Chapter 1
The EMADS Extendible Multi-Agent Data Mining Framework

Kamal Ali Albashiri, and Frans Coenen

Abstract In this chapter we describe EMADS, an Extendible Multi-Agent Data mining System. The EMADS vision is that of a community of data mining agents, contributed by many individuals and interacting under decentralized control, to address data mining requests. EMADS is seen both as an end user platform and a research tool. This chapter details the EMADS vision, the associated conceptual framework and the current implementation. Although EMADS may be applied to many data mining tasks; the study described here, for the sake of brevity, concentrates on agent based Association Rule Mining and agent based classification. A full description of EMADS is presented.

Keywords Agent-Driven Data Mining (ADDM), Classifier Generation, Meta Association Rule Mining (MARM).

1.1 Introduction

Agent-Driven Data Mining (ADDM), also known as multi-agent data mining, seeks to harness the general advantageous of MAS to the application domain of DM. It is clear that MAS technology has much to offer DM, particularly in the context of various forms of distributed and cooperative DM. MAS have a clear role in both these areas. MAS technology also offers some further advantageous for ADDM, namely:

- extendibility of DM frameworks
- resource and experience sharing,

Kamal Ali Albashiri, and Frans Coenen
Department of Computer Science, The University of Liverpool,
Ashton Building, Ashton Street, Liverpool L69 3BX, United Kingdom
e-mail: {ali,frans}@csc.liv.ac.uk

- greater end-user accessibility, and
- the addressing of privacy and security issues.

Consequently the MAS approach would support greater end user access to DM techniques. In the context of privacy and (to an extent) security, by its nature data mining is often applied to sensitive data; the MAS approach would allow data to be mined remotely. Similarly, with respect to DM algorithms, MAS can make use of algorithms without necessitating their transfer to users, thus contributing to the preservation of intellectual property rights.

It is suggested in this chapter that a method of addressing the communication requirement of ADDM is to use a system of *mediators* and *wrappers* coupled with an ACL such as FIPA ACL, and that this can more readily address the issues concerned with the variety and range of contexts to which any ADDM system should be applicable.

To investigate and evaluate the expected advantageous of wrappers and mediators, in the context of the disparate nature of ADDM, the authors have developed and implemented (in JADE) an ADDM platform, EMADS (the Extendible Multi-Agent Data mining System). Extendibility is seen as an essential feature of ADDM primarily because it allows the functionality of EMADS to grow in an incremental manner. The vision is of an anarchic collection of agents, contributed to by a community of EMADS users, that exist across an *internet space*, that can negotiate with each other to attempt to perform a variety of data mining tasks (or not if no suitable collection of agents can come together) as proposed by other (or the same) EMADS users. An EMADS demonstrator is currently in operation.

In the context of EMADS three categories of data mining tasks are considered to exist: classification, clustering and Association Rule Mining (ARM). The current EMADS demonstrator includes ARM and classification agents. To evaluate the operation of EMADS two data mining scenarios are considered in this chapter. The first (Sub-Section 1.5.1) is a distributed merge-mining scenario where EMADS agents are used to merge the results of a number of ARM operations, a process referred to as meta-ARM, to produce a global set of Association Rules (ARs). The challenge here is to minimise the communication overhead, a significant issue in distributed DM (regardless of whether it is implemented in an agent framework or not); this is also an issue in parallel DM. The second scenario (reported in Sub-Section 1.5.2) is a classification scenario where the objective is to generate a classifier (predictor) fitted to a, EMADS user, specified dataset. The aim of this second scenario is to identify a "best" classifier given a particular dataset.

In summary the chapter describes an operational ADDM framework, EMADS. The framework is currently in use and is providing a useful facility, not only to achieve ADDM, but as a platform for conducting ADDM research.

The rest of this chapter is organized as follows. A brief review of some related work on ADDM is presented in Section 1.2. The conceptual framework for EMADS is presented in Section 1.3. The current implementation of EMADS, together with an overview of the wrapper principle is given in 1.4. The operations of EMADS are illustrated in Section 1.5 with a Meta ARM and classification scenarios. Some conclusions are presented in Section 1.6.

1.2 Related Work

Agent-based systems have shown much promise for flexible, fault-tolerant, distributed problem solving. Much of the foundational work on agent technology has focused on inter-agent communication protocols, patterns of conversation for agent interactions, and basic facilitation capabilities. Some ADDM frameworks consider agents to be relatively trivial models for single platform execution. Others focus on developing complex features for specific DM task, while providing little support in the context of usability or extendibility. The success of peer-to-peer systems and negotiating agents has engendered a demand for more generic, flexible, robust frameworks.

There have been only few ADDM systems directed at such a generic framework. An early example was IDM [7], a multi-agent architecture for direct DM to help businesses gather intelligence about their internal commerce agent heuristics and architectures for KDD. In [4] a generic task framework was introduced, but designed to work only with spatial data. The most recent system was introduced in [11] where the authors proposed a multi-agent system to provide a general framework for distributed DM applications. In this system the effort to embed the logic of a specific domain has been minimized and is limited to the customization of the user. However, although its customizable feature is of a considerable benefit, it still requires users to have very good DM knowledge.

1.3 The EMADS Conceptual Framework

Conceptually EMADS is a hybrid peer to peer agent based system comprising a collection of collaborating agents that exist in a set of containers. Agents may be created and contributed by any EMADS user/contributor. The implementation includes a “main container” that houses a number of housekeeping agents that have no particular connection with ADDM, but provide various facilities to maintain the operation of EMADS. In particular the main container holds an Agent Management System (AMS) agent and a Directory Facilitator (DF) agent. The terminology used is taken from the JADE (Java Agent Development) framework [5] in which EMADS is implemented. Briefly the AMS agent is used to control the life cycles of other agents in the platform, and the DF agent provides an agent lookup service. Both the main container and the remaining containers can hold various DM agents. Note that the EMADS main container is located on the EMADS host organisation site (currently the University of Liverpool in the UK), while other containers may be held at any other sites worldwide.

EMADS agents are responsible for accessing local data sources and for collaborative data analysis. EMADS includes: (i) data mining agents, (ii) data agents, (iii) task agents, (iv) user agents, and (v) mediators (JADE agents) for agents coordination. The data and mining agents are responsible for data accessing and carrying through the data mining process; these agents work in parallel and share information

through the task agent. The task agent co-ordinates the data mining operations, and presents results to the user agent. Data mining is carried out by means of local data mining agents (for reasons of privacy preservation).

1.3.1 EMADS End User Categories

EMADS has several different modes of operation according to the nature of the participant. Each mode of operation (participant) has a corresponding category of user agent. Broadly, the supported categories are as follows:

- **EMADS Users**: Participants, with restricted access to EMADS, who may pose data mining requests.
- **EMADS Data Contributors**: Participants, again with restricted access, who are prepared to make data available to be used by EMADS mining agents.
- **EMADS Developers**: Developers are EMADS participants, who have full access and may contribute data mining algorithms.

Note that in each case, before interaction with EMADS can commence, appropriate software needs to be downloaded and launched by the participant. Note also that any individual participant may be a user as well as a contributor and/or developer at the same time.

Conceptually the nature of EMADS data mining requests, that may be posted by EMADS users, is extensive. In the current implementation, the following types of generic request are supported:

- Find the “best” classifier (to be used by the requester at some later date in off line mode) for a data set provided by the user.
- Find the “best” classifier for the indicated data set (i.e. provided by some other EMADS participant).
- Find a set of Association Rules (ARs) contained within the data set(s) provided by the user.
- Find a set of Association Rules (ARs) contained within the indicated type of data set(s) (i.e. provided by other EMADS participants).

The Association Rule Mining (ARM) style of request is discussed further in Sub-Section 1.5.1. The idea was that an agent framework could be used to implement a form of Meta-ARM where the results of the parallel application of ARM to a collection of data sets, with not necessarily the same schema but conforming to a global schema, are combined. Details of this process can be found in Albashiri et al. [2, 3]. A “best” classifier is defined as a classifier that will produce the highest accuracy on a given test set (identified by the mining agent) according to the detail of the request. To obtain the “best” classifier EMADS will attempt to access and communicate with as many classifier generator DM agents as possible and select the best result. The classification style of user request will be discussed further in Sub-Section 1.5.2 to illustrate the operation of EMADS in more detail.

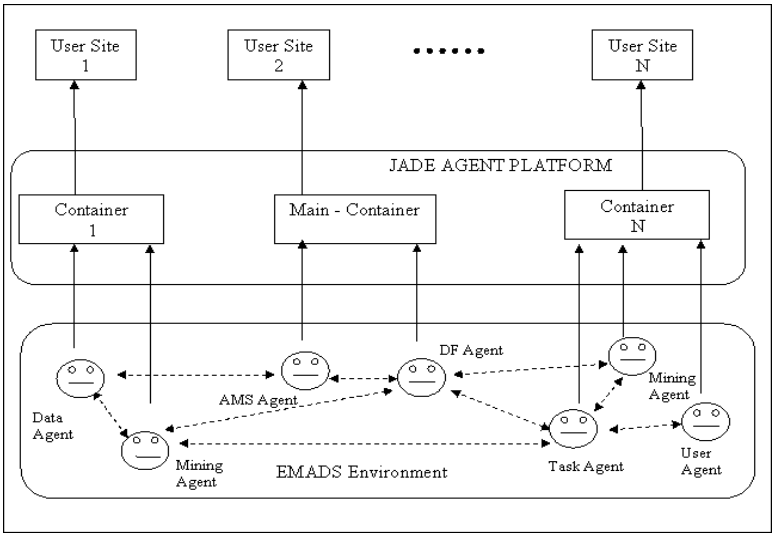


Fig. 1.1 EMADS Architecture as Implemented in Jade

1.4 The EMADS Implementation

EMADS is implemented using the JADE framework. JADE is FIPA (Foundation for Intelligent Physical Agents) [10] compliant middleware that enables development of peer to peer applications based on the agent paradigm. JADE defines an agent platform that comprises a set of containers, which may be distributed across a network as in the case of EMADS. A JADE platform includes a main container in which is held a number of mandatory agent services. These include the AMS and DF agents whose functionality has already been described in Section 1.3. Recall that the AMS agent is used to control the lifecycles of other agents in the platform, while the DF agent provides a lookup service by means of which agents can find other agents. When a mining or data agent is created, upon entry into the system, it announces itself to the DF agent after which it can be recognized and found by other agents.

Fig. 1.1 gives an overview of the implementation of EMADS using JADE. The figure is divided into three parts: at the top are listed N user sites. In the middle is the JADE platform holding the main container and N other containers. At the bottom a sample collection of agents is included. The solid arrows indicates a “belongs to” (or “is held by”) relationship while the dotted arrows indicate a “communicates with” relationship. Thus the data agent at the bottom left belongs to container 1 which in turn belongs to User Site 1; and communicates with the AMS agent and (in this example) a single mining agent. The principal advantage of this JADE architecture is that it does not overload a single host machine, but distributes the processing load among multiple machines.

1.4.1 EMADS Wrappers

One of the principal objectives of EMADS is to provide an easily extendible framework that can readily accept new data sources and new data mining techniques. In general, extendibility can be defined as the ease with which software can be modified to adapt to new requirements or changes in existing requirements. Adding a new data source or data mining techniques should be as easy as adding new agents to the system. The desired extendibility is achieved by a system of wrappers. EMADS wrappers are used to "wrap" up data mining artifacts so that they become EMADS agents and can communicate with other EMADS agents. As such EMADS wrappers can be viewed as agents in their own right that are subsumed once they have been integrated with data or tools to become data or data mining agents. The wrappers essentially provide an application interface to EMADS that has to be implemented by the end user, although this has been designed to be a fairly trivial operation. Two broad categories of wrapper have been defined:

- **Data wrappers:** Data wrappers are used to "wrap" a data source and consequently create a data agent. Broadly a data wrapper holds the location (file path) of a data source, so that it can be accessed by other agents; and meta information about the data. To assist end users in the application of data wrappers a data wrapper GUI is available. Once created, the data agent announces itself to the DF agent as consequence of which it becomes available to all EMADS users.
- **Tool wrappers:** Tool wrappers are used to "wrap" up data mining software systems and thus create mining agents. Generally the software systems will be data mining tools of various kinds (classifiers, clusters, association rule miners, etc.) although they could also be (say) data normalization/discretization or visualization tools. It is intended that EMADS will incorporate a substantial number of different tool wrappers each defined by the nature of the desired I/O which in turn will be informed by the nature of the generic data mining tasks that it is desirable for EMADS to be able to perform.

Currently the research team has implemented two tool wrappers: the binary valued data, single label, classifier generator and the data normalization/discretization wrapper. However, many more categories of tool wrapper can be envisaged. Mining tool wrappers are more complex than data wrappers because of the different kinds of information that needs to be exchanged.

In the case of a *binary valued, single label, classifier generator* wrapper the input is a binary valued data set together with meta information about the number of classes and a number slots to allow for the (optional) inclusion of threshold values. The output is then a classifier expressed as a set of Classification Rules (CRs). As with data agents, once created, the data mining agent announce themselves to the DF agent after which they will becomes available for use to EMADS users.

In the case of the data normalization/discretization wrapper, the LUCS-KDD (Liverpool University Computer Science - Knowledge Discovery in Data) ARM

DN (Discretization/ Normalization) software [1] is used to convert data files, such as those available in the UCI data repository [6], into a binary format suitable for use with Association Rule Mining (ARM) applications. This tool has been "wrapped" using the data normalization/discretization wrapper.

1.5 EMADS Operations

In the following two sub-sections the operation of EMADS is illustrated using two DM scenarios: Meta Association Rule Mining and Classification.

1.5.1 Meta ARM (Association Rule Mining) scenario

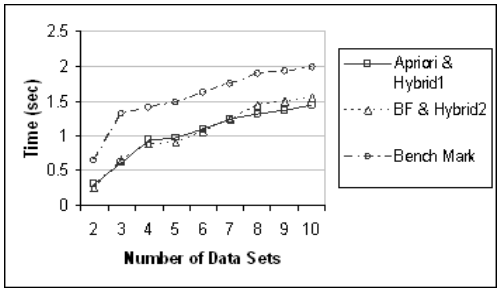
The term Meta Mining is defined, in the context of EMADS, as the process of combining individually obtained results of N applications of a DM activity. The motivation behind the scenario is that data relevant to a particular DM application may be owned and maintained by different, geographically dispersed, organizations. Information gathering and knowledge discovery from such distributed data sources typically entails a significant computational overhead; computational efficiency and scalability are both well established critical issue in data mining [12]. One approach to addressing problems, such as the Meta ARM problem, is to adopt a distributed approach. However this entails expensive computation and communication costs. In distributed data mining, there is a fundamental tradeoff between accuracy and computation cost. If we wish to improve the computation and communication costs, we can process all the data locally obtaining local results, and combine these results centrally to obtain the final result. If our interest is in the accuracy of the result, we can ship all the data to a single node (and apply an appropriate algorithm to produce this desired result). In general the latter is more expensive while the former is less accurate. The distributed approach also entails a critical security problem in that it reveals private information; privacy preserving issues [1] are of major concerns in inter enterprise data mining when dealing with private databases located at different sites.

1.5.1.1 Dynamic Behaviour of EMADS for Meta ARM operations

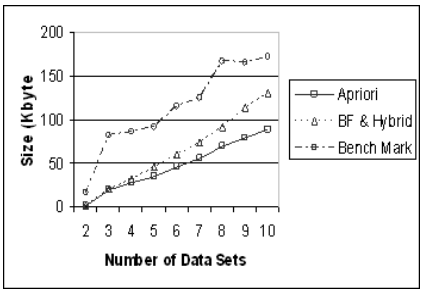
The meta ARM scenario comprises a set of N data agents, N ARM mining agents and a meta ARM agent. Note that each ARM mining agent could have a different ARM algorithm associated with it, however, it is assumed that a common data structure is used to facilitate data interchange. For the scenario described here the

[1] $http://www.csc.liv.ac.uk/\tilde{f}rans/KDD/Software/$

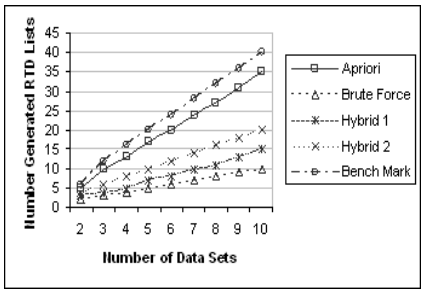
common data structure is a T-tree [8], a *set enumeration* tree structure for storing item sets. Once generated the N local T-trees are passed to the Meta ARM agent which creates a global T-tree. Each of the Meta ARM algorithms makes use of *return to data* (RTD) lists, one per data set, to contain lists of itemsets whose support was not included in the current T-tree and for which the count is to be obtained by a return to the raw data. During the global T-tree generation process the Meta ARM agent interacts with the various ARM agents in the form of the exchange of RTD lists. There are a number of strategies that can be adopted with respect to when in the process the RTD lists should be exchanged; the research team identified four distinct strategies (Apriori, Brute Force, Hybrid 1 and Hybrid 2). A full description of the algorithms can be found in [2].



(a) Processing Time



(b) Total size of RTD lists



(c) Number of RTD lists

Fig. 1.2 Effect of number of data sources.

1.5.1.2 Experimentation and Analysis

To evaluate the five Meta ARM algorithms, in the context of EMADS, a number of experiments were conducted. The experiments were designed to analyze the effect of the following: (i) the number of data sources (data agents), (ii) the size of the

datasets (held at data agents) in terms of number of records, and (iii) the size of the datasets (held at data agents) in terms of number of attributes.

Experiments were run using two Intel Core 2 Duo E6400 CPU (2.13GHz) computers with 3GB of main memory (DDR2 800MHz), Fedora Core 6, Kernel version 2.6.18 running under Linux except for the first experiment where two further computers running under Windows XP were added. For each of the experiments we measured: (i) processing time (seconds/mseconds), (ii) the size of the RTD lists (Kbytes), and (iii) the number of RTD lists generated.

Fig. 1.2 shows the effect of adding additional data sources using the four Meta ARM algorithms and the bench mark algorithm. For this experiment ten different artificial data sets were generated and distributed among four machines using T = 4 (average number of items per transactions), N = 20 (Number of attributes), D=100k (Number of transactions). Note that the slight oscillations in the graphs result simply from a vagary of the random nature of the test data generation. For other experiments results readers are referred to [3].

Fig. 1.2 also indicates, at least with respect to Meta ARM, that EMADS offers positive advantages in that all the Meta ARM algorithms were more computationally efficient than the bench mark algorithm. The results of the analysis also indicated that the Apriori Meta ARM approach coped best with a large number of data sources, while the Brute Force and Hybrid 1 approaches coped best with increased data sizes (in terms of column/rows).

1.5.2 Classifier Generation scenario

In this section the operation of EMADS is further illustrated in the context of a classifier generation task; however much of the discussion is equally applicable to other generic data mining tasks such as clustering and ARM. The scenario is that of an end user who wishes to obtain a "best" classifier founded on a given, pre-labeled, data set; which can then be applied to further unlabelled data. The assumption is that the given data set is binary valued and that the user requires a single-label, as opposed to a multi-labeled, classifier. The request is made using the individual's user agent which in turn will spawn an appropriate task agent. For this scenario the task agent interacts with mining agents that hold *single labeled classifier* generators that take binary valued data as input. Each of these mining agents is then accessed and a classifier, together with an accuracy estimate, requested. Once received the task agent selects the classifier with the best accuracy and returns this to the user agent. The data mining agent wrapper in this case provides the interface that allows input of: (i) the identifier for the data set to be classified, and (ii) the number of class attributes (a value that the mining agent cannot currently deduce for itself); while the user agent interface allows input for threshold values (such as support and confidence values). The output is a classifier together with an accuracy measure. To obtain the accuracy measures the classifier generators (data mining agents) build

their classifiers using the first half of the input data as the "training" set and the second half of the data as the "test" set.

From the literature there are many reported techniques available for generating classifiers. For the scenario reported here the authors used implementations of eight different algorithms[2]:

1. FOIL (First Order Inductive Learner) [15]: The well established inductive learning algorithm for the generation of Classification Association Rules (CARs).
2. TFPC (Total From Partial Classification): A CAR generator [9] founded on the P and T-tree set enumeration tree data structures.
3. PRM (Predictive Rule Mining) [16]: An extension of FOIL.
4. CPAR (Classification based on Predictive Association Rules) [16]: A further development from FOIL and PRM.
5. IGDT (Information Gain Decision Tree) classifier: An implementation of the well established C4.5 style of decision tree based classifier using information gain as the "splitting criteria".
6. RDT (Random Decision Tree) classifier: A decision tree based classifier that uses most frequent current attribute as the "splitting criteria".
7. CMAR (Classification based on Multiple Association Rules): A well established Classification Association Rule Mining (CARM) algorithm [13].
8. CBA (Classification Based on Associations): Another well established CARM algorithm [14].

These were placed within an appropriately defined tool wrapper to produce eight (single label binary data classifier generator) data mining agents. This was a trivial operation indicating the versatility of the wrapper concept.

Thus each mining agent's basic function is to generate a classification model using its own classifier and provide this to the task agent. The task agent then evaluates all the classifier models and chooses the most accurate model to be returned to the user agent to be presented to the user.

1.5.2.1 Experimentation and Analysis

To evaluate the classification scenario, as described above, a sequence of data sets taken from the UCI machine learning data repository [6] were used (preprocessed by data agents so that they were discretized/normalized into a binary valued format). The results are presented in Table 1.5.2 Each row in the table represents a particular request and gives the name of the data set, the selected best algorithm as identified from the interaction between the EMADS agents, the resulting best accuracy and the total EMADS execution time from creation of the initial task agent to the final "best" classifier being returned to the user agent. The naming convention used in the Table is that: D equals the number of attributes (after discretization/normalization),

[2] Taken from the LUCS-KDD software repository at *http://www.csc.liv.ac.uk/~frans/KDD/Software/*

Data Set	Classifier	Accuracy	Generation Time (sec)
connect4.D129.N67557.C3	RDT	79.76	502.65
adult.D97.N48842.C2	IGDT	86.05	86.17
letRecog.D106.N20000.C26	RDT	91.79	31.52
anneal.D73.N898.C6	FOIL	98.44	5.82
breast.D20.N699.C2	IGDT	93.98	1.28
congres.D34.N435.C2	RDT	100	3.69
cylBands.D124.N540.C2	RDT	97.78	41.9
dematology.D49.N366.C6	RDT	96.17	11.28
heart.D52.N303.C5	RDT	96.02	3.04
auto.D137.N205.C7	IGDT	76.47	12.17
penDigits.D89.N10992.C10	RDT	99.18	13.77
soybean-large.D118.N683.C19	RDT	98.83	13.22
waveform.D101.N5000.C3	RDT	96.81	11.97

Table 1.1 Classification Results

N the number of records and C the number of classes (although EMADS has no requirement for the adoption of this convention).

The results demonstrate firstly that EMADS can usefully be adopted to produces a best classifier from a selection of classifiers. Secondly that operation of EMADS is not significantly hindered by agent communication overheads, although this has some effect. Generation time, in most cases does not seem to be an issue, so further classifier generator mining agents could easily be added. The results also reinforce the often observed phenomena that there is no single best classifier generator suited to all kinds of data set.

1.6 Conclusions

This chapter describes EMADS, a multi-agent framework for data mining. The principal advantages offered are that of experience and resource sharing, flexibility and extendibility, and (to an extent) protection of privacy and intellectual property rights.

This chapter presented the EMADS vision, the associated conceptualization and the JADE implementation. Of particalar note is the use of wrappers to incorporate existing software into EMADS. Experience indicates that, given an appropriate wrapper, existing data mining software can be very easily packaged to become an EMADS data mining agent. The EMADS operation was illustrated using Meta ARM and classification scenarios.

A good foundation has been established for both data mining research and genuine application based data mining. It is acknowledged that the current functionality of EMADS is limited to classification and Meta-ARM. The research team is at present working towards increasing the diversity of mining tasks that EMADS can address. There are many directions in which the work can (and is being) taken

forward. One interesting direction is to build on the wealth of distributed data mining research that is currently available and progress this in an MAS context. The research team is also enhancing the system's robustness so as to make it publicly available. It is hoped that once the system is live other interested data mining practitioners will be prepared to contribute algorithms and data.

References

1. Aggarwal, C. and Yu, P., "A Condensation Approach to Privacy Preserving DataMining". Lecture Notes in Computer Science, Vol. 2992, pp. 183-199, (2004).
2. Albashiri, K., Coenen, F., Sanderson, R. and Leng. P., "Frequent Set Meta Mining: Towards Multi-Agent Data Mining". In Bramer, M., Coenen, F.P. and Petridis,M. (Eds.), Research and Development, Springer, London, pp139-151, (2007).
3. Albashiri, K., Coenen, F., and Leng. P., "Agent Based Frequent Set Meta Mining: Introducing EMADS". AI in Theory and Practice II, Proceedings IFIP, Springer, pp23-32, (2007).
4. Baazaoui H., Faiz S., Hamed R., and Ghezala H., "A Framework for data mining based multi-agent: an application to spatial data". 3rd World Enformatika Conference, Istanbul, (2005).
5. Bellifemine, F. Poggi, A. and Rimassi, G., "JADE: A FIPA-Compliant agent framework". Proceedings Practical Applications of Intelligent Agents and Multi-Agents, pg 97-108, (1999). (See http://sharon.cselt.it/projects/jade for latest information).
6. Blake, C. and Merz, C. "UCI Repository of machine learning databases". (1998). Irvine, CA: University of California, Department of Information and Computer Science. http://www.ics.uci.edu/mlearn/MLRepository.html
7. Bose, R. and V. Sugumaran, "IDM: An Intelligent Software Agent Based Data Mining Environment". In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 2888-2893 San Diego, CA: IEEE Press, (1998).
8. Coenen, F., Leng, P., and Goulbourne, G., "Tree Structures for Mining Association Rules". Journal of Data Mining and Knowledge Discovery, Vol 8, No 1, pp25-51, (2004).
9. Coenen, F., Leng, P. and Zhang, L. "Threshold Tuning for Improved Classification Association Rule Mining". Proceeding PAKDD, LNAI3158, Springer, pp216-225, (2005).
10. Foundation for Intelligent Physical Agents, FIPA 2002 Specification. Geneva, Switzerland. (See http://www.fipa.org/specifications/index.html).
11. Giuseppe, D., Giancarlo, F., "A customizable multi-agent system for distributed data mining". Proceedings of the 2007 ACM symposium on applied computing, Pages: 42 47, (2007).
12. Kamber, M.,Winstone, L.,Wan, G., Shan, S. and Jiawei, H., "Generalization and Decision Tree Induction: Efficient Classification in Data Mining". Proceedings of the Seventh International Workshop on Research Issues in Data Engineering, pp.111-120, (1997).
13. Li W., Han, J. and Pei, J., "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules". Proceedings ICDM, pp369-376, (2001).
14. Liu, B. Hsu, W. and Ma, Y., "Integrating Classification and Assocoiation Rule Mining". Proceedings KDD-98, New York, 27-31 August. AAAI. pp80-86, (1998).
15. Quinlan, J. R. and Cameron-Jones, R. M., "FOIL: A Midterm Report". Proceedings ECML, Vienna, Austria, pp3-20. (1993).
16. Yin, X. and Han, J., "PAR: Classification based on Predictive Association Rules". Proceedings SIAM Int. Conf. on Data Mining (SDM'03), San Fransisco, CA, pp. 331-335, (2003).