The algorithm complexity cost and the message negotiation cost are considered as very important performance issues for the distributed algorithms.

3.6.1. Algorithm Complexity Cost of DMAAR

The algorithm Complexity (sometimes referred to as Big O Notation) indicates a rough estimate for the number of steps performed by the algorithm in terms of the size of the input data and the number of algorithm loops (Kudryavtsev and Andreev, 2010, Goldreich, 2008).

Due to the all-to-all broadcasting of candidate itemsets support counts from every site to all other sites, ***Count Distribution algorithm (CD)*** requires a complexity of $O(n^2)$ (Cheung et al., 2002). This can be calculated as follows:

Algorithm Complexity Cost of CD = $O(|C_k| \cdot n \cdot (n-1))$

$= O(|C_k| \cdot (n^2-n))$

$= O(|C_k| \cdot n^2)$

$= O(n^2)$

Where n is the number of sites and C_k is the candidate itemsets support counts at k iteration

Due to the all-to-all broadcasting of candidate itemsets from every site to all other sites, ***Distributed Mining of Association Rules algorithm (DMA)*** requires a complexity of $O(n^2)$ (Cheung et al., 2002). This can be calculated as follows:

Algorithm Complexity Cost of DMA= $O(|C_k| \cdot n \cdot (n-1)) =$

$= O(|C_k| \cdot (n^2-n))$

$= O(|C_k| \cdot n^2)$

$= O(n^2)$

However, due to generation of less number of candidate itemsets than CD, DMA claims that the complexity is less than CD, thus less than $O(n^2)$ (Cheung et al., 2002), thus:

Algorithm Complexity Cost of DMA<= $O(n^2)$

The complexity cost of the ***Proposed DMAAR Algorithm*** is calculated as the cost of sending the generated candidate itemsets from the main site to all local sites in addition to the cost of returning back their support counts to the main site. This is calculated as follows:

Cost of sending the generated candidate itemsets to local sites is:

$O(|C_k| \cdot n) = O(n)$

Cost of sending local support counts to main site is:

$O(|C_k| \cdot n) = O(n)$

Total Algorithm Complexity Cost of DMAAR is:

$O(|C_k| \cdot n) + O(|C_k| \cdot n)) =$

$= O(n) + O(n) = O(n)$

From the above analysis, it is clear that the algorithm complexity cost of the proposed DMAAR algorithm is less than that of CD and DMA algorithms.

3.6.2. **Message Negotiation Cost of DMAAR**

The total message exchange cost of the ***Count Distribution algorithm*** (CD) is calculated as the cost of broadcasting the itemsets support counts to all other sites

Thus, the Total Message Exchange Cost of CD is:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|SC_k^i\right|.(N-1) \qquad (3.1)$$

Where N is the number of sites, K is the number of iterations and SC_k^i is the support counts of the candidate itemsets at iteration k for site i.

Assume that:

$$SC_{\max} = \max\left(\left|SC_k^i\right|\right)$$

Thus, Total Message Exchange Cost of CD is:

$$N.(N-1).K.SC_{\max} \qquad (3.2)$$

Similarly, the total message exchange cost of the ***Distributed Mining of Association Rules algorithm (DMA)*** is calculated as the total costs of message negotiations between sites. This is calculated as:

The cost of sending local frequent itemsets for counting the support

+ Cost of replying with the local frequent itemsets support counts

+ Cost of sending global frequent itemsets

+ Cost of sending the global frequent itemsets support count.

Thus, Total Message Exchange Cost of DMA is:

$$\begin{aligned} &\sum_{i=1}^{N}\sum_{k=1}^{K}\left|C_k^i\right|.(N-1)+ \\ &\sum_{i=1}^{N}\sum_{k=1}^{K}\left|SC_k^i\right|.(N-1)+ \\ &\sum_{i=1}^{N}\sum_{k=1}^{K}\left|C_k^i\right|.(N-1)+ \\ &\sum_{i=1}^{N}\sum_{k=1}^{K}\left|SC_k^i\right|.(N-1) \end{aligned} \qquad (3.3)$$

Assume that:

$$C_{\max} = \max(\left|C_k^i\right|)$$

$$SC_{\max} = \max(\left|SC_k^i\right|)$$

Total Message Exchange Cost of DMA is:

$$2.N.(N-1).K.SC_{\max} + 2.N.(N-1).K.C_{\max} \quad (3.4)$$

The message Exchange Cost of the ***Proposed DMAAR Algorithm*** is the cost of sending the generated candidate itemsets from the main site to all local sites in addition to the cost of returning back their support counts to the main site as follows:

Message exchange cost of sending local support counts to main site is:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|SC_k^i\right|$$

Message exchange cost of sending the generated candidate itemsets from the main site to the local sites is:

$$\sum_{i=1}^{N}\sum_{k=2}^{K}\left|C_k^i\right|$$

Total Message Exchange Cost of DMAAR is:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|SC_k^i\right| + \sum_{i=1}^{N}\sum_{k=2}^{K}\left|C_k^i\right| \quad (3.5)$$

For the maximum size of the candidate itemsets, assume that:

$$C_{\max} = \max(\left|C_k^i\right|)$$

$$SC_{\max} = \max(\left|SC_k^i\right|)$$

Total Message Exchange Cost of DMAAR is:

$$N.K.C_{\max} + N.(K-1).C_{\max} \qquad (3.6)$$

From the above analysis, it is clear that the message negotiation cost of the proposed DMAAR algorithm is less than that of DMA algorithm. Moreover, it is less than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

In terms of scalability, the message negotiation cost of CD and DMA is directly proportion to the square of the number of sites unlike DMAAR which is directly proportion to the number of sites. This makes DMAAR more scalable than CD and DMA when the number of sites increases.

3.7. Performance Evaluation

The experiments include the implementation of two algorithms against five different benchmark datasets at five different supports. Algorithms implemented are the Count Distribution algorithm CD and the proposed DMAAR algorithm. The five benchmark datasets from UCI machine learning repository are related to different application domains and are commonly used in the literature to test and compare the Association Rules algorithms. Datasets are described in Table 3.1.

Table 3.1 UCI Benchmark Dataset

Dataset	# of instances	# of attributes	year
Abalone	4177	8	1995
Car Evaluation	1728	6	1997
Mammographic Mass	961	6	2007
Blood Transfusion Service Center	748	5	2008
Iris	150	4	1988

The results obtained are illustrated in Figure 3.4 to Figure 3.8.

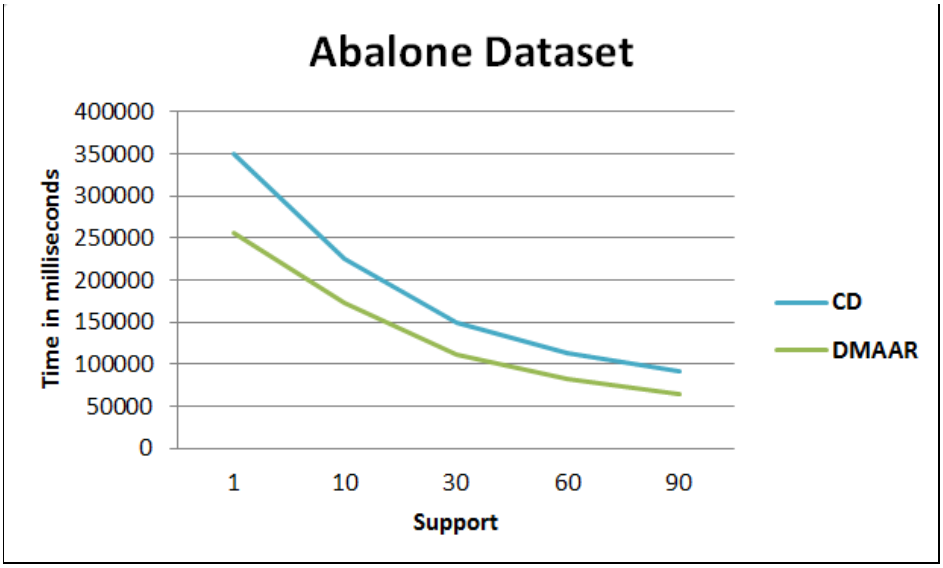

Figure 3.4 Comparison of DMAAR against CD on Abalone Dataset

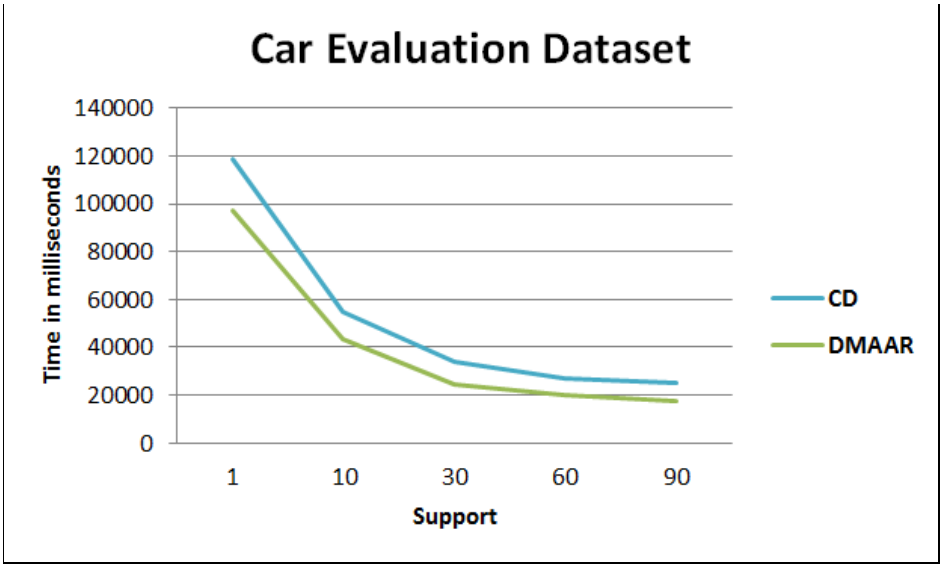

Figure 3.5 Comparison of DMAAR against CD on Car Evaluation Dataset

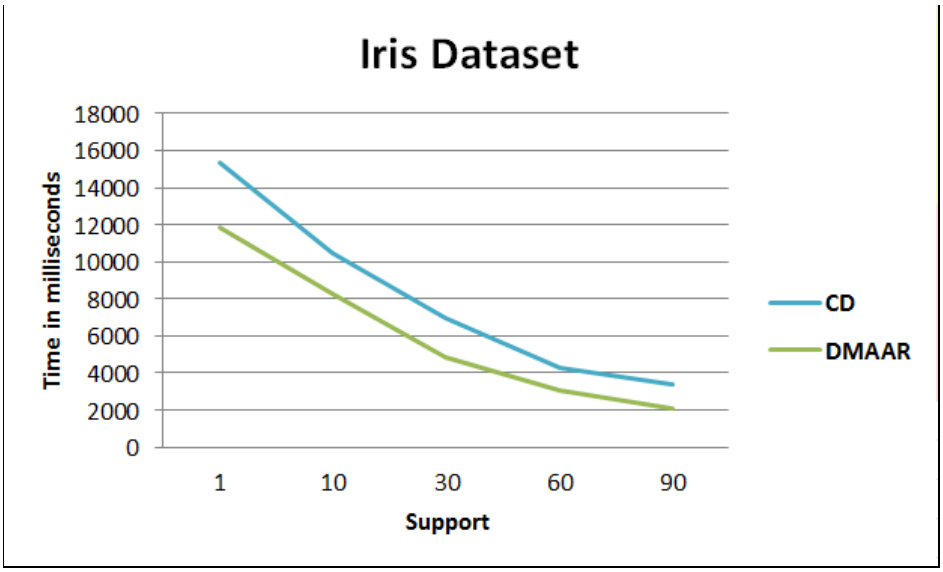

Figure 3.6 Comparison of DMAAR against CD on Iris Dataset

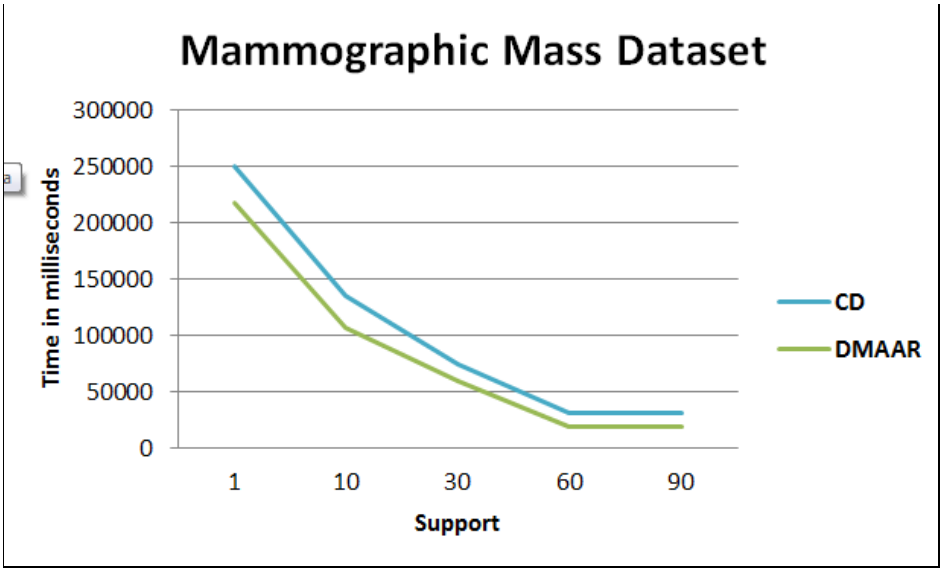

Figure 3.7 Comparison of DMAAR against CD on Mammographic Dataset

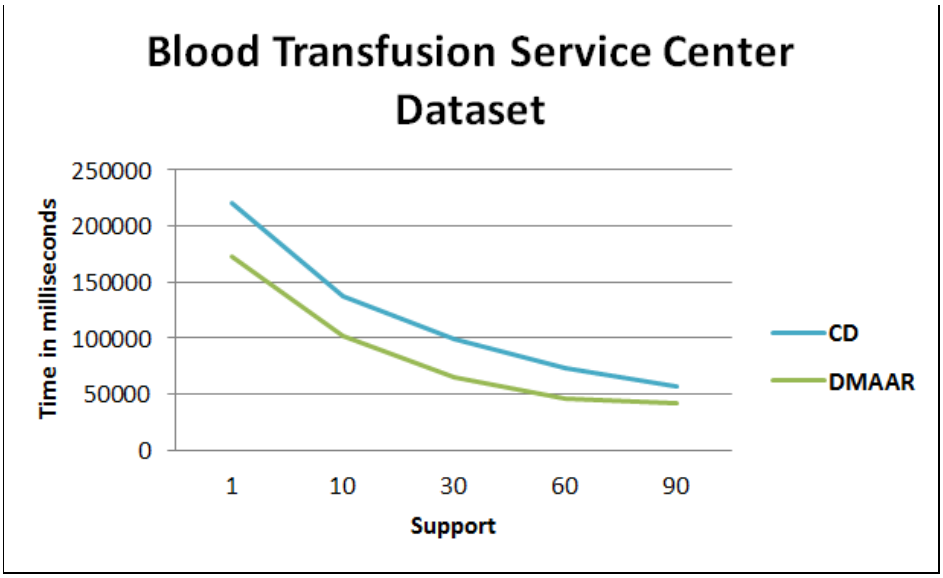

Figure 3.8 Comparison of DMAAR against CD on Blood Transfusion Dataset

The comparative study for the performance evaluation of DMAAR with Count Distribution shows the following:

1- Figure 3.4 to Figure 3.8 show that DMAAR outperforms Count Distribution algorithm. This is due to the proper distribution of tasks across the agents which maximized the parallelism of the mining tasks, the avoidance of the all-to-all broadcasts which reduced the message negotiation between agents and the order of the processes which reduced the waiting time of the Main and the Local Agents, consequently, the total time needed for the execution of the algorithm.

2- Figure 3.4 to Figure 3.8 also show that the total execution time for both algorithms increases when the support value decreases. This is

because when the support value decreases, the number of generated candidate itemsets increases, thus, increasing the time needed to generate these candidates and to count their supports. These limitations are addressed in the following chapters.

3.8. Summary and Conclusion

Research presented in this Chapter tackled some of the imposed challenges when applying Data Mining techniques on distributed databases. The objectives of the research in this chapter were to avoid the all-to-all broadcasting between sites, minimize the algorithm complexity, minimize the waiting time, maximize the tasks parallelism and minimize the message negotiation cost.

In this chapter, the Distributed Multi-Agent Association Rules algorithm (DMAAR) for mining Association Rules in distributed databases was presented. The proposed algorithm combined different types of technologies, namely the Association Rules as a Data Mining technique and the Multi-Agent systems to build a model that can be applied to distributed databases rather than to centralized databases only.

In order to overcome the challenges of applying Data Mining techniques to distributed databases, DMAAR employed a multi-agent based framework and a standard communication mechanism.

Three kinds of agents have been proposed in the algorithm, the Interface Agent, the Main Agent and the Local Agents. DMAAR avoided the all-to-all

broadcasting between distributed sites by including a Main Agent as the main controller with a global view of all other agents.

In summary, the proposed DMAAR algorithm presented the following contributions:

1- The Multi-Agent system framework of DMAAR enabled the agents to operate cooperatively with each other and with other different external agents. This offered a generic platform and a basic infrastructure that can deal with other Data Mining techniques.
2- Unlike CD, PDM, DMA and FPM which generate global frequent itemsets only, the generation of local and global frequent itemsets by DMAAR provided useful knowledge to local and global decision makers.
3- Message compliance with the global communication standard, the Foundation for Intelligent Physical Agents (FIPA), enabled the agent cooperation with other standard agents.
4- The proposed solution architecture avoided the all-to-all broadcasting implemented by CD, DMA and FPM algorithms by including a Main Agent which has a global view of all other agents.
5- The sequence and the activity diagrams presented in Figure 3.2 and Figure 3.3 respectively showed that the proposed algorithm tasks are functionally distributed between the Main and the Local Agents. This helped in reducing the total time required for the whole mining process.
6- The sequence and the activity diagrams presented in Figure 3.2 Figure 3.3 respectively also showed that the order of the algorithm

processes maximized the parallelism and reduced the waiting time for the Main and the Local Agents

7- Analytical calculations presented in Section 3.6.1 showed that the proposed Multi-Agent based algorithm reduced the algorithm complexity from $O(n^2)$ to $O(n)$.

8- Analytical calculations showed also that the message negotiation cost of DMAAR is less than that of DMA algorithm. Moreover, it is less than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

9- In terms of scalability, the message negotiation cost presented in Section 3.6.2 showed that the costs of CD and DMA algorithms are directly proportion to the square of the number of sites unlike that of DMAAR which is directly proportion to the number of sites. This showed that DMAAR is more scalable than CD and DMA when the number of sites increases.

10- The performance evaluation for the proposed DMAAR algorithm was compared with existing algorithms on five different benchmark datasets related to different domains. Results presented in Figure 3.4 to Figure 3.8 showed that DMAAR achieved better performance and execution time.

However, the proposed DMAAR algorithm suffered the following limitations:

1- Mining Association Rules is an iterative process. In each iteration, the database is scanned in order to perform support counting and pattern matching for candidate itemsets. For huge amount of data,

the database transactions become too large to be stored in memory. One of the limitations of the proposed DMAAR algorithm is that it did not address the memory problems nor the techniques required to fit the database transactions in the local memory of the main and the local sites.

2- The huge number of database transactions increases the time needed for the mining process. However, DMAAR did not investigate the techniques that can trim the database transactions. Reducing the number of transactions can help in reducing the number of iterations and the time needed to finish the whole mining process.

3- Generating candidate itemsets is considered a time consuming process. However, DMAAR did not address the techniques that can reduce the candidate generation process.

4- Counting the supports of the generated candidate itemsets requires scanning the database and counting the number of occurrences of these itemsets. This is a lengthy process that needs to be addressed by the proposed algorithm.

Due to the aforementioned limitations of DMAAR, the BitTable Multi-Agent Association Rules Algorithm (BMAAR) is proposed in the next chapter to address and improve these limitations.

Chapter Four

4. The BitTable Multi-Agent Association Rules Algorithm (BMAAR)

4.1. Chapter Overview

In the previous chapter, DMAAR presented many contributions which include the avoidance of all-to-all broadcasts, the proper functional distribution of the mining processes across the Main and the Local Agents which helped in maximizing the parallelism and minimizing the waiting time, the reduction of the complexity and message negotiation costs, the compliance with the Foundation for Intelligent Physical Agents communication standard. However, the algorithm suffered the following limitations.

First, DMAAR did not address techniques for compressing the database to fit into the memory of the main and the local sites. Second, DMAAR did not include approaches to improve the time needed for candidate itemsets generations and support counting. Finally, DMAAR did not include techniques to trim the number of database transactions in order to decrease the number of mining iterations and consequently the total time needed for the mining process.

In this chapter, the proposed BitTable Multi-Agent Association rules algorithm (BMAAR) which is an extension for DMAAR is presented.

BMAAR algorithm uses the efficient BitTable data structure proposed in(Dong and Han, 2007). The BitTable data structure was proved to have better performance and less memory than hash trees data structure used by CD, PDM, DMA and FPM. The proposed algorithm also includes two Bitwise AND/OR operations proposed in the BitTableFI algorithm (BT) (Dong and Han, 2007) for quick candidate itemsets generation and quick support counting. The BitTable data structure and the Bitwise AND/OR operations are adopted in (Thi et al., 2010, Nawapornanan and Boonjing, 2011, Chen and Xiao, 2010, Song et al., 2008, Yang and Yang, 2010).

Moreover, BMAAR includes three transaction trimming techniques to reduce the number of database transactions. These techniques are the Direct Hashing and Pruning (DHP) proposed in (Park et al., 1997), the proposed Grouping Identical Transactions technique and the Non Frequent 1-Itemset Removal Technique proposed in (Park et al., 1997). DHP and the Non Frequent 1-Itemset Removal techniques are adopted in (Najadat et al., 2011, Özel and Güvenir, 2001).

Experiments were conducted to compare the performance of the proposed BMAAR algorithm with CD, BT and DMAAR. Other experiments were conducted to evaluate the effect of including the transaction trimming techniques on the Data Mining process.

4.2. The proposed BMAAR algorithm

As the communication between agents plays a very important role in the proposed algorithm, the role of each kind of agents is explained in details.

4.2.1. The Local agents

The message negotiation between agents is initiated when the Main Agent accepts the support and confidence from the user and sends them to the Local Agents at all sites. The proposed algorithm for the Local Agents at the first iteration is explained in Algorithm 4.1. Each Local Agent converts its local database into bit vectors as in step 2 in Algorithm 4.1. The Database conversion process is explained in details in algorithm 4.4. Conversion of the database into bit vectors helps to compress the database at each local site to fit better into its local memory. It also helps to speed up the candidate itemsets generation and support counting processes. After the database conversion, the Local agents use the proposed transaction trimming technique named “The Grouping Identical Transactions” to minimize the number of database transactions in their local database (see section 4.10).

Support counts for the itemsets are calculated by Local Agents using BitWise AND/OR operations. BitWise operations has proven to be more efficient when used with BitTable data structure (Dong and Han, 2007). This is explained in details in Algorithms 4.5, 4.6 for k=1 and for k iteration respectively. Only the support counts for the itemsets are sent to the Main Agent. Itemsets are not sent to the Main Agent. This reduces the message negotiation cost of the proposed algorithm. Itemsets whose support counts are greater than the local minimum support are saved in the local frequent itemsets knowledge base.

At phase one only, non-frequent 1-itemsets are removed from the database as proposed in (Park et al., 1997). Then, identical transactions are grouped together to decrease number of transactions. "Removing non-frequent 1-itemsets" and "Grouping Identical Transactions" techniques and how they are applied to the BitTable data structure are explained in section 4.10.

Local agents sleep and wait until the Main Agent sends the global candidate itemsets for the next phase. When Local Agents receive the global candidate itemsets, they apply the Transaction Trimming technique presented in DHP. The technique and how it is applied to the BitTable data structure is explained in (Algorithm 4.8). The algorithm implemented by the Local Agents at the first iteration is presented in Algorithm 4.1.

Algorithm 4.1 The Proposed Algorithm at the Local Agents (at the first iteration)

```
1:  begin
2:      Convert_Database_into_bit_vectors () (Algorithm 4.4);
3:      Group Identical Transactions technique;
4:      // Perform BitWise AND operations;
5:      Calculate support count for items bit vector (Algorithm 4.5);
6:      Send local support counts only to the main site;
7:      for each item bit vector Ib_i in the items bit table do
8:          If total support (Ib_i) >= s^i then
9:              Add Ib_i to set of local large itemsets Lb^i_1;
10:         end;
11:     end;
12:     Save local large 1-itemsets Lb^i_1 in local knowledge database;
13:     Remove non-Frequent 1-itemsets from DB^i;
14:     Group Identical Transactions;
15:     Receive global 2-candidate itemsets C_2 from main site;
16: end;
```

The algorithm implemented by the Local Agents at k-iteration is presented in Algorithm 4.2.

Algorithm 4.2 The Proposed Algorithm at the Local Agents (at k iteration)

```
1:  Begin
2:      // Perform BitWise AND operation operations;
3:      Calculate support count for global candidate itemsets Ck
        (Algorithm 4.6);
4:      Sends local support counts only of C^i_k to the main site;
5:      for each itemset C^i_k in the global candidate itemsets Ck do
6:          If total support (C^i_k) >= s^i then
7:              Add C^i_k to set of large itemsets Lb^i_k;
8:          end;
9:      end;
10:     Save Lb^i_{k+1} in local knowledge database;
11:     Receive global candidate itemsets C^i_{k+1} from main site ;
12:     Apply PDM Transaction Trimming Technique (Algorithm 4.8);
13:     If global k+1 candidate itemsets C^i_{k+1} exists then
14:             go to step 3;
15:     end;
16: end;
```

4.2.2. **The Main agent**

The Main agent is the main coordinator of the Local Agents. It sends the user support and confidence to all Local Agents. When the Main Agent receives the local k-itemsets support counts from all Local Agents, it calculates the global counts. If the itemset global count is greater than the global minimum support, the itemset is saved in the global frequent itemsets knowledge base. Otherwise, it is discarded for the next phase.

At k iteration, candidate itemsets are generated from the previous pass (Algorithm 4.7). Candidate itemsets are sent to all Local Agents. Main agent sleeps and waits for the itemsets support counts from Local Agents. The algorithm implemented by the Main Agent at k-iteration is presented in Algorithm 4.3.

Algorithm 4.3 The Proposed Algorithm at the Main Agent (at k iteration)

```
1:  Begin
2:      Sends the required support and confidence to Local Agents;
3:      Receives k-itemsets support counts from all Local Agents;
4:      Calculates total counts for k-itemsets;
5:      for each itemset X in the k-itemsets do
6:          If X.supp>= MinimumSupport then
7:              Mark X as frequent itemsets;
8:          end;
9:      end;
10:     // Perform BitWise AND/OR operations;
11:     Generates candidate itemsets C_{k+1} from Lb_k (Algorithm 4.7);
12:     If k+1 candidate itemsets C_{k+1} exists then
13:         Sends global candidate itemsets C_{k+1} to Local Agents;
            Saves Frequent itemsets in global knowledge database L_k;
14:         go to step 3;
15:     end;
16: end;
```

4.3. Message Negotiation between the Main and the Local Agents

The message negotiation between the Main and the Local Agents is presented in Table 4.1.

Table 4.1 Message Negotiation between the Main and the Local Agents for BMAAR algorithm

Main Agent	Local Agents
	1. Convert DB into Bit Table 2. Apply Group Identical Transactions technique. 3. Calculate support counts for local candidate 1-itemsets using BitWise AND/OR operations.
1. Receives and sums all support counts for local candidate 1-temsets from Local Agents.	4. Send local support counts only to main site ←
2. Find global Frequent 1-itemsets	5. Find local frequent1-itemsets greater than minsup.
3. Generates the set of candidate 2-itemsets using BitWise AND/OR operations.	6. Save frequent1-itemsets in local knowledge base.
	7. Eliminate non-frequent 1-itemsets. 8. Apply Group Identical Transactions technique.
4. Send candidate 2-itemsets to all local sites →	9. Receive candidate 2-itemsets.
5. Save global Frequent 1-itemsets in global knowledge base	10. At iteration k do the following:

	11. Calculate support count for local candidate k-itemsets using BitWise AND/OR operations.
6. Receives and sums all support counts for local candidate k-itemsets from all local sites. ←	12. Sends support count only to main site
7. Find global Frequent k-itemsets	13. Find local Frequent k-itemsets greater than minsup
8. Generate the set of candidate k+1itemsets using BitWise AND/OR operations.	14. Save frequent k-itemsets in local knowledge base.
	15. Apply PDM transaction trimming technique.
9. Send generated candidate k+1itemsets to all local sites →	16. Receive candidate k+1itemsets. If candidate itemsets exist, then go to step 3
10. Save global Frequent k-itemsets in global knowledge base	

4.4. The Sequence Diagram of BMAAR algorithm

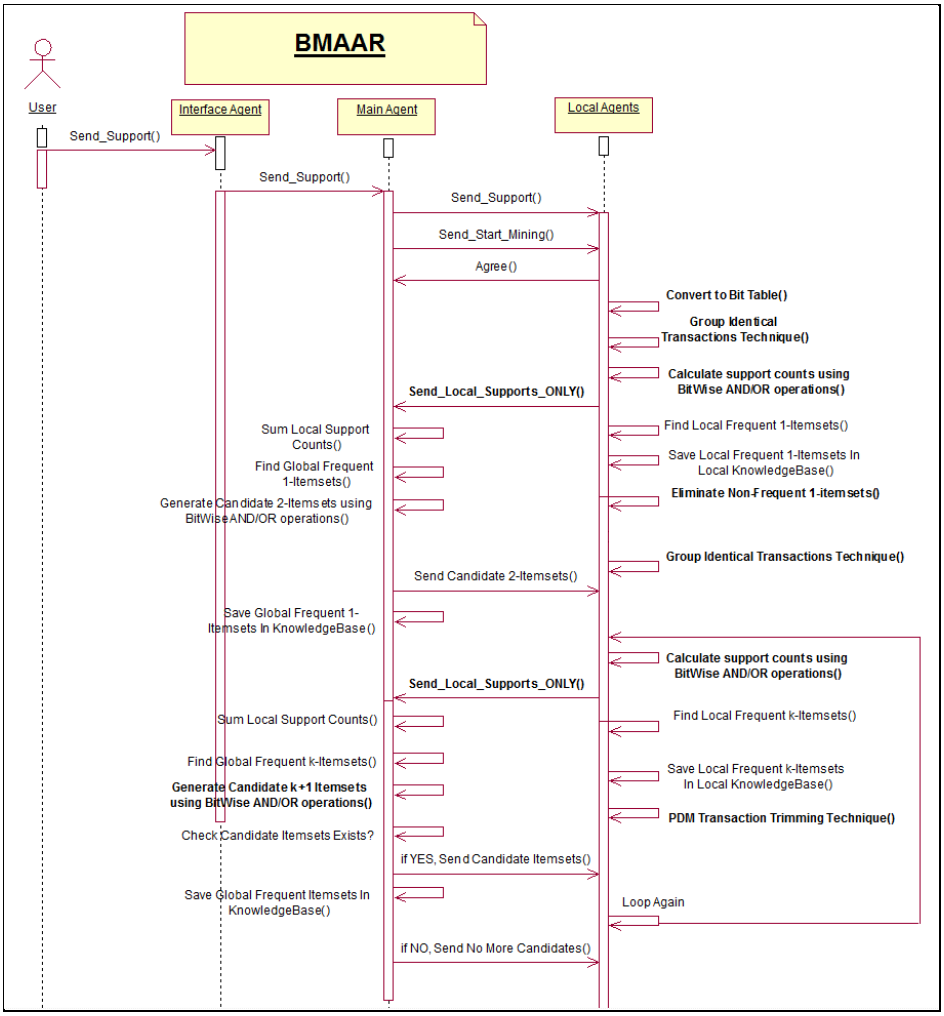

Figure 4.1 The Sequence Diagram of the proposed BMAAR algorithm

According to the discussion in Section 3.4, BMAAR Sequence diagram presented in Figure 4.1 shows how the distribution of the mining tasks maximized the tasks parallelism. Moreover, the Sequence diagram highlights the tasks that enhanced the performance of BMAAR.

4.5. The Activity Diagram of BMAAR algorithm

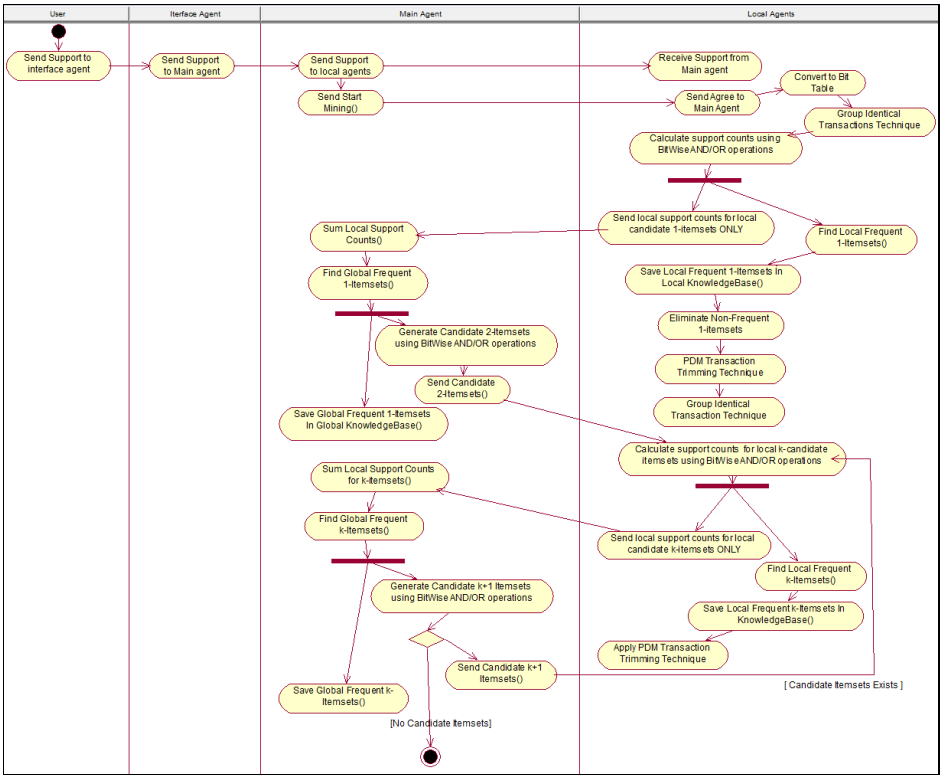

Figure 4.2 The Activity Diagram of the proposed BMAAR algorithm

According to the discussion in Section 3.5, BMAAR Activity diagram presented in Figure 4.2 shows how the order of the mining tasks minimized the waiting time of the Main and the Local agents.

4.6. Database Conversion Algorithm

The proposed BMAAR algorithm converts the local databases into the BitTable format rather than the Apriori format. Unlike the BitTableFI algorithm (Dong and Han, 2007) which applies the database conversion after the second iteration, the proposed algorithm converts the database before the first iteration. The proposed algorithm is implemented in distributed databases unlike the BitTableFI algorithm which was implemented in centralized database.

In the database conversion algorithm, every item is checked for existence in the transaction. If the item exists, the item is replaced by 1 otherwise it is replaced by 0. For instance, assume the items ABCDE and the transaction ACD, the BitVector representation of the transaction is 10110. The conversion of the database into the BitTable format is described in details in Algorithm 4.4.

Let $I = \{i_1, i_2, \ldots i_n\}$ be a finite set of items and D is a dataset containing N transactions, where each transaction $t \in D$ is a list of distinct items $t = \{i_1, i_2, \ldots i_j\}$ where $i_j \in I$ $(1 \leq j \leq |t|)$, and each transaction is identified by a distinct Transaction Identifier (TID).

Algorithm 4.4 Convert_Database_into_bit_vectors (Local Agents)

```
1:  begin
2:      Let {I} = the finite set of all items {i1,i2, … In};
3:    for every item ij in the set of items {I} where  (1 ≤  j ≤ n) do
4:        If ij=ik where ik∈ t ={i1, i2 ... i|t|}, ik∈ I, (1 ≤  k ≤ |t|) then
5:            tbitvector += 1;
6:        Else
7:            tbitvector += 0;
8:        End If;
9:    next;
10:     Output the database transactional bit vector tbitvector;
11: End;
```

4.7. Support Counting using BitWise AND/OR operations

Support counting for the candidate itemsets in Apriori like algorithms is time consuming process. Apriori like algorithms use hash trees for the counting process. In (Dong and Han, 2007), it was proved that applying BitWise AND/OR operations on the BitTable data structure to count the supports of the candidate itemsets is much faster than the support counting using the hash trees.

For every itemset in the candidate itemsets, a BitWise AND operation is performed with the transactions bit table. The itemset support count is increased if the resultant bit vector is not 0. If the support count is greater than the minimum support, then the candidate itemset is considered as frequent. Otherwise, it is discarded the next iteration. Algorithms 4.5 and 4.6 show how the Local Agents use the BitWise AND/OR operations to count the supports for the k=1 and for k iteration respectively.

Algorithm 4.5 Support counting process using BitWise AND/OR operations at k=1 (Local Agents)

```
1:  begin
2:      for each item bit vector I^b_i in the items bit table do
3:          for each transaction in the database do
4:              Perform BitWise AND operation with t_bitvector;
5:              If I^b_i AND t_bitvector = I^b_i then
6:                  Increment the support(I^b_i) using OR operation ;
7:              end;
8:          end;
9:          If total support (I^b_i) >= minsupp then
10:             Add I^b_i to set of large itemsets Lb_1 ;
11:         end;
12:     end;
13:     Output set of 1-Frequent itemsets ;
14: end;
```

Algorithm 4.6 BitWise AND/OR operations for counting process at k iteration (Local Agents)

```
1:  begin
2:      for each candidate k-itemset bit vector C^i_bitvector in the candidates
        bit table (C_k) do
3:          for each transaction t_bitvector in the database do
4:              Perform BitWise AND operation for C^i_bitvector with t_bitvector;
5:              If C^i_bitvector AND t_bitvector = C^i_bitvector then
6:                  Increment the support(C^i_bitvector) ;
7:              end;
8:          end;
9:          If total support (C^i_bitvector) >= minsupp then
10:             Add C^i_bitvector to set of large itemsets Lb_k;
11:         end;
12:     end;
```

```
13:         | Output set of k-Frequent itemsets ;
14:     end;
```

4.8. Candidate itemsets generation using BitWise AND/OR operations

In Apriori like algorithms, candidate (k+1) itemsets are generated using the frequent k-itemsets calculated in the previous iteration. This is done using hash trees data structure.

In the BitTable data structure, candidate (k+1) itemsets are generated as follows. For each k-frequent itemset, a middle variable (MID) is created by replacing the last bit into 0. For each frequent itemset following the MID, if the result of the Bitwise AND operation between this itemset and the MID has the same value as the MID, a BitWise OR operation is performed between the original bit vector (i.e. before generating the MID) and the itemset. The result bit vector is added to the list of candidate (k+1) candidate itemsets for the next iteration. This process is performed for all itemsets in the list of k-frequent itemsets. Applying BitWise AND/OR operations to the BitTable data structure in order to generate the candidate itemsets is very efficient and very quick. The process is presented in Algorithm 4.7.

Algorithm 4.7 BitWise AND/OR operations for candidate k+1itemsets generation (Main Agent)

```
1:  begin
2:      for each frequent k-itemset bit vector F^i_bitvector in the set of
        frequent itemsets bit table(F_bk) do
3:          Get the Mid of F^i_bitvector = (set of items with the last bit =
            1 change to 0);
4:          for each frequent k-itemsets F^j_bitvector where (i+1 ≤ j ≤
            number of frequent k-itemsets) do
5:              Perform BitWise AND operation with F^j_bitvector;
6:              If Mid of F^i_bitvector AND F^j_bitvector = Mid of F^i_bitvector then
7:                  Generate Candidate k-itemset bit vector
                    C^(k+1)_bitvector = F^i_bitvector OR F^j_bitvector;
8:                  Add C^(k+1)_bitvector to the set of candidate k+1
                    itemsets bit table (Cb_k+1);
9:              end;
10:         end;
11:     end;
12:     Output set of Candidate k+1itemsets bit table (Cb_k+1) ;
13: end ;
```

4.9. Database Transaction Trimming

An item in a transaction t can only be trimmed if it is not in at least k of the candidate k-itemsets in t. In other words, for any transaction containing frequent (k+1) itemsets, any item contained in these frequent (k+1) itemsets must be in at least k times of the candidate k-itemsets in C_k.

More trimming is proposed for extra transaction trimming by checking each (k+1) subsets of the resultant transaction t. For each subset, all k subsets must be contained in C_k to be included in the transaction else it is trimmed from the transaction.

The whole transaction can also be trimmed if the length of the transaction is less than the iteration k.

The technique is explained in Algorithm 4.8.

Algorithm 4.8 Database Transactions Trimming (Local Agents)

```
1:  begin
2:      for each t ∈ DB^i do
3:          for each c ⊂ C_k^i do
4:              If c ⊂ t then
5:                  Increase the count support of c;
6:                  for all subsets x of c do
7:                      count[x]++;
8:                  end;
9:              end;
10:         end;
11:         for each item t_i ∈ t where i<= |t| do
12:             If count[t] < k then
13:                 Remove item t_i from transaction t;
14:             end;
15:         end;
16:         for all k+1 subsets y of the resultant transaction t do
17:             If all k-itemsets subsets z of y⊂C_k^i then
18:                 Add z to the new transaction t_new;
19:             end;
20:         end;
21:         If |t| <=k then trim the whole transaction t;
22:         Output the new transaction t_new;
23:     end;
24: end;
```

4.10. Removing non-frequent 1-itemsets and Grouping Identical Transactions

One of the transaction trimming techniques included in the proposed algorithm is the removal of non-frequent 1- itemsets. The technique reduces the number of database transactions. We claim that combining this technique with the proposed technique “Grouping Identical transactions” reduces the number of database transactions to a great percentage. Assume the database transaction records as shown in Table 4.2 given that the minimum support value is 45%.

Table 4.2 The Initial Database Transactions

TID	Items
100	ABCD
101	ACEF
102	ACE
103	AD
104	ABCDF
105	AB
106	ACEF
107	ABD
108	ADE
109	ABCDF
110	ABD
111	ABCD

The proposed Grouping Identical Transactions technique groups the identical database transactions in the database. For each transaction, the related support is calculated as the count of these transactions in the database. This is shown in Table 4.3.

Table 4.3 Grouping Identical Transactions Technique

Transactions		
TID	Items	Count
100,111	ABCD	2
101,106	ACEF	2
102	ACE	1
103	AD	1
104,109	ABCDF	2
105	AB	1
107, 110	ABD	2
108	ADE	1

After the removal of the non-frequent 1-itemset {E,F}, the “Grouping Identical Transactions technique” is applied again to the resultant database transactions. Result is displayed in Table 4.4.

Table 4.4 Eliminating non-frequent 1-itemsets and Grouping Identical Transactions

Transactions (after k=1)		
TID	Items	Count
100, 111, 104, 109	ABCD	4
101,106,102	AC	3
103, 108	AD	2
105	AB	1
107, 109	ABD	2

After elimination of the non-frequent 1-itemsets and applying the proposed Grouping Identical Transactions technique again, number of database transactions is reduced to 42% of the original database transactions.

4.11. Analytical Evaluation for the proposed BMAAR algorithm

In this section, the algorithm complexity and the message negotiation costs for BMAAR algorithm are evaluated.

4.11.1. Algorithm Complexity Cost of BMAAR

It was proved in section 3.6.1 that the ***Count Distribution algorithm (CD)*** requires a complexity of $O(n^2)$ (Cheung et al., 2002). Where n is the number of sites and C_k is the candidate itemsets support counts at k iteration.

Section 3.6.1 also proved that the ***Distributed Mining of Association Rules algorithm (DMA)*** requires a complexity of $O(n^2)$. Moreover, the algorithm claimed that the cost can reach less than $O(n^2)$ due to the generation of less number of candidate itemsets than CD.

The complexity cost of the proposed ***BMAAR algorithm*** is calculated as the cost of sending local candidate itemsets support counts to main site (O (n)) in addition to the cost of sending global candidate itemset support counts to local sites (O (n)).

Total Algorithm Complexity Cost = O (n) + O (n) = O (n)

From the above costs calculation, it is obvious that the algorithm complexity cost of the proposed ***BMAAR algorithm*** is the same as that of DMAAR algorithm and is better than that of CD and DMA algorithms.

4.11.2. **Message Negotiation Cost of BMAAR**

As mentioned in section 3.6.2, the total message exchange cost of the ***Count Distribution algorithm (CD)*** is:

$$N.(N-1).K.SC_{\max}$$

Where N is the number of sites, K is the number of iterations and SC_k^i is the support counts of the candidate itemsets at iteration k for site i.

Similarly, as mentioned in section 3.6.2, the total message exchange cost of ***Distributed Mining of Association Rules algorithm (DMA)*** is:

$$2.N.(N-1).K.SC_{\max} + 2.N.(N-1).K.C_{\max}$$

Message Exchange Cost of the proposed ***BMAAR algorithm*** is the cost of sending the generated candidate itemsets from the main site to all local sites in addition to the cost of returning back their support counts to the main site. The cost is the same as that of DMAAR as both algorithms have the same paradigm. For this reason, the total Message Exchange Cost of BMAAR is:

$$N.K.C_{\max} + N.(K-1).C_{\max}$$

From the above analysis it is obvious that the message negotiation cost between agents is the same as that of DMAAR and is less than that of DMA algorithm. Moreover, the cost is better than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

Similar to DMAAR, the proposed BMAAR algorithm is more scalable than CD and DMA as the message negotiation cost is directly proportion to the number of sites rather than the square of the number of sites as in CD and DMA.

4.12. Transaction Trimming Evaluation on Benchmark datasets

The experiments were conducted to emphasize the importance of applying the Removal of non-frequent 1-itemsets and the Grouping of the Identical Transactions techniques to the database. Experiments were conducted on the same five benchmark datasets from UCI machine learning repository at four different supports. The five benchmark datasets from UCI machine learning repository related to different application domains are described in Table 3.1. Total number of transactions and the number of transactions after applying the trimming techniques were recorded.

Results obtained are presented in Table 4.5 and Figure 4.3 to Figure 4.7.

Table 4.5 shows the databases used in the experiment, the total number of database transactions in each database and the number of transactions after applying the transaction trimming techniques.

Table 4.5 Total number of database transactions before and after transaction trimming

Database	Number of transactions before trimming	Number of transactions after applying the transaction trimming techniques at different Supports			
		5	10	25	50
Abalone	4177	2339	2130	2005	1671
Cars	1728	1002	916	812	726
Iris	150	90	83	74	68
Mammography	961	567	529	461	404
Blood Transfusion	748	426	381	359	307

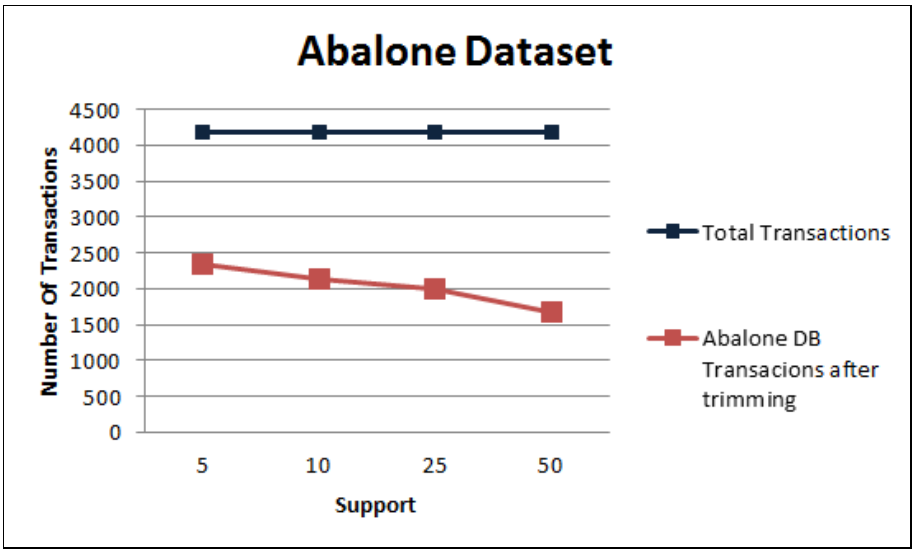

Figure 4.3 Number of transactions for Abalone dataset before and after applying the transaction trimming techniques.

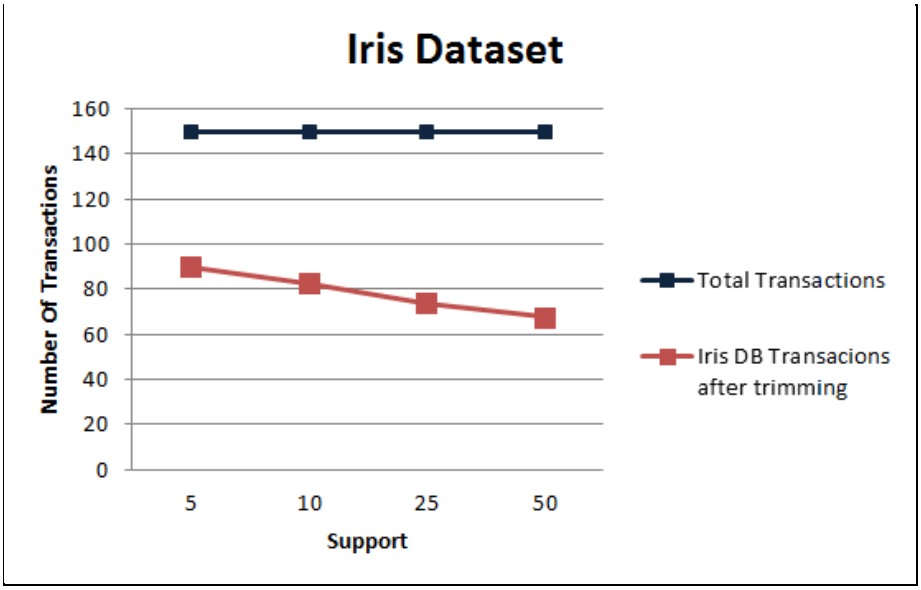

Figure 4.4 Number of transactions for Iris dataset before and after applying the transaction trimming techniques.

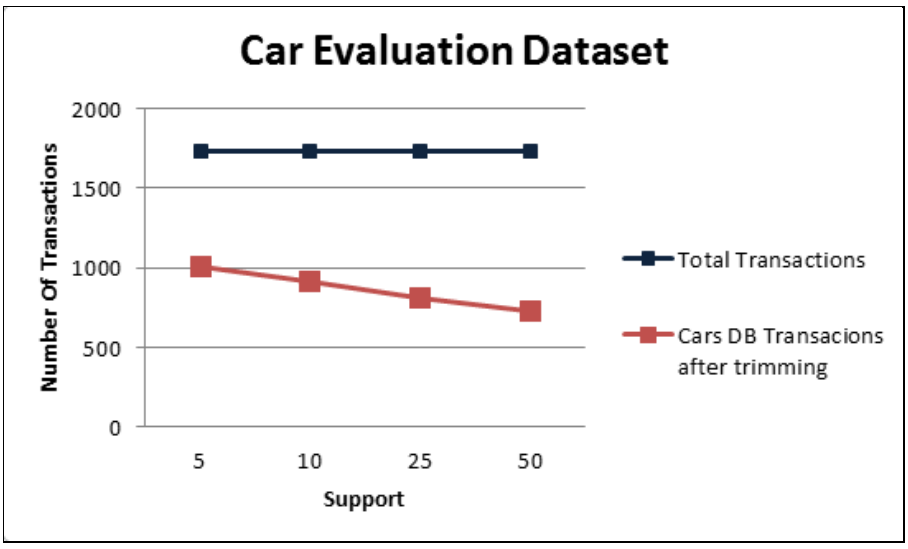

Figure 4.5 Number of transactions for Car Evaluation dataset before and after applying the transaction trimming techniques.

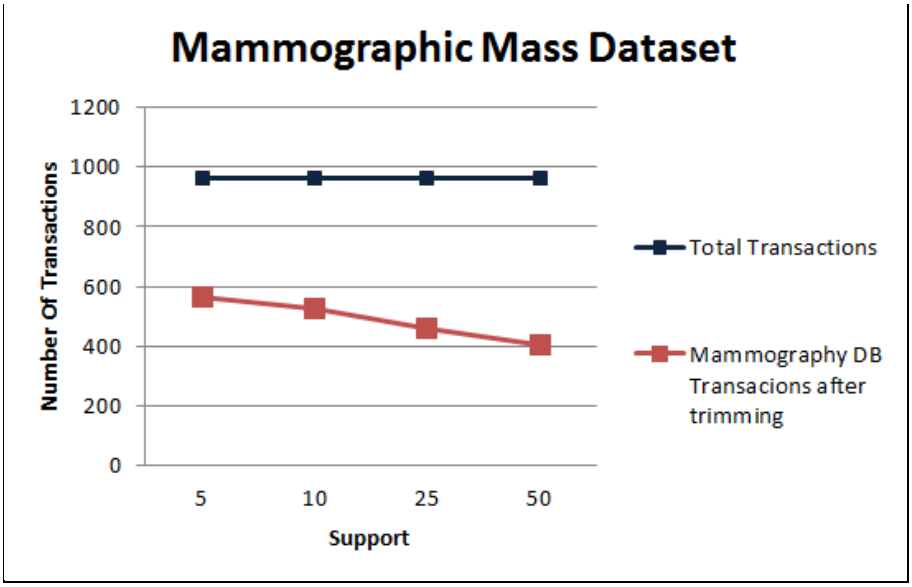

Figure 4.6 Number of transactions for Mammographic Mass dataset before and after applying the transaction trimming techniques.

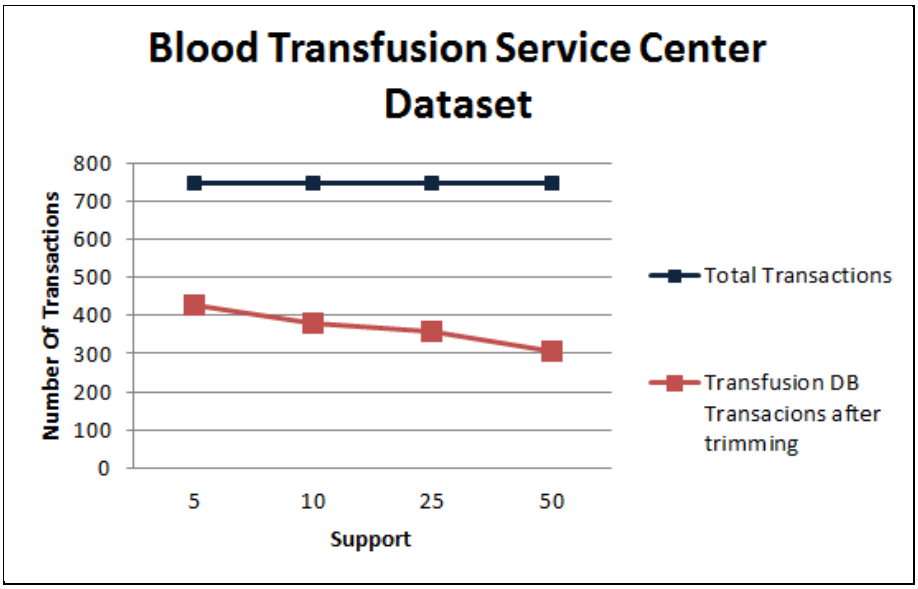

Figure 4.7 Number of transactions for Blood Transfusion Service Center dataset before and after applying the transaction trimming techniques.

Results in Figure 4.3 to Figure 4.7 and Table 4.5 show the following:

1. Figure 4.3 to Figure 4.7 show that applying the transaction trimming techniques reduces the number of database transactions, consequently, the total execution time for the proposed algorithm.
2. The Figures also show that when the support value decreases, the number of non-frequent itemsets decreases. Thus, the number of the trimmed transactions due to the removal of these non-frequent itemsets decreases. On the other hand, when the support value increases, the number of trimmed transactions increases.
3. Table 4.5 showed that the reduction in the database transactions due to applying the transactions trimming techniques averages between 40% and 60%.

4.13. Performance Evaluation on Benchmark datasets

The experiments included the implementation of four algorithms against five different real world datasets at five different supports with total of 100 values. Algorithms implemented are the BitTableFI algorithm (BT), the Count Distribution algorithm CD, DMAAR and BMAAR. The five benchmark datasets from UCI machine learning repository related to different application domains are described in Table 3.1.

The results obtained are illustrated in the following Figures.

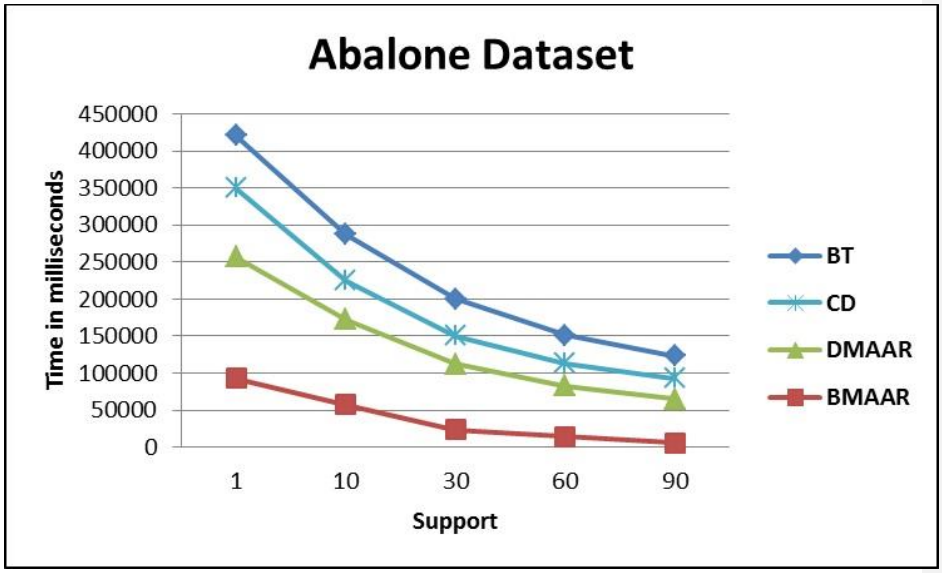

Figure 4.8 Testing algorithms on Abalone Dataset

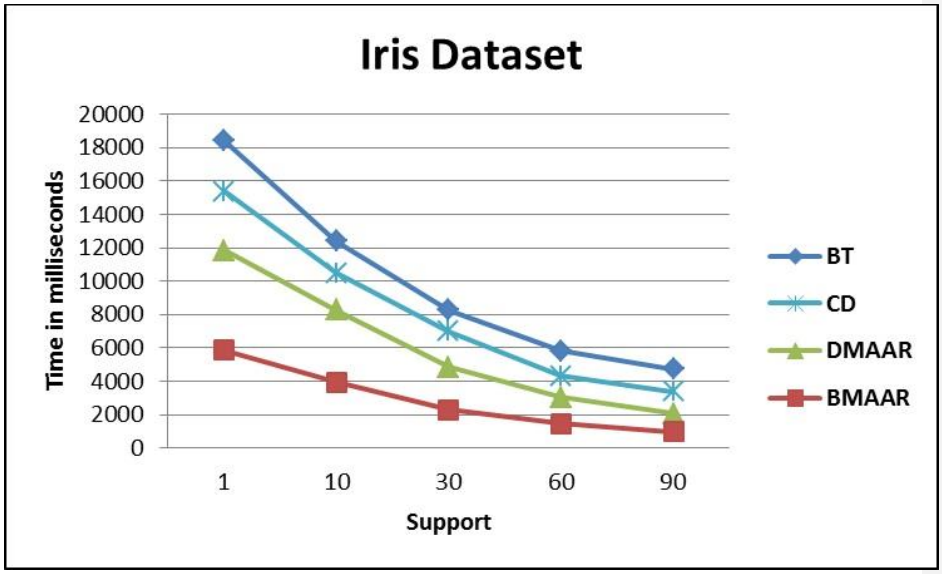

Figure 4.9 Testing algorithms on Iris Dataset

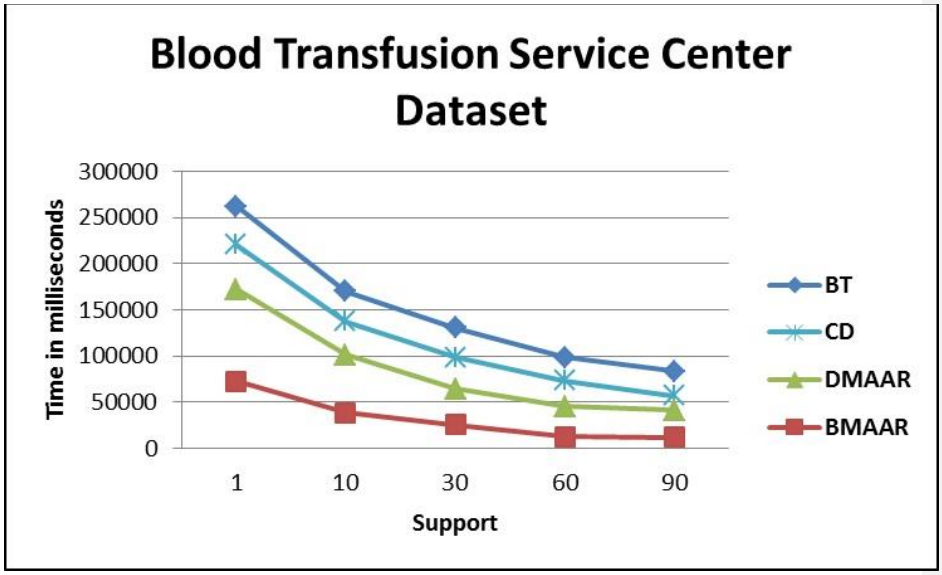

Figure 4.10 Testing algorithms on Blood Transfusion Service Center Dataset

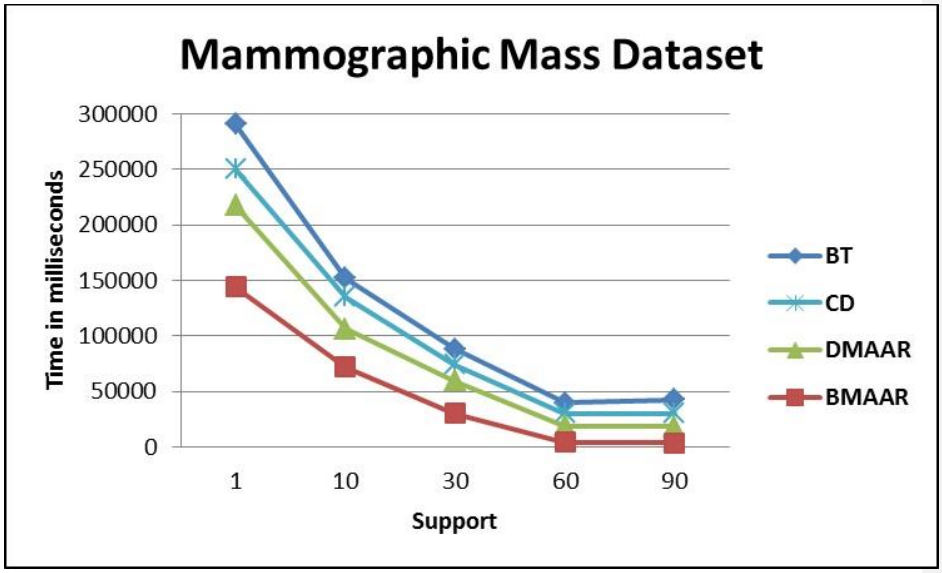

Figure 4.11 Testing algorithms on Mammographic Dataset

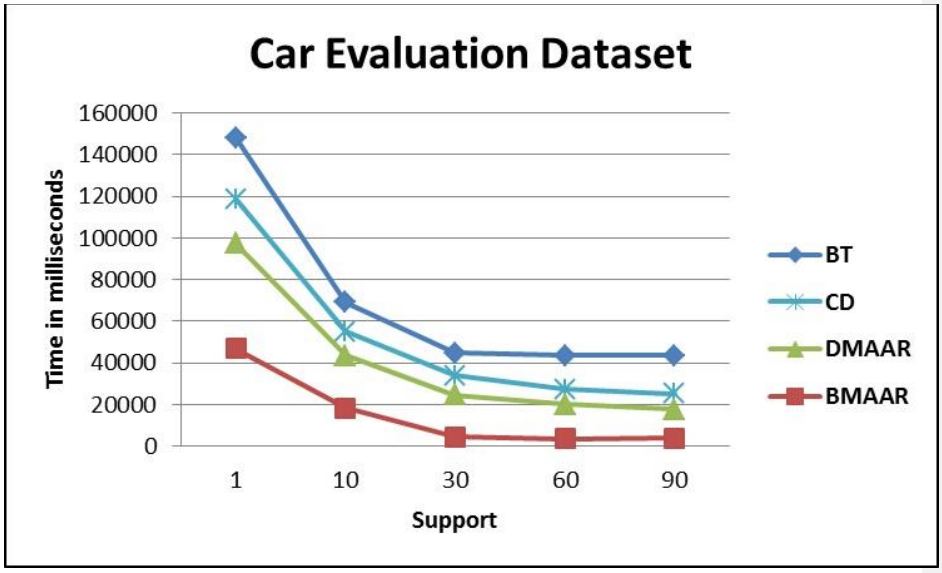

Figure 4.12 Testing algorithms on Car Evaluation Dataset

The performance evaluation of BMAAAR shows the following:

1- Results presented in Figure 4.8 to Figure 4.12 show that BMAAR outperforms the Count Distribution algorithm, the BitTableFI algorithm and the previously proposed DMAAR algorithm.

2- BMAAR outperformed DMAAR and existing algorithms because it included three transaction trimming techniques which reduced the large number of database transactions. It also used the efficient BitTable data structure which was proved to have better performance and less memory than hash trees data structure. Moreover, BMAAR included two BitWise AND/OR operations for quick candidate itemsets generation and quick support counting.

4.14. Summary and Conclusion

In this chapter, BMAAR algorithm was proposed to improve the performance of DMAAR algorithm by tackling the limitations presented in 3.8.

The proposed BMAAR algorithm used the BitTable data structure to minimize the memory needed to store the transactions at the main and the local sites. It has been proved in (Dong and Han, 2007) that the BitTable data structure provided better performance and less memory than hash trees data structure used by CD, PDM, DMA and FPM. Moreover, BMAAR algorithm applied two BitWise AND/OR operations proposed in the BitTableFI algorithm for quick candidate itemsets generation and quick support counting.

A comparative study has been conducted in order to evaluate the effect of including the BitTable data structure and the Bitwise AND/OR operations on the performance of the proposed algorithm. BMAAR has been compared with the BitTableFI algorithm (BT), the Count Distribution algorithm (CD) and DMAAR on five benchmark datasets from UCI machine learning repository that are related to different application domains. Experiments have been conducted at five different supports with total of 100 values. Results showed that BMAAR outperforms other algorithms.

To further improve the efficiency of the proposed algorithm, BMAAR included three transactions trimming techniques to reduce the huge number of database transactions, consequently, the time needed for the mining process. These techniques were the Direct Hashing and Pruning technique, the proposed Grouping Identical Transactions technique and the Removing Non Frequent 1-Itemset technique.

A comparative study has been conducted in order to evaluate the effect of including the three transaction trimming techniques in the proposed algorithm. Experiments proved that applying the transactions trimming techniques helped in reducing the number of database transactions and the response time of the proposed algorithm. Transactions reduction due to applying the trimming techniques averaged between 40% and 60%.

Analytical evaluation showed that the proposed BMAAR algorithm achieved better algorithm complexity cost (O(n)) when compared with the Count Distribution algorithm ($O(n^2)$) and DMA (less than $O(n^2)$).

Analytical evaluation also showed that the BMAAR message negotiation cost is less than that of DMA algorithm and is better than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

In summary, the proposed BMAAR algorithm presented the following contributions:

1- The proposed BMAAR algorithm helped in reducing the number of database transactions and consequently the time needed for the whole mining process by including three transaction trimming techniques.
2- Applying the efficient Bit data structure which was proved to have better performance and less memory than hash trees data structure helped to fit the database transactions into the memory of the Main and the Local sites.
3- Applying the BitWise AND/OR operations helped in quick candidate itemsets generation which is a time consuming process.

4- Applying the BitWise AND/OR operations helped in quick support counting for the generated candidate itemsets.

5- According to the analysis presented in this chapter and since BMAAR was based on the same Multi-Agent based framework of DMAAR, the proposed BMAAR algorithm provided the same contributions of DMAAR such as reducing the algorithm complexity cost, reducing the message negotiation cost, maximizing the tasks parallelism, minimizing the waiting time of the agents, avoiding the all-to-all broadcasting between sites, complying with the FIPA communication standard and providing scalability with the increase in the number of sites.

However, BMAAR algorithm suffered the following limitation:

1- BMAAR did not address the problem of generating large number of candidate itemsets. This is a very time consuming process. Moreover, the increase in the number of generated candidate itemsets increases the time needed to count their supports.

Due to the aforementioned limitation of BMAAR, the Pruned Multi-Agent Association Rules Algorithm (PMAAR) is proposed in the next chapter to address and improve BMAAR.

Chapter Five

5. The Pruning Multi-Agent Association Rules Algorithm (PMAAR)

5.1. Chapter Overview

In Count Distribution algorithm (CD), each site generates the candidate itemsets based on the frequent itemsets generated from the previous iteration. One of the limitations of CD algorithm is that it generates large number of candidate itemsets which incurs a large amount of communication (Cheung et al., 2002).

In the previous chapter, BMAAR used the BitTable data structure to minimize the memory needed to store the transactions at the main and the local sites, included two BitWise AND/OR operations for quick candidate itemsets generation and quick support counting and implemented transaction trimming techniques to reduce the large number of database transactions. However, BMAAR did not include pruning techniques to reduce the large number of generated candidate itemsets.

In this chapter, the Pruning Multi-Agent Association Rules Algorithm (PMAAR) is proposed. PMAAR applies three pruning techniques to dramatically reduce the large number of generated candidate itemsets. The distribution of the mining tasks presented by DMAAR and BMAAR is different than that of PMAAR. This is because PMAAR reassigns the mining tasks across the agents in order to achieve efficient pruning for the generated candidate itemsets. The experimental results showed that the pruning techniques included in PMAAR helped in reducing the number of generated candidate itemsets.

PMAAR has been compared with the existing algorithms and has proved better performance and execution time.

Analytical evaluation showed that PMAAR reduced the algorithm complexity from $O(n^2)$ to $O(n)$. Analytical evaluation also showed that the algorithm complexity cost of CD and DMA are directly proportion to the square of the number of sites unlike the cost of PMAAR which is directly proportional to the number of sites only. This showed that PMAAR is more scalable than CD and DMA.

5.2. Introduction

Generating candidate itemsets is a nontrivial problem. If the number of items (m) and the database (*D)* are large, the number of possible distinct candidate itemsets is 2^m-1.When the number of generated candidate itemsets increases, the time needed to count their supports increases. This is because, for each candidate itemset, the database needs to be scanned once to calculate the support count. This is a very time consuming process.

One of the limitations of CD algorithm is that it generates large number of candidate itemsets. The support counts of these candidate itemsets are broadcasted from each site to all other sites causing a large amount of communication between the distributed sites (Cheung et al., 2002).

The number of candidate itemsets ($|C_k|$) generated by Apriori can be calculated by following equation (Park et al., 1997):

$$\left|C_k\right| = \binom{\left|L_{k-1}\right|}{2} = \frac{\left|L_{k-1}\right|!}{(\left|L_{k-1}\right| - 2)!\times 2!} = \frac{\left|L_{k-1}\right| * \left(\left|L_{k-1}\right| - 1\right)}{2}$$

Where (L_{k-1}) is the set of frequent itemsets generated at iteration (k-1). For instance, if we have 100 large itemsets at iteration (k-1), 4950 candidate itemset are generated for the next iteration (k) which is a very large number.

In this chapter, the Pruned Multi-Agent Association Rules algorithm (PMAAR) which is considered as further improvement for BMAAR is presented.

PMAAR applies three pruning techniques that have been modified to work on the BitTable data structure rather than the hash trees in order to reduce the number of generated candidate itemsets. These techniques are the Global Pruning Technique proposed in (Cheung et al., 2002), Distributed Pruning Technique proposed in (Cheung et al., 2002) and Apriori Pruning Technique proposed in (Agrawal and Shafer, 1996). The Global Pruning Technique and the Distributed Pruning Technique have been adopted in (Rui and Zhiyi, 2011) while the Apriori Pruning Technique has been adopted in (Li and Li, 2010, Chong and Yanqing, 2011). The three pruning techniques applied in PMAAR

have been selected due to their efficiency in reducing the number of candidate itemsets.

In the ***Distributed Pruning Technique***, local candidate itemsets are generated in each site rather than at the main site. This reduces the number of generated candidate for the next iteration. For instance, assume we have 4 large k-itemsets in site 1, 3 large k-itemsets in site 2 and 3 large k-itemsets in site 3with minimum support = 4.This can be shown in Table 5.1.

Table 5.1 Number of k-Frequent itemsets at each site

Itemset	Site 1	Site 2	Site 3
A	12	0	0
B	7	5	0
C	8	4	0
D	0	12	0
E	0	0	12
F	0	0	12
G	0	0	12
H	12	0	0

Local candidate itemsets generated are {AB,AC,AH,BC,BH,CH}, {BC,BD,CD}, {EF,EG,FG} for sites 1,2 and 3 respectively. Total number of candidate itemsets generated by all local sites is 12. Moreover, when the Main Agent receives the local candidates from all sites, redundant candidate itemsets like {BC} are removed, reducing the total number of global candidates to 11 candidates. On the other side, due to the all-to-all broadcasting in Apriori like algorithms, the number of generated candidate itemsets is 28 candidates.

From the example above, we can see that applying the Distributed Pruning technique prunes the number of generated candidate itemsets to less than 40% of original size.

In the ***Global Pruning Technique*** the expected maximum support for the k itemset is calculated based on the support counts of the k-1 subsets for the itemset. If the sum of the maximum supports from all sites is less than the global minimum support, this itemset is pruned. Assume that the frequent itemsets support counts are as shown in Table 5.2 given that the global minimum support is 15.

Table 5.2 Frequent Itemsets Support count at (k-1) iteration

Site 1	Site 2	Site 3
C=1	C=12	C=2
D=2	D=34	D=1

In the above example, the maximum expected supports for the itemset {CD} at site 1, 2 and 3 are the minimum of the count supports of the k-1 itemsets which are 1, 12 and 1 respectively. Since the total number of supports =14 is less than the global minimum support. Hence, the itemset {CD} is pruned from the candidate itemsets for the next k-iteration.

Apriori Pruning technique is based on the fact that all subsets of any frequent itemset must be frequent. In this technique, all items whose (k-1) subsets are not in L_{k-1} are deleted from $C_{k.}$

Where C_k is the candidate itemset at iteration k and L_{k-1} is the frequent itemsets at iteration (k-1).

5.3. The Proposed PMAAR algorithm

5.3.1. Local agents

The Main Agent accepts the support and confidence from the user and sends them to the Local Agents at all sites. Each Local Agent converts its local database into bit vectors as in step 2 in algorithm 5.1. The Database conversion is explained in Section 4.6. This helps to compress the database at each local site to fit better in local memory. It helps also to speed up later processes including candidates generation and candidates support counting. Local agents then use the “Grouping Identical Transactions trimming technique” to minimize the number of database transactions in their local databases (see Section 4.10).

Support counts for the candidate itemsets are calculated by Local Agents using BitWise AND/OR operations (see Section4.7). At k iteration, candidate itemsets are generated by Local Agents from previous pass (see Section 4.8). Two pruning techniques are applied to the generated candidate itemsets: First, the Apriori Pruning technique, which states that the subsets of large itemsets must be large. Second, the Distributed Pruning technique which reduces the number of generated candidate itemsets. The efficiency of the technique and how it is applied to the BitTable data structure is explained in Section 5.2. Local candidate itemsets generated for the second phase are sent to the main site. Itemsets whose support counts are greater than the local minimum support are saved in the local frequent itemsets knowledge base.

At phase one only, non-frequent 1-itemsets are removed from the database using the “Removing non-frequent 1-itemsets”technique. Then, “Grouping Identical Transactions” is applied to further decrease the number of transactions. “Removing non-frequent 1-itemsets”and “Grouping Identical

Transactions" techniques and how they are applied to the BitTable data structure are explained in Section 4.10. Local agents sleep and wait until the Main Agent sends the global pruned candidate itemsets for the next phase. When Local Agents receive the global pruned candidates, they apply the DHP transaction trimming technique. The technique and how it is applied to the BitTable data structure is explained in Section 4.9. The algorithm implemented by the Local Agents at the first iteration is presented in Algorithm 5.1.

Algorithm 5.1 The Proposed Algorithm at the Local Agents (at the first iteration)

```
1:  begin
2:      Convert_Database_into_bit_vectors () ;
3:      Group Identical Transactions technique;
4:      // Perform BitWise AND operation operations;
5:      Calculate support count for items bit vector;
6:      Send support counts only to the main site;
7:      for each item bit vector Ib_i in the items bit table do
8:          If total support (Ib_i) >= s^i then
9:              Add Ib_i to set of local large itemsets Lb^i_1;
10:         end;
11:     end;
12:     // Perform BitWise AND operation operations;
13:     Generate Local candidate itemsets C^i_2;
14:     Apply Apriori_Prune() function on C^i_2 (see Section 5.2);
15:     Use Distributed Pruning Technique (see Section 5.2);
16:     Send Local candidate itemsets C^i_2 to the main site;
17:     Save local large 1-itemsets Lb^i_1 in local knowledge base;
18:     Remove non-Frequent 1-itemsets from DB^i;
19:     Group Identical Transactions;
20:     Receive global pruned candidate itemsets C_2 from main site;
21: end;
```

The algorithm implemented by the Local Agents at k-iteration is presented in Algorithm 5.2.

Algorithm 5.2 The Proposed Algorithm at the Local Agents (at k iteration)

```
1:  Begin
2:      // Perform BitWise AND operation operations;
3:      Calculate support count for global candidate itemsets Ck;
4:      Sends local support counts only of C^i_k to the main site;
5:      for each itemset C^i_k in the global candidate itemsets Ck do
6:          If total support (C^i_k) >= s^i then
7:              Add C^i_k to set of large itemsets Lb^i_k;
8:          end;
9:      end;
10:     // Perform BitWise AND operation operations;
11:     Generate Local candidate itemsets C^i_k+1;
12:     Apply Apriori_Prune() function on C^i_k+1;
13:     Use Distributed Pruning Technique;
14:     Send local candidate itemsets C^i_k+1 to the main site;
15:     Save Lb^i_k+1 in local knowledge database;
16:     Receive global pruned candidate itemsets C^i_k+1 from main site ;
17:     Apply PDM Transaction Trimming Technique;
18:     If global k+1 candidate itemsets C^i_k+1 exists then
19:             go to step 1;
20:     end;
21: end;
```

5.3.2. **Main Agent**

The Main agent sends the user support and confidence to all Local Agents. When the Main Agent receives the local candidate k-itemsets support counts from all Local Agents, it calculates the global counts. If the itemset global count is greater than the global minimum support, the itemset is saved in the global frequent itemsets knowledge base. Otherwise, it is discarded for the next iteration.

When the Main Agent receives the local candidate (k+1) itemsets from all Local Agents, it applies the Global Pruning Technique to reduce the number of generated global candidate itemsets for the next iteration. Details of the technique and how it is applied is described in Section 5.2. Global pruned candidate itemsets are sent to all Local Agents. Main agent sleeps and waits for the support counts and the local candidate (k+1) itemsets from Local Agents. The algorithm implemented by the Main Agent at k-iteration is presented in Algorithm 5.3.

Algorithm 5.3 The Proposed Algorithm at the Main Agent (at k iteration)

```
1:  Begin
2:      Sends the required support and confidence to Local Agents;
3:      Receives local candidates C^i_k support counts from all sites;
4:      Calculates global counts for candidate itemsets C^i_k;
5:      for each itemset X in the global candidate itemsets C_k do
6:          If X.supp>= MinimumSupport then
7:              Saves X and X.supp in the global knowledge database
                Lb_k;
8:          end;
9:      end;
10:     Receives local candidate itemsets C^i_{k+1} from all sites;
11:     Merges global pruned candidate itemsets C_{k+1} from C^i_{k+1};
12:     Applies Global Pruning;
13:     If global k+1 candidate itemsets C_{k+1} exists then
14:         Sends global pruned candidate itemsets C_{k+1} to local sites;
15:         go to step 3;
16:     end;
17: end;
```

5.4. Message Negotiation between the Main and the Local agents

The message negotiation between agents is presented in Table 5.3.

Table 5.3 Message Negotiation between Main and Local Agents for PMAAR algorithm

Main Agent	Local Agents
	1. Convert transactions into Bit Table 2. Apply Grouping Identical Transactions technique. 3. Calculate support counts for local 1-candidate itemsets using BitWise AND/OR operations.
1. Receives and sums all support counts for local 1-candidate itemsets from Local Agents. ←	4. Send local support counts only to main site
2. Find global 1-Frequent itemsets	5. Calculate the local 1-Frequent itemsets.
3. Save global 1-Frequent itemsets in global knowledge base	6. Generate the set of local 2 candidate itemsets using BitWise AND/OR operations. 7. Apply Apriori_Prune() function. 8. Apply Distributed Pruning Technique.
4. Receives the set of local k+1 candidate itemsets from all local sites. ←	9. Sends the set of local 2 candidate itemsets to Main Agent
5. finds global 2-candidate itemsets from all local sites	10. Save 1-Frequent itemsets in local knowledge base.
6. Apply Global Pruning	11. Eliminate non-frequent 1-itemsets. 12. Apply Group Identical Transactions technique.
7. Send global pruned candidate 2-itemsets to all local sites →	13. Receive global 2candidate itemsets.

	14. At iteration k do the following:
	15. Calculate support count for local k-candidate itemsets using BitWise AND/OR operations.
8. Receives and sums all support counts for local k-candidate itemsets from all local sites.	16. Sends support count only to main site
9. Find global k-Frequent itemsets	17. Calculate local k-Frequent itemsets greater than minsup
10. Save global k-Frequent itemsets in global knowledge base	18. Generate the set of local k+1 candidate itemsets using BitWise AND/OR operations.
	19. Apply Apriori_Prune() function.
	20. Apply Distributed Pruning Technique.
11. Receives the set of local k+1 candidate itemsets from all local sites.	21. Sends the set of local k+1 candidate itemsets to Main Agent
12. finds global k+1candidate itemsets from set of local k+1candidate itemsets	22. Save k-Frequent itemsets in local knowledge base.
13. Apply Global Pruning	23. Apply PDM Transaction Trimming technique
14. Send generated global pruned k+1candidate itemsets to all local sites	24. Receive global pruned k+1candidate itemsets. If candidate itemsets exist, then go to step 3

5.5. The Sequence Diagram of PMAAR algorithm

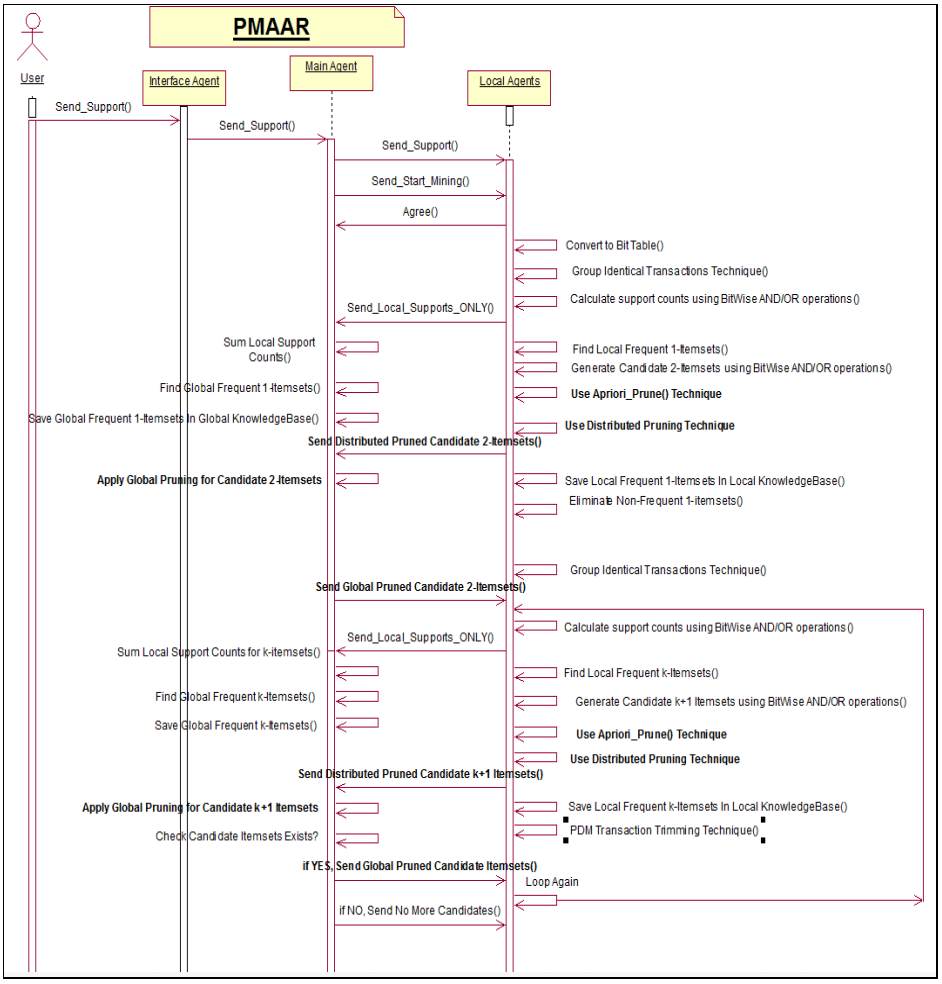

Figure 5.1 The Sequence Diagram of the proposed PMAAR algorithm

According to the discussion in Section 3.4, the Sequence diagram presented in Figure 5.1 shows the sequence of operations during the execution of the proposed PMAAR algorithm. Moreover, it shows how the distribution of the mining tasks maximized the tasks parallelism. The Sequence diagram highlights the tasks that enhance the performance of PMAAR.

5.6. The Activity Diagram of PMAAR algorithm

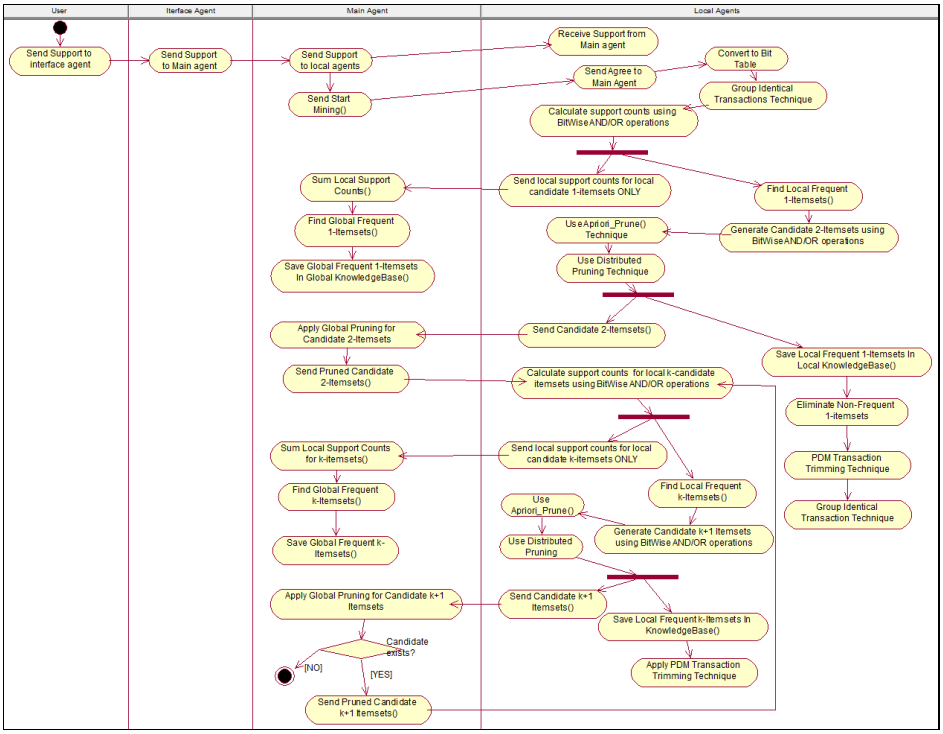

Figure 5.2 The Activity Diagram of the proposed PMAAR algorithm

According to the discussion in Section 3.5, PMAAR activity diagram presented in Figure 5.2 shows how the order of the mining processes has been designed to give higher priority to those subtasks that other agents are waiting for. The design helps in minimizing the waiting time of the whole algorithm.

5.7. Analytical Evaluation for the proposed PMAAR algorithm

5.7.1. Algorithm Complexity Cost of PMAAR

It has been proved in section 3.6.1 that due to the all-to-all broadcasting of candidate itemsets, ***Count Distribution algorithm (CD)*** requires a complexity of $O(n^2)$ (Cheung et al., 2002). Where n is the number of sites and C_k is the candidate itemsets at k iteration.

Section 3.6.1also proved that due to the all-to-all broadcasting of candidate itemsets from every site to all other sites, ***Distributed Mining of Association Rules algorithm (DMA)*** requires a complexity of $O(n^2)$ and it claimed that it may reach <= $O(n^2)$ due to the generation of less number of candidate itemsets than CD.

The complexity cost of the proposed ***PMAAR algorithm*** is calculated as the cost of sending the generated local candidate itemsets together with their support counts from all Local Agents to the main site in addition to the cost of returning back the global candidate itemsets to all local sites. This is calculated as follows:

Cost of sending local candidates support counts to the main site is:

$$O(|C_k|. n) = O(n)$$

Cost of sending generated local candidate itemsets to main site is:

$$O(|C_k|. n) = O(n)$$

Cost of sending global candidate itemsets to local sites is:

$$O(|C_k|. n) = O(n)$$

Total Algorithm Complexity Cost = O ($|C_k|$. n) + O ($|C_k|$. n)) + O ($|C_k|$. n))

= O (n) + O (n) + O (n) = O (n)

From the above analysis, it is obvious that the algorithm complexity cost of PMAAR is better than that of CD and DMA algorithms.

5.7.2. **Message Negotiation Cost of PMAAR**

As mentioned in section 3.6.2 the total message exchange cost of the ***Count Distribution algorithm (CD)*** is calculated as:

Total Message Exchange Cost of CD is:

$$N.K.SC_{\max}.(N-1)$$

Where N is the number of sites, K is the number of iterations and SC_k^i is the support counts of the candidate itemsets at iteration k for site i.

Similarly, as mentioned in section 3.6.2, the total message exchange cost of the ***Distributed Mining of Association Rules algorithm (DMA)*** is calculated as follows:

Total Message Exchange Cost of DMA is:

$$2.N.(N-1).K.SC_{\max} + 2.N.(N-1).K.C_{\max}$$

Message Exchange Cost of the proposed ***PMAAR algorithm*** is the cost of sending the local pruned candidate itemsets and the support counts of the

global pruned candidate itemsets from the Local Agents to the Main Agent in addition to the cost of sending the global pruned candidate itemsets from the Main Agent to all Local Agents as follows:

Message Exchange Cost of sending local pruned candidate itemsets and the support counts of the global pruned candidate itemsets to the Main Agent is:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|LPC_k^i\right| + \sum_{i=1}^{N}\sum_{k=1}^{K}\left|GPCSC_k^i\right|$$

Where LPC_k^i is the local pruned candidate itemsets at iteration k for site i and $GPCSC_k^i$ is the support count of the global pruned candidate itemsets at iteration k for site i.

Message Exchange Cost of sending the global pruned candidate itemsets to local sites is:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|GPC_k^i\right|$$

Where GPC_k^i is the global pruned candidate itemsets at iteration k for site i.

Total Message Exchange Cost of PMAAR is:

Field Code Changed

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\left|LPC_k^i\right| + \sum_{i=1}^{N}\sum_{k=1}^{K}\left|GPCSC_k^i\right| + \sum_{i=1}^{N}\sum_{k=1}^{K}\left|GPC_k^i\right| \quad (5.1)$$

For the maximum size of the candidate itemsets, assume that:

$$LPC_{\max} = \max(|LPC_k^i|)$$

$$GPCSC_{\max} = \max(|GPCSC_k^i|)$$

$$GPC_{\max} = \max(|GPC_k^i|)$$

Total Message Exchange Cost of PMAAR is:

$$N.K.LPC_{\max} + .N.K.GPCSC_{\max} + N.K.GPC_{\max} \quad (5.2)$$

Consider the example presented in Section5.2 for the three sites Site1, Site2 and Site3.Assume that the local large 1-itemsets are {A,B,C,H} , {B,C,D} and {E,F,G} for sites 1,2 and 3 respectively.

The local candidate 2-itemsets generated by PMAAR for the three sites are {AB,AC,AH,BC,BH,CH},{BC,BD,CD}, {EF,EG,FG} respectively. Total number of local candidate itemsets generated by all local sites is 12.At k=1, the cost of sending the support counts of the 1-itemsetsfrom the local agents to the main agent is (8 x 3=24 messages). At k=2, the cost of sending the local pruned candidates from the local agents to the main agent is (12 messages). When the Main Agent receives the local pruned candidates from all sites, redundant candidate itemset {BC} is removed and globally pruned candidate itemsets (11 candidates) are sent back to all local sites. This costs additional 33 messages. The cost of sending the support counts of these globally pruned candidates from the local agents to the main agent is 33 messages with total of 102 messages.

For Count Distribution algorithm, the support counts of the 8 items are broadcasted to all sites with total of 3 x 2 x8= 48 messages at the first iteration. At the second iteration, number of candidate itemsets is 28 candidates. The message negotiation cost is calculated as 3 x 2 x 28= 168 messages with total of 216 messages.

The above calculations show that the message negotiation costs of PMAAR and CD at the first iteration are24and 48 messages respectively. At the second iteration, the costs are 78 and 168 messages respectively. The total message costs for both algorithms are 102and 216 messages respectively. Thus, the analysis shows that the message negotiation cost of PMAAR is less than that of CD. This is because of the pruning techniques which dramatically reduce the number of candidates, consequently, the number of messages.

Equation 5.2 shows that the message negotiation cost is less than DMA. Moreover, the pruning techniques implemented and presented in this chapter decreases dramatically the number of messages between agents making the number of messages negotiation at the proposed PMAAR algorithm less than that of count distribution algorithm.

The above equations show that the message negotiation costs of CD and DMA algorithms are directly proportional to the square of the number of sites unlike PMAAR which is directly proportional to the number of sites only. This makes PMAAR more scalable than CD and DMA algorithms when the number of sites increases.

5.8. Candidate itemsets Pruning Evaluation on Benchmark Datasets

In order to compare the number of candidate itemsets generated by the PMAAR, BMAAR and CD algorithms, experiments were conducted on the same five benchmark datasets from UCI machine learning repository at five different supports.

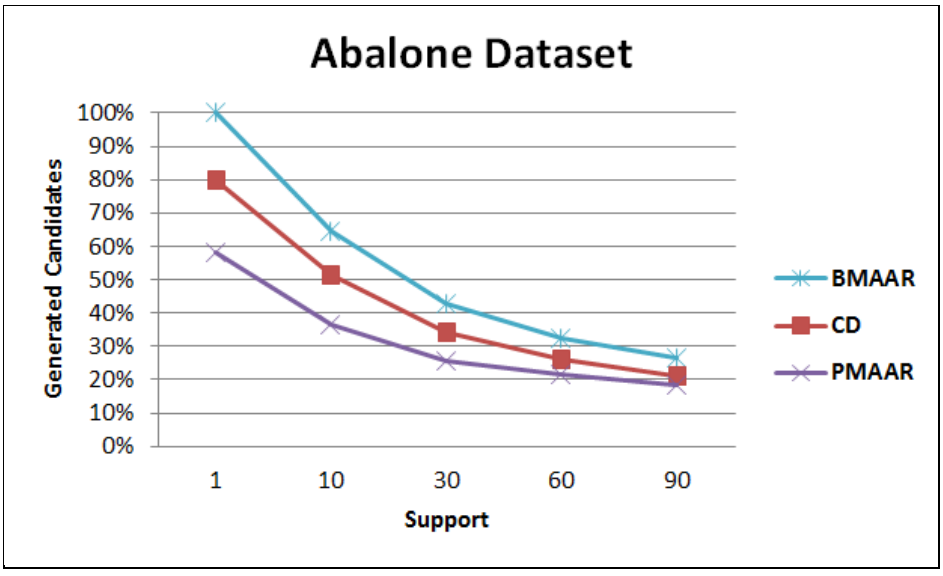

Figure 5.3 Percentage of generated candidate itemsets for abalone dataset.

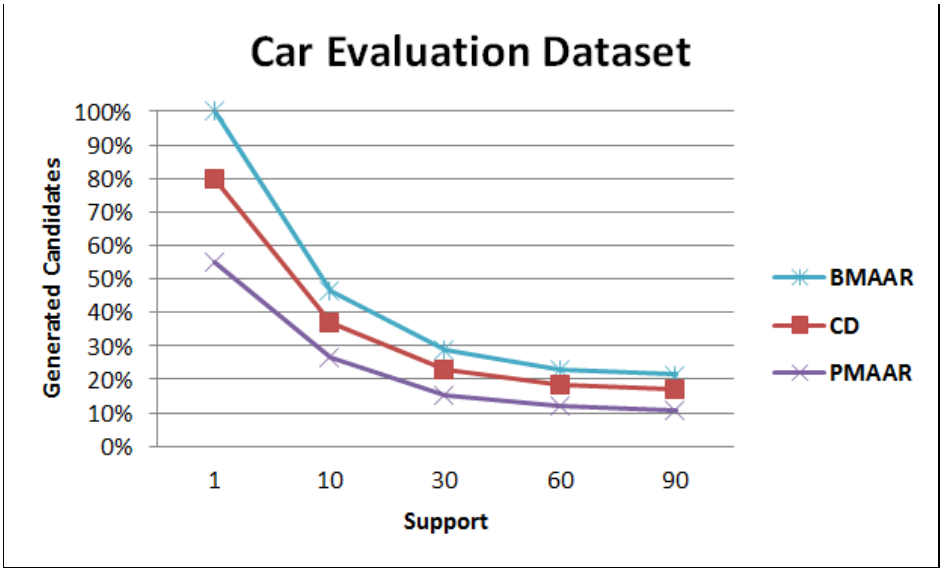

Figure 5.4 Percentage of generated candidate itemsets for car evaluation dataset

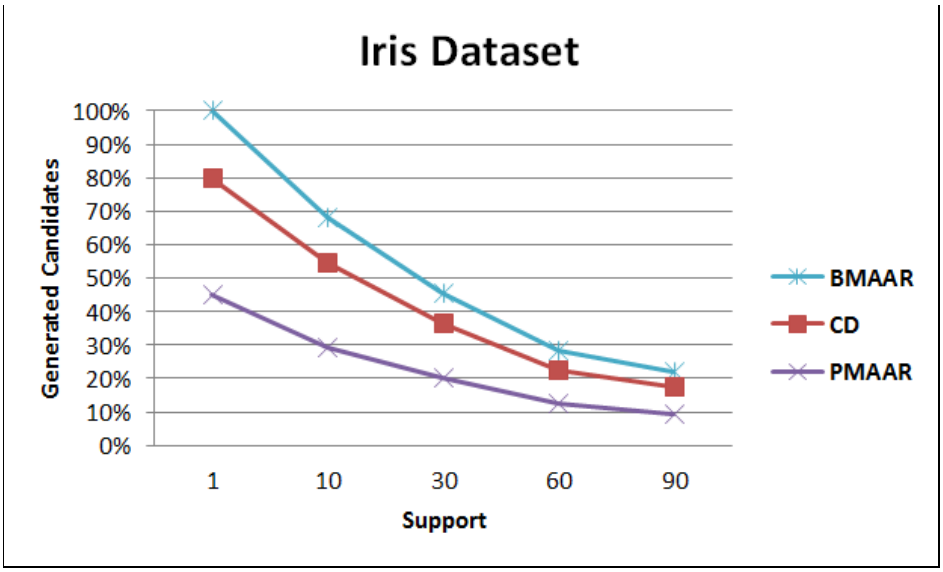

Figure 5.5 Percentage of generated candidate itemsets for iris dataset

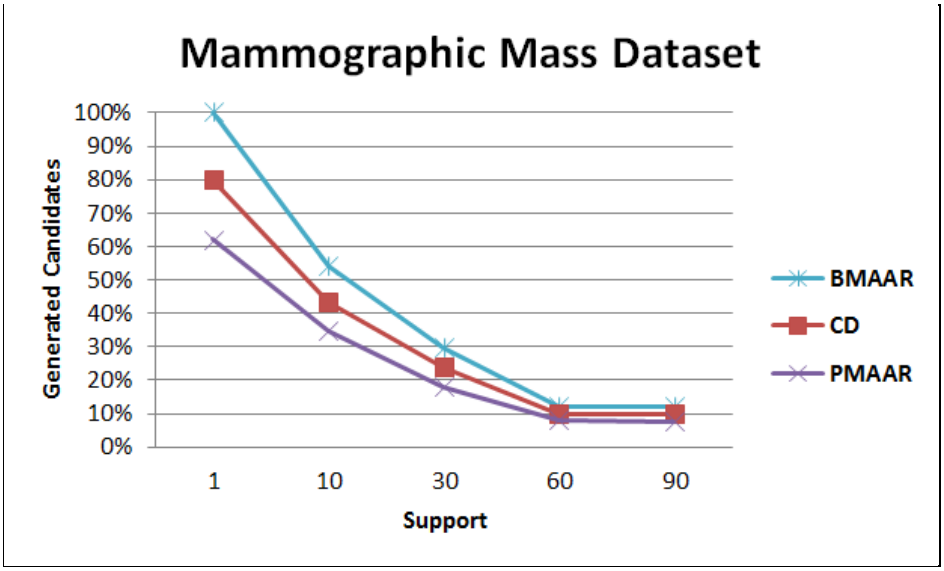

Figure 5.6 Percentage of generated candidate itemsets for mammographic mass dataset

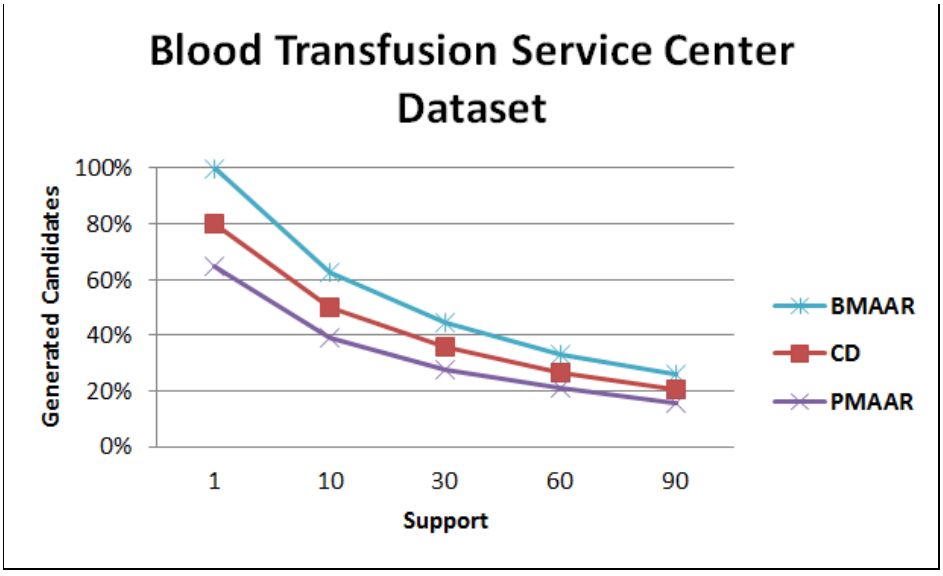

Figure 5.7 Percentage of generated candidate itemsets for data transfusion service center dataset

Results presented in Figure 5. to Figure 5.7 show the following:

1. The number of candidate itemsets generated by PMAAR is less than that generated by BMAAR and CD. This is due to the three pruning techniques included in PMAAR which prunes the number of generated candidate itemsets, consequently, the number of message negotiations between the sites.
2. When the support value increases the number of generated candidate itemsets decreases.
3. When the support value decreases, the difference in the number of generated candidate itemsets increases, which means that PMAAR performs better than other algorithms at low supports.
4. The number of candidate itemsets generated by CD is less than that generated by BMAAR. This is because BMAAR applies the candidate generation process without applying any pruning technique unlike CD which applies Apriori_Prune() function that reduces the number of generated candidate itemsets.

5.9. Performance Evaluation on Benchmark Datasets

The experiments include the implementation of six algorithms against five datasets at five different supports with total of 150 values. Algorithms implemented are the BitTableFI algorithm (BT), the Count Distribution algorithm CD, the Message Passing Interface algorithm (MPI), DMAAR, BMAAR and PMAAR. The five benchmark datasets from UCI machine learning repository are related to different application domains. Datasets are described in Table 3.1. The results obtained are illustrated in Figure 5.8 to Figure 5.12.

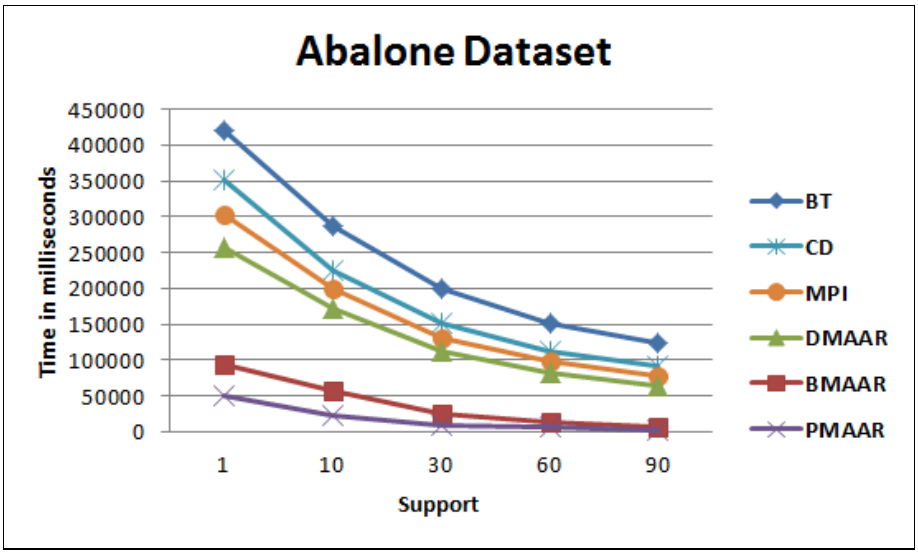

Figure 5.8 Testing algorithms on Abalone Dataset

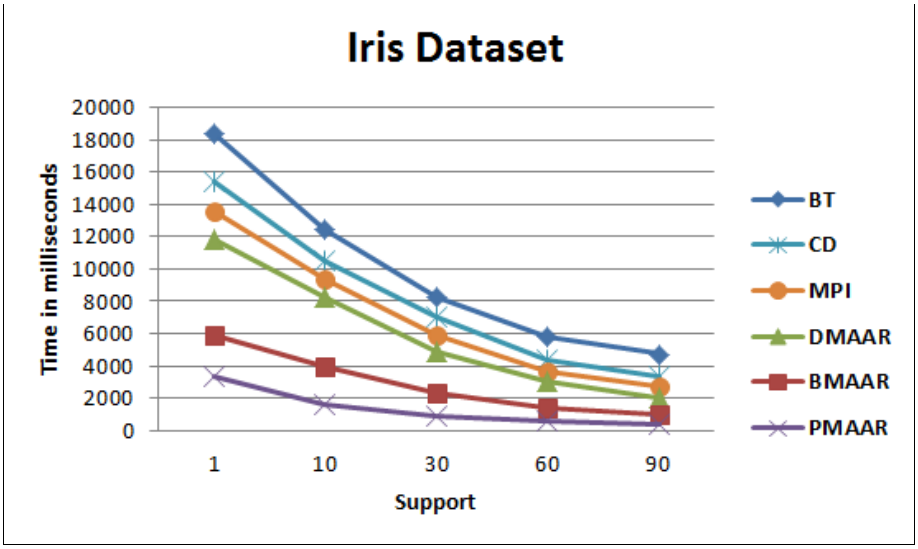

Figure 5.9 Testing algorithms on Iris Dataset

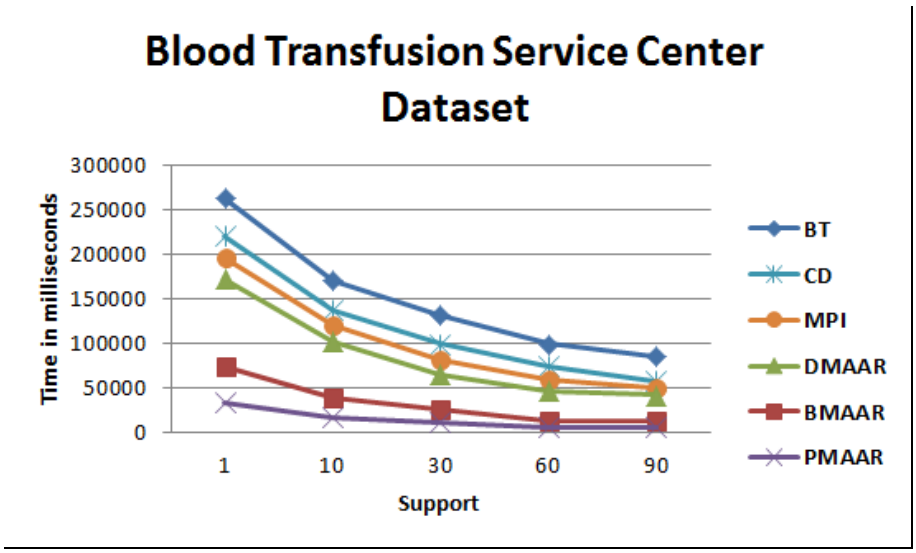

Figure 5.10 Testing algorithms on Blood Transfusion Service Center Dataset

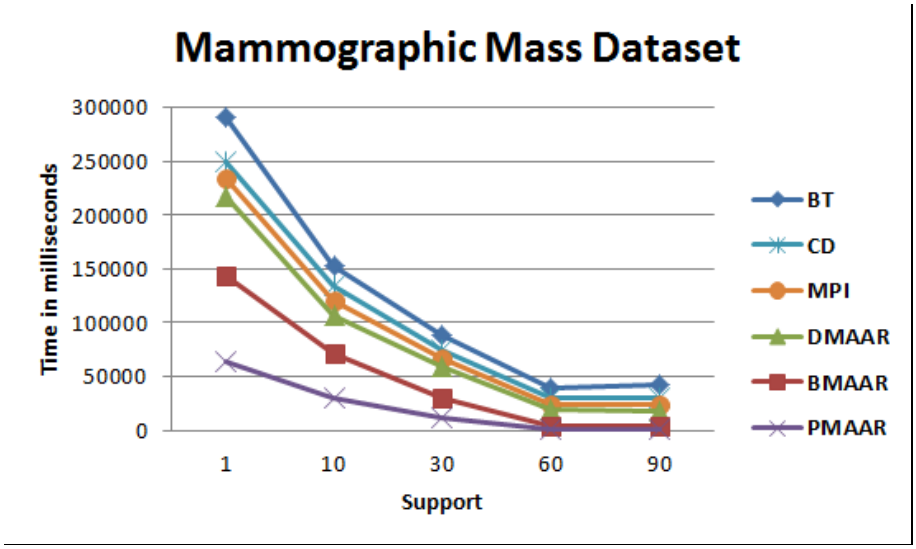

Figure 5.11 Testing algorithms on Mammographic Mas Dataset

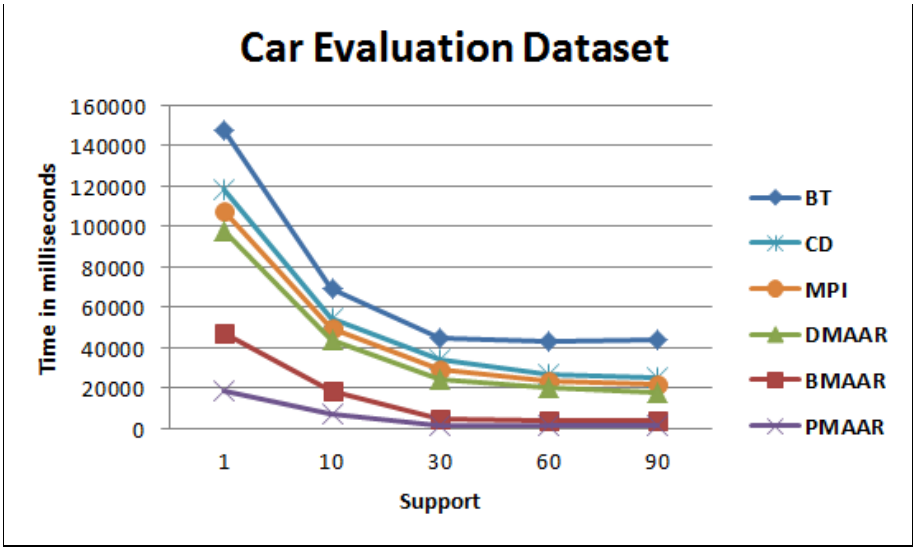

Figure 5.12 Testing algorithms on Car Evaluation Dataset

The comparative study for the performance evaluation of PMAAR with existing algorithm shows the following:

1- Results presented in Figure 5.8 to Figure 5.12 show that the PMAAR outperforms the algorithms CD, MPI, BT, DMAAR and BMAAR.

2- Although low support values are considered as one of the limitations of the Association Rules algorithms, yet, the difference in execution time increases when the support value decreases. This shows that the PMAAR algorithm outperforms other algorithms at low support values.

3- The proposed PMAAR algorithm outperformed other algorithms for the following reasons: (i) PMAAR included three pruning techniques which dramatically reduced the large number of generated candidate itemsets. (ii) PMAAR included three transaction trimming techniques which reduced the large number of database transactions at local sites. (iii) PMAAR used the efficient BitTable data structure which was proved to have better performance and less memory than hash trees data structure. (vi) PMAAR performed two BitWise AND/OR operations for quick candidate itemsets generation and quick support counting. (v) PMAAR implemented a Multi-Agent based solution architecture which avoided the all-to-all broadcasting between distributed sites. (vi) The proper distribution of the mining tasks across the agents maximized the mining tasks parallelism. (vii)The order of the mining processes reduced the waiting time of the Main and the Local Agents.

5.10. Summary and Conclusion

In this chapter, the Pruning Multi-Agent Association Rules algorithm (PMAAR) is presented. The proposed PMAAR algorithm further enhances the performance of BMAAR algorithm.

One of the limitations of the existing association rules algorithms is the large number of generated candidate itemsets which causes a large amount of communication when broadcasted to all other sites (Cheung et al., 2002).

The number of candidates (C_k) generated by Apriori is calculated as follows (Park et al., 1997):

$$\left|C_k\right| = \binom{\left|L_{k-1}\right|}{2} = \frac{\left|L_{k-1}\right|!}{(\left|L_{k-1}\right| - 2)! \times 2!} = \frac{\left|L_{k-1}\right| * \left(\left|L_{k-1}\right| - 1\right)}{2}$$

In order to reduce the number of generated candidate itemsets, PMAAR included three candidate itemsets pruning techniques, namely, Apriori Pruning technique, Distributed Pruning Technique and Global Pruning Technique. Reducing the number of generated candidate itemsets reduces the message negotiations between sites, and consequently, the total time of the mining process.

Experiments have been conducted to evaluate the effect of including the pruning techniques in PMAAR. Results in Figure 5. to Figure 5.7 showed that the number of generated candidate itemsets in PMAAR is less than that of BMAAR and CD.

In order to apply the pruning techniques efficiently, PMAAR allocated the generation process to the Local Agents rather than the Main Agent as proposed in DMAAR and BMAAR. Figure 5.1 and Figure 5.2 showed that the PMAAR maximized the tasks parallelism for the whole mining algorithm and minimized the waiting time of the Main and the Local Agents.

Analytical evaluation showed that the proposed PMAAR algorithm achieves better algorithm complexity Cost (O(n)) when compared with the Count Distribution algorithm ($O(n^2)$) and DMA (less than $O(n^2)$). Analytical evaluation also showed that the message negotiation costs of CD and DMA algorithms are directly proportional to the square of the number of sites unlike PMAAR which is directly proportional to the number of sites only. This shows that PMAAR is more scalable than other algorithms.

Experiments have been conducted to evaluate the overall performance of PMAAR and the existing algorithms. PMAAR was compared with DMAAR, BMAAR, the BitTableFI algorithm (BT), the Message Passing Interface algorithm (MPI) and the Count Distribution algorithm (CD) at different supports on five benchmark datasets from UCI machine learning repository that are related to different application domains. Results presented in Figure 5.8 to Figure 5.12 showed that PMAAR outperformed other algorithms.

In summary, the proposed PMAAR algorithm presented the following contributions:

1- Applying Apriori Pruning technique, Distributed Pruning Technique and Global Pruning Technique reduced the number of generated candidate itemsets, consequently, the number of message negotiation between sites and the time needed for the whole mining process.

2- The proposed PMAAR algorithm reduced the number of database transactions and consequently the time needed for the whole mining process by including three transaction trimming techniques.

3- Applying the efficient Bitable data structure helped to fit the database transactions into the memory of the Main and the Local sites. BitTable data structure was proved to have better performance and less memory than hash trees data structure.

4- For quick candidate itemsets generation and support counting, BMAAR applied the Bitwise AND/OR operations. Candidate itemsets generation and support counting using BitWise operations were proved to be faster than that using the hash trees data structure.

5- The Multi-Agent system framework of PMAAR enabled the agents to operate cooperatively with each other and with other different external agents. This offered a generic platform and a basic infrastructure that can deal with other Data Mining techniques.

6- Messages compliance with the global communication standard, the Foundation for Intelligent Physical Agents (FIPA), enabled the agent cooperation with other standard agents.

7- The proposed solution architecture avoided the all-to-all broadcasting implemented by CD, DMA and FPM algorithms by including a Main Agent which has a global view of all other agents.

8- The sequence and the activity diagrams presented in Figure 5.1 and Figure 5.2 respectively showed that the order of the algorithm processes maximized the parallelism and reduced the waiting time for the Main and the Local Agents. Thus, reducing the total time required for the whole mining process.

9- Analytical calculations presented in 5.7.1 showed that the proposed Multi-Agent based algorithm reduced the algorithm complexity from $O(n^2)$ to $O(n)$.

10- Analytical calculations presented in 5.7.2 showed that the message negotiation cost of PMAAR is less than that of DMA algorithm. Moreover, it is less than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

11- In terms of scalability, the message negotiation cost presented in Section 5.7.2 showed that the costs of CD and DMA algorithms are directly proportion to the square of the number of sites unlike that of PMAAR which is directly proportion to the number of sites. This showed that PMAAR is more scalable than CD and DMA when the number of sites increases.

Chapter Six

6. A Case Study for the early prediction of urinary bladder inflammation disease

6.1. Chapter Overview

Many algorithms have been proposed for the discovery of Association Rules. The efficiency of these algorithms needs to be improved to handle real-world large datasets, especially, when the data is stored in geographically distributed sites. In the previous chapter, the multi-agent based PMAAR algorithm for mining Association Rules in distributed databases was presented. The proposed algorithm has been tested on UCI machine learning repository datasets at different supports and has proved good performance and execution time.

In order to demonstrate the applicability of the proposed algorithm to real world databases, this chapter presents the implementation of the proposed Multi-Agent based approach in distributed medical databases for three hospitals in Egypt. In particular, to generated rules to build a patient diagnostic knowledge base for two chronic diseases. The knowledge base can be used for

the early detection of Inflammation of urinary bladder and Nephritis of renal pelvis origin diseases.

Data preprocessing for the raw data obtained from the three hospitals included the removal of patient's records containing missing values and the conversion of the hospitals raw data into Apriori format, then storing the new format in the hospitals local warehouses.

Two main contributions are presented in this chapter. First, the addition of a Rule Agent to the previously proposed Multi-Agent based algorithm. Second, the implementation of the proposed algorithm on distributed medical databases in order to generate the hidden and the interesting rules inside the data. Using the cooperative agents of the proposed algorithm, data is analyzed and useful Association Rules that can help decision makers are generated.

The proposed model improves the diagnostic knowledge of the medical staff and discovers the previously mentioned diseases based on the minimum number of effective tests. The knowledge base constructed provides accurate medical decisions based on cost effective treatments, thus improving the medical service for the patients. Moreover, it allows the medical staffs to predict the existence or the absence of the diseases for patients.

6.2. The Case Study motivations

Many researchers have adopted different Data Mining techniques for mining Association Rules in medical databases. These techniques have been applied to medical application domains such as skin cancer image data (Chung and Qing, 2001), protein data (Bahamish et al., 2004) and diabetic patients (Dua et al., 2008).

In order to increase the scalability and the flexibility of these techniques, Multi-Agent systems for medical domain have been proposed by many researchers (Xing et al., 2003, Cao, 2009). The proposed techniques included the Multi-Agent systems for patient monitoring such as checking the patient status at real time using Body Area Network (BAN) tools as in (Byung-Mo et al., 2006) or using mobile wireless communication infrastructure as in (Talaei-Khoei et al., 2009) or collecting patients feedback as in (Yupeng et al., 2009).

The previously mentioned techniques provided a rich source of data for other researchers to apply Data Mining in order to discover the hidden patterns that are useful for decision makers. The proposed techniques included the construction of decision support systems in breast cancer domain(Siddiqa et al., 2009) and Schizophrenia disease (Ouali et al., 2003). However, the agents negotiation in these techniques did not comply with the global communication standards such as FIPA (FIPA, May, 2002) to cooperate with other agents.

Healthcare information systems in Egypt provide a wide range of services such as patient admission, scheduling, registration system, medical records system, laboratory information system, etc. However, the increasing trends in the occurrence of diseases present serious problems in providing

suitable healthcare within the existing medical structure. Medical experts and doctors provide health-care diagnosis and address these anticipated problems. However, the large number of patients and their related data cannot be efficiently processed with existing systems, especially, if the patient medical records are distributed in many hospitals and an efficient processing approach is needed for real-time response requirements for critical health situations.

The complexities of healthcare information systems in Egypt besides the huge amount of data collected every day have led to an overgrowing demand for the use of artificial intelligent tools. These tools can be used for data analysis to improve the efficiency of patient treatments and avoid the overall costs by minimizing the risks of false diagnosis. It is important to integrate Artificial Intelligence tools in everyday medical applications. This has motivated many researchers to investigate various distributed Association Rules techniques for mining distributed medical databases.

One of the motivations of this chapter is to extend PMAAR by adding a Rule Agent in order to discover the hidden association rules from the previously generated frequent itemsets. The other motivation is to apply the whole model to the distributed medical databases obtained from the three hospitals in Egypt and to generate useful Association Rules that can help the decision makers.

6.3. PMAAR algorithm with the Rule Agent

6.3.1. Solution Architecture

The solution architecture of the extended model is based on the proposed PMAAR algorithm together with an additional agent named as the Rule agent. The Rule Agent is based on Apriori Algorithm presented in (Agrawal and Srikant, 1994) and adopted in (Hanguang and Yu, 2012). However, some parts of this algorithm were modified to work in distributed and Multi-Agent environment and to deal with the newly proposed bit-table data structure rather than the hash trees used by Apriori. The message negotiation between the Interface Agent, the Main Agent, the Rule Agent and the Local Agents complies with FIPA communication standards. The solution architecture of the proposed algorithm is shown in Figure 6..

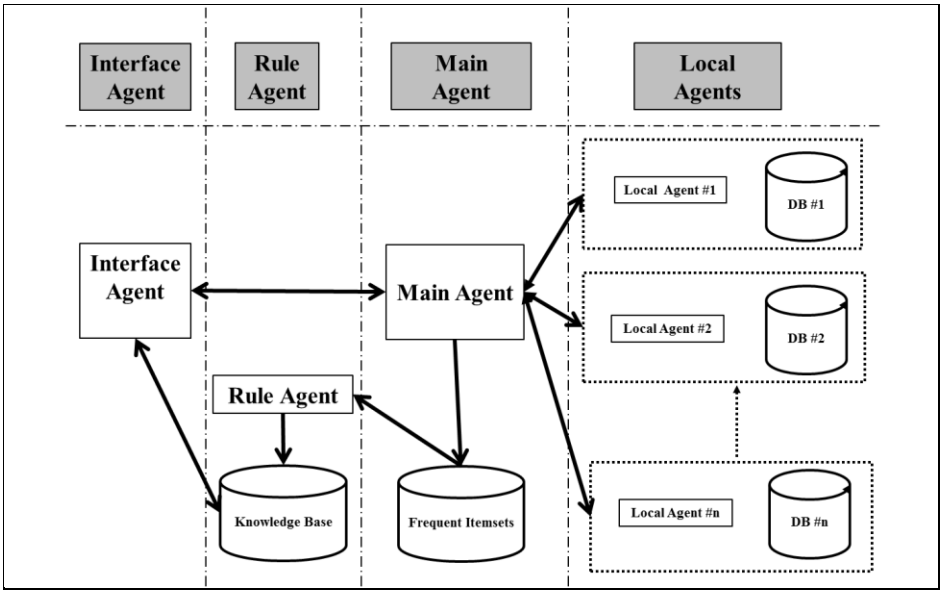

Figure 6.1 The Solution architecture of the proposed model

The rule generation process is presented in Table 6.1

Table 6.1 The rule generation process

Main Agent	Rule Agent
Sends a proposal for the rule generation process including the required confidence. →	
	Sends an “Agree” FIPA message accepting the proposal ←
	The Rule Agent loops on the list of frequent itemsets, and for every non-empty subset (a) of the frequent itemset (L), the following steps are done:
	1. Sends a “Request” FIPA asking for the support value of the itemset (a) ←
Sends an “Inform” FIPA message containing the support value of the itemset (a) →	
	2. Sends a “Request” FIPA asking for the support value of the itemset (L) ←
Sends an “Inform” FIPA message containing the support value of the itemset (L) →	
	1. The Rule Agent calculates the ratio of the support (L) to the support (a).
	2. If this value is greater than or equal to the minimum confidence, a rule in the form of $a \rightarrow (L - a)$ is generated.

The previous steps are iterative until there are no more frequent itemsets.	
	Sends an “inform” FIPA message with the finish of the rule generation process

6.4. Message negotiation between agents

The message negotiation between agents is described as follows:

1. The Interface Agent accepts the required support and confidence from the user and sends them to the Main Agent which in turn sends them to the Local Agent in each of the local hospitals.
2. The local Agent together with the Main Agent start the distributed mining process through some negotiation FIPA messages, and finally it generates the global large frequent itemsets.
3. When the global large itemsets are generated, the Main Agent sends a FIPA message with a "propose per formative" to the newly implemented Rule Agent as follows:

   (Propose
   :sender (agent-identifier :name main_agent)
   :receiver (set (agent-identifier :name rule_agent))
   :content "Start mining with confidence = 80%"
   :reply-with rule_generation_proposal)
4. The Rule Agent replies with an "agree performative" to the Main Agent as follows:

   (**Agree**
   :sender (agent-identifier :name rule_agent)
   :receiver (set (agent-identifier :name main_agent))
   :content "proposal approved for rule generation"
   :in-reply-to rule_generation_proposal)
5. The Rule Agent starts the generation of the rules according to the required support and confidence.
6. The Rule Agent loops on the list of frequent itemsets generated by

the Main and the Local Agents.

7. For each frequent itemset, the Rule Agent starts finding all non-empty subsets of this frequent itemset (All subsets of the itemset are considered since the Rule Agent generates all rules with multiple consequents).
8. Loop for every subset (a) for the itemset (L)
   The Rule Agent sends a "Request performative" FIPA message to the Main Agent to get the support of the itemset (a) as follows:

   (**Request**
   :sender (agent-identifier :name rule_agent)
   :receiver (set (agent-identifier :name main_agent))
   :content "get itemset support for itemset (a)"
   :reply-with itemset_a).
9. The Main Agent sends the support value for the itemset (a) to the Rule Agent using a FIPA message with an "Inform performative" as follows:

   (**Inform**
   :sender (agent-identifier :name main_agent)
   :receiver (set (agent-identifier :name rule_agent))
   :content "The value of the support for itemset (a)"
   :in-reply-to itemset_a)
10. The Rule Agent sends a "Request performative" FIPA message to the Main Agent to get the support of the itemset (L) as follows:

    (**Request**
    :sender (agent-identifier :name rule_agent)
    :receiver (set (agent-identifier :name main_agent))
    :content "get itemset support for itemset (L)"
    :reply-with itemset_L).
11. The Main Agent sends the support value for the itemset (L) to the Rule Agent using a FIPA message with an "Inform performative" as follows:

    (**Inform**
    :sender (agent-identifier :name main_agent)
    :receiver (set (agent-identifier :name rule_agent))

:content "The value of the support for itemset (L)"
:in-reply-to itemset_L).

12. The Rule Agent calculates the ratio of the support (L) to the support (a), if this value is greater than or equal to the minimum confidence supplied by the Interface Agent, the Rule Agent generates a rule in the form of $a \rightarrow (L - a)$.
13. If there are more non-empty subsets of the itemset (L), then go to step 8.
14. If there are more frequent itemsets, then go to step 7.
15. After the medical rules are generated, the Rule Agent sends a FIPA message with an "Inform performative" to the Main Agent as follows:

(**Inform**
:sender (agent-identifier :name rule_agent)
:receiver (set (agent-identifier :name main_agent))
:content "rules are successfully generated"
:in-reply-to rule_generation_proposal)

16. The Main Agent sends all rules generated to the Interface Agent for graphical representation to the doctors.

6.5. Applying the proposed model to the medical database

Inflammatory lesions of urinary bladder and kidneys exist in 3 - 5% of the population. Women suffer more often due to anatomic peculiarities of their urogenital system. There are conditions to transfer adjacently the bacterial contamination from vagina. Pregnancy is also a factor due to compression of urethra by the fetus and retention of urine. By advancing of age and appearance of prostate adenoma in some males, infections of urinary bladder and kidney become more frequent in them. Nephrolithiasis is also the reason for development of inflammatory of urinary bladder and kidneys. When the inflammatory process has covered the kidneys, the status of patients is significantly damaged, and it is possible to develop more severe complications like inflammation of adjacent tissues or acute renal failure (Nkudic, 2006). It is necessary to carry out many examinations for the patients under investigation. These examinations may be costly and time consuming. Our goal is to construct a model that can discover the effective minimum number of tests to identify the previously mentioned diseases. Moreover, the model should construct a knowledge base that can help doctors in future prediction, thus saving time and effort which is very critical for the patients.

6.5.1. Data gathering and preprocessing

The medical data used by the proposed Multi-Agent based algorithm contains patients' symptoms related to the Inflammation of urinary bladder and Nephritis of renal pelvis origin diseases. These symptoms are Burn or itch of urethra and swell for its outlet, Lumbar pain, Micturition pains, Occurrence of nausea and Urine pushing (continuous need for urination). We should note that

although the algorithm was implemented on the aforementioned diseases, yet, it can be applied to other items related to the patient medical records.

Data was obtained anonymously from three distributed hospitals in Egypt without patient's personal details. 55,993 medical records have been obtained from the first hospital, 59,367 medical records have been obtained from the second hospital and 56,000 medical records have been obtained from the third hospital with total of 171,360 medical records. Raw data for the patients' medical records are presented in Figure 6.2.

BURN	LUMBAR	MICTURITION	NAUSEA	URINE_PUSHING	URINARY_BLADDER	RENAL_PELVIS
no	yes	no	no	no	no	no
yes	no	yes	no	yes	yes	no
no	yes	no	no	no	no	no
yes	no	yes	no	yes	yes	no
no	yes	no	no	no	no	no
no	yes	no	no	no	no	no
yes	no	yes	no	yes	yes	no
no	yes	no	no	no	no	no
yes	no	yes	no	yes	yes	no
yes	no	yes	no	yes	yes	no

Figure 6.2 Raw data for the patients' medical records before preprocessing

One of the most important stages in data preprocessing is removing the missing data which is considered as a common problem for data quality in real world databases (Peyre et al., 2011). Prevention and treatment methods of missing values in the medical domain have been proposed in (Little et al., 2012). The medical data records containing missing values (about 8% of the total amount of data) were removed to improve the quality of the data. The symptoms and the diseases were extracted from the remaining data and were stored in a separate lookup table as in Figure 6.3.

item_code	item_name
A	No Burn /itch of urethra & No swell for its outlet
B	Burn, itch of urethra and swell for its outlet
C	No Lumbar pain
D	No Micturition pains
E	No Occurrence of nausea
F	No Urine pushing (No cont. need for urination)
L	Lumbar pain
M	Micturition pains
N	Occurrence of nausea
U	Urine pushing (continuous need for urination)
W	No Inflammation of urinary bladder
X	No Nephritis of renal pelvis origin
Y	Inflammation of urinary bladder
Z	Nephritis of renal pelvis origin

Figure 6.3 The symptoms and diseases codes and names

Finally, the remaining data was converted to the Apriori-like format as in Table 6.2 and was loaded into the hospital local data warehouse.

Table 6.2 Sample of the medical data after being converted to the Apriori like format

TID	Transaction
1	A-L-D-N-F-W-Z
2	B-C-M-N-U-Y-X
3	A-L-D-N-F-W-X
4	B-C-M-N-U-Y-Z
5	B-L-D-N-F-W-X

The distributed structure of the medical data warehouses is presented in Figure 6.4.

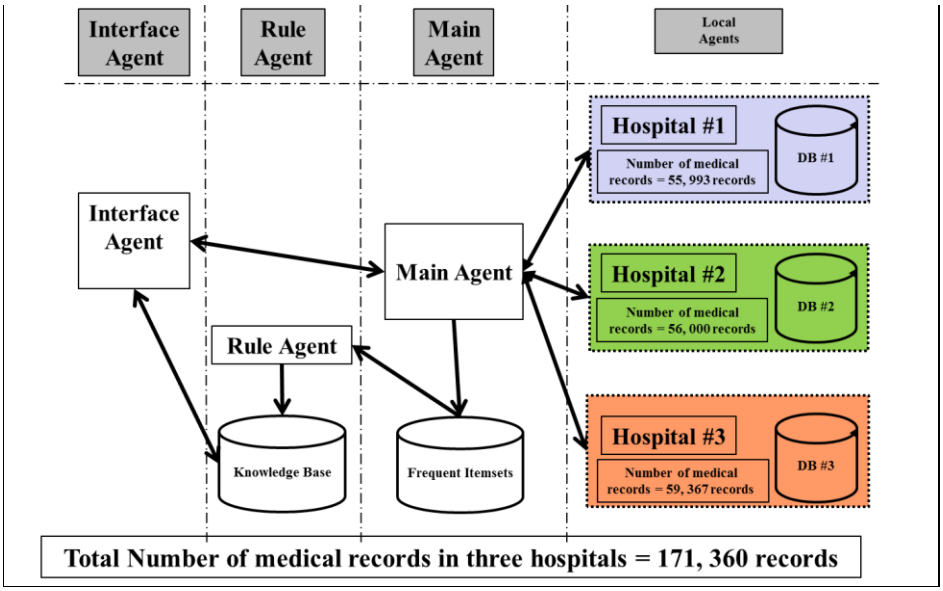

Figure 6.4 Distributed structure of the medical records in the hospitals

6.5.2. **Model Validation**

In order to validate the model, ten runs of 10-fold cross-validation technique were conducted. The cross-validation technique is a widely accepted standard way for predicting the error estimates of a learning technique given a simple fixed sample of data (Airola et al., 2011).

Local database in every site was randomly partitioned into ten partitions. Nine of these partitions were used as training data, and the remaining partition was used as the validation data for the model.

The cross-validation process was repeated ten times (folds) where each partition was used exactly once as the validation data. In each fold, the predicted output values obtained from the model was compared with the real

output values in order to calculate the error rate (in percentage). To produce a single error rate, all error rates have been averaged. The goal of conducting the cross-validation and calculating the error was to estimate the degree of model fitness with the data.

The following evaluation measures have been widely used to assess the prediction models (Kelley and Lai, 2011, Garofalo et al., 2011, Diel et al., 2011, Leung et al., 2011):

- **Mean Absolute Error (MAE)** $= \frac{1}{N}\sum_{i=1}^{N}\left(\left|PV_i - RV_i\right|\right)$
- **Mean Squared Error (MSE)** $= \frac{1}{N}\sum_{i=1}^{N}\left(PV_i - RV_i\right)^2$
- **Root Mean Squared Error (RMSE)** $= \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(PV_i - RV_i\right)^2}$
- **Positive Predictive Value (PPV)** $= \frac{TP}{TP + FP}$
- **Negative Predictive Value (NPV)** $= \frac{TN}{TN + FN}$

Where PV_i is the predicted output value for the record *i*, RV_i is the real output value the record *i*, *N* is the total number of comparison records, *TP* is the number of true positives, *TN* is the number of true negatives, *FP* is the number of false positives, *FN* is the number of false negatives.

The mean absolute error (MAE) measures the arithmetic average value of all prediction errors. It is calculated by summing the absolute error values between the real and the predicted values divided by the number of comparison records. MAE is considered as the most common measure of the average error.

The calculation of the Root Mean Squared Error (RMSE) in illustrated in two steps. First, the Mean Squared Error (MSE) is calculated as the squared difference between the real and predicted values divided by the number of comparison records. Second, the RMSE is calculated as the square root of the MSE. It should be noted that RMSE is always greater than or equal to MAE. RMSE detects the error variations. High RMSE indicates large variations and vice versa. If RMSE is closer to MAE, this means the model is consistent (Willmott and Matsuura, 2005, Witten and Frank, 2005).

In the Confusion Matrix (also called contingency table) (Bunkar et al., 2012), the Positive Predictive Value (PPV) indicates the percentage of the positive results that is correctly diagnosed as true positives by the model. PPV is considered as a critical measure for the model performance as it measures the probability that a positive test reflects the underlying condition being tested for. The same is applied for the Negative Predictive Value.

Evaluation measures obtained due to applying the 10-fold cross-validation technique are presented in Table 6.3, Figure 6.5 and Figure 6.6.

Table 6.3 Evaluation measures due to applying PMAAR to the medical data

Evaluation Measures	inflammation of urinary bladder disease	Nephritis of Renal pelvis origin disease	Both Diseases
MAE	0.03739	0.0899	0.06365
MSE	0.03739	0.0899	0.06365
RMSE	0.19338	0.29984	0.25229
PPV	0.98001	0.97456	
NPV	0.95	0.8649	

		Real data		
		Positive	Negative	
Proposed Model	Outcome Positive	True Positive (TP)=70533	False Positive (Type I Error) (FP)=1439	**Positive Predictive value** = TP / (TP + FP) = 70533 / (70533 + 1439) = 0.98001= 98%
	Outcome Negative	False Negative (Type II Error) (FN)=4969	True Negative (TN)=94419	**Negative Predictive value** = TN / (TN +FN) = 94419 / (94419 + 4969) = 0.95= 95%

Figure 6.5 Positive and Negative Predictive values for PMAAR when applied to the medical data for the inflammation of urinary bladder disease

		Real data		
		Positive	Negative	
Proposed Model	Outcome Positive	True Positive (TP)=68833	False Positive (Type I Error) (FP)=1797	**Positive Predictive value** = TP / (TP + FP) = 68833/ (68833+ 1797) = 0.97456= 97%
	Outcome Negative	False Negative (Type II Error) (FN)=13609	True Negative (TN)=87121	**Negative Predictive value** = TN / (TN +FN) = 87121/ (87121+ 13609) = 0.8649= 86%

Figure 6.6 Positive and Negative Predictive values for PMAAR when applied to the medical data for the Nephritis of renal pelvis origin disease

Table 6.3 shows that the total MSE for both diseases is (0.06365). However, deeper analysis shows that the model is more accurate in case of the inflammation of urinary bladder disease (MSE=0.03739) when compared to the Nephritis of Renal pelvis origin disease (MSE=0.0899) due to lower MSE.

Deeper analysis using the Confusion Matrix has been conducted (see Figure 6.5 and Figure 6.6) in order to explore and explain why the MSE of the Nephritis of Renal pelvis origin disease is higher than that of the inflammation of urinary bladder disease.

The relatively small Negative Predictive Value (NPV = 86%) indicates that many of the negative results for the Nephritis of Renal pelvis origin disease are false negatives. Thus, it will be necessary to follow up any negative result with a more reliable test to obtain more accurate assessment as to whether the disease is present. The strength of the Nephritis of Renal pelvis origin disease is in its Positive Predictive Value (PPV=95%), which means that if the predicted output is positive; this gives us a high confidence that the patient has the disease.

6.5.3. **Rules generation**

The experiments involved the implementation of the algorithm using different support and confidence values. Increasing the value of the confidence led to a small number of generated rules. On the other side, decreasing the value of the confidence led to huge number of rules, some of them were neither relevant nor interesting. Much research has been conducted to eliminate the huge number of redundant (ZAKI, 2004) and non-interesting (Xu and Li, 2005)

Association Rules. Although we did not tackle the problems of mining interesting and non-redundant rules as they are out of our scope in this research, yet, we found that the best confidence for our medical data was 80%. Table 6.4 describes some of the medical rules generated by Rule Agent at confidence = 80%

Table 6.4 Sample of the generated medical rules (at confidence = 80%)

Serial	Antecedent	→	Consequent	Confidence
1	A-D-E	→	X	100
2	A-D	→	X	88.37
3	D-E-L	→	W	100
4	F-C	→	W	86.48
5	D-M	→	X	93.21
6	L-U	→	Z	100
7	D-N	→	Y	95.54
8	E-C	→	Z	97.58
9	M-U	→	Y	100
10	M-N	→	W	84.72
11	D-Z	→	Y	91.24
12	C-Y	→	Z	95.89
13	C-E-X	→	Y	82.67
14	…….	→	….	……

6.5.4. **Analysis and Findings**

Association rules generated by the Rule agent are stored in the knowledge base in the form:

Antecedent Itemset → Consequent Itemset

With Confidence = c

When PMAAR was applied to the medical data to discover the hidden medical association rules, large amount of rules have been generated.

These rules were classified into three categories. a) Rules where the diseases are part of the antecedent imply that a disease can indicate a symptom. These rules are not useful for the domain experts and were omitted from the knowledge base (Rules 11 and 12). b) Rules where the confidence is less than 100 have been saved in the knowledge base to be presented to the decision maker (Rules 2, 4, 5, 7 8, 10 and 13). However, it is worth mentioning it is up to the medical expert to rely on the confidence of the rule in assessing the existence or the absence of the disease especially in critical patient situation. c) Rules where the confidence is equal to 100 have been saved into the knowledge base to be presented to the decision maker (Rules 1, 3, 6 and 9). When these rules were further analyzed, it was found that they can identify the existence or the absence of the diseases based on the minimum number of effective tests. The following is a sample of these rules:

1. L-U → Z

   IF Lumbar pain

   AND Urine pushing (continuous need for urination)

   THEN Nephritis of renal pelvis origin

   WITH CONFIDENCE 100%

2. M-U → Y

   IF Micturition pains

   AND Urine pushing (continuous need for urination)

   THEN Inflammation of urinary bladder

   WITH CONFIDENCE 100%

3. D-E-L → W

   IF No Micturition pains

   AND No Occurrence of nausea

   AND Lumbar pain

   THEN No Inflammation of urinary bladder

   WITH CONFIDENCE 100%

4. A-D-E → X

   IF No Burn /itch of urethra

   AND No Micturition pains

   AND No Occurrence of nausea

   THEN No Nephritis of renal pelvis origin

   WITH CONFIDENCE 100%

From the medical point of view, the model provided many benefits.

The model has constructed a knowledge base containing many interesting rules that can be very useful for the medical experts. This knowledge base architecture can improve the real time response for the health system by identifying the features that indicate the presence or the absence of the disease. For instance, rule (1) and (2) can help to identify the presence of the Inflammation of urinary bladder AND Nephritis of renal pelvis origin respectively based on the minimal number of symptoms/tests for the patient, thus reducing time to identify the disease and reducing the cost of examining other tests.

On the other side, rules (3) and (4) identify the features that indicate the absence of the disease. Thus, if these symptoms do not exist, patients do not need to examine further symptoms or tests.

The model has presented a diagnostic platform that can help in investigating patients diagnosis based on symptoms or tests done in different hospitals or within the hospital departments. Based on the previously discovered Association Rules, the model can help doctors in the early prediction for the existence or the absence of the disease based on the minimum number of effective tests. Thus, the model minimizes the number of tests needed by the doctors, thus minimizing the effort, the cost and the time needed for the patients to do other tests or to be checked for other symptoms.

6.5.5. **Performance Evaluation**

This section presents the evaluation of the system from the technical point of view. The experiments included the implementation and testing of six algorithms on three distributed real world medical databases for the inflammation of urinary bladder disease. These algorithms are BT, CD, MPI, DMAAR, BMAAR and PMAAR. The test bed used was Windows 7 Enterprise edition, 64-bit operating system, Intel Core™2 Duo CPU and 4 gigs ram. Figure 6.7 presents the implementation of the six algorithms at five different supports.

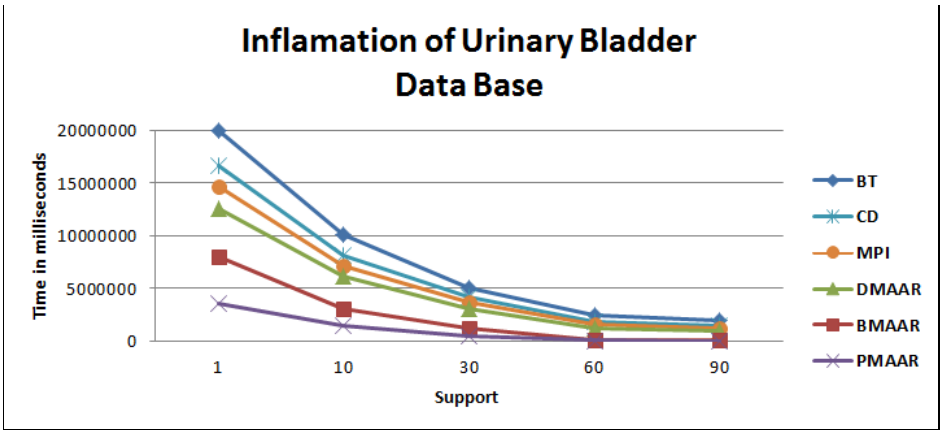

Figure 6.7 Implementation of the algorithms on Inflammation of Urinary Bladder Data at different supports

From Figure 6.7 we can observe that the proposed PMAAR algorithm outperforms the previously implemented algorithms BT, CD, MPI, DMAAR and BMAAR. Moreover, this performance increases when the support values decrease.

The performance of PMAAR has been achieved due to the use of the BitTable data structure, the Bitwise And/OR operations for quick candidate generation and itemsets support counting, the three transaction trimming techniques and the three pruning techniques. Moreover, the avoidance of the all-to-all broadcasting led to the decrease of the complexity and the message negotiation costs.

6.6. Summary and Conclusion

Association rules algorithms needs to be evaluated when applied to real world datasets. In this chapter, the Pruning Multi-Agent Association Rules Algorithm (PMAAR) was applied to real world medical databases obtained from three hospitals in Egypt. The medical data is related to the inflammation of urinary bladder and the Nephritis of renal pelvis origin diseases.

In order to mine the Association Rules from the generated frequent itemsets, a Rule Agent was added to the previously proposed Main Agent, Interface Agent and Local Agents. The main purpose of the Rule Agent was to discover the useful medical information that is hidden inside the medical data.

The goals of the implementation involved the discovery of the medical Association Rules inside the medical databases, constructing a medical knowledge base for the extracted medical rules, identifying the minimum effective number of tests for the Inflammation of urinary bladder and the Nephritis of renal pelvis origin diseases, predicting the existence or the absence of the diseases for future patients, improving the diagnostic knowledge for the doctors and improving the quality of clinical decision making.

The Pruning Multi-Agent Association Rules Algorithm (PMAAR) has been tested against the BitTableFI algorithm (BT), the Count Distribution algorithm (CD), Message Passing Interface algorithm (MPI), DMAAR and BMAAR on the medical distributed databases. Experiments have been conducted at five different supports. Results presented in Figure 6.7 showed that PMAAR outperformed other algorithms.

Chapter Seven

7. Conclusion and Future Work

7.1. Summary and Conclusion

The research investigated the Association Rules mining techniques in distributed databases. The objective of the research was to mine the distributed databases efficiently without loading the data into centralized location so as not to violate data privacy nor impose network overheads. For this reason, an incremental approach for building an efficient multi-agent based algorithm for mining geographically distributed databases has been proposed.

Much research has focused on enhancing the performance of Association Rules algorithms by enhancing the data structure, the candidate generation process, the candidate counting process, etc. However, little research focused on defining the list of performance criteria that can be considered as standard measures for the efficiency of the Association Rules algorithms.

For this reason, the list of the performance criteria that affects the overall performance for the Association Rules algorithms has been collected from the literature and proposed in Chapter 1. In order to create the performance criteria list, the following activities have been achieved:

(i) Techniques such as Association Rules (AR), Data Mining (DM) and Multi-Agent Systems (MAS) have been studied.

(ii) Previous work related to the implementation of these techniques on distributed databases has been studied.

(iii) Extensive literature review for existing distributed Association Rules algorithms has been conducted.

(iv) The strength and the weakness of existing distributed Association Rules algorithms have been collected and evaluated to build the performance criteria affecting the efficiency of the distributed Association Rules algorithms.

The list of performance criteria for the distributed Association Rules algorithms (presented in Section 1.5) can provide lots of benefits:

(i) It defines the performance standards by which the distributed Association Rules can be evaluated. This provides the researchers with a target or a goal to strive for.

(ii) The efficiency of the existing and the future Association Rules algorithms can be measured by evaluating the performance criteria. This can help comparing the efficiency of different Association Rules algorithms.

(iii) The performance criteria provided a roadmap for the research regarding the features that are to be enhanced in order to improve the proposed algorithm and to make it more efficient.

The review of the state of the art literatures indicated that applying Data Mining techniques to distributed databases imposes a number of challenges. These challenges include:

(i) Avoiding the all-to-all broadcasting cost between sites.

(ii) Minimizing the waiting time of the distributed sites.

(iii) Maximizing the tasks parallelism of the mining processes.

(iv) Minimizing the algorithm complexity and the message negotiation costs.

(v) Flexibility of the Association Rules algorithm to the changes in requirements.

(vi) Algorithm compliance with the global communication standards.

(vii) Scalability of the algorithm with the increase in the number of sites.

The proposed Distributed Multi-Agent Association Rules algorithm (DMAAR) presented in Chapter 3 employed a multi-agent based framework and a standard communication mechanism to overcome these challenges.

DMAAR avoided the all-to-all broadcasting between distributed sites by including a Main Agent as the main controller with a global view of all other agents.

Analytical calculations presented in Section 3.6.1 showed that the proposed Multi-Agent based algorithm reduced the algorithm complexity from $O(n^2)$ to $O(n)$. Analytical calculations also showed that the message negotiation cost of DMAAR is less than that of DMA algorithm and is less than that of the Count Distribution algorithm when the number of sites is greater than or equal to three.

The appropriate order and the distribution of the mining tasks minimized the waiting time and maximized the tasks parallelism. The research presented in this thesis used the sequence and the activity diagrams presented in Figure 3.2 and Figure 3.3 respectively to model the interactions between different mining tasks and to illustrate the parallelism between the mining processes.

In terms of scalability, the message negotiation cost presented in Section 3.6.2 showed that the costs of CD and DMA algorithms are directly proportion to the square of the number of sites unlike that of DMAAR which is directly proportion to the number of sites. This showed that DMAAR is more scalable than CD and DMA when the number of sites increases.

Messages between agents are compliant to the global communication standard, the Foundation for Intelligent Physical Agents (FIPA), thus enabling the agent cooperation with other standard agents.

DMAAR algorithm generates global and local frequent itemsets unlike CD, PDM, DMA and FPM which generate global frequent itemsets only.

The comparative study for the performance evaluation of DMAAR with existing algorithms on five different benchmark datasets related to different domains showed that DMAAR achieved better performance and execution time. Results have been presented in Figure 3.4 to Figure 3.8.

However, the proposed Distributed Multi-Agent Association Rules algorithm (DMAAR) had the following four limitations:

(i) The large number of database transactions which increases the number of the mining iterations and consequently increases the total time needed for the mining process.

(ii) The memory problems due to the huge amount of transactions at the main and local sites.

(iii) The cost of generating the candidate itemsets

(iv) The cost of counting the generated candidate itemsets.

To address the above limitations, the BitTable Distributed Multi-Agent Association Rules algorithm (BMAAR) has been proposed in Chapter 3. BMAAR is the enhanced version of DMAAR.

In order to tackle the first problem, BMAAR applied three transaction trimming techniques, namely Direct Hashing and Pruning proposed in (Park et al., 1997), the proposed Grouping Identical Transactions technique and the Non Frequent Itemset Removal Technique proposed in (Park et al., 1997) which significantly reduced the number of database transactions.

Experiments were conducted to show the effect of including the three transaction trimming techniques in the proposed algorithm. The total number of transactions before and after applying the three transactions trimming techniques has been calculated and compared with the existing algorithms. Section 4.10 and the results obtained in Figure 4.3 to Figure 4.7 showed that the average reduction in the database transactions due to the use of the transaction trimming techniques averages between 40% and 60%.

In order to tackle the second problem, BMAAR used the efficient BitTable data structure proposed in (Dong and Han, 2007). The BitTable data structure

was proved to have better performance and less memory than hash trees data structure used by CD, PDM, DMA and FPM.

In order to tackle the third and the fourth problems, BMAAR algorithm applied two BitWise AND/OR operations proposed in BitTableFI algorithm (Dong and Han, 2007) for quick candidate itemsets generation and quick support counting.

In order to evaluate the effect of applying the BitTable data structure and the Bitwise AND/OR operations on the performance of the proposed algorithm, a comparative study for the performance evaluation of BMAAR and the existing algorithms on five different benchmark datasets related to different domains was conducted. Results presented in Figure 4.8 to Figure 4.12 showed that BMAAR achieved better performance and execution time.

Analytical evaluation presented in Section 4.11showed that the algorithm complexity and the message negotiation costs of BMAAR are the same as DMAAR.

However, the proposed BitTable Multi-Agent Association Rules algorithm (BMAAR) did not address the problem of the generation of large number of candidate itemsets. This is a very time consuming process. Moreover, when the number of generated candidate itemsets increases, the time needed to count their supports increases.

In order to tackle this problem, the Pruned Multi-Agent Association Rules algorithm (PMAAR) which is considered as further improvement for BMAAR has been presented in Chapter 5.

PMAAR applied three pruning techniques, namely, Global Pruning Technique proposed in (Cheung et al., 2002), Distributed Pruning Technique proposed in (Cheung et al., 2002) and Apriori Pruning Technique proposed in (Agrawal and Shafer, 1996) in order to reduce the number of generated candidate itemsets. Moreover, the proposed PMAAR algorithm allocated the generation process to Local Agents to apply the pruning techniques efficiently rather than generating the candidate itemsets by the Main Agent as in DMAAR and BMAAR.

A comparative study to evaluate the effect of applying the pruning techniques has been conducted. Results presented in Figure 5. to Figure 5.7 showed that the number of generated candidate itemsets in PMAAR is less than that of BMAAR and CD. This implied that the pruning techniques presented in this chapter helped in reducing the number of generated candidate itemsets, consequently, the number of message negotiations between sites and the time needed for the whole mining process.

In spite of the reallocation of the mining tasks to different agents, yet, the order and the distribution of these tasks in the sequence and the activity diagrams (Figure 5.1 and Figure 5.2 respectively) showed that the proposed PMAAR algorithm minimized the waiting time of the Main and the Local Agents and maximized the tasks parallelism of the whole mining algorithm.

Similar to BMAAR and DMAAR presented in Chapter 3 and Chapter 4 respectively, Section 5.7.1 showed that PMAAR reduced the algorithm complexity from $O(n^2)$ to $O(n)$.

Analytical calculation presented in Section 5.7.2 showed that the message negotiation costs for Count Distribution and DMA algorithms are in order of the square of the number of sites unlike PMAAR which is in order of the number of sites only. This showed that PMAAR is more scalable than Count Distribution and DMA algorithms when the number of sites increases.

A comparative study to evaluate the overall performance of PMAAR and the existing algorithms was conducted. PMAAR was compared with DMAAR, BMAAR, the BitTableFI algorithm (BT), the Message Passing Interface (MPI) and the Count Distribution algorithm (CD) at different supports on five benchmark datasets from UCI machine learning repository that are related to different application domains. Results presented in Figure 5.8 to Figure 5.12 showed that PMAAR outperformed other algorithms.

Association rules algorithms need to be evaluated when applied to real-world large datasets, especially, when these datasets are stored in geographically distributed sites.

In order to demonstrate the capabilities of the proposed PMAAR algorithm, Chapter 6 presented the implementation of the algorithm on real world distributed medical databases related to three hospitals in Egypt. Medical data is related to the Inflammation of urinary bladder and the Nephritis of renal pelvis origin diseases. A Rule Agent was added to the existing agents in order to discover the hidden Association Rules in the medical databases. The proposed Rule Agent was based on Apriori Algorithm with some modifications to work in distributed and multi-agent environments.

From the medical point of view, the extraction of the medical Association Rules provided lots of benefits:

(i) Building a medical knowledge base for the extracted medical rules.

(ii) Identifying the minimum effective number of tests for the Inflammation of urinary bladder and the Nephritis of renal pelvis origin diseases.

(iii) Predicting the existence or the absence of the diseases for future patients.

(iv) Improving the diagnostic knowledge for the doctors.

(v) Reducing the real time response for the health system.

(vi) Improving the quality of clinical decision making.

From the technical point of view, the performance evaluation for PMAAR algorithm has been conducted. PMAAR has been compared with the BitTableFI algorithm (BT), the Count Distribution algorithm (CD), the Message Passing Interface algorithm (MPI), DMAAR and BMAAR on the medical distributed databases. Experiments have been conducted at five different supports. Results presented in Figure 6.7 showed that PMAAR outperformed other algorithms.

7.2. Future Work

In this section, we present some suggestions for the future work based on the research outcomes. The following are some suggestions that can extend our model further:

7.2.1. Frequent Pattern mining for weighted items

Frequent pattern mining plays a very important role in mining Association Rules in huge amounts of data. The main objective of the frequent pattern mining is to mine associations, correlations and other hidden relationships between data. Frequent itemsets are those who are available in a dataset with support count greater than the minimum threshold supplied by the user. For instance, assume that the milk and bread items are frequently bought together. The two items are considered as frequent itemsets and can be part of an association rule.

However, traditional algorithms including the proposed PMAAR algorithm presented in the research did not consider the different semantic significances (weights) of these items. In real world databases, finding frequent patterns shields the knowledge related to the low frequent, yet, high valued (weighted) patterns. For example, in a business database, knowledge about gold watches with low frequency and high value is not discovered compared to items like pens which have high frequency, yet, low value. This problem has been proposed in (Jie and Yu, 2011, Chang, 2011).

Future research can involve: (i) Assigning weights to the database items. (ii) Investigating the enhancement of the proposed multi-agent based PMAAR

algorithm to discover the hidden Association Rules taking into account the weights of the items.

7.2.2. **Fuzzy Association rules**

One of the limitations of the traditional Association Rules techniques is that they cannot be applied to all kinds of data. Extracted rules can be discovered only from binary data where the item either exists in the transaction (1) or does not exist (0). Mining quantitative data requires including other techniques to convert these data into crisp values. The straight forward solution is to split each quantitative attribute into intervals. Every single value falls into one of the intervals depending on its value and the range of the intervals. The solution converts the quantitative data into binary data. However, this can lead to incorrect estimations for the values that are very close to the borders.

For this reason, fuzzy Association Rules techniques were investigated (Shen and Liu, 2012, Bai et al., 2012) where items can have a partial membership in more than one interval using a membership function and fuzzy set operations in order to calculate the quality measures of the discovered rules.

7.2.3. **Elimination of redundant rules**

One of the problems of the Association Rules technique is that it extracts overwhelming number of Association Rules that can be very confusing for decision makers. Large number of these extracted rules is redundant. Redundant Association Rules affect the quality of the presented knowledge and

reduce the time needed to manipulate or use the discovered rules. Previous work presented different approaches to solve this problem (Gupta et al., 2012, Vo and Le, 2011, Shaw et al., 2009, Li et al., 2009).

Future research can involve the addition of a new agent to the proposed PMAAR algorithm for eliminating the redundant rules in order to improve the quality of the generated rules.

7.2.4. **Association Rules ranking**

Techniques used to eliminate redundant rules are based on the support and the confidence of the itemsets. The goal of applying these techniques is to eliminate the large number of redundant rules. However, in business applications, even after removing redundant rules, only small number of the rules may be interesting and useful for the business users. Evaluating interesting and useful rules have been widely investigated by many researchers (Toloo et al., 2009, Szczech et al., 2011, Greco et al., 2012, Ghanem et al., 2011).

Future research can involve the addition of a new agent to the proposed PMAAR algorithm for ranking (sometimes referred to as prioritization) the generated Association Rules. The process can be divided into two steps: (i) First, the Redundant Rules agent eliminates the redundant rules. (ii) Second, the Ranking Rules agent ranks the rules generated from the previous step according to multiple user specified criteria. This can help the decision makers to easily find the relevant domain specific rules.

References

AGRAWAL, R. & SHAFER, J. (1996). Parallel mining of association rules. *IEEE Transactions on Knowledge and Data Engineering,* 8(6)**,** pp. 962-969.

AGRAWAL, R. & SRIKANT, R. (1994). Fast algorithms for mining association rules in large Databases. *In: Proceedings of the 20th International Conference on Very Large Data Bases*.

AIROLA, A., PAHIKKALA, T., WAEGEMAN, W., DE BAETS, B. & SALAKOSKI, T. (2011). An experimental comparison of cross-validation techniques for estimating the area under the ROC curve. *Computational Statistics & Data Analysis,* 55(4)**,** pp. 1828-1844.

ALHAJJ, R. & KAYA, M. (2005). *Multiagent Association Rules Mining in Cooperative Learning Systems* Springer.

ANSARI, E., DASTGHAIBIFARD, G., KESHTKARAN, M. & KAABI, H. (2008). Distributed frequent itemset mining using trie data structure. *IAENG International Journal of Computer Science,* 35(3)**,** pp. 377-381.

BAHAMISH, H. A. A., SALAM, R. A., ABDULLAH, R., OSMAN, M. A. & RASHID, N. A. (2004). Mining protein data using parallel/distributed association rules. *In: Proceedings of the*

International Conference on Information and Communication Technologies: From Theory to Applications, 19-23 April 2004, pp. 461-462.

BAI, Y. M., MENG, X. Y. & HAN, X. J. (2012). Mining Fuzzy Association Rules in Quantitative Databases. *Applied Mechanics and Materials,* 182(1)**,** pp. 2003-2007.

BODON, F. (2003). A fast apriori implementation. *In: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations (FIMI'03)*.

BODON, F. (2005). A trie-based APRIORI implementation for mining frequent item sequences. *In: Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations,* pp. 65.

BODON, F. & RÓNYAI, L. (2003). Trie: An alternative data structure for data mining algorithms. *Mathematical and Computer Modelling,* 38(7)**,** pp. 739-751.

BUNKAR, K., SINGH, U. K., PANDYA, B. & BUNKAR, R. (2012). Data mining: Prediction for performance improvement of graduate students using classification. *In: Proceedings of the Ninth International Conference on Wireless and Optical Communications Networks (WOCN), 20-22 Sept. 2012,* pp. 1-5.

BYUNG-MO, H., SEUNG-JAE, S., KYU MIN, L., KYUNG-SOO, J. & DONG-RYEOL, S. (2006). Multi-agent system based efficient healthcare service. *In: Proceedings of the 8th International Conference in Advanced Communication Technology, ICACT, 20-22 Feb. 2006,* pp. 5-51.

CAO, L. (2009). *Data Mining and Multi-agent Integration.* Springer.

CHANG, J. H. (2011). Mining weighted sequential patterns in a sequence database with a time-interval weight. *Knowledge-Based Systems,* 24(1)**,** pp. 1-9.

CHEN, J. & XIAO, K. (2010). BISC: A bitmap itemset support counting approach for efficient frequent itemset mining. *ACM Transactions on Knowledge Discovery from Data (TKDD),* 4(3)**,** pp. 12.

CHEUNG, D. W., LEE, S. D. & XIAO, Y. (2002). Effect of data skewness and workload balance in parallel data mining. *IEEE Transactions on Knowledge and Data Engineering,* 40(1)**,** pp. 498-514.

CHEUNG, D. W., NG, V. T., FU, A. W. & FU, Y. (1996). Efficient mining of association rules in distributed databases. *Knowledge and Data Engineering, IEEE Transactions on,* 8(6)**,** pp. 911-922.

CHIA-CHU, C. & SHEN, L. (2010). Distributed Association Mining on Message Passing Systems. *In: Proceedings of the International*

Symposium on Parallel and Distributed Processing with Applications (ISPA), 6-9 Sept. 2010, pp. 208-214.

CHONG, W. & YANQING, W. (2011). Discovering consumer's purchasing behavior based on efficient association rules. *In: Proceedings of the Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 26-28 July 2011,* pp. 937-941.

CHUANSHEN, Z., BAOXIAN, J., YUHUA, L. & LIXIA, C. (2010). Mining global frequent subtrees. *In: Proceedings of the Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 10-12 Aug. 2010,* pp. 2275-2279.

CHUNG, S. M. & QING, W. (2001). Content-based retrieval and data mining of a skin cancer image database. *In: Proceedings of the International Conference on Information Technology: Coding and Computing, Apr 2001,* pp. 611-615.

CIOS, K. J., PEDRYCZ, W., SWINIARSKI, R. W. & KURGAN, L. A. (2010). *Data mining: a knowledge discovery approach.* Springer Publishing Company, Incorporated.

DA SILVA, J. C., GIANNELLA, C., BHARGAVA, R., KARGUPTA, H. & KLUSCH, M. (2005). Distributed data mining and agents. *Engineering Applications of Artificial Intelligence,* 18(7)**,** pp. 791-807.

DEAN, J. & GHEMAWAT, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM,* 51(1)**,** pp. 107-113.

DI FATTA, G. & FORTINO, G. (2007). A customizable multi-agent system for distributed data mining. *The 22nd Annual ACM Symposium on Applied Computing,* 22(1).

DIEL, R., LODDENKEMPER, R., NIEMANN, S., MEYWALD-WALTER, K. & NIENHAUS, A. (2011). Negative and Positive Predictive Value of a Whole-Blood Interferon-γ Release Assay for Developing Active Tuberculosis An Update. *American journal of respiratory and critical care medicine,* 183(1)**,** pp. 88-95.

DONG, J. & HAN, M. (2007). BitTableFI: An efficient mining frequent itemsets algorithm. *Knowledge-Based Systems,* 20(4)**,** pp. 329-335.

DUA, S., JAIN, V. & THOMPSON, H. W. (2008). Patient classification using association mining of clinical images. *In: Proceedings of the 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 14-17 May 2008,* pp. 253-256.

DUNHAM, M. H. (2006). *Data mining: Introductory and advanced topics. India*: Pearson Education.

FAYYAD, U., PIATETSKY-SHAPIRO, G. & SMYTH, P. (1996a). *Advances in knowledge discovery and data mining.* MIT Press.

FAYYAD, U., PIATETSKY-SHAPIRO, G. & SMYTH, P. (1996b). The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM,* 39(11)**,** pp. 27-34.

FIPA. (May, 2002). *Foundation for Intelligent Physical Agents* [Online]. Available: http://www.fipa.org/ [Accessed 1/11/2012].

FLORES-MENDEZ, R. A. (1999). Towards a standardization of multi-agent system framework. *Crossroads,* 5(4)**,** pp. 18-24.

GAROFALO, V., PETRA, N. & NAPOLI, E. (2011). Analytical calculation of the maximum error for a family of truncated multipliers providing minimum mean square error. *Computers, IEEE Transactions on,* 60(9)**,** pp. 1366-1371.

GHANEM, S. M., MOHAMED, M. A. & NAGI, M. H. (2011). EDP-ORD: Efficient distributed/parallel Optimal Rule Discovery. *In: Proceedings of the IEEE Symposium on Computers and Communications (ISCC), June 28 2011,* pp. 956-963.

GOLDREICH, O. (2008). Computational complexity: a conceptual perspective. *SIGACT News,* 39(3)**,** pp. 35-39.

GONZÁLEZ, E. J., HAMILTON, A. F., MORENO, L., MARICHAL, R. L. & MUÑOZ, V. (2006). Software experience when using ontologies in a multi-agent system for automated planning and scheduling. *Software: Practice and Experience,* 36(7)**,** pp. 667-688.

GRECO, S., SŁWINSKI, R. & SZCZECH, I. (2012). Properties of rule interestingness measures and alternative approaches to normalization of measures. *Information Sciences,* 10(1).

GUPTA, A., KUMAR, N. & BHATNAGAR, V. (2012). Mining of multiobjective non-redundant association rules in data streams. *In: Proceedings of the Artificial Intelligence and Soft Computing,* pp. 73-81.

HAHSLER, M., GRÜN, B. & HORNIK, K. (2005). A computational environment for mining association rules and frequent item sets. Vienna: WU Vienna University of Economics and Business.

HAN, E., KARYPIS, G. & KUMAR, V. (1997). *Scalable parallel data mining for association rules.* ACM.

HAN, J., CHENG, H., XIN, D. & YAN, X. (2007). Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery,* 15(1)**,** pp. 55-86.

HAN, J. & PEI, J. (2000). Mining frequent patterns by pattern-growth: methodology and implications. *ACM SIGKDD Explorations Newsletter,* 2(2)**,** pp. 14-20.

HANGUANG, L. & YU, N. (2012). Intrusion Detection Technology Research Based on Apriori Algorithm. *Physics Procedia,* 24(2)**,** pp. 1615-1620.

HONG, T. P., KUO, C. S. & WANG, S. L. (2004). A fuzzy AprioriTid mining algorithm with reduced computational time. *Applied Soft Computing,* 5(1)**,** pp. 1-10.

HUA-JIN, W., CHUN-AN, H. & JIAN-SHENG, L. (2010). Distributed Mining of Association Rules Based on Privacy-Preserved Method. *In: Proceedings of the International Symposium on Information Science and Engineering (ISISE), 24-26 Dec. 2010,* pp. 494-497.

IMBERMAN, S. P. (2002). Effective use of the kdd process and data mining for computer performance professionals. *In: Proceedings of the Computer Measurement Group Conference,* pp. 611-620.

JIAYI, Z., KUN-MING, Y. & BIN-CHANG, W. (2010). Parallel frequent patterns mining algorithm on GPU. *In: Proceedings of the IEEE International Conference on Systems Man and Cybernetics (SMC), 10-13 Oct. 2010,* pp. 435-440.

JIE, W. & YU, Z. (2011). DSWFP: Efficient mining of weighted frequent pattern over data streams. *In: Proceedings of the Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 26-28 July 2011,* pp. 942-946.

KANTARDZIC, M. (2011). *Data mining: concepts, models, methods, and algorithms.* Wiley-IEEE Press.

KAOSAR, M. G., XU, Z. & YI, X. (2009). Distributed Association Rule Mining with Minimum Communication Overhead. *In:*

Proceedings of the 8th Australasian Data Mining Conference (AusDM'09), pp. 17-23.

KELLEY, K. & LAI, K. (2011). Accuracy in Parameter Estimation for the Root Mean Square Error of Approximation: Sample Size Planning for Narrow Confidence Intervals. *Multivariate Behavioral Research,* 46(1)**,** pp. 1-32.

KUDRYAVTSEV, V. & ANDREEV, A. (2010). On algorithm complexity. *Journal of Mathematical Sciences,* 168(1)**,** pp. 89-122.

KUMAR, V. & ZAKI, M. (2005). High performance data mining. *High Performance Computing for Computational Science***,** pp. 111-125.

LEI, Z. & REN-HOU, L. (2007). An Algorithm for Mining Fuzzy Association Rules Based on Immune Principles. Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering, 2007. BIBE 2007., 2007. IEEE, 1285-1289.

LEUNG, C. C., YAM, W. C. & YEW, W. W. (2011). The Positive Predictive Value of T-Spot. TB and Tuberculin Skin Test in Patients with Silicosis. *American journal of respiratory and critical care medicine,* 183(2)**,** pp. 277-278.

LI, J., XU, Y., WANG, Y.-F. & CHU, C.-H. (2009). Strongest Association Rules Mining for Personalized Recommendation. *Systems Engineering,* 29(8)**,** pp. 144-152.

LI, L. & ZHANG, M. (2011). The strategy of mining association rule based on cloud computing. *In: Proceedings of the International Conference on Business Computing and Global Informatization (BCGIN),* pp. 475-478.

LI, T. & LI, X. (2010). Novel alarm correlation analysis system based on association rules mining in telecommunication networks. *Information Sciences,* 180(16)**,** pp. 2960-2978.

LITTLE, R. J., D'AGOSTINO, R., COHEN, M. L., DICKERSIN, K., EMERSON, S. S., FARRAR, J. T., FRANGAKIS, C., HOGAN, J. W., MOLENBERGHS, G. & MURPHY, S. A. (2012). The Prevention and Treatment of Missing Data in Clinical Trials. *New England Journal of Medicine,* 367(14)**,** pp. 1355-1360.

LIU, B., CAO, S. G. & HE, W. (2011). Distributed data mining for e-business. *Information Technology and Management,* 12(2)**,** pp. 67-79.

MAIMON, O. & ROKACH, L. (2010). *Data mining and knowledge discovery handbook.* Springer.

MARÍK, V., STEPÁNKOVA, O., KRAUTWURMOVA, H. & LUCK, M. (2002). *Multi-agent systems and applications II: 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001 , HoloMAS 201 , Selected Revised Papers.* Springer.

MARUKATAT, R. (2007). Structure-based rule selection framework for association rule mining of traffic accident data. *Computational Intelligence and Security,* 12(4)**,** pp. 231-239.

MENON, R., TONG, L. H. & SATHIYAKEERTHI, S. (2005). Analyzing textual databases using data mining to enable fast product development processes. *Reliability Engineering and System Safety,* 88(2)**,** pp. 171-180.

MINGHUA, H., JENNINGS, N. R. & HO-FUNG, L. (2003). On agent-mediated electronic commerce. *Knowledge and Data Engineering, IEEE Transactions on,* 15(4)**,** pp. 985-1003.

MITRA, S., PAL, S. K. & MITRA, P. (2002). Data mining in soft computing framework: A survey. *IEEE transactions on neural networks,* 13(1)**,** pp. 3-14.

MOHAN, S. R., PARK, E. K. & HAN, Y. (2005). Association rule based data mining agents for personalized Web caching. *In: Proceedings of the 29th Annual International Conference in Computer Software and Applications, COMPSAC,* pp. 37-38.

NAJADAT, H., SHATNAWI, A. & OBIEDAT, G. (2011). A New Perfect Hashing and Pruning Algorithm for Mining Association Rule. *Communications of the IBIMA,* 2011(6)**,** pp. 1-9.

NAWAPORNANAN, C. & BOONJING, V. (2011). A new share frequent itemsets mining using incremental BitTable knowledge.

In: Proceedings of the 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT), 29 Nov. 2011, pp. 358-362.

NKUDIC. (2006). *National Kidney and Urologic Diseases Information Clearinghouse:Prostate Enlargement* [Online]. Available: http://kidney.niddk.nih.gov/kudiseases/pubs/prostateenlargement/ [Accessed 1/11/2012].

OLSON, D. L. & DELEN, D. (2008). *Advanced data mining techniques.* Springer.

OUALI, A., RAMDANE-CHERIF, Z., RAMDANE-CHERIF, A., LEVY, N. & KREBS, M. O. (2003). Agent paradigm in clinical large-scale data mining environment. *In: Proceedings of the The Second IEEE International Conference on Cognitive Informatics,* pp. 143-150.

ÖZEL, S. A. & GÜVENIR, H. A. (2001). An algorithm for mining association rules using perfect hashing and database pruning. *In: Proceedings of the 10th Turkish Symposium on Artificial Intelligence and Neural Networks,* pp. 257-264.

PANAIT, L. & LUKE, S. (2005). Cooperative Multi-Agent Learninig: The State of the Art. *Autonomous Agents and Multi-Agent Systems,* 11(3).

PARK, J., CHEN, M. & YU, P. (1995). Efficient parallel data mining for association rules. *In: Proceedings of the fourth international conference on Information and knowledge management,* pp. 31-36.

PARK, J., CHEN, M. & YU, P. (1997). Using a hash-based method with transaction trimming for mining association rules. *Knowledge and Data Engineering, IEEE Transactions on,* 9(5)**,** pp. 813-825.

PEYRE, H., LEPLÈGE, A. & COSTE, J. (2011). Missing data methods for dealing with missing items in quality of life questionnaires. A comparison by simulation of personal mean score, full information maximum likelihood, multiple imputation, and hot deck techniques applied to the SF-36 in the French 2003 decennial health survey. *Quality of Life Research,* 20(2)**,** pp. 287-300.

REGLI, W. C., MAYK, I., DUGAN, C. J., KOPENA, J. B., LASS, R. N., MODI, P. J., MONGAN, W. M., SALVAGE, J. K. & SULTANIK, E. A. (2009). Development and Specification of a Reference Model for Agent-Based Systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* 39(5)**,** pp. 572-596.

RENJIT, J. A. & SHUNMUGANATHAN, K. (2010). Mining The Data From Distributed Database Using An Improved Mining Algorithm. *arXiv preprint arXiv:1004.1677,* 9(1).

ROMERO, C. & VENTURA, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications,* 33(1)**,** pp. 135-146.

RUI, C. & ZHIYI, L. (2011). An improved apriori algorithm. *In: Proceedings of the International Conference on Electronics and Optoelectronics (ICEOE), 29-31 July 2011,* pp. 476-478.

RUSSELL, S. & NORVIG, P. (2003). *Artificial Intelligence : A Modern Approach.* Prentice Hall.

SCHEUER, O. & MCLAREN, B. M. (2012). Educational data mining. *Encyclopedia of the sciences of learning, Springer, New York,* 9(3)**,** pp. 1075-1079.

SHAW, G., XU, Y. & GEVA, S. (2009). Eliminating redundant association rules in multi-level datasets. *In: Proceedings of the 4th International Conference on Data Mining, 14-17 July 2008*.

SHEN, L. & LIU, S. (2012). A New Fuzzy Association Rules Mining in Data Streams. *In: Proceedings of the 3rd International Conference on Teaching and Computational Science (WTCS 2009),* pp. 163-172.

SHINTANI, T. & KITSUREGAWA, M. (1996). Hash based parallel algorithms for mining association rules. *In: Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems,* pp. 19-30.

SIDDIQA, A., NIAZI, M., MUSTAFA, F., BOKHARI, H., HUSSAIN, A., AKRAM, N., SHAHEEN, S., AHMED, F. & IQBAL, S. (2009). A new hybrid agent-based modeling and simulation decision support system for breast cancer data analysis. *In: Proceedings of the International Conference on Information and Communication Technologies, ICICT, 15-16 Aug. 2009,* pp. 134-139.

SONG, W., YANG, B. & XU, Z. (2008). Index-BitTableFI: An improved algorithm for mining frequent itemsets. *Knowledge-Based Systems,* 21(6)**,** pp. 507-513.

SUH, S. (2011). *Practical Applications of Data Mining.* Jones and Bartlett Learning.

SUMITHRA, R. & PAUL, S. (2010). Using distributed apriori association rule and classical apriori mining algorithms for grid based knowledge discovery. *In: Proceedings of the International Conference on Computing Communication and Networking Technologies (ICCCNT),* pp. 1-5.

SZCZECH, I., GRECO, S. & SLOWINSKI, R. (2011). New property for rule interestingness measures. *In: Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), 18-21 Sept. 2011,* pp. 103-108.

TALAEI-KHOEI, A., RAY, P. & PARAMESWARAN, N. (2009). An Awareness Framework for Agent-Based Mobile Health Monitoring. *In: Proceedings of the third International Conference on Next Generation Mobile Applications, Services and Technologies, (NGMAST), 15-18 Sept. 2009,* pp. 108-113.

TAN, P. N. (2007). *Introduction to data mining.* Pearson Education India.

THI, L., THI, N. & TAE CHONG, C. (2010). BitApriori: An Apriori-Based Frequent Itemsets Mining Using Bit Streams. *In: Proceedings of the International Conference on Information Science and Applications (ICISA), 21-23 April 2010,* pp. 1-6.

TOLOO, M., SOHRABI, B. & NALCHIGAR, S. (2009). A new method for ranking discovered rules from data mining by DEA. *Expert Systems with Applications,* 36(4)**,** pp. 8503-8508.

VLASSIS, N. (2007). A concise introduction to multiagent systems and distributed artificial intelligence. *Artificial Intelligence and Machine Learning,* 1(1)**,** pp. 1-71.

VO, B. & LE, B. (2011). Mining minimal non-redundant association rules using frequent itemsets lattice. *International journal of intelligent systems technologies and applications,* 10(1)**,** pp. 92-106.

WANG, H., HU, C. & LIU, J. (2010). Distributed Mining of Association Rules Based on Privacy-Preserved Method. *In: Proceedings of the International Symposium on Information Science and Engineering (ISISE),* pp. 494-497.

WEI, H., EL-DARZI, E. & LI, J. (2007). Extending the Gaia Methodology for the Design and Development of Agent-based Software Systems. *In: Proceedings of the Computer Software and Applications Conference, (COMPSAC),* pp. 159-168.

WHITE, T. (2012). *Hadoop: The definitive guide.* O'Reilly Media.

WILLMOTT, C. J. & MATSUURA, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research,* 30(1)**,** pp. 79.

WITTEN, I. H. & FRANK, E. (2005). *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann.

WU, G., ZHANG, H., QIU, M., MING, Z., LI, J. & QIN, X. (2012). A decentralized approach for mining event correlations in

distributed system monitoring. *Journal of parallel and Distributed Computing,* 7(3).

XIE, M. C. & TACHIBANA, A. (2007). Cooperative Behavior Acquisition for Multi-agent Systems by Q-learning. *In: Proceedings of the Foundations of Computational Intelligence, (FOCI),* pp. 424-428.

XING, Y., MADDEN, M. G., DUGGAN, J. & LYONS, G. (2003). A Multi Agent System for Context Based Distributed Data Mining. Galway: National University of Ireland.

XUEPING, Z., YANXIA, Z. & NAN, H. (2010). Improved paralled algorithm for mining frequent item-set used in HRM. *In: Proceedings of the Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 10-12 Aug. 2010,* pp. 283-289.

YANG, J. & YANG, Y. (2010). A parallel algorithm for mining association rules. *In: Proceedings of the 2nd International Conference on Networking and Digital Society (ICNDS),* pp. 475-478.

YANG, X. Y., LIU, Z. & FU, Y. (2010). MapReduce as a programming model for association rules algorithm on Hadoop. *In: Proceedings of the 3rd International Conference on Information Sciences and Interaction Sciences (ICIS), 23-25 June 2010,* pp. 99-102.

YE, Y. & CHIANG, C. C. (2006). A parallel apriori algorithm for frequent itemsets mining. *In: Proceedings of the Fourth International Conference on Software Engineering Research, Management and Applications,* pp. 87-94.

YIN, Y. (2009). A proximate dynamics model for data mining. *Expert Systems with Applications,* 36(6)**,** pp. 9819-9833.

YUPENG, Z., LEE, M. & GATTON, T. M. (2009). Agent-Based Web Healthcare Systems for Real-Time Chronic Disease. *In: Proceedings of the World Conference on Services, 6-10 July 2009,* pp. 14-21.

ZGHAL, H. B., FAIZ, S. & GHEZALA, H. B. (2005). A Framework for Data Mining Based Multi-Agent An Application to Spatial Data. *World Academy of Science, Engineering and Technology,* 6(2).