A Grid Data Mining Architecture for Learning Classifier Systems

SANTOS, M.F [1*]., MATHEW, W [1#]., KOVACS, T [2+]., SANTOS, H. [1⌗]

[1] Department of Information System
University of Minho,
4800-058 Guimarães, Portugal.
[*] mfs@dsi.uminho.pt [#] wesley@dsi.uminho.pt, [⌗] hsantos@dsi.uminho.pt

[2] Department of Computer Science
University of Bristol
Merchant Venturers Building, Bristol BS8 1UB
[+] kovacs@cs.bris.ac.uk

Abstract: - Recently, there is a growing interest among the researchers and software developers in exploring Learning Classifier System (LCS) implemented in parallel and distributed grid structure for data mining, due to its practical applications. The paper highlights the some aspects of the LCS and studying the competitive data mining model with homogeneous data. In order to establish more efficient distributed environment, in the current work, Grid computing architecture is considered a better distributed framework in Supervised Classifier System (UCS). The fundamental structure of this work allows each site of the distributed environment to manage independent UCS and such local sites transmit learning models to the global model for making complete knowledge of the problem. The Boolean 11-multiplexer problems are used for the execution. Hence, the main objective of this work is to keep the average accuracy of distributed mode without loosing accuracy rate compared to models. The experimental results showed that the testing accuracy of distributed mode is higher than other models.

Key-Words: - Learning Classifier Systems, UCS, Genetic Algorithm, Fitness, Accuracy, Data Mining, Grid computing, Cloud computing, Grid Data Mining.

1 Introduction

During last decade, the increasing applications of data mining techniques have been receiving more concentration from the researchers and computer professionals. The data mining on localized computer environment no longer meet the modern scientific and industrial complex problem solving environment [1]. This is mainly due to the storage of digital data in a geographically distributed sites and alarming increase in the size of data. "*Classification*" is one of the data mining technologies. Classification can be defined as the process of assigning a class label to a given problem, given a set of problem previously defined [2]. Potential of grid technology is very proactive in developing distributed data mining environment on grid platform.

The intention of this work is to carry out global model from all local models in the distributed site of grid structure. The given system has diverse distributed sites (agents) and a global (central) site. The learning processes are carried out in local site and send learned model to global model. All the learning models in the distributed sites are unified to form a single learning model in the global site. This single unified model shows the knowledge of complete distributed database.

In this work, supervised learning classifier system (UCS) is used for Grid Data Mining to induce global data mining models from distributed agents. This system consolidates such knowledge from different site that assist in improving the learning classifier system performance and permit the parallelization and distribution of classifiers in an adaptable way. The benefit of the current system is that the global data mining has better accuracy compared to other data mining methods.

The remainder of this paper is organized as follows. Section 2 describes some background of supervised learning classifier systems for grid data mining. This include learning classifier system, UCS, grid computing, cluster computing, data mining and distributed data mining. Section 3 provides details of proposed work. This section demonstrates working of the supervised learning classifier system for grid data mining. The section 4 includes details of experimental results. Section 5 illustrated basic level discussion of two methods and different models of supervised classifier system, and related work. Finally, section 6 summarizes the main conclusions from this work.

2 Background

The size of digital data, has been increasing at a rapid rate in recent times, doubling every three years. These data are typically stored in geographically distributed repositories. Distributed data mining is a valuable technique to analyze such data.
Classification is one of the technologies used for data mining. Common types of classifier systems include LCS, XCS, SB-XCS, ACS, and UCS. This paper addresses a supervised learning classifier system, which mainly includes LCS and UCS to perform distributed data mining classification tasks.

In distributed data mining, which implements data mining in a distributed fashion, the technologies available for distribution include distributed computing, cluster computing, grid computing and cloud computing. In this work, grid computing is employed, because of its usefulness in implementing data mining environments on grid platform by developing grid services. Since cloud computing is an important field in distributed environment, a brief description of cloud computing is also provided.

2.1 Data Mining

Data mining as a one step in the multi step process of knowledge discovery in database (KDD). [3, 4] The KDD process starts with row data such as recorded financial or medical data and final produces some potentially useful pattern. The steps of KDD processes are selection, pre-processing, transformation, data mining, interpretation, evaluation [4]. During selection understanding the domain and goal of the end user and useful features extract from the row data. The pre-processing step cleans row data such as remove all noise and irrelevant data; fixing incorrect format of data, etc. The data transformation converts cleaned data to another form for better understanding of the relationship among the features. The data mining face search patterns of interest by using data mining algorithms. The interpretation and evaluation step interpret pattern from data mining to human understandable form.

Generally, data mining involves computer assisted analysis of large amount of data from different identities and converting it into useful information. The data mining process has many components such as machine learning, artificial intelligence, statistics, signal processing, mathematical optimization, and pattern recognition. Data mining software allows users to analyze data from many different angles or aspects to categorize data, and recapitulate the relationships identified. In principle, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

2.2 Distributed Data Mining

The main objective of distributed data mining is the parallel implementation of data mining in distributed sites to extract and combine useful knowledge [4]. There are two patterns in distributed data mining Distributed Mode (DMod) and Centralised Mode (CMod). In the DMod, data mining is applied to each distributed site and collects local models from each agent combining them in a central site where a final model is obtained. In the CMod, row data are collected from distributed locations and stored in a central location, and then data mining techniques are applied in the central location. The first pattern has the advantage that it needs to send only small amounts of data from local models thus reducing the bandwidth required. The second pattern needs to send row data that needs more bandwidth. Another advantage of the first pattern is that it is more secure than the second one.

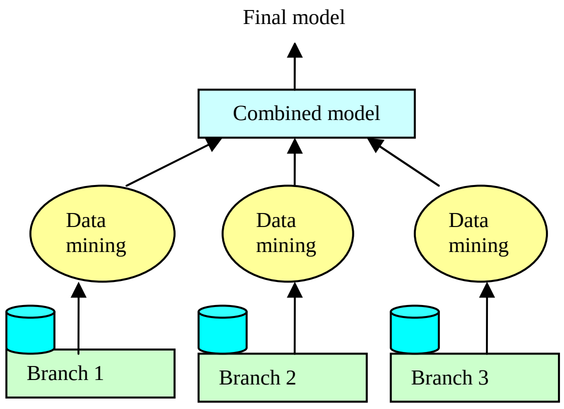

Figure 1 - Distributed data mining.

As an example, consider a company that has different branches in different geographical locations and one central location, with each branch having a separate database. Figure 1 illustrates the distributed data mining process for this company. Data mining in the client extracts knowledge (local model) from that particular database and sent it to the central location. The central location combines all local knowledge from the results obtained in each local site. The final model is derived from the combined model. Hence

the central office of the company gets the over all knowledge.
A distributed environment involves large volumes of several sources of data. Analysing and monitoring these distributed data requires data mining techniques designed for distributed application. The quality of performance in data mining needs to be improved in distributed environments. The best example of a distributed environment is the Internet, where growing more databases and data streams appear. More over internet consist of communication media for distributed information system, as for example the earth observing system of NASA [4].

2.3 Learning Classifier Systems

Learning Classifier Systems (LCSs) are a machine learning paradigm that can be used as a data mining technique [3, 6]. LCS uses the concept of evolutionary computation and reinforcement learning. The classifiers consist of a set of condition action rules, which is the knowledge of the LCS. The classifier evolves by learning from unknown environment.

LCSs were introduced by John H Holland in 1970 [5]. There was considerable research in the 1980s but in the early 1990s, LCSs introduced only a few successful applications [6]. In the mid 1990s, the field appeared almost at a dead end. But, during the last 10 years, new models have been developed and new applications have been presented which caused a great movement of this area.

The researches in the LCS area originated various types of LCS. We can differentiate two main approaches: the Michigan approach and the Pittsburg approach [4]. The main difference between these two approaches is that in the Michigan approach each individual is a single rule that represent a partial solution (the approach subscribed in this work), whereas in the Pittsburg approach, each individual is a set of rules that represents a complete solution to a learning problem [5]. The canonical LCS consists of four main components:

(i) The classifiers - which consists of condition action rules that represent the current knowledge of the system;

(ii) The performance component - which controls the interaction with the environment;

(iii) The reinforcement component (credit assignment component) - which receives reward from environment and delivers it to classifiers;

(iv) The genetic component - which is responsible for generating better rules and improving existing ones through a Genetic Algorithm (GA).

Classifiers have two associated measures: the strength and the fitness. Strength is the prediction of the classifier utility in terms of the amount of reward that the system will receive if the classifier is used. Fitness calculates the quality of the action about the problem that the classifier conveys, and it is performed by the discovery component for generating new population. The high fitness means that the classifier conveys more appropriate action about the problem and therefore it should generate more classifier through the genetic algorithm; a low fitness means that the classifier conveys little appropriate action about the problem and therefore should generate less classifier [5].

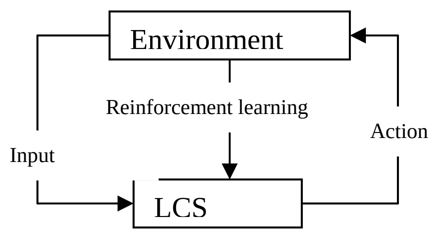

Figure 2 - LCS interact with environment.

A LCS learns by interacting with an environment from which it receives reward in the form of numerical compensation. Learning is accomplished by trying to maximize the amount of reward received. On each discrete time step (system life cycle) the environment present a situation or problem in the form of a string or a binary code, the system receives as input the current situation of the environment and builds a match set [M] that consist of all the classifiers in the population [P] whose condition satisfies the condition parts of input situation. Then, the system evaluates the usefulness of the actions appearing in the match set; an action A is selected from those in the match set [M], and sent to the environment to be performed. Depending on the current situation and on the performance of the action A the system then receives a reward (or a punishment). The credit assignment component distributes the reward value among the rules accountable of the incoming rewards. This can be either implemented with an algorithm specifically designed for LCS (e.g., the bucket brigade algorithm).

The genetic component in the learning classification system randomly selects two classifiers (in mono-step systems; for multi-step systems this value doubles the number of steps) from the population with probability proportional to their fitness's. By applying crossover and mutation, two new child classifiers are generated and subsumed by parent for check the generality of the new classifiers. If new classifiers are more general, then new classifiers are appended to the population, while two others are deleted in order to keep the population size constant.

The objective of the learning is to generate more accurate rules from previously unknown situations. The classification problem environment is a training set of pre-classified examples (situations). Each example is a vector of attributes and class labels.

2.4 Supervised Classifier System

Supervised Classifier System (UCS) is a learning classifier system derived from XCS, one of the most successful implementation of LCS [2]. UCS is specifically designed for supervised learning problems, while XCS follows a reinforcement learning scheme. The basic format of the supervised learning is based on the fact that after the learner chosen an action, the environment indicates what the correct action was [4]. This means that the error can be measured and used to optimize the model to be learned. UCS is then more suitable for data mining applications.

Moreover, UCS develops classifiers by learning. Population contains set of classifiers. Each classifier consists of rules and a set of parameters. A classifier can be defined as a tuple:

<condition, action, fitness, accuracy, numerosity, number of correct, number of match, correct set size, last time this was in the GA >

Fitness and accuracy are the quality of classifier, numerosity is the number of copies of classifier in the set, "number of match" is the number of occurrences of the classifier in the match set, "number of correct" is the number of times that the classifier is in the correct set, "correct set size" is the average size of all the correct sets where the classifier participate.

Fitness of the UCS is based on the accuracy of the classifier [2, 4]. We can group classifiers into two categories: correct and incorrect classifiers. The correct classifiers are those that get highest payoff if they are fired, and incorrect classifiers are the ones that get lowest payoff. The existences of classifiers in the population have more chance to correct one because incorrect classifiers will receive less fitness. Therefore, they can not enter into the evolutionary process and these classifiers have more chance for deletion than correct one [7].

The UCS has two mode of operation: testing and training. During training, inputs (examples) coming from the training set are provided to UCS. Then, UCS forms a match set containing the classifiers in the population whose conditions match the condition part of the input example. From the match set, the classifiers that predict class c form the correct set [8]. If the correct set is empty, the covering operator is executed. The covering is used for generating new classifiers. The new classifiers condition and class same as in the example which matches with population. Then, the parameters of classifiers are updated and eventually, the GA is applied. The genetic algorithm is invoked when the average experience of classifiers in correct set is higher than a GA threshold defined by the user. In test mode, prediction is carried in match set for getting predicted class. For the prediction, the sum of fitness in each classifier in the match set is used. The class with highest value is chosen as predicted class. The test mode genetic operation is disabled.

The parameters of classifiers are updated only in the training mode and only those belonging to the match set. The "number of match" (NM) updated corresponds to number of input examples correctly matched to the classifier condition part. The "number of correct" (NC) updated correspond to the number of the input class correctly matched to that classifiers' class part.

The accuracy of a classifier is the ratio of number of correct by number of match:

$$Accuracy = \frac{NC}{MN} \quad (1)$$

The fitness of a classifier depends on accuracy:

$$\text{Fitness} = \text{Accuracy}^{v} \quad (2)$$

Where v it is a pre-defined constant [3, 6]. A typical value is 20.

The genetic algorithm is used to generate classifiers in the population as the search mechanism. The GA is applied to the correct set [C]. The GA is triggered when the average of experience (last time this was in the GA) in the correct set is greater than a GA threshold defined by user. Genetic algorithm first selects two classifiers from the correct set with a probability proportional to fitness. These classifiers are considered as parent classifiers. The parent classifiers pass through crossover and mutation with probability "**λ**" and **"μ"** respectively. The resulting offspring are checked for subsumption with their parents [2, 4]. If new offspring is more general and accurate than the parent then it's added to population; otherwise the numerosity of parent is increased by one. The UCS population has a fixed size of N. If the size of population is greater than N, then one classifier is deleted from the population.

2.5 Grid computing

Grid computing is a special kind of distributed computing system. In distributed computing, different computers within the same network share one or more resources; but in grid computing, every resource is shared [26]. Grid computing is a geographically-distributed computer network and heterogeneous computing, storage, and network resources, providing secure and pervasive access to their combined capabilities. Therefore, Grid platforms enable sharing, exchange, discovery, selection and aggregation of distributed heterogeneous resources such as computers, databases, visualization devices and scientific instruments.
Grid computing system connects all computer resources together in a way that gives the user advantages like a supercomputer [26] because user can access and leverage the collected power of all the computers in the system. Grid computing has the potential to support a wide range of applications that includes computer-intensive applications and applications requiring distributed services.
The problems found in the grid environment are not easily handled by executing a specific ad-hoc program. Instead, that needs several interacting software components, to run separately or concurrently over a given set of inputs. In this field, until now not much work has been done to build high-level design facilities for complex grid applications in which many programs and data sets are involved. These types of applications are most widely used in several domains, such as knowledge management, computational science, and e-business. Data mining tools for use in grid environment is necessary due to databases that tend to have pentabytes of stored data in geographically different places, which demands high processing power in a distributed way.
The increasing interest, application, and power demanding of data mining techniques in one side, and the grid technologies processing potential, it is very useful to develop data mining environments on grid platforms by developing grid services for the extraction of knowledge from distributed data repositories.
This paper proposes the development of an Agent-Based Leaning Classifier System for Grid Data Mining.

2.6 Cloud computing

Cloud computing is an internet based development [28, 29]. For the computation over cloud, it combines different networks, software and services by the internet. The collection of networks and services are together called "the cloud" [29]. The cloud environment gives supercomputing access power to users. Using a thin client or other access point, like an iPhone, Blackberry or laptop, users can reach into the cloud for resources as they need them. For this reason, cloud computing has also been described as on demand computing.
Large processing ability is made possible though distributed cluster computing, often in concert with server virtualization software and parallel processing [30]. Cloud computing can process tens of trillions of computations in a second.

Cloud computing has many applications such as intelligence agencies, military, research labs, universities and major corporations to handle calculation and complex work like designing airplanes, and simulation of nuclear explosion. In addition, cloud computing can be used for sorting enormous amounts of data. For example: Google has an initial edge in cloud computing precisely because of its need to produce instant, accurate results for millions of incoming search inquires every day, parsing through the terabytes of Internet data cached on its servers [30].

3 Gridclass system

Gridclass is a project that aims to construct an UCS based grid/cloud environment to support distributed data mining tasks. At the first stage, only classification tasks are under consideration. In the next lines more details are presented.
The computational environment consists of physically distributed computer systems. The current work deals with a physically distributed agents and merging of all agent classifiers into a non distributed environment (global model). Both parts are responsible for building complete knowledge as a whole from local mining systems.

Figure 3 displays the architecture of a grid classifier system. The grid environment consists of multiple individual autonomous systems. The second environment (non distributed) is responsible for combining all knowledge from the grid environment. This architecture corresponds to a many-to-one relationship in which all agents in the grid environment are required to communicate with the global model occasionally in a parallel fashion. For the development of the grid version of the supervised learning classifier system, it is expected a grid infrastructure like the Globus toolkit middleware [27].
The toolkit of a given architecture gives input instance to the local UCS systems. This toolkit is connected to all local UCS in the parallel structure. Therefore, all local UCS get the same input instance at same time.

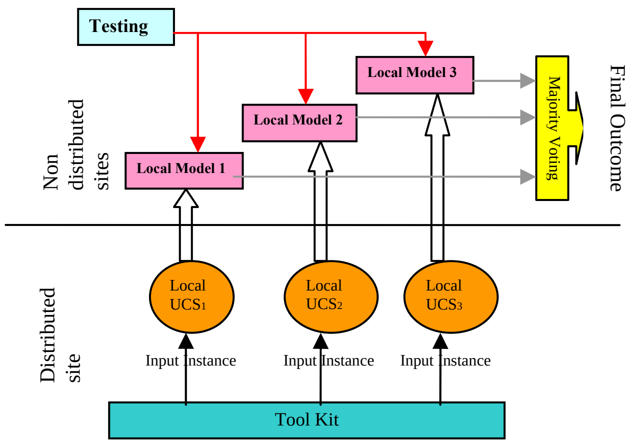

Figure 3 – Grid class System.

3.1 The Grid Environment
The grid environment contains distributed agents, corresponding to an agency. Agents might or might not communicate with each other for exchanging their knowledge. The communication between agents might help to speed up the learning at each agent [8].
Each agent is an independent component, which is able to operate on its own right. The parameter settings of each agent are, in principle, the same. A complete distribution is employed at each agent in order to obtain local patterns coming from local data sources. Data arrive at each agent in stream fashion from the toolkit, so its learning model updates the knowledge on the fly.
Each agent is responsible for updating its knowledge at the global model. Input instance of each agent must be the same but the output instance may be different. Outputs of each agent depend on the population of the individual. So, the size of each local model (knowledge) may vary. The communication frequency is decided by the user. In general, more frequent communication provides faster updates at the global model, but it results in more traffic load.

3.2 Global Model
Global model contains all the local models in its memory. Three local models are maintained at the global model, as shown in Figure 3. These models represent local knowledge of agents. From the manageability point of view, a company might want to build a single knowledge source rather than several independent sources of information.
Each local model consists of a set of individuals. Each and every individual consists of an IP address of the local system and an input instance ID, classifier and their parameters. The memory of the global site is equally accessed by all the local agents and the global system. The agents are responsible for updating their models regularly. Suppose M different local agents in the distributed site provide M different set of models at the global model. The responsibility of the global model is to integrate these local models and choose a final model from the integrated one.
The global model has two main phases: the first one is the testing phase and the other is voting phase. The testing events directly contact with all local models.
During the testing, it applies cut-off among the local models. It is mainly for keeping a minimum grade (standard) in all classifiers in the global model. It is assumed that the total classifier in the M models have N classifiers. The cut-off threshold (T) can be defined by the average fitness of local models:

$$T = \frac{\sum_{c\in[GM]} Fc}{N} \qquad (3)$$

Here GM is the global model, c is the classifier in the global model and Fc is the fitness of the classifier.

$$f(Fc,T) = \begin{Bmatrix} 1\; if\; (Fc \geq T \Rightarrow Fc) \\ 0\; otherwise \end{Bmatrix} \qquad (4)$$

If Fc is greater than and equal to T then that represent 1 (classifier exists in the global model); otherwise 0 (classifier is removed from global model). This assists to improve the efficiency of the global model. In the first stage, the value of cut-off threshold is defined by the user. During the each iteration, the system can update its value depending of the average fitness of local models. After the testing phase subsets of local models are appended to the global model. Then, the final model is taken from this global model. In the current work the voting technique is employed to find the final model. By the voting approach it is possible to combine all local models to

form a single and coherent model. The combination of local models is not a physical process but logical because we consider all local models are in a single structure by using input example ID. Any conflict in the output of local models is resolved through voting [4].
For the selection of final model, two patterns can be used. First pattern is the system prediction employed in separate models based on IP address and takes the separate class from each. Voting can be applied among these classes. In the above example there are 3 local models in the global model. All the individuals in the local model-1 have the same IP address. Select one class from local model-1 by system prediction using the same methodology. The separate class are formed for other two local models. From the three classes we can select one as final class. Alternatively, in the second pattern, a single model is formed from all models, based on input example ID and system prediction applied over these models. The main advantage of such arrangement is that if any system has failed during the iteration that does not affect overall system performance. In the current project, the second method is preferred because it is more efficient and it has short processing time.
To calculate the System Prediction (SP) firstly the summation of fitness (F) and numerosity (Num) of each class should be calculated by:

$$Fclass(i) = \sum_{c\in[GM]class(i)} Fc \qquad (5)$$

$$Num\,class(i) = \sum_{c\in[GM]\,class(i)} Num\,c \qquad (6)$$

Then find the ratio of “F” by “Num”:

$$SP\,class(i) = \frac{F\,class\,(i)}{Num\,class(i)} \qquad (7)$$

These values are taking for voting. The majority value of these votes i.e. Max ($SP_{class\ (i)}$), is considered as the final class.

3.3 Majority Voting

The majority voting is a simple but quite efficient approach for combining knowledge from multiple models. In this approach, there is no requirement of training and an acceptable performance is observed [4]. The final outcome of global model is the effect of all local models impending from the testing process in the global model. The global model applies voting among all predicted output classes.
Let’s assume that a particular grid environment has a set L of m local models L = {l1; l2; :: lm}. Each testing instance gets new sets of local models. Such new set of local models consider a single model for the voting process. Voting applies to the output class of system prediction. The winning class is the one whose vote is majority among result set.
It is worth mentioning that the paper reports only a preliminary approach. More aspects will be included in the future work by considering other possible solutions.

4 Experimental Work

The experiments took under consideration 3 different systems:
1. *The distributed system* -The distributed mode system has six different clients (population). The same input instance used for training and testing of these six different clients. During the testing phase, it calculates separate accuracy of each model in the global site.
2. *The unified distributed system* – The unified distributed mode system also has six different learning models. The learning models from the distributed sites are then combined in the global site by majority voting technique. In order to have a more accurate classifier in the global model system applied cut off threshold in the global model. This global model used for calculating testing accuracy.
3. *The single UCS system* –The single UCS system has only one population. The testing accuracy of the system based on the single learning model.

For the execution of these systems the input instance (environmental problems) were represented by the Boolean 11-multiplexer and the action (target class) was represented as 0 or 1. In this experiment 2/3 of records were used for training and the rest of 1/3 of records were used for testing. For the learning process the initial population was considered empty and a total of 10,000 iterations have been used for training. During the testing process the average accuracy (AccTesting) was considered to assess the models:

$$AccTesting = \frac{Number\ of\ Correct}{Number\ of\ Example} \qquad (8)$$

Where Number of Correct denotes how many predicted classes are matched with target class. Number of example denotes number of input instance used for testing process. Hence, by estimating the average testing accuracy helps to understand the efficiency of the system and to compare with other systems (benchmarking).

4.1 Problem considered in the demonstration

In this set of experiments, environmental problems are defined by using Boolean 11- multiplexer. The Boolean 11- multiplexer is a tuple of 11 bits, where the first 3 bits (a0 - a2) determine the address and next eight (d0- d7) determine the answer. For the learning classification each example are represent {a1, a2, a3, d0, d1, d2, d3, d4, d5, d6, d7} [31]. There are 2^{11} = 2048 combinations for the 11-multiplexer problems.

In terms of learning classification, multiplexer are considered as a class of addressing problems [32]. The target class is combined with each example as demonstrated in the following classifier population:

- 00101100110 =>class 1
- 01001100110 =>class 0
- 10001100110 =>class 0
- 10101100110 =>class 1
- 01101100110 =>class 0

4.2 The experimental conditions

The experiment was conducted on a personal computer and it is configured on Pentium 4 CPU 2.00 GHz, 1 GB RAM and 40 GB Hard disk, Microsoft Windows XP as the operating system was used for execution of the system. The Supervised Learning classifier system (UCS) was developed in JAVA language.

In the testing phase, distributed system containing 6 different populations (M1, M2, M3, M4, M5, M6) was considered. The unified distributed model has been induced from the combination of the six different models applying majority voting strategy. A cut off threshold was applied to global model to remove the less fitness classifiers. The single UCS has only one population (model).
All the models where trained using 2/3 of the available data and tested with the rest 1/3 of data. Ten different test runs were executed in order to compare the performance among the three approaches.

4.3 Results

The results are presented in the Table 1. The Figure 3 shows that the average of accuracy of the 6 distributed models M1,…,M6 (0.985) is less than the average accuracy of unified Distributed model (0.998). The unified distributed model attained an average accuracy of 0.998. The single UCS obtained an average accuracy of 0.986.

Table 1 - Testing Accuracy of Distributed Mode, Unified Distributed Mode And Single UCS.

Execution	Testing Average Accuracy of Distributed Mode with 6 Different Models						Unified Distributed Model	Single UCS
	M 1	M2	M3	M4	M5	M6		
1	1	.98	.99	.98	1	1	0,99	1
2	.99	.99	.93	1	.97	1	1	.97
3	1	1	.96	1	1	.97	0,99	1
4	.99	.99	.99	1	.99	.97	1	.99
5	.96	.99	.93	.96	.99	.99	1	.96
6	.98	1	.97	.98	.99	.99	1	.96
7	.99	.94	1	.93	1	1	1	1
8	1	.97	1	.99	.98	.96	1	.98
9	1	.98	1	1	1	.99	1	1
10	.97	.99	.99	1	1	1	1	1
Average	.988	.983	.976	.984	.992	.987		
	0,985						**0.998**	**0,986**

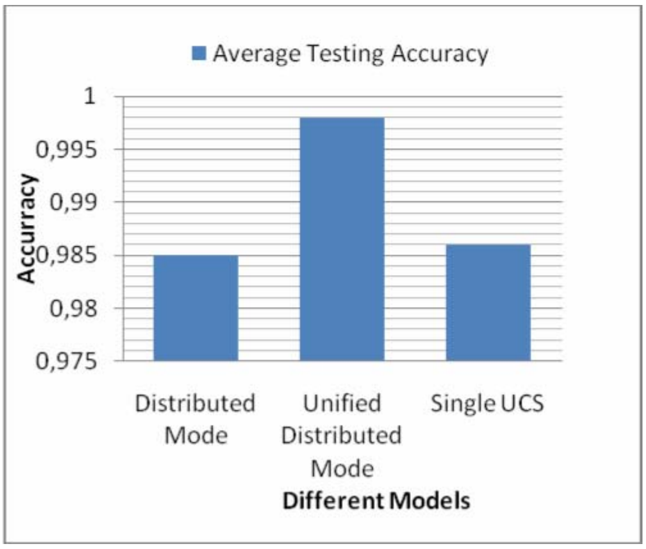

Fig 3 – The accuracy difference between 3 systems.

5 Discussion

5.1 Discussion on central or distributed approach

The option on how the data mining models should be induced in a distributed environment is not yet clear. Both approaches present advantages and limitations as resumed in the Table 2, where:

DMod - Stands for the Distributed Model induction. Each site has a local model that contributes to the global model;
CMod – Refers to the Centralized Model induction. The data is sent to a central repository and then used to induce the global model.

Table 2 – Advantages and limitations of the model induction approaches.

	Advantages	Limitations
DMod	• Speed up on the global model induction • The co-existence of local and centralized models • Accuracy of the models	• The need of efficient merging methods. This is where the LCS technology can play an important role
CMod	• The global model represents all the data	• Communication costs to transfer big quantities of data • Computational costs to process a bigger volume of data

Majority voting can be seen as a contribution in favour of the DMod approach allowing an efficient global model construction from local models during the training phase. The main objection against this approach is the parameter setting (e.g., the threshold T) which tuning can be difficult.

5.2 Comparison of the different models

As mentioned before, this work includes three different models, namely, distributed model, unified distributed model, and single model. Table 1 shows the accuracy of different 10 executions of these 3 different models. All the three models presented good results even though the accuracy of unified distributed model showed a best rate.

It should be noted that in the distributed and unified distributed mode, the population size of each client was the same and they were learned from the same source of data. Moreover all the clients of the distributed site should pass through same generalization process. Even though the learning model of the each client should be depending on the population, each distributed model might have differences. The basic difference between distributed mode and unified distributed mode is the unified distributed mode combined all the learning models from the client but distributed mode not combined learning models. The size of models in distributed system is smaller than the global model of unified distributed model. This implies that unified distributed model is a more knowledgeable model.

The results clearly show that average accuracy of unified distributed model is slightly better than other two models which mean that the distributed approach represents an efficient solution for grid data mining environments using UCS. This approach permits the induction of efficient and locally adjusted models (e.g., to predict local costumer behaviour) and, at the same time, allowing the induction of a global model from the local distributed models (e.g., to predict global costumer behaviour). Furthermore, the global model is induced without the need of a centralized data gathering and processing.

5.3 Related Work

H H. Dam et al [4] introduced an adaptive neural classification system in 2008. The proposed system could overcome the problem of large population as the neural network is employed to represent the action in UCS. In fact, the adaptive version of the ensemble frame work dynamically controls the population size of the system. Hence the results/outcomes do not require any parameter settings. However, supervised learning classifier system has fixed the population size and predefined parameters of all the agents in grid environment.

6 Conclusion and Future work

In this paper, an attempt has been made to study the applicability of supervised learning classifier system in grid data mining applications. The following conclusions can be derived:

- UCS was chosen as a base system for investigation because supervised learning is more suited for data mining applications;
- A comprehensive analysis of UCS in grid computing for data mining indicate that it is useful;
- It is potentially very useful to implement a data mining environment on a grid platform by developing grid services for learning classifier systems.

Two methods have been identified to induce the global data mining model and a set of advantages and limitations for each one of them. A majority voting

approach was proposed to incrementally induce a global model. A comparative study has been conducted in order to compare three approaches to induce a global data mining model. The distributed approach proved to be slightly better in terms of accuracy and more suitable in terms of a qualitative analysis.
Current system only considers homogeneous data. In the future work heterogeneous data by more dynamic learning behaviour integration will be included. In addition, other types of problems will be considered in order to generalize the results obtained so far.

Acknowledgement

The authors would like to express their gratitude's to FCT (Foundation of Science and Technology, Portugal), for the financial support through the contract GRID/GRI/81736/2006.

Reference:

[1] J. Luo., M. Wang., J. Hu., Z. Shi., Distributed data mining on Agent Grid: Issues, Platform and development toolkit. *future Generation computer system 23 (2007) 61-68.*

[2] A. Orriols., EsterBernado-Mansilla.. Class Imbalance Problem in UCS Classifier System: Fitness Adaptation..,*Evolutionary Computation, 2005. The 2005 IEEE congress on, 604-611 Vol.1 2005.*

[3] M. F. Santos., H. Quintela ., J. Naves., Agent-Based Learning Classifier System for Grid Data Mining. *GECCO 2006, July 8-12, 2004, Seattle, WA, USA.*

[4] H. H. Dam., A scalable Evolutionary Learning Classifier System for Knowledge Discovery in Stream Data Mining, M.Sci. University of Western Australia, Australia, B.Sci. (Hons) Curtin University of Technology, Australia. *Thesis work 2008.*

[5] http://mlkd.csd.auth.gr/ddm.html.Consulting on 13/ 3/2009.

[6] J. H. Holmes., P. L.. Lanzi., S. W. Wilson., W. Stolzmunn., Learning Classification System: New Models Successful Applications., *Information Processing Letters, 2002.*

[7] T. Kovacs., Strength or Accuracy: Credit Assignment in Learning Classifier Systems. D*istinguished Dissertation,. Springer-Verlag London limited 2004.*

[8] M. F. Santos. Learning Classifier System in Distributed environments, University of Minho School of Engineering Department of Information System. *PhD Thesis work 1999.*

[8] A. J. Bagnall., G. C. Cawley., Learning Classifier Systems for Data Mining: A Comparison for XCS with other Classifiers for the Forest Cover Data Set. *Neural Networks 2003. Proceeding of the International Joint Conference on.*

[10] G. Brown., T. Kovacs., J. Marshall., UCSpv: Principled Voting in UCS Rule Populations. *GECCO 2007, Genetic and Evolutionary Computation Conference proceeding of the 9th annual conference on Genetic evolutionary computation*.

[11] A. M. Fard., H. Kamyar. Intelligent Agent based Grid Data Mining using Game Theory and Soft Computing., *thesis work 2007.*

[12] A.Silva., P. Cortez, M. F. Santos., Mortality assessment in intensive care unit via adverse events using artificial neural networks. *Artificial intelligence in medicine ISSN 0933.36:3 (2006) 223-234*

[13] M. Y. Santos., A. Moreira., Automatic Classification of Location Contexts with Decision Tree. *Proceeding of the Conference on Mobile and Ubiquitius System, 2006.*

[14] M. V. Butz., S. W. Wilson., An algorithmic description of XCS *Springer Verlag 2002.*

[15] J. Abonyi., A. Abraham., Special Issue Computation Intelligence in Data mining. *Informatica An International Journal of Computing and Informatics 2005 ISSN 0350-5596.*

[16] M. Riedmiller., Supervised Learning in Multilayer Perceptrons - from Backpropagation to Adaptive Learning Techniques. *Computer standards and Interfaces 16(1994).*

[17] L. Bull.,: Application of learning classification system. *Springer-Verlag Berlin Heidelberg 2004.*

[18] C. Clifton., M. Kantarcioglu., J. Vaidya.,: Tool for privacy preserving Distributed Data Mining, *SIGKDD exploration volume 2.*

[19] A. Zanasi., D. A. Gomar, N F F Ebecken., C A Brebbia.,: Data Mining 9, *WIT press.*

[20] M. V. Butz.,: Anticipatory Learning Classifier System, *Klumar Academic publishers.*

[21] M. Plantie., M. Roche., G. Dray., P. Poncelet.,: Is voting Approach Accurate for Opinion Mining? *Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery 2008.*

[22] T. Takei., E. Nakano., Introduction to the Grid Computing. *SX series technical highlight 2003.*

[23] H. Lu., E. Pi., Q. Peng., L. Wang., C. Zhang.,: A particle swarm optimization-aided fuzzy cloud classifier applied for plant numerical taxonomy based on attribute similarity. *Expert system with applications, volume 36, pages 9388-9397 (2009).*

[24] B. Medlock., T. Briscoe., Weakly supervised Learning for Hedge Classification in scientific literature. *Proceedings of the 45th Annual Meeting of the Association of computational Linguistics 2007.*

[25] O. Unold., K. Tuszynski., Mining knowledge from data using Anticipatory Classifier System. *Knowledge-Based System 21(2008) 363-370.*

[26] http://communication.howstuffworks.com/grid-computing.htm .Consulted on 10/3/2009.

[27] http://en.wikipedia.org/wiki/Grid_computing .Consulted on 10/3/2009.

27] http://en.wikipedia.org/wiki/ Cloud computing .Consulted on 10/3/2009.

[29]http://www.dirjournal.com/computers-journal/utility-based-cloud-computing-power Consulted on 10/ 3 /2009.

[30]http://searchenterprisedesktop.techtarget.com/sDefinition/0,,sid192_gci1287881,00.html Consulted on 12/3/ 2009.

[31] http://www.gene-expression-programming.com/webpapers/FerreiraCS2001/Section6/SS5/SSS2.htm Consulted on 08/05/2009.

[32] http://www.cs.bris.ac.uk/Teaching/Resources/COMSM0302/lectures/gp_examples06.p.pdf Consulted on 08/05/2009.

[33] A Orriols.; Further look at UCS Classifier System. *GECCO'06, 2006, Seattle, Washington, USA.*

[34] K. Shafi.; T. Kovacs.; H A. Abbass.; W Zhu.; Intrusion detection with evolutionary learning classifier systems. *Natural computing An International Journal, Springer Science and Business media B.V. 2007.*

[35] M. Caramia., S. Giordani., Resource Allocation in Grid Computing: An Economic Model *WSEAS TRANSACTIONS on COMPUTER RESEARCH ISSN:1991 -8755, Volume 3 2008.*

[36] D P Brown., S W Lee., C Draganova., M Kang., Model Learning Neural Networks. *WSEAS TRANSACTIONS on COMPUTERS, ISSN 1109-2750, Issue 2, Volume 8, 2009.*

[37] J. LI., A Sampling –based Method for Dynamic Scheduling in Distributed Data Mining Environment. *WSEAS TRANSACTIONS on COMPUTERS, ISSN 1109-2750, Issue 1, Volume 8, 2009.*