Multi-Agent Data Mining with Negotiation: A Study in Multi-Agent Based Clustering

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy
by
Santhana Chaimontree

June 2012

Abstract

Multi-Agent Data Mining (MADM) seeks to harness the general advantages offered by Multi-Agent System (MAS) with respect to the domain of data mining. The research described in this thesis is concerned with Multi-Agent Based Clustering (MABC), thus MADM to support clustering. To investigate the use of MAS technology with respect to data mining, and specifically data clustering, two approaches are proposed in this thesis. The first approach is a multi-agent based approach to clustering using a generic MADM framework whereby a collection of agents with different capabilities are allowed to collaborate to produce a "best" set of clusters. The framework supports three clustering paradigms: *K*-means, *K*-NN and divisive hierarchical clustering. A number of experiments were conducted using benchmark UCI data sets and designed to demonstrate that the proposed MADM approach can identify a best set of clusters using the following clustering metrics: F-measure, Within Group Average Distance (WGAD) and Between Group Average Distance (BGAD). The results demonstrated that the MADM framework could successfully be used to find a best cluster configuration. The second approach is an extension of the proposed initial MADM framework whereby a "best" cluster configuration could be found using cooperation and negotiation among agents. The novel feature of the extended framework is that it adopts a two-phase approach to clustering. Phase one is similar to the established centralised clustering approach (except that it is conducted in a decentralised manner). Phase two comprises a negotiation phase where agents "swap" unwanted records so as to improve a cluster configuration. A set of performatives is proposed as part of a negotiation protocol to facilitate intra-agent negotiation. It is this negotiation capability which is the central contribution of the work described in this thesis. An extensive evaluation of the extended framework was conducted using: (i) benchmark UCI data sets and (ii) a welfare benefits data set that provides an exemplar application. Evaluation of the framework clearly demonstrates that, in the majority of cases, this negotiation phase serves to produce a better cluster configuration (in terms of cohesion and separation) than that produced using a simple centralised approach.

Contents

List of Figures

List of Tables

Acknowledgement

It is with immense gratitude that I acknowledge the support and guidance of my main supervisor, Dr. Frans Coenen. His enthusiasm, constructive criticism, research ideas and always making time for discussion have made the completion of my PhD possible and highly enjoyable. I would also like to express my gratitude to my second supervisor, Dr. Katie Atkinson, for her assistance, suggestions and constructive feedback.

In addition, I would also like to convey thanks to all who have provided me with financial assistance. My PhD research has been funded by the Ministry of Science and Technology of the Royal Thai Government and I am most grateful for this support. Additionally, the Department of Computer Science at the University of Liverpool within the Faculty of Science and Engineering has provided me with financial assistance to enable me to attend a number of conferences/workshops and present my work throughout my years of studying. I am grateful to all parties for this assistance.

Special thanks also to all my colleagues, especially group members; Stephanie Chua, Kamal Ali Albashiri, Maya Wardeh, Puteri Nor Ellyza Nohuddin, Mohd Hanafi bin Ahmad Hijazi, Chuntao (Geof) Jiang, Matias Garcia Constantino and Ayesh Alshukri for sharing the literature and invaluable assistance.

Finally, I wish to express my love and gratitude to my beloved family for their understanding and endless love, though the duration of my study.

Chapter 1

Introduction

An agent is a software entity located in some environment where it is able to operate in an autonomous manner to achieve some specific end goal [89]. By combining a number of such agents to interact with each other, the problem-solving power of agents can be significantly enhanced. Such a system is referred to as a Multi-Agent System (MAS). An important aspect of MASs is the necessary communication between agents so that they can cooperate and negotiate in order to carry out collectively some allotted task. MASs are seen to offer a number of advantages over more conventional distributed and parallel systems. These include:

- Decentralise control
- Autonomy
- Expertise sharing
- Resource sharing
- Scalability
- Enhance performance

Knowledge Discovery in Databases (KDD) and Data Mining play an important role in the information industry and society because of the availability of large amounts of data [44]. For many commercial and civic institutions the quantity of data that has been collected is substantial. KDD is concerned with the discovery of hidden but worthwhile knowledge in data. The term describes a process that includes data warehousing (cleaning) and interpretation and/or visualisation of results. Data mining is the central phase within the overall KDD process concerned with the discovery of hidden knowledge. Knowledge in this context can be patterns, rules or relationships. The field of data mining can be divided into two subfields: *supervised learning* and *unsupervised learning*. In supervised learning, labelled data is used to inform the learning process. A typical example is the generation of classifiers using pre-labelled data. The need for pre-labelled data imposes limitations on the use of supervised techniques. Unsupervised learning, in turn, does not require the provision of pre-labelled data to support the data mining process. One common example of unsupervised learning is *data clustering* where the data is grouped into a number of clusters according to some similarity metrics, similar data objects are placed in the same cluster while the dissimilar objects are placed in different clusters. It is interesting to note that given a definition of a set of clusters (for example in terms of a set of prototypes), this definition can be used for classification purposes.

Over the last twenty years data mining has become a well established field of study which has found application in many domains [44]. However research challenges still remain. Current issues include [91]: (i) the ability to deal with high dimensional data, high speed data streams, complex data, sequence data, time series data and distributed data; (ii) the maintenance of the security/privacy of data; (iii) data integration and (iv) scalability.

It is conjectured that data mining and MAS technology have much to offer one another [23, 22, 24]. There are two main paradigms for this integration:

- Data mining-driven agents: The use of data mining to support the abilities of agents with respect to (say) adaptation, coordination, learning and/or reasoning.
- Agent-driven data mining (also known as Multi-Agent Data Mining (MADM) or Agent Assisted Data Mining (AADM)[1]): The use of collections of agents to perform collaborative data mining tasks.

This thesis is concerned with MADM, that is the use of MAS technology to support data mining. More specifically, to act as focus for the work, the research described in this thesis is directed at Multi-Agent Based Clustering (MABC).

The rest of this chapter is organised as follows. Section 1.1 provides an overview of the motivation for the research. The research question and associated issues to be addressed are then described in Section 1.2. The adopted research methodology is then presented in Section 1.3, followed by an overview of the contribution of the work described in Section 1.4. The structure of the remainder of the thesis is then described in Section 1.5. A brief summary is given in Section 1.6.

1.1 Motivation

This section presents the motivation for the described work. In particular this section seeks to establish the advantages that MASs can offer data mining. In essence MADM seeks to harness the general advantages offered by MASs with respect to the domain of

[1]In this thesis the acronym MADM will be adopted

data mining. In the context of this thesis these advantages are considered with respect to Multi-Agent Based Clustering (MABC).

Recall the generally accepted advantages offered by MASs with respect to distributed cooperative computing and resource sharing listed previously. From a high level viewpoint these advantages all have something to offer data mining of which the decentralisation property of MAS can be argued to offer the most significant advantages with respect to data mining processes where it is often necessary to process very large amounts of geographically distributed data. Of course, for many applications it may be appropriate to move the data to a centralised storage structure (such as a data warehouse) and perform data mining in a centralised manner. However, in many cases this can be inefficient and/or result in data security and privacy preserving issues. MADM provides a potential solution. An example of a MABC application that preserves data privacy and minimise communication between sites can be found in [51]; where an agent based distributed data clustering mechanism is described founded on a distributed density based clustering algorithm whereby density estimation samples are transmitted, instead of actual data values.

Autonomy is an important advantage offered by MAS that allows agents to operate in a self directed manner. This advantage is equally applicable to MADM. There are three main capabilities that support autonomy; reactive, proactive and social capability [89]. Reactive capabilities allows agents to perform tasks (such as data mining tasks) and produce results when requested to do so. Proactive capabilities allow agents to predict likely requests. Social capabilities allow agents to communicate and negotiate with one another, exchange information and share expertise (for example expertise in the context of clustering algorithms) and resources (for example clustering results). Central to the work described in this thesis is the ability of agents to be able to negotiate with one another.

The advantages offered by MAS with respect to expertise and resource sharing are self-evident. The scalability advantage that is a feature of MAS is also of particularly significant to data mining as the amount of data available for mining increases rapidly year on year, and new techniques or variations of existing techniques constantly become available. One solution to the increase in the quantity of data available for mining is to adopt a parallel or distributed approach. This can also be achieved using a MAS approach, but with the additional other benefits that MAS has to offer (as noted above). MASs are well suited to supporting distributed/parallel data mining because of the "added value" that they provide in terms of mechanisms to support the interchanging of information and coordinating activities. With respect to the evolution of new data mining techniques MAS architectures readily support the inclusion of additional agents supporting new technology. MAS architectures provides an extremely versatile platform.

Finally, MASs have the ability to enhance performance in terms of: (i) *computational efficiency* because of the concurrency of computation; (ii) *reliability* in that MASs can auto-recover, in the event of "crashed" agents, by dynamically finding redundant agents or appropriate alternative agents; (iii) *extensibility*, by allowing new agents to be added to the systems; (iv) *maintainability* because of the modularity of MASs; and (v) *flexibility*, because agents with different abilities can contribute to disparate tasks.

There is no a generic clustering algorithm that suited to every data set, hence different clustering algorithms perform differently and produce different clustering results depending on the characteristic of a given data set. The clustering results often are suboptimal. Thus the proposed intra-negotiation mechanism offers the additional advantage that it can be used to refine an initial clustering result in order to improve the cluster configuration in terms of cluster cohesion and separation. This intra-agent negotiation, when incorporated into an MABC framework, such as that proposed in this thesis, serves to harness the true potential of MASs rather than simply using the concept of MASs to achieve a form of distributed clustering.

1.2 Research Issues

The aim of the research described in this thesis is to investigate the use of MAS technology with respect to data mining, and more specifically clustering, so as to establish the advantages that MAS can offer data mining. The research question to be addressed is:

> "How do we obtain a best cluster configuration using the full power of MAS to support unsupervised learning and specifically clustering?"

The provision of an answer to this research question, in turn, entails the resolution of a number of research issues:

1. How can a MAS identify the most appropriate number of clusters with respect to a given data set?
2. What is the nature of a MAS framework to support data mining and clustering in particular?
3. How do a group of agents know when they have generated a "best" cluster configuration?
4. What are the specific MAS mechanisms that can usefully be adopted to support MADM?
5. What are the most appropriate cooperation and negotiation protocols for MABC?

1.3 Methodology

The broad research methodology that has been adopted was to first investigate and build a multi-agent platform to support the investigation of MABC. Once this platform was established, the incorporation of a sequence of clustering paradigms was considered: K-means [57], K-NN [27] and divisive hierarchical clustering [50]. These paradigms were selected because they are well established, feature extensively in the literature, and have been used as the foundation for many enhancements and variations. Two communication mechanisms were considered. The first was an ontology based communication mechanisms used with respect to simple scenarios to identify a best cluster configuration. The second was founded on the concept of dialogues as found in, for example, work on argumentation. It was found that the second mechanism much more readily supported intra-agent communication and especially negotiation. The basic idea behind the negotiation was that when a group of agents had established an initial cluster configuration they could negotiate to improve this cluster configuration by exchanging records between them. A negotiation protocol was therefore defined and implemented to support the desired intra-agent communication.

Evaluation of the initial framework, the inclusion of the different paradigms and the negotiation mechanism, was conducted using benchmark data sets and a welfare benefit data set that was used as the exemplar application. The evaluation was conducted in terms of the cluster configuration accuracy with respect to known "ground-truth" configurations.

1.4 Contributions

The main contributions of the research work considered in this thesis can be summarised as follows:

- A generic approach to multi-agent based clustering to identify a best set of clusters using a collection of "clustering agents".
- The design of a MABC framework, a generic MADM environment that uses a "bespoke" data mining ontology.
- The use of clustering metrics to evaluate the quality of cluster configurations generated by a MABC and hence identifying the best one.
- A comparison of two measures, Within Group Average Distance (WGAD) and Between Group Average Distance (BGAD) which are based on cluster cohesion and cluster separation measures respectively, to identify the most appropriate set of clusters for a given clustering problem.

- Mechanisms to determine the most appropriate parameters for a given clustering algorithm using clustering metrics.
- A versatile multi-agent based clustering framework.
- A set of communication performatives specifically to support negotiation within multi-agent based clustering.
- A negotiation protocol whereby agents can interact with one another with the express aim of exchanging records to improve an initial cluster configuration.

The most significant contributions of the work is the negotiation capability, to support MABC, that has been built into the proposed MADM framework.

1.5 Structure of Thesis

The remainder of this thesis is structured into seven chapters as follows. Chapter 2 presents a literature survey of existing research relevant to the contributions presented in this thesis. Chapter 3 introduces a generic Multi-Agent Data Mining (MADM) framework, that can be used for MABC, that allows a collection of agents to collaborate to produce a "best" cluster configuration. Chapter 4 proposes an extension of the proposed initial MADM framework introduced in Chapter 3 whereby a "best" cluster configuration could be found using cooperation and negotiation among agents. This chapter also gives a detailed description of a set of communication performatives which is proposed as part of a negotiation protocol to facilitate intra-agent negotiation. The operation of the proposed extended framework is described in terms of both the communication performatives used and from an algorithmic perspective. Some examples of the operation of the proposed extended framework are also presented in this chapter. Chapter 5 presents the evaluation of the extended MADM framework, and especially the negotiation mechanism that is incorporated into the extended framework. Chapter 6 presents a number of conducted experiments to automatically identifying clustering parameters. A summary of the main findings and a discussion of possible avenues for future research work are then presented in Chapter 7.

1.6 Summary

This chapter has provided the necessary context and background to the research described in this thesis. The motivations for the research, the specific research objectives, and the methodology for realising these objectives, have been described. A literature review of the relevant previous work, with respect to this thesis, is presented in the following chapter.

Chapter 2

Literature Review

This chapter presents a discussion and a critical review of the current research relating to Multi-Agent Data Mining (MADM). The organisation of this chapter is as follows. Section 2.1 presents a general description of agents and MADM systems including a review of intra-agent communication and the JADE (Java Agent DEvelopment) platform used to implement the two proposed frameworks, MADM, so as to both illustrate and evaluate the ideas suggested in this thesis. Section 2.2 then gives an overview of data mining and Distributed Data Mining (DDM). The section also reviews three specific clustering paradigms: K-means, K-Nearest Neighbour (K-NN) and hierarchical clustering; which are all used for demonstration purposes later in this thesis. The previous work concerning MAS for data mining is reviewed in Section 2.3, where recent research in this field is summarised and some established approaches are presented. More specifically, the section considers the use of MADM for data clustering and presents a discussion of the use of negotiation in MADM to improve data mining results. This chapter is concluded, in Section 2.4, with a brief summary.

2.1 Agents and Multi-Agent Systems

Over recent years the computing trend has changed from a focus on individual standalone computer systems to a situation in which the real power of computers is realised through distributed, open and dynamic systems. In addition, the trend has been towards ever more intelligent systems which can automatically perform ever more complex tasks. However, in open, distributed and dynamic environments, the ability to cooperate and reach agreements with other systems is a necessary and overriding requirement. Agent technology provides an infrastructure that enhances our ability to solve problems in a collaborative manner by providing a “scaffolding” to support the required cooperation. At its simplest, a Multi-Agent System (MAS) is a software system that comprises a collection of software agents. These agents are capable of independent action, on behalf of their users or owners with some degree of autonomy. In other words, each agent can “figure out for itself” what it needs to do in order to satisfy its design objectives. The agents in MASs can interact with one another by exchanging messages using network protocols. The agents in a MAS typically have different goals and motivations thus there is a chance of conflict with regard to the aims of individual agents. Hence the operational requirement that to successfully interact and resolve such conflicts agents must have the ability to cooperate, coordinate and negotiate with each other.

The rest of this section is organised into four subsections. Subsection 2.1.1 gives a brief overview of agents and their capabilities with respect to data mining activities. Subsection 2.1.2 presents a general overview of MAS. Subsection 2.1.3 describes intra-agent communication and includes a discussion about Agent Communication Languages (ACLs). Finally, JADE (Java Agent DEvelopement platform), the platform used to implement the MADM frameworks that features the ideas proposed in this thesis, is presented in Subsection 2.1.4.

2.1.1 Agents

Agents are usually defined as computer entities that are suited to some environment, and that are able to carry out their tasks in an autonomous manner. The environments in which agents operate may be dynamic, unpredictable, and uncertain; thus it is desirable for agents to exhibit some form of computer intelligence in order to interact with their environment. There are a number of capabilities that intelligent agents are expected to display [89]:

- *Reactivity.* Intelligent agents should be able to anticipate tasks, according to their environment, so that they can respond in a timely fashion.
- *Proactiveness.* Intelligent agents should be able to explore alternatives to achieve their design objectives.
- *Social ability.* Intelligent agents should be capable of interacting with other agents (and possible humans) in order to satisfy their design objectives.

It is self-evident that the above capabilities can increase the degree of autonomy with which agents can perform tasks.

2.1.2 Multi-Agent Systems (MASs)

A Multi-Agent System (MAS) is a software system that employs a number of interactive agents to solve a problem in open and decentralised uncertain environments. A central feature of MAS is that there is no centralised control mechanism; agents are required to collaborate to achieve the design objective of a given MAS. A MAS has collective capabilities that an individual agent does not have. Thus, as listed in [81]:

- In a MAS computational resources and capabilities are distributed across a network of interconnected agents to solve problems that are too large for an individual agent. A centralised system may be plagued by resource limitations, performance bottlenecks, or critical failures.

- A MAS allows for the interconnection and interaction of multiple existing legacy systems; by building an agent wrapper around such systems, so that they can be incorporated into an agent society.

- In a MAS problems are modelled in terms of autonomous interacting component-agents that allows these agents to operate in self directed manner.

- In a MAS information from sources that are spatially distributed is efficiently retrieved, filtered, and globally coordinated.

- Use of a MAS provides solutions in situations where expertise is spatially and temporally distributed. With respect to the social ability of agents, expertise and resources can be shared.

- Use of MAS enhances overall system performance, especially along the dimensions of: computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility and reuse due to its distributed nature.

In the introduction to this thesis the advantages that MAS can offer data mining were listed. The decentralised control and autonomy properties of MAS can be argued to offer the most significant advantages with respect to the data mining process in that they allow individual agents to collectively process large amounts of data. Individual agents can typically process sub-sets of the data and then combine the local results to produce the desired global result, thus achieving computational efficiency advantages. This local processing also offers the advantage of privacy preservation in data mining situations where this is a requirement. The advantages offered by MAS with respect to expertise and resource sharing are clearly of significance. In the context of the work described in this thesis the agent interaction capabilities supported by MAS are also considered to be beneficial. As will be noted in Section 2.3, although there is reported work on the use of MAS for data mining, to the best knowledge of the author there is no reported work on using intra-agent negotiation to enhance data mining results as proposed later in this thesis. For many data mining activities it is well established that there is no single "best" algorithm suited to all types of data. A data mining MAS where individual agents are equipped with different data mining algorithms which produce separate results from which the best can be selected, such as that described in the following chapter, is therefore clearly also of benefit.

2.1.3 Intra Agent Communication

An obvious aspect of the interaction within a MAS is the ability for the agents to be able to communicate effectively. The development of agent communication mechanisms has been influenced by work from philosophy on speech act theory, most notably Austin's work [5] and Searle' work [78]. Speech act theory conceptualises agent communication in terms of "performatives" which are derived from the actions which an agent may wish to "perform" in order to achieve a desired task [5]. Thus performatives were defined as a class of natural language utterances which have the characteristics of actions. A number of performative verbs were identified to correspond to different types of speech act (for example *request*, *inform* and *promise*).

Agent Communication Languages (ACLs) were devised to allow agents to communicate by exchanging information and knowledge, especially in the form of complex objects, such as shared plans and goals or even shared experience and long-term strategies. Other means of exchanging information and knowledge among applications are Remote Procedure Call and Remote Method Invocation (RPC and RMI), and COBRA. Using these methods software modules (objects) are able to communicate with one another regardless of where they are located in a network. ACLs provide additional functionality over the above for two reasons: (i) ACLs handle propositions, roles and actions (which have semantic associations), whereas RPC and RMI or COBRA have no such semantic association, and (ii) an ACL message describes a desired state in a declarative rather than a procedural form [55].

There is no definitive agent communication language appropriate to all applications. There are some notable examples of popular languages that have been developed for use in multi-agent systems, with two of the most prominent proposals being: (i) the Knowledge Query and Manipulation Language (KQML) and the associated Knowledge Interchange Format (KIF) [69] and (ii) the FIPA (Foundation for Intelligent Physical Agents[1]) ACL [37].

Firstly, regarding KQML and KIF, KQML is defined as a message-based "outer" language for communication; classifying messages into particular groups, called *performatives*, to establish a common format for the exchanges. KIF, in turn, is concerned with providing a representation for "inner" content of the communication, thus the knowledge applicable in a particular domain. Table 2.1 illustrates an example KQML message using the *tell* performative. In the example agent i tells to agent j that agent i believes the price of pizza is £5. Table 2.2 indicates the semantics of the *tell* performative for sender i and receiver j. The semantics of KQML were described in terms of preconditions, postconditions and completion conditions for each performative [54]. Preconditions indicate the necessary states for a sender to send the performative and for a receiver to accept and successfully process the performative. If the preconditions

[1]the association responsible for multi-agent system protocols to support agent interoperability

do not hold, an *error* or *sorry* performative will be generated to report the unsuccessful message processing. A completion condition describes the final state that corresponds to the fulfilment of the intention that starts the conversation. Preconditions, postconditions and completion conditions describe mental states of agents (*belief*, *knowledge*, *desire* and *intention*) and the action description used for sending and processing a messages.

Table 2.1: An example of the *tell* performative in a KQML message where agent *i* "tells" agent *j* that the pizza price is £5. [53]

```
(tell
   :sender agent-i
   :receiver agent-j
   :content
      (price pizza 5)
   :language kif
   :ontology x-auction
)
```

Table 2.2: KQML semantics for *tell* [53].

tell(i,j,X) i states to j that i believes the content to be true.
-BEL(i,X)
-*Pre(i)*: BEL(i, X) $\wedge$ KNOW(i, WANT(j, KNOW(j,S)))
Pre(j): INT(j, KNOW(j,S))
where S may be any of BEL(j,X), or $\neg$(BEL(j,X)).
-*Post(i)*: KNOW(i, KNOW(j, BEL(i,X))
Post(j): KNOW(j, KNOW(i,X))
-*Completion*: KNOW(j, BEL(i,X))

The notion of conversation, is a sequence of KQML messages in agent interaction using a set of performatives and the rules that describe permissible conversations among KQML speaking agents, can be found in [9] and [54]. In [9] the notion of structured "conversation" amongst agents was proposed to underpin coordination in MASs using the extended version of KQML language. The conversation was defined as a conversation class and an actual conversation. The conversation class specifies the available rules, their control mechanism and the local database that maintains the state of the conversation. The actual conversations are processes that create instances of conversation classes and are used to maintain: (i) the current state of the conversation, (ii) the actual values of the conversation's variable and (iii) the historical information during conversation execution. Thus, interactions amongst agents can be handled by carrying out "conversations". In [54] rules for conversations were defined as conversation policies using the Definite Clause Grammars (DCGs) formalism which was extended from the use of Context Free Grammars (CFGs) [70]. The conversation policies describe both the sequences of performatives and the constraints. The conversation policies depend on values of the reserved parameters of the performatives involved in the conversations. Each KQML message was defined as a *terminal* in the DCG. A terminal is a list of the following values: *performative_name, sender, receiver, in-reply-to, reply-with, language, ontology, IO, content.* The *IO* is used for setting the type of the message; *I* is an incoming message and O is an outgoing message. The *content* is the performative itself, thus the content is also a list. The list is closed with *"["* and *"]"*. An example of a terminal is $[[ask-if, A, B, id1, id2, prolog, onto, price(X)]]$. A benefit of the use of conversation policies is that they can be used to devise a software component that monitors an agent's incoming and outgoing messages and ensures that it only engages in valid KQML conversations. Additionally, an agent's multiple interaction (conversation) with other agents can be kept track of when it goes wrong, and can be recovered from unforeseen situations. However, with respect to the work presented in this thesis a set of performatives was defined to be used as part of the protocol (a set of rules) that govern the exchange information between agents and to support negotiation within multi-agent based clustering. Transition diagrams are used to illustrate the state of agents when receiving a performative.

Although KQML has been influential among agent developers and has formed the basis of a number of implementations, numerous criticisms have been directed at it on a number of grounds including [21, 53]: (i) its interoperability, (ii) insufficient semantics support, and (iii) the omission of certain classes of message to handle particular expressions.

Table 2.3: An example of the $inform$ performative in a FIPA ACL message that agent i "informs" agent j that the pizza price is £5 [66].

```
(inform
   :sender agent-i
   :receiver agent-j
   :content
      (price pizza 5)
   :language sl
   :ontology x-auction
   :protocol contract-net
)
```

The criticisms levelled at KQML have led to the development of alternative (although in many ways similar) agent communication languages, such as FIPA ACL, which were aimed at establishing a standard communication format for use by autonomous agents. FIPA ACL is similar to KQML in terms of syntax (see examples in Table 2.1 and Table 2.3). FIPA ACL uses an outer language to enable message passing that is separate from the inner content, which may be expressed in any suitable logical language. For the outer language, FIPA ACL provides twenty two *performatives* to distinguish between the different kinds of messages that can be passed between agents. Examples of FIPA ACL performatives are: (i) *inform*, and (ii)*request*. The first is used to pass information from one agent to another; the second to ask for a particular action to be performed. The full specification of FIPA ACL performatives can be found in [37]. In order to avoid some of the criticisms that were directed against KQML, the developers of FIPA ACL provided a comprehensive formal semantics for their language. These semantics made use of the work on speech act theory, as mentioned earlier, through the definition of a formal language called Semantic Language (SL). SL enables the representation of agents' beliefs, uncertain beliefs, desires, intentions and actions available for performance. To ensure that agents using the language are conforming to it, SL contains constraints (pre-conditions), in terms of formulae mapped to each ACL message, that must be satisfied in order for compliance to hold. For example agents must be sincere, and they must themselves believe the information they pass on to others. Additionally, SL enables the rational effects of actions (post-conditions) to be modelled, which state the intended effect of sending the message. For example, that one agent wishes another to believe some information passed from the first to the second.

Although, the FIPA ACL gives a precise and standardised language for inter-agent communication, as discussed in [61], the FIPA ACL is rather broad and not universally applicable to all types of dialogue interaction that may take place within different domains. There are some performatives, such as proposal, Calling For Proposals (CFP) and so on, for general agent negotiation processes, but they are not sufficient for negotiation specific to the data mining domain such as that considered in this thesis. Thus, instead of using the generic FIPA ACL performatives, a set of "bespoke" communication performatives designed specifically to support negotiation within multi-agent based clustering will be proposed later in this thesis (Chapter 4).

Negotiation is a key form of interaction in MAS whereby agents are able to exchange offers. The agents may accept these offers or respond in some other way to the offers made to them. With respect to MAS negotiation, a negotiation framework should specify a negotiation protocol in order to govern the interaction among participants (agents) [47]. The protocol specifies which an agent is allowed to say what and actions that an agent can make. It also includes possible communication scenarios among individual agents. For example, one agent makes a proposal, the other agent may be able to accept it, reject it or criticise it, but the agent might not be allowed to ignore it by making a counterproposal. The operation of the protocol rules might be dependent on the last utterance (or the last message) that has been made or the message series exchanged between participants.

In the MAS negotiation literature, there are many sets of performatives and protocols which have been proposed, for example Persuasive Negotiation (PN) [72] and the work described in [79, 4, 60, 77]. The specification of negotiation protocols in [79] was carried out using *finite state machines* while the work proposed in [4, 60, 77] was carried out using *dialogue games*[2].

The work in [72] focused on a reasoning mechanism and a protocol for agents to engage in persuasive negotiation in situations where the agents need to negotiate with one another repeatedly. The mechanisms allow agents to influence the outcomes of future negotiations through promises of rewards, rather than exchanging offers and counter offers that only impact on the outcome of the current a negotiation. Agents try to give rewards or ask for rewards in order to get their opponent(s) to accept a particular offer. Rewards are about giving a higher current utility outcome to an opponent or a higher utility to the agent in the future. The Persuasive Negotiation (PN) protocol structures the interactions between agents as it allows them to understand messages exchanged and the commitments that agents make to each other when engaging in persuasive negotiations using rewards. The protocol was specified using dynamic logic and provided detail about how commitments arise or get retracted as a result of agents promising rewards or making offers. The protocol consists a set of illocutionary actions that agents interchange. A set of axioms was defined to express the constraints and applied to the protocol.

In [79] an argument-based negotiation was presented for a scenario in business process management. The communication language contains locutions in the form of *locution(sender, receiver, content, time)*. In addition, the locutions express offers, requests, acceptances, rejections and withdrawals. The communication language expresses three types of locutions: (i) *threats*, (ii) *rewards*, and (iii) *appeals*. The negotiation protocol was specified as a finite state machine which terminate when an agent utters an *accept* or *withdraw* locution.

The work proposed in [3] was based on MacKenzie's system DC [56], a game for two players applying the same rules. DC allowed the participants to argue about the truth of a proposition. A subset of the moves from DC which was useful in agent dialogues, was taken and augmented with some new moves. Each move was defined in terms of arguments that agents can build, and defined the moves that an agent is allowed to make at a given point in time (determining from the set of acceptable arguments). A system enabling persuasion, inquiry and information-seeking dialogues was proposed. The agent interaction protocol of [3] was based on four locutions allowed by DC: *assert, accept, question* and *challenge*. Three additional locutions: *request, promise* and *refuse* for negotiation were proposed in [4].

In [77], the work focused on one-to-one agent negotiation for achieving goals in a

[2]Dialogue games are interactions between two or more players, where each player "moves" by making utterances in a common communication language, according to certain rules [59].

system with limited resources, thus the negotiation was used for exchanging resources. Agents were equipped with a planner including a given goal. The agents were able to generate a sequence of actions and an associated set of needed resources required to carry out a plan. A single dialogue was used to obtain a resource. An agent dialogue cycle was used to trigger a new dialogue after one dialogue was terminated in order to collect all the needed resources to achieve some goal. A similar protocol to that giving in [4] was proposed but with fewer locutions. The proposed locutions for negotiation are: *request, accept, refuse, challenge, justify* and *promise.* The content of *request* and *promise* are resources, while the content for the other locutions were any of the defined six locutions. The *justify* locution allows for the utterance of some support by means of a reason for a previous locution. Because the proposed framework is based on declarative logic programming, the protocol was presented in the form of if-then rules, called *a set of dialogue constraints.* Each protocol rule was represented in the form $p(T) \bigwedge C \Rightarrow p'(T+1)$; which meant that if the agent received the performative p at a certain time t, and condition C was satisfied at that time then the agent (receiver) must use the performative p' on the next occasions.

The work in [60] presented an overview of agent interaction protocols based on formal dialogue games. A dialogue game is an interaction among two or more participants who make "moves" by uttering locutions according to some predefined rules. Dialogue game rules are specified in terms of a dialogue game protocol. The protocol consists of a number of different type of rule: *commencements, locutions, combinations, commitments* and *terminations.* Commencement and termination rules specified when a dialogue commenced and ended. Locutions indicated what utterances were permitted in the construction of dialogue moves. Commitment rules specify how agents'commitments should be managed. Finally, combination rules defined the permitted sequences of the dialogue moves.

The negotiation protocol presented in this thesis, Chapter 4, is defined in terms of a set of performatives. This set of performatives includes the name of the performatives, pre-conditions and post-conditions. The pre-conditions describe the necessary state for an agent to send a given performative. The post-conditions indicate the state of the sender after successful utterance of a performative and the state of the receiver after obtaining and processing a performative. Transition diagrams are used to represent the dialogue moves (performatives that can be uttered) by the agents and the subsequent choice of moves which are then available in the new state.

2.1.4 JADE (Java Agent DEvelopment framework)

To facilitate the rapid development of MASs using FIPA ACL, several platforms have been developed that support FIPA messaging. One of these is the JADE (Java Agent Development Environment) platform [11], a software framework to develop agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. The JADE framework is based on middleware that provides a set of generic services for the development of multi-agent systems based on the Peer-to-Peer (P2P) communication architecture. In P2P systems, the system is required to provide suitable services that allow "peers" to enter, join, or leave the network at any time, and search for or discover other peers. These services are implemented in the form of a set of virtual "yellow pages" that allow for the publishing and discovering of services provided by peers. In JADE, a peer is an agent. The services provided by agents can be registered and modified through the yellow page service. JADE allows agents to communicate by exchanging synchronous messages using the FIPA ACL.

JADE includes software packages and tools to allow Java developers to construct and deploy FIPA agent systems. JADE also includes a run-time environment that provides the services that are required to support the operation of MASs. Figure 2.1 illustrates the relationship between the main architecture elements in a JADE platform. Each instance of JADE runtime is called a *container* and a set of containers is called a *platform*. A JADE platform includes a *main container*. The main container is the "bootstrap" container and all containers in the system must register with the main container. The main container holds a number of agents providing basic services. These include: (i) the AMS (Agent Management System) agent responsible for managing and controlling the life cycle of other agents in the platform, (ii) the DF (Directory Facilitator) agent that provides the "yellow pages" service to allow agents to register their capabilities, and (iii) the RMA (Remote Management Agent) which provides the general (GUI) management console for the JADE agent platform. Functions provided by the RMA include: new agent creation, agent removing and container terminating. The RMA obtains the information about the platform by interacting with the AMS.

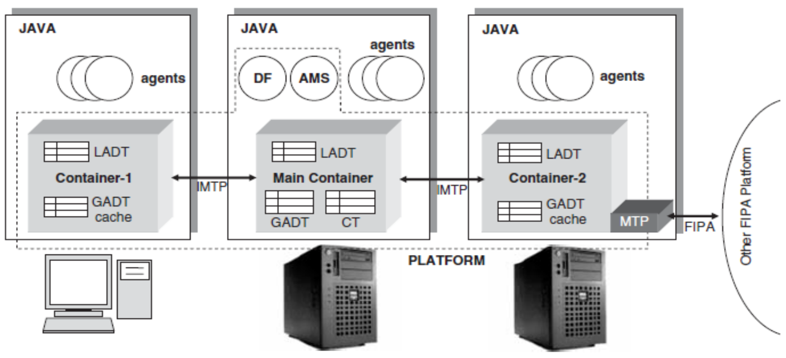


Figure 2.1: Relationship between the main elements in the JADE architecture [12].

There are three main services that an agent platform is required, according to the FIPA specification, to provide: (i) an Agent Management Service (AMS), (ii) Directory Facilitator (DF) service and (iii) a Message Transport Service (MTS). As noted above, in JADE, the agent management service and the directory facilitator service are provided by the AMS agent and the DF agent respectively. The MTS is intended to manage all messages exchanged within and between platforms. In the case of JADE, to support interoperability between platforms (including, none-JADE platforms), JADE implements all the standard Message Transport Protocols (MTPs) defined by FIPA. Each MTP includes the definition of a particular transport protocol and a standard encoding of the message envelope. HTTP (HyperText Transfer Protocol) is the default MTP. JADE always starts a HTTP-based MTP when the main container is initialised, as a result a server socket is created on the main container host. The HTTP-based MTP listens for incoming HTTP connections at the URL specified. When an incoming connection is established and a valid message received over the connection, the MTP routes the messages to its destination agent. A "single-hop routing" table, which provides direct IP visibility among the containers, is used when JADE performs message routing for both incoming and outgoing messages.

Agents apply an asynchronous message passing mechanism, ACL message, to exchange messages through the computer infrastructure. Through the content language a particular expression is communicated from a sender to a recipient. The content expression might be encoded in several ways; JADE provides three types of content language encoding as follows [12]:

1. *SL.* The SL (Semantic Language) is a human-readable string encoded content language, suitable for open-based applications where agents provided by different developers, and running on different platforms, are required to communicate.

2. *LEAP.* The LEAP (Lightweight Extensible Agent Platform) is a non-human readable byte-code content language. Therefore, LEAP is "lighter" than SL and typically adopted for agent-based applications that have a memory constraint.

3. *User Defined.* Any user-defined content language that is consistent with the languages handled by the resources, for example SQL, XML, RDF, etc.

FIPA does not define a specific content language but recommends using the SL language when communicating with the AMS and DF agents. JADE creates and manages a queue of incoming ACL messages which is private to each agent. This provides a communication architecture which is flexible and efficient. Agents can access their queues via a combination of several modes: blocking, polling, timeout and pattern matching. The full FIPA communication model has been fully incorporated into JADE. JADE also provides features to facilitate the automatic conversion of message content into a suitable exchange content format such as XML, RDF or Java objects.

With respect to the work described in this thesis, the JADE framework was selected as the agent development toolkit with which to construct an MADM system to act as an evaluation medium to support the analysis of the ideas concerning MADM proposed in this thesis. The reasons for choosing JADE were as follows [13]:

- *Standard compliance.* JADE complies with the FIPA specifications that enable end-to-end interoperability between agents in different platforms.
- *Well known and widely used.* JADE is the best-known and widely used toolkit for developing MAS applications. JADE is used by a various communities of users both with respect to research activity and commercial applications. There is evidence for the use of JADE in many research papers, such as in [2, 25, 71].
- *Mature software framework.* The JADE framework was released in 2003 and has been in use (although sometimes updated and maintained) ever since. There has been a continual improvement in JADE; while other development tools were created as part of short-time research activity, their further development and maintenance has thus typically ended after two or three years.
- *Extensibility.* The JADE framework is distributed under an Open Source License, therefore JADE code can be extended to enhance the platform with respect to some specific application or requirement. For example, the jade.mtp API allows the users to add new message transport protocols.
- *User support and documentation.* JADE code is distributed together with a programming guide and an administrator guide, a number examples and tutorials, and the *javadocs* for all the APIs.
- *Versatility.* JADE provides APIs which are independent from underlying networks. With respect to JAVA technology, JADE also provides multi-platform support; the agent platform can be distributed across machines regardless of the operating system used.
- *Information hidding.* Each JADE API is independent and provides some specific purpose-functionality, therefore developers need to understand only the required functionalities used with respect to some specific implementation.
- *Transparent communication.* The communication in the JADE framework is handled by the JADE run-time environment, thus programmers need not be concerned with the mechanism used to actually deliver messages. A number of GUI based tools are also provided in order to allow users to monitor and control run-time system activity.

A further reason why JADE was selected with respect to the work described in this thesis, is the extensibility property of the FIPA ACL that allows the definition of user-defined performaitves in order to support intra-agent communication in specific domains. User defined performatives can makes the syntax of the communication used by the agents more specific and descriptive of the particular task that the agents are communicating about (as will be demonstrated in Chapter 4).

2.2 Data Mining

Data mining refers to the process of discovering hidden but interesting patterns and knowledge from large amounts of data. The data sources are typically held in databases or data warehouses, however the Web and other information repositories can be used, or data that is streamed into the system dynamically or distributed across many sites. Data mining methods may be classified into three categories:

1. **Association Rule Mining (ARM)** or frequent pattern mining is concerned with the discovery of interesting associations and correlations between itemsets in transactional and relational databases. A popular frequent pattern mining application is market basket analysis, which is the process of analysing the buying habits of customers by finding associations between items that are frequently purchased together.

2. **Classification** is the process of finding a model (a classifier) that can be used to predict (classify) objects according to some set of predefined labels. The classification process comprises two major steps: (i) the learning step (or training phase) where the training data is used to generate a classifier using some classification algorithm, and (ii) the classification step where the classifier is applied to unseen data. Usually, to obtain some measure of the effectiveness of a generated classifier, it is first tested prior to the classification step using test data. If the accuracy is acceptable, the classifier is used to classify new data. Classifiers may be formatted in different ways, popular formats include: classification rules, decision trees, mathematical models and neural networks. Because a pre-labelled training set is required, classification is sometimes referred to as a type of *supervised learning.*

3. **Clustering** is the process of partitioning a set of data into categories ("clusters") according to characteristics found in the data. Objects (records) in a cluster are similar to one another, while dissimilar objects are located in alternative clusters. Clustering has a long and rich history in a variety of scientific fields. No prior knowledge concerning which records should belong to which cluster is required, thus clustering is sometimes referred to as a type of *unsupervised learning.*

Two approaches to data mining can be identified: (i) centralised processing and (ii) decentralised processing. Centralised processing involves moving all data to a central data repository (a data warehouse) and then analysing it with a single data mining system. Even though the centralised processing approach tends to be the most straight forward approach, it might be infeasible in cases where the size of the data is too large to store at a single location or where the data is naturally distributed and for reasons of confidentiality cannot be combined (such as in the case of cross border policing applications). A hybrid form of processing can also be identified where data can be locally analysed and the results combined at some central site to generate the final result. Surveys of Distributed Data Mining (DDM) techniques can be found in [8, 31, 46, 67]. The decentralised processing paradigm is clearly of interest with respect to MADM. However a recognised problem with this approach is the "message passing" overhead. To address this weakness, many researchers attempt to achieve the integration of knowledge discovered from different sites with a minimum amount of network communication and a maximum amount of local computation. Agent based architectures are well suited to supporting DDM because of the "added value" that they provide in terms of mechanisms to support the interchanging of information and coordinating activities. There is also evidence to suggest that the use of agent architectures can speed up the data mining task (see for example [75]).

The remainder of this section is organised into two subsections. Subsection 2.2.1 gives some background concerning a number of clustering techniques used to investigate the use of MADM/MABC as reported in this thesis, and to evaluate the proposed solutions to the research issues identified in Chapter 1. Subsection 2.2.2 then consider a number of cluster evaluation metrics that may be used for assessing the quality of cluster configurations.

2.2.1 Clustering

Clustering is one of the most prevalent of data mining activities; as already noted the aim is to group data into a set of categories (clusters). The grouping is accomplished by finding similarities between data according to characteristics found in the actual data without any prior knowledge concerning the nature of the clusters. Thus, data objects in the same cluster should be similar to each other, while data objects in different clusters should be dissimilar from one another. Four basic steps can be identified within the overall clustering process [45]:

- **Feature selection**. Feature selection is the sub-process of finding a subset of good features (variable subset) that are likely to form high quality clusters. The most appropriate subset will contain the least number of dimensions and make the highest contribution to accuracy; the remaining and unimportant attributes are discarded. In practice the features with minimum information redundancy, not affected by noise and easy to extract and interpret, are typically selected. Feature selection can reduce the amount of computation and simplify the clustering process. To obtain the highest degree of confidence with respect to some clustering result, choosing appropriate and meaningful features can greatly reduce the burden of subsequent design and result evaluation.

- **Clustering algorithm application**. This step includes the specification of the necessary parameters and cluster criterion functions and, of course, the selection of an appropriate clustering algorithm. Recall that many different clustering algorithms have been developed and there is no single best clustering algorithm suited to all possible applications.

- **Cluster validation**. In this third step, various metrics are used to evaluate the quality of the generated cluster configuration. A review of established metrics used to evaluate clustering results is given in Subsection 2.2.2 below.

- **Result interpretation**. In many cases, end users (typically experts in the relevant field) must integrate the clustering result with experimental evidence so as to analyse the outcomes.

Clustering algorithms may be broadly categorised into two types: (i) *partitional* and (ii) *hierarchical*. Partitional clustering algorithms generate clusters by allocating them to a "flat" set of clusters, while hierarchical clustering algorithms iteratively subdivide/combine a given data set into a set of clusters.

Two well-known partitional clustering algorithms are K-means [57] and K-Nearest Neighbour (K-NN) [27]. The K-means algorithm partitions a given data set into a user-specified set of K clusters. The algorithm operates as follows: (i) select the K first records to define K distinct clusters (a cluster is defined according to its *centroid*, which in turn is defined by the mean values of the records contained within it, if there is only one record then that record represents the centroid); (ii) assign each remaining record to its "closest" cluster centroids (some Euclidean distance measure is usually used); (iii) when all records have been assigned, recalculate the cluster centroids and reassign the records; (iv) continue in this manner until the cluster centroids become "fixed". The performance of K-means is strongly influenced by the value of K used and the nature of the first K records used to define the initial centroids.

The K-NN algorithm does not require specification of a predefined number of clusters; instead a user supplied threshold, τ, is used to determine nearest neighbours. The algorithm starts with one cluster whose centroid is represented by the first record. Further records are then added. If the "distance" between a new record and an existing cluster centroid is less than τ the new record is placed into the appropriate existing cluster, otherwise a new cluster is created. The chosen value of τ thus greatly influences the number of clusters.

Hierarchical clustering groups data into a sequence of nested partitions. Hierarchical clustering can be conducted in a top-down (*divisive*) or a bottom-up (*agglomerative*) manner [90]. Top-down hierarchical clustering commences with a single cluster representing the entire data set which is then divided (split) into two sub-clusters. The cluster with the largest diameter, defined as the largest distance between any pair of records, is selected for further division. The algorithm continues until an appropriate cluster configuration is reached (according to some given metrics) or each cluster contains only one record. The bottom-up hierarchical clustering works in the reverse order. The process starts with a number of clusters equivalent to the number of records. The two clusters with the largest "similarity" are then combined to form a merged cluster. The process again continues until an appropriate cluster configuration is arrived at or all nodes are merged into a single cluster. Hierarchical clustering may involve excessive computational time and storage, because the algorithms used must make many iterations for each level in the hierarchy.

Three of the above clustering algorithms; K-means, K-NN and divisive hierarchical clustering, were selected for the proposed of evaluating the work described later in this thesis. Agglomerative hierarchical clustering was not selected because of its similarity to divisive hierarchical clustering.

2.2.2 Clustering Metrics

Given a data set, any clustering algorithm can always generate a set of clusters, although the quality may not be as desired. Moreover, different clustering algorithms will produce different clustering results depending on: (i) the characteristics of the input data sets, (ii) the presentation and ordering of records in the input data and (iii) the input parameters used to define the nature of the desired clusters (this in turn will be dependent on the nature of the adopted clustering algorithm). Thus, given the above, effective evaluation standards or criteria are important to provide users with a degree of confidence in the clustering results derived from using a particular clustering algorithms.

Cluster configuration metrics are typically used to evaluate cluster configurations. Clearly any such metric should be objective and have no preferences for any specific algorithm. The available metrics can be divided into three categories: (i) external criteria (supervised metrics), (ii) internal criteria (unsupervised metrics) and (iii) relative criteria [42, 84]. *External criteria* metrics require pre-labelled data sets, with "known" cluster configurations, and measure how well a given clustering technique performs with respect to these known clusters (often referred to as "a ground-truth partition"). On the other hand, *Internal criteria* metrics are used to evaluate the "goodness" of a cluster configuration without any prior knowledge of the nature of the clusters. Internal criteria metrics use only the quantities and features inherent in the data sets, and tend to be founded on statistical methods. *Relative criteria* metrics are used to compare clustering results, generated by different clustering algorithms, and so as to decide which of them is better. Surveys of well established metrics, with respect to the foregoing characterisation, can be found in [42, 43].

Internal criteria metrics are typically founded on the the concepts of cluster *cohesion* and *separation* [83]. Cohesion is used to measure the compactness ("tightness") of clusters, whilst separation is a measure of the distinctiveness of a cluster with respect to other clusters. There are a number of different methods whereby cohesion and separation can be calculated, these include: (i) *density-based*, (ii) *graph-based* and (iii) *prototype-based* approaches. The density based approach evaluates the density distribution within and between clusters. Using the graph-based approach the cohesion of clusters is defined as the sum of the weights of the links among points in the cluster. Separation is then measured by computing the sum of the weights of the links from all points in one cluster to all points in another cluster. Using the prototype-based approach the cohesion of a cluster is measured in terms of the sum of the weights of the links from the prototype to points in the cluster. The separation between two clusters may then be measured in terms of the sum of the weights of the links among the prototypes of two clusters. This is illustrated in Figure 2.2 where an asterisk ($*$) represents a centroid of a cluster. In the context of the work described later in this thesis the prototype (centroid or medoid) based approach has been adopted with respect to the internal criteria metrics because it has a lower computational overhead associated with it compared to the other two approaches.

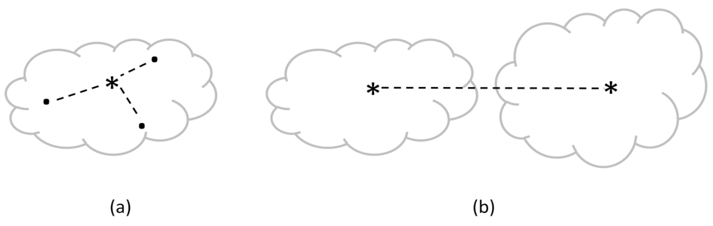


Figure 2.2: (a) Prototype-based Cohesion (b) Prototype-based Separation

With respect to the work described in this thesis both external and internal criteria were used to evaluate the "goodness" of a cluster configuration produced as a result of the application the proposed MABC mechanisms. More specifically the F-measure, which is an external metric, was only used with respect to the initial MADM framework when applied to MABC. While two internal criteria metrics: (i) Within Group Average Distance (WGAD), and (ii) Between Group Average Distance (BGAD), were used both with respect to the initial MADM framework and the extended MADM framework (in the context of MABC). In the context of the operation of the proposed MADM framework, to allow the system to identify the most appropriate clustering parameter (as described in Chapter 6), three clustering metrics were considered: (i) the Silhouette

Coefficient (Sil. Coef.) [76], (ii) the Davies-Bouldin (DB) index [28], and (iii) the combination of WGAD and BGAD (WGAD-BGAD). Details concerning each of these metrics are presented in the remainder of this subsection.

The F-measure (F1 measure) is an external criteria metric and is popularly used in the domain of Information Retrieval [85]. The metric measures how well cluster labels match externally supplied class labels. The F-measure combines the probability that a member of a cluster belongs to a specific partition (*precision*) and the extent to which a cluster contains all objects of a specific partition (*recall*) [83]. Let $C = \{C1, C2, ..., Ck\}$ be a clustering result, $P = \{P1, P2, ..., Pk\}$ be the ground-truth partition of the input data set. The precision of cluster i with respect to partition j is $precision(i,j) = |Ci \cap Pj|/|Ci|$. The recall for cluster i with respect to partition j is defined as $recall(i,j) = |Ci \cap Pj|/|Pj|$. The F-measure for cluster i with respect to partition j is then defined as:

$$F_{ij} = \frac{2 \times precision(i,j) \times recall(i,j)}{precision(i,j) + recall(i,j)} \quad . \tag{2.1}$$

and the overall F-measure of the cluster is calculated using:

$$F = \sum_{i=1}^{m} \frac{|P_i|}{|P|} \times \max(F_{ij}) \quad . \tag{2.2}$$

where m is the number of partitions.

Out of the two internal criteria metrics used for evaluation purposes (and used with respect to this thesis), WGAD is used to determine cohesion [73] while BGAD to determine separation [83]. WGAD is the sum of the average distance of a cluster centroid, c_i, to each data point (x) in the cluster. The lower the WGAD the greater the compactness (cohesiveness) of the cluster. The WGAD of a given cluster configuration is defined as:

$$WGAD = \sum_{i=1}^{i=K} \frac{\sum_{j=1}^{j=X_i} dist(x_j, c_i)}{|X_i|} \quad . \tag{2.3}$$

where K is the number of clusters and $|X_i|$ is the number of data points in cluster i.

BGAD, in turn, is the average distance of each cluster centroid, c_i, to the overall centroid, c. The higher the BGAD of a cluster configuration the greater the separation of the clusters from one another. The BGAD of a given cluster configuration is defined as:

$$BGAD = \frac{\sum_{i=1}^{i=K} dist(c_i, c)}{|K|} \quad . \tag{2.4}$$

Thus, to identify a "best" cluster configuration, WGAD should be minimised and BGAD should be maximised to achieve a best degree of cohesion and separation.

The combination of WGAD and BGAD (WGAD-BGAD) is the difference between a pair of WGAD and BGAD values to express the overall validity of a given cluster configuration. This metric was derived by the author and used, later in this thesis, to measure the “goodness” of an obtained cluster configuration.

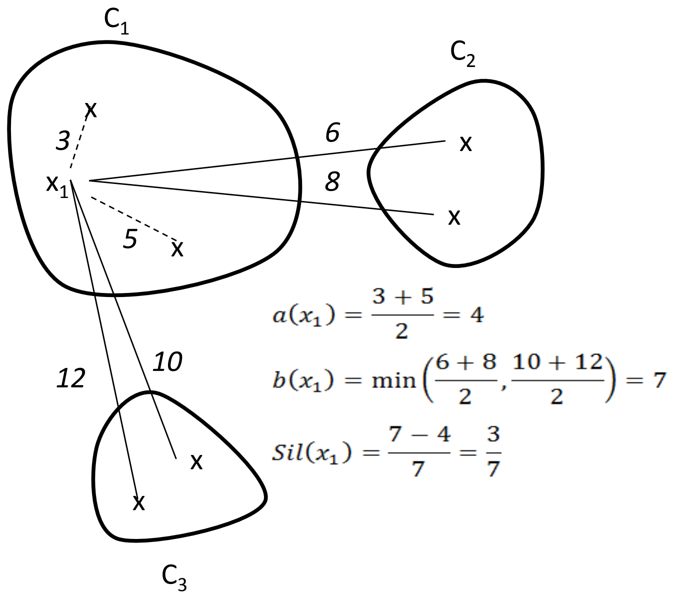

C1
C2
C3
x1
3
5
6
8
10
12
x
$a(x_1) = \frac{3+5}{2} = 4$
$b(x_1) = \min\left(\frac{6+8}{2}, \frac{10+12}{2}\right) = 7$
$Sil(x_1) = \frac{7-4}{7} = \frac{3}{7}$

Figure 2.3: Derivation of the Silhouette Coefficient ($Sil.\ Coef.$)

The Silhouette Coefficient ($Sil.\ Coef.$) of a cluster configuration is a measure of both the cohesiveness and separation of a configuration. It is determined by first calculating the *silhouette* (Sil) of each individual point x_j within the configuration as follows:

$$Sil(x_j) = \frac{b(x_j) - a(x_j)}{max(a(x_j), b(x_j))} \quad . \tag{2.5}$$

where $a(x_j)$ is the average intra-cluster distance of the point x_j to all other points within its cluster, and $b(x_j)$ is the minimum of the average inter-cluster distances of x_j to all points in each other cluster (see Figure 2.3 for further clarification). The $Sil.\ Coef.$ value is then calculated as follows:

$$Sil.\ Coef. = \frac{\sum_{i=1}^{i=K} \frac{\sum_{j=1}^{j=|C_i|} sil(x_j)}{|C_i|}}{K} \quad . \tag{2.6}$$

The resulting value is thus a real number between -1.0 and 1.0. If the silhouette coefficient is close to -1, it means the cluster configuration is undesirable because the average distance to points in the clusters, is greater than the minimum average distance to points in the other cluster(s). The overall silhouette coefficient can thus be used to

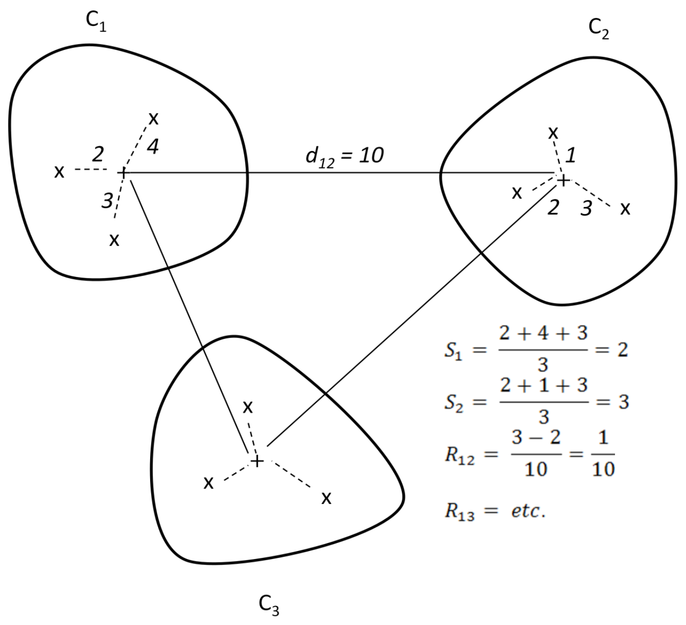

C1
C2
C3
$d_{12} = 10$
2
4
3
1
2
3
$S_1 = \frac{2+4+3}{3} = 2$
$S_2 = \frac{2+1+3}{3} = 3$
$R_{12} = \frac{3-2}{10} = \frac{1}{10}$
$R_{13} = etc.$

Figure 2.4: Derivation of the Davies-Bouldin (DB) index

measure the goodness of a cluster configuration. The larger the coefficient the better the cluster configuration.

The Davies-Bouldin (DB) index is the sum of the maximum ratios of the intra-cluster distances to the inter-cluster distances for each cluster i:

$$DB = \frac{1}{K} \sum_{i=1}^{i=K} R_i \ . \tag{2.7}$$

where R_i is the maximum of the ratios between cluster i and each other cluster j (where $1 \leq j \leq K$ and $j \neq i$). The lower the DB index value the better the associated cluster configuration. The individual ratio of the intra-cluster distances to the inter-cluster distances for cluster i with respect to cluster j is given by:

$$R_{ij} = \frac{S_i - S_j}{d_{ij}} \ . \tag{2.8}$$

where d_{ij} is a distance between the centroid of cluster i and the centroid of cluster j, and S_i (S_j) is the average distance between the points within cluster i (j):

$$S_i = \frac{1}{|C_i|} \sum_{n=1}^{n=|C_i|} d(x_n, c_i) \ . \tag{2.9}$$

where c_i is the centroid of cluster i, and x is a point (object) within cluster i. The derivation of DB index is illustrated in Figure 2.4.
With respect to clustering applications, cluster metrics are as important as the clustering algorithms.

2.3 Multi-Agent Data Mining (MADM)

This section presents a review of some work related to that described in this thesis. The section commences by first considering data mining driven agents and then continues with agent driven data mining or Multi-Agent Data Mining (MADM). The discussion includes consideration of MADM directed at Association Rule Mining (ARM), classification and clustering; and highlights the distinctions between this previous work and that proposed in this thesis.

As already noted, MASs provide a powerful technology for distributed, autonomous problem solving. Given the popularity of MAS the idea of adopting MAS technology for data mining is an attractive one. Reviews of agent mining interaction and integration can be found in [17, 18, 19, 20]. As already noted there are two main paradigms with respect to integration of MAS and data mining:

- *Data mining-driven agents.* The use within MASs of knowledge models, derived through data mining, to enhance agent intelligence including learning/reasoning, coordination, adaptation, planning and behaviour analysis.
- *Agent-driven data mining.* The utilisation of collections of agents, within some MAS, to perform collaborative data mining tasks (referred to in this thesis using the acronym MADM).

From the literature a number of MASs that use data mining driven agents have been reported ([62, 58, 65]). In [62] the Agent Academy development framework was described. Agent Academy is a MAS development framework that uses data mining techniques to mine application specific data from which rules can be generated, which in turn can be embedded in a rule based reasoning mechanism to drive agent behaviour. In [58] data clustering was employed to support automated negotiation in an e-commerce environment. The history of previous negotiations, buyers' behaviours and strategies were grouped in order to increase the experience of sellers to handle the negotiation process between the buyer and the seller. There were two main phases within the learning process: (i) the learning model generation and (ii) the use of the learning model for the seller agent to generate an interaction strategy. In the first phase, data clustering was used to classify buyers into clusters by considering negotiation history information. Each cluster was then used to define a strategy by using an association rule based technique. The knowledge generated in the first phase, was then used to prepare for any future negotiation. In the second phase, the knowledge extracted in the first phase, was mapped and matched to find an appropriate negotiation strategy given a new buyer. Using this approach the seller agent was able to revise and adapt its negotiation strategies in the light of newly gained experience. In [65] data mining was integrated with an intelligent tutoring agent designed to provide relevant feedback to students using CanadarmTutor, a system used to teach astronauts how to operate a robot manipulator aboard the International Space Station.

MADM is concerned with using MAS technology to enhance data mining processes. Typically a collection of agents is used to perform a data mining task in an autonomous manner. As noted earlier in this chapter, MADM allows for distributed data to be mined effectively without the need to first move the data into a data warehouse. This offers advantages where it is not easy or not possible for data to be first collated; for example because of security and privacy reserving issues or because of technical issues. MADM also supports the creation of frameworks that can be allowed to grow in an anarchic manner such that additional agents can be easily incorporated as long as they comply with whatever protocols have been specified [2]. However, the true power of agents is their ability to behave in an autonomous manner and interact (negotiate) in order to provide a solution to a problem. The proposed MADM approach described in this thesis seeks to harness this power in the context of clustering problems. In this thesis the acronym MABC (Multi-Agent Based Clustering) is used to refer to the application of MADM to clustering.

From the literature there have been a number of MADM systems reported. Many of these can be considered to be simply distributed systems founded on agent architecture (they do not make full use of the capabilities offered by MASs). To use MAS for Distributed Data Mining (DDM), where data sources are located at different sites, it is necessary to develop mechanisms to achieve acceptable time and space performance. DDM offers a way of dealing with data sets regardless of their physical locations. The "standard" approach is to process the data at each site to produce local results which are then combined to generate a global result. MAS technology "dovetails" nicely with DDM because MAS is well suited to distributed problem solving. Surveys of agent-based DDM can be found in [26, 39, 52, 63, 74]. In many of these reported cases, unlike the approach reported in this thesis, the reported MADM systems do not use MASs to their full potential, control tends to be centralised and very little "negotiation" (agent interaction) takes place between agents.

An early MADM is described in [29, 30] in which a collection of agents was used to enhanced the Knowledge Discovery in Databases (KDD) process. The system epitomised the high level concept of MADM whereby communities of data mining agents perform data mining tasks. In this early system, Inductive Logic Programming (ILP) was adopted in order to find first-order relations in data. The MADM in this case comprised a coordinator agent designed to support coordination between agents and provide an end user interface to allow users to interact with the system and assigning data sources to mining agents. At the final stage, individual knowledge was generated and integrated into a "globally coherent theory".

Other MADM systems with similar purpose are described in [6, 71]. In [6] a MADM directed at geographically dispersed data, that used pattern mining technique to populate a Knowledge Based (KB), is described. This KB was then used to support decision making by end users. In [71] the AgentDiscovery system is described which comprises a collection of agents, each associated with a phase in the KDD process including business understanding, data understanding, data preparation, modelling, evaluation and deployment. The added feature of the system was the ability to provide a suggestion for the best knowledge model or the best algorithm(s) to solve a problem depending on user preference (such as the quickest, the most accurate, the most scalable etc.). However, the above proposed approach view of MADM was that individual agents should perform specific data mining tasks, not that a collection of agents should collaboratively undertake such a task.

Fatta et al. [32] proposed an agent-based distributed computing system intended for large-scale, non-dedicated and heterogeneous computing environments like Grids. The proposed system used a load balancing policy for distributing subtasks among the agents to reduce the communication cost and the computational overhead. More specifically the MADM deployed frequent subgraph mining techniques, in a distributed manner, to identified molecular structures. Two types of agents were involved in the process: mining agents and coordinator agents. The coordinator agent, as in the case of similar systems, is responsible for: (i) coordinating the processing of tasks, (ii) collecting the results and (iii) providing a user interface.

Although there has been a significant amount of reported work on distributed/parallel clustering ([8, 31, 46, 67]) there are few reported examples of MABC. What systems there are tend to be very much founded on the established distributed or parallel approaches to clustering. Examples of early (c1990) reported MABC systems include [49] and [8]. In [49] the PADMA (PArallel Data Mining Agents) system is described. PADMA was broadly directed at the integration of knowledge discovery from different sites with a minimum amount of network communication and a maximum amount of local computation. PADMA adopted a "standard" approach to achieving MABC founded on a hierarchical clustering model in the context of document categorisation. The clustering was aided by what the authors term a "relevance feedback-based supervised learning technique". However, the focus of their work was to present the benefits of agent based parallel data mining (rather than MABC). In [8] a MABC system, called Papyrus, was described that used mobile agents. The system was intended for DDM, handling clusters and meta-clusters distributed over heterogeneous sites. In this system mobile data mining agents were used to transport data and intermediate results between clusters for local processing or from local sites to a central site which produced the final result with the aim of reducing the network communication load.

In [52] two schemes for MABC, founded on the concept of *kernel density estimation based clustering* (a clustering mechanism directed at the identification of dense regions in a given feature space) were described. The first MABC scheme adopted a “standard” agent based approach and the second a mobile agent approach. The claimed advantage was that by estimating the global densities from local data the communication between agents can be minimised.

In [80] a MABC was presented for clustering environment data by comparing the geometrical shape of each environment feature with neighbouring features. Agglomerative hierarchical clustering was employed to demonstrate the process. The system incorporated two types of agent: Map Agents and Cluster Agents. A Map Agent was responsible for controlling the entire process of clustering and managing the life cycle of all agents and for dictating a plan to be used by the Cluster Agents. A Cluster Agent represented a cluster of several single geometries. A two phase process was described. In the first phase the Map and Cluster Agents were initialised and statistical constraints concerning the input data set were calculated. In the second phase merging or dividing of clusters, according to the constraints identified in the first phase, was undertaken. A “happiness” threshold was used to determine the desired cluster configuration. Thus the Map Agent controlled the clustering process by dividing or merging, while the Cluster Agents were responsible for performing the merging and dividing of clusters, and calculating statistics concerning the data belonging to each cluster.

The above systems all successfully used agents to facilitate distributed data clustering (mining). However there is little reported work on agent based clustering systems that support intra-agent negotiation in order to undertake (or refine) the desired clustering, as proposed in the work described in this thesis. There has been some work on the implementation of MABC systems that use an ensemble approach to clustering (the combination of the results from several clustering algorithms to generate a “best” cluster configuration). Two examples are presented in [1] and [7]. In [1] each agent is assigned a small subset of the data and votes on which final cluster its records should belong to. The final clustering is then generated by a global utility, but computed in a distributed way. In [7] an MABC is proposed where a number of agents (each operating on a subset of the input data) generated a cluster configuration (where each cluster is defined by a “prototype”). The results are then combined, by another agent, to produce the desired best cluster configuration. These “ensemble” based MABC bear some similarity with the system described in this thesis in that they seek to improve on a basic cluster configuration of the form that would be generated through application of a “stand alone” (centralised) clustering algorithm. However, the mechanisms described in this thesis differ in that they seek to improve on a basic configuration by allowing agents to negotiate with one another so as to improve on this basic configuration; in this thesis the decision making concerning improving the basic cluster configurations is distributed amongst the agents and is not assigned to a single “control” agent. The approach proposed in this thesis is therefore quite distinct from these other approaches.

There has been previous work that has adopted a negotiation mechanism with respect to MADM systems. One example is described in [33] with respect to the classification of bioinformatics data. Bioinformatics databases tend to be very large, therefore the aim of this work was to divide the data into several data subsets and allow a collection of agents to negotiate to produce a global classifier. A negotiation process was used to integrate the local classifiers. There were two main types of agent involved in the negotiation process: (i) Learning Agents and (ii) Mediator Agents. Learning Agents were responsible for generating a local classifier and evaluating it, whereas the Mediator Agent was responsible for controlling the communication among Learning Agents and finalising a negotiation. The Learning Agents could use different classification algorithms to generate their local classifiers. The final classifier was generated on conclusion of the negotiation process. The negotiation process was described as follows:

1. The Mediator Agent asks the Learning Agents to send their overall accuracies.
2. The Learning Agent who holds the poorest overall accuracy makes a proposal containing the first classification rule in its classifier.
3. The Mediator Agent distributes the identifies classification rule to other Learning Agents.
4. Each Learning Agent evaluates the proposal which contains the identified classification rule and the rule accuracy, and searches for an equivalent rule with better accuracy. An equivalent rule is one that describes the same concept and shares at least one attribute with the original rule. If there is no equivalent rule or no equivalent rule with better accuracy, then the agent accepts the proposal. Otherwise, other Learning Agents who have rules with better accuracy than the rule in the proposal, send their rules to the Mediator. The Mediator identifies the rule that gives the best accuracy and sends the selected rule to all agents in order to inform them of the newly accepted rule. The new rule will be marked as an *evaluated rule*. The accepted rule is added in the global classifier.
5. The negotiation process is repeated until all rules have been evaluated.

Another example of the use of negotiation in MADM for classification is described in [68]. In this case, the operational process covered two KDD phases: data preparation and data mining. Classification was applied in the data mining phase. The proposed framework comprised a collection of KD (Knowledge Discovery) agents. KD agents used the same input data sets but different classification methods, therefore different classifiers were produced. A negotiation process was then used with the aim of identifying the agent who produced the best classification with respect to a given record. When a new record was presented for classification each agent, using its classifier, produced a set of "likelihood" values for each potential class. The authors referred to these values as *confidence values*, but these should not be confused with the confidence values commonly used in association rule mining. If all agents agreed on a classification, no negotiation was undertaken and this classification was selected as the class for the new record. If there was no consensus the agents negotiated with the aim of reducing each others' confidence values by applying a penalty. The negotiation was conducted according to the differences between the *sensitivity values* associated with each attribute in the new record and each class, and the *training mean* sensitivity values associated with the classifier belonging to an individual agent. Attributes with a high difference indicated that a particular classification might be inaccurate, and these were used as reasons for reducing the confidence values between agents that "disagree". The penalty applied was calculated according to the difference value. This process continued in a round by round manner until all agents agreed or no further negotiation could take place, in which case the classification with the highest confidence value was selected.

There is a clear distinction between the earlier work on MADM that featured negotiation, as described above [33, 68] and the work described in this thesis. The previous work was directed at supervised learning and it is suggested here that negotiation for unsupervised learning is more challenging (hence the lack of reports in the literature). Typically, unsupervised learning is more difficult and challenging problem than supervised learning because in supervised learning *labelled data* (training patterns with known category labels) is involved, while in unsupervised learning *unlabelled data* is involved. In clustering the notion of a "cluster" is not precisely defined, as a consequence many clustering methods have been developed and there is no best clustering method. Hence, the use of negotiation in unsupervised learning to improve the quality of cluster configurations is challenging. To summarise; the main aim of the negotiation described in [33] was to integrate multi-classifiers, while the aim of the negotiation described in [68] was to identify a best classifier with respect to a given data set. The negotiation proposed in this thesis aims to improve cluster configurations.

It should also be noted in this section on MADM that the author's first experiments directed at MABC [22, 23], described in Chapter 3, built on some of the ideas featured in EMADS (Extendible Multi-Agent Data Mining System) [2], a reported example of a generic MADM system directed at classification and frequent pattern mining. EMADS was designed to support the "organic" growth of a MADM. However, a disadvantage of EMADS was that fixed protocols were used to achieve agent communication. Furthermore EMADS did not feature the concept of intra-agent negotiation.

The author's first MABC experiments used a collection of agents, running different clustering algorithms to determine a best cluster configuration. Later work, described in Chapter 4 added a negotiation capability so as to realise the full potential offered by the use of MAS technology. The work described in this thesis is therefore distinct from the previous work on MADM and MABC described above.

2.4 Summary

The literature review presented in this chapter has given the essential background knowledge underpinning the research work proposed in this thesis. A brief overview of agents and MASs in relation to data mining activities was first presented. Intra-agent communication was then discussed in the context of ACLs; some reasons for selecting the JADE framework as the development toolkit for the work described in this thesis were also discussed. A brief review of data mining, including the clustering algorithms used for evaluation purposed later in this thesis, was then presented. The chapter was concluded with a review of the related research on MADM and MABC that has had an influence on the work described in this thesis; here it was noted that there have been only a few MADM systems that used negotiation to improve data mining results and that these were directed at supervised learning. The literature review presented in this chapter has shown that there is little reported work on agent based systems that support intra-agent negotiation in order to enhance data mining results, especially in the context of MABC, which is what the work reported in the forthcoming chapters addresses.

Chapter 3

A Multi-Agent Data Mining Framework

As described in Chapter 1, the aim of the research presented in this thesis is to investigate the use of MAS technology with respect to data mining and specifically data clustering. This chapter presents a multi-agent based approach to clustering using a generic MADM framework that harnesses the processing power of a collection of "clustering" agents, to produce a "best" set of clusters given a particular clustering problem. The motivation for the application is that there is no clear, general purpose, best clustering algorithm suited to all data. It is suggested that a Multi-Agent System (MAS) based approach provides a good solution to the generic problem of finding a best set of clusters. The approach allows for a collection of clustering agents to collaborate to produce a "best" cluster configuration. With reference to Chapter 1, Section 1.2, the establishment of a MADM framework entails the resolution of a number of research issues. The first is the nature of the operation of the desired MAS based clustering, i.e. the structure of the MAS environment, and the coordination and communication mechanisms to be used. The second is how to define what is meant by a "best" cluster configuration, and how this definition may be used within a MAS framework.

To address the first issue, a Multi-Agent Data Mining (MADM) environment, was investigated. A full description of the proposed MADM architecture is included in this chapter. The second issue is more difficult to address. The accuracy of a set of clusters (a cluster configuration) can of course be evaluated by comparing the derived results with a set of known results, as in the case of supervised learning. However, this requires provision of pre-labelled data so that a *training* set can be presented to the clustering system, from which labelled clusters can be generated, which can then be evaluated using a *test set.* With respect to many applications the pre-labelling of data entails an unacceptable overhead, unsupervised learning is therefore frequently deemed to be more desirable. However, in this chapter, for evaluation purposes both supervised and unsupervised metrics were adopted to determine the appropriateness of a generated cluster configuration. In the case of supervised metrics, the F-measure was used; while two unsupervised measures, Within Group Average Distance (WGAD) and Between Group Average Distance (BGAD), were considered and compared. The main contributions of this chapter are thus as follows:

- The design of a proposed MABC framework, a generic MADM environment.
- A generic approach to multi-agent based clustering to identify a best set of clusters using a collection of "clustering agents".
- The use of clustering metrics to evaluate the quality of cluster configurations and hence identify the best one.
- A comparison of two measures, WGAD and BGAD, to identify the most appropriate set of clusters for a given clustering problem.

Note that there are two implementations of the MADM framework used to illustrate the investigation of the use of MAS technology with respect to MABC in this thesis. The first implementation of the proposed MADM framework is presented in this chapter. A second implementation of the proposed MADM framework used to support a proposed intra-agent negotiation mechanism for cluster configuration refinement, is described in Chapter 4.

The rest of this chapter is organised as follows. The nature of the proposed generic framework for MADM is presented in Section 3.1. The proposed OntoDM ontology is then presented in Subsection 3.1.1. Measures to identify a best cluster configuration for a given data set, are considered in Section 3.3. Section 3.4 then presents a discussion of the proposed approach, followed by some conclusions to end this chapter.

3.1 The Proposed Multi-Agent Data Mining Framework

To achieve the desired MADM functionality, the start point for the design of an appropriate MADM framework was the identification of the nature of the required agents. To this end, five basic types of agent, each of which may have several sub-types, were identified as follows:

1. **User Agents**: Agents that provide the interface between the end users, who wish to conduct some data mining activity, and the MADM environment. Typically there will be one User Agent per end user.
2. **Task Agents**: Agents that facilitate (manage) the performance of data mining tasks. Three distinct sub-types of Task Agent are identified corresponding to the three high level data mining tasks as described in Chapter 2, Section 2.2: Association Rule Mining (ARM), Classification, and Clustering. Task Agents are spawned by User Agents in response to an end user data mining request;

they exist until the data mining task they are directed to coordinate, whatever this might be, is completed. Task Agents are responsible for coordinating the response to an end user data mining request by interacting with existing agents within the MADM environment. Typically Task Agents do not generate a solution themselves. They identify, using a "yellow pages" service, suitable agents who may be able to complete a data mining task and consequently provide a solution.

3. **Data Mining Agents**: Agents that are responsible for performing data mining activities and generating results to be passed back to a Task Agent. Data Mining Agents are typically equipped with data mining algorithms. With respect to this chapter the Data Mining Agents are equipped with various kinds clustering algorithms.

4. **Data Agents**: Data Agents are the "owners" of data resources. More specifically, Data Agents act as communication conduits to and from data resources.

5. **Validation Agents**: Agents that are responsible for evaluating and assessing the "validity" of data mining results. Several different sub-types of Validation Agent are identified, each associated with different generic data mining tasks (in a similar manner to the sub-types identified for Task Agents): Association Rule Mining (ARM), Classification, and Clustering. For example, in the case of a clustering scenario Validation Agents will be able to measure the "goodness" of a proposed cluster configuration (using, for example, the F-measure, WGAD and BGAD metrics as described in Chapter 2, Subsection 2.2.2).

In addition to the above listed agents, some "house keeping" agents will also be required in order to provide various facilities to maintain the operation of the proposed framework. To evaluate the above suggested agent configuration, a demonstration framework was implemented using the Java Agent Development Environment (JADE). It should be noted that JADE comes with a number of house keeping agents; namely: the AMS (Agent Management System) Agent and the DF (Directory Facilitator) Agent. The first is used to control and manage the lifecycle of other agents in the *platform*, the second provides a "lookup service" to allow agents to register their services. Thus the lookup service allows a MADM Task Agent to identify the appropriate Data Mining, Data and Validation Agents required to complete a given data mining task.

A typical agent configuration is given in Figure 3.1. The figure includes a User Agent, a Task Agent, several Data Mining Agents, a Data Agent, a Validation Agent and some house keeping agents. The directed arcs indicate communication between agents; communication can be bidirectional or unidirectional. Note that the MADM agent configuration given in Figure 3.1 actually describes a clustering scenario.

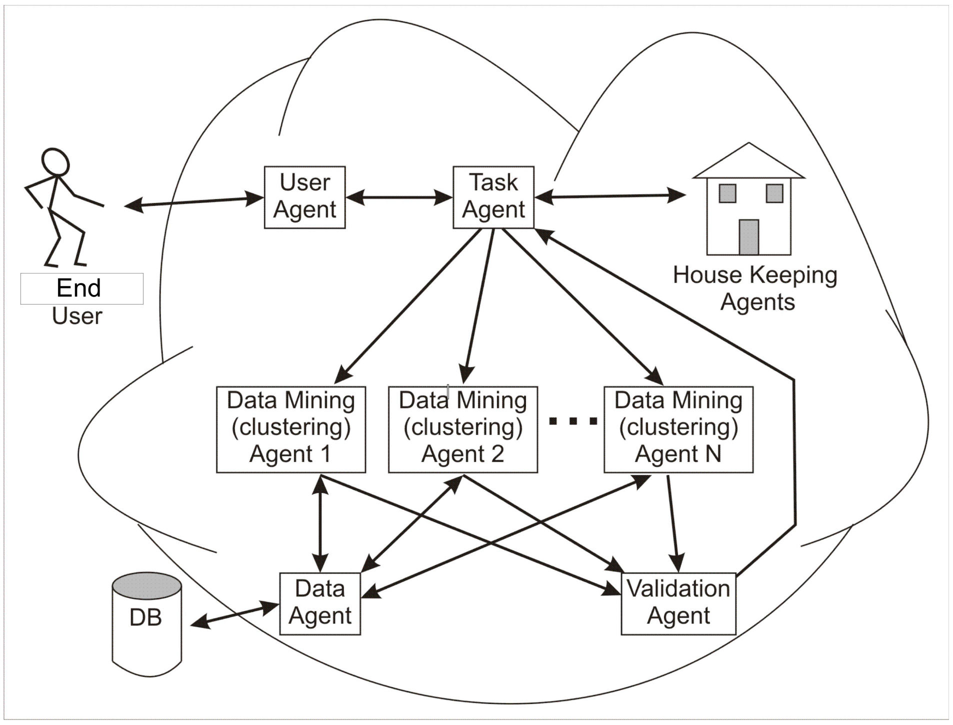


Figure 3.1: Architecture of the proposed MADM system

3.1.1 OntoDM

An ontology is a particular conceptualisation of a set of concepts and relationships between classes, properties and instances in a domain of knowledge. An ontology consists of terms, their definitions and axioms relating them [40, 41]. Some possible ways to describe ontologies are as follows [35]:

- A thesaurus describing the relationships between a vocabulary that represents a set of concepts.
- A taxonomy representing how the concepts of a knowledge domain are related using a classification based on similarities of structure.
- A detailed database schema describing the concepts (entities and attributes) and relationships.
- A logical theory, founded on mathematical logic, that specifies the concepts and relationships.

The benefits of the use of ontologies in the context of MAS are as follows:

- Shared understanding.
- Supporting semantic interoperability among agents.

- Suitable for open domains.

Ontologies have proven to be a useful mechanism to help in the understanding and the analysis of information flow among agents when attempting to describe a certain domain. Typically, each agent in MAS carries out a specific type of tasks and services with different aims. One agent does not perform all steps in the task. Agents may collaborate with different agents, using the content of messages exchanged among agents. In order for a pair of agents to understand each other, a basic requirement is that they speak the same language and talk about the same things. Ontologies provide a mechanism for structuring the concepts, relationships and constraints contained in a specific domain so that the description can be used by, for example, software agents, so that the agents have a shared understanding of the domain. This shared understanding then can be used to facilitate communication, negotiation and interaction among agents.

In closed domains, agents are able to interact to achieve their goals according to fixed protocols with fixed participants. However, in this case, agents are unable to deviate from expected behaviours. The use of ontologies in open and flexible systems allow them to adapt to the participation of heterogeneous agents with different agendas. Thus the use of ontologies provides a more flexible mode of operation for MASs (used in open domains) than fixed protocols.

Examples of multi-agent systems that used ontologies to achieve a shared understanding include Ontolingua [36], InfoSleuth [10] and the work presented in [82]. Ontolingua was a web-based service that provided a set of tools and services to support ontology development by different groups so that the ontology could be shared. The InfoSleuth project aimed at developing technologies to facilitate interoperation among agent systems in a dynamic and open environment. The project built on KQML and KIF. A common service ontology was defined in order to match an agent to the requirements of a task. A matchmaking agent used this ontology to reason over agent capabilities to recommend agents for specific tasks. The ontology provided a dictionary of meta-information about agent capabilities, and a shared set of prescriptive conversation policies to provide a structure for basic agent dialogue. In [82] the proposed approach allowed agents to negotiate in any type of marketplace regardless of the negotiation protocol. To support a wide variety of of negotiation mechanisms an ontology was used to represent the negotiation protocol, rather than using hard-coded in the agents.

In the context of work presented in this chapter a taxonomy, called OntoDM, was used rather than a fully operational ontology. Intra agent messaging is conducted using a sequence of communicative acts. A taxonomy based approach was selected because it offers several advantages over hard coding. With respect to the Agent Communication Language (ACL), agents are able to understand the structure of exchanged messages and the associated *performatives*. However, the ACL does not support semantic interoperability, hence the requirement for an taxonomy such as OntoDM. The main aim of the use of a taxonomy based approach is to provide a pragmatic approach to the sharing of "terms" among agents when passing parameters in agent communications concerning, for example data set information, data mining techniques, required parameters or clustering metrics.

A sample set of communicative acts, directed at clustering operations, is presented as a context free grammar in BackusNaur form (BNF) in Table 3.1. In the table the ... notation is used to indicate that there are further alternatives not included in the table (for ease of understanding). From Table 3.1 it can be seen that, at a high level, an OntoDM *utterance* comprises one of the following:

1. A **data informative**, indicating a data set, coupled with a **data mining request** to perform a specific data mining task with respect to that data; typically sent by a Task Agent to a Data Mining Agent.
2. A **validation request**, comprising information regarding the nature of the validation and the data on which the validation is to be performed; typically sent by a Data Mining Agent to a Validation Agent.
3. A **data request** typically sent by a Data Mining Agent to a Data Agent.
4. A **data response informative**, the response from a Data Agent to a successful data request from a Data Mining Agent.
5. A **results informative** used to return results to the originating Task Agent.
6. Some form of **error informative** used to handle situations where agents are unable to respond to an utterance because of some technical malfunction.

3.2 Multi-Agent Based Clustering Demonstration

The proposed MADM framework introduced above was designed to act as a platform to support generic data mining. With reference to Figure 3.1, the desired data mining process commences with an end user instructing his/her User Agent to perform a specific data mining task. In the implementation of the MADM framework, this instruction was facilitated by a GUI included as part of the User Agent. The User Agent then spawns a specific Task Agent as directed by the end user's instruction. The framework is facilitated with a number of kinds of generic Task Agent. The generated Task Agent then interacts with the house keeping agents to identify those agents that may best contribute to the resolution of the given data mining task. This generic process is illustrated in Figure 3.1 where the directed arcs indicate communications between agents.

Table 3.1: The OntoDM.

\∗ Top level utterances ∗\		
< UTTERANCE >	::=	< DATABASE >< TASKTYPE > | < VALIDATION > | | < DATA_REQ > | < DATA > | < RESULTS > | < ERROR > ;
\∗ Data Informative ∗\		
< DATABASE >	::=	“Database” URL| “Database” URL < DATA_PARAMS >;
< DATA_PARAMS >	::=	“All” | < FROM_REC_NO >< TO_REC_NO > | < FROM_ATT_NO >< TO_ATT_NO > | < FROM_REC_NO >< TO_REC_NO > | < FROM_ATT_NO >< TO_ATT_NO > ;
\∗ Data Mining Request ∗\		
< TASKTYPE >	::=	< CLUSTERING > |... ;
< CLUSTERING >	::=	< NAME_OF_ALGO >| < NAME_OF_ALGO >< CLUSTERING_PARAM > ;
< NAME_OF_ALGO >	::=	“K-means”|“K-NN”|“Divisive HC” ;
< CLUSTERING_PARAM >	::=	int | double | double int ;
\∗ Validation Request ∗\		
< VALIDATION >	::=	< UNSUPERVISED_VALIDATION > |... ;
< UNSUPERVISED_ VALIDATION >	::=	< APPROACH_NAME > | < APPROACH_NAME >< VALIDATION_PARAM > ;
< APPROACH_NAME >	::=	“WGAD” | “BGAD” | “F-measure” | ... ;
< VALIDATION_PARAM >	::=	< RESULTS > | ... ;
\∗ Data Request or Informative ∗\		
< DATA_REQ >	::=	< DATA_PARAMS > ;
< DATA >	::=	array_of_data ;
\∗ Results Informative ∗\		
< RESULTS >	::=	< CLUSTER_LIST_OF_LISTS >< ACCURACY > | ... ;
< CLUSTER_LIST_OF_LISTS >	::=	“[”< CLUSTERLIST >“]”|“[ ]” ;
< CLUSTERLIST >	::=	“[”< CLUSTER >“]”| ; “[”< CLUSTER >“]”< CLUSTERLIST > ;
< CLUSTER >	::=	< RECNO >|< RECNO >< CLUSTER > ;
< RECNO >	::=	int ;
\∗ Error Informative ∗\		
< ERROR >	::=	“null” ;
\∗ Miscellaneous ∗\		
< FROM_REC_NO >	::=	int ;
< TO_REC_NO >	::=	int ;
< FROM_ATT_NO >	::=	int ;
< TO_ATT_NO >	::=	int ;
< ACCURACY >	::=	double ;

Note that in the chapter subsequent to this one various refinements to the framework, as described in this chapter, will be suggested specifically in the context of MABC.

With respect to MABC, once the Task Agent has identified appropriate clustering Data Mining Agents (there are a sequence of N of these in Figure 3.1) the Task Agent requests the identified Clustering Agents to conduct the desired clustering with respect to the indicated data. Each Clustering Agent then communicates with the appropriate Data Agent (the Data Agent that has access to the indicated data source). As a consequence the Data Agent passes the data back to the Clustering Agent which then generates a set of clusters according to its specific clustering algorithm. The generated results are then passed to the Validating Agent, which determines the most appropriate set of clusters, according to some metrics (experiments using F-measure, WGAD and BGAD are reported in Section 3.3), and returns the "best" configuration to the Task Agent. The Task Agent then returns the result to the user, via the User Agent, after which the process is ended and the Task Agent ceases to exist (but the remaining agents persist).

This process is illustrated in Table 3.2, in terms of the agent communicative acts presented in Table 3.1 that are used at each stage. Note that the table does not include communication with the house keeping agents or communication from the User Agent to the Task Agent as neither is conducted using OntoDM. Communication with house keeping agents is as dictated by JADE. Communication from the User Agent to the Task Agent is integral to the Task Agent spawning process.

Table 3.2: Clustering Process in Terms of the Agent Communications using OntoDM.

Step	From Agent	To Agent	Message Content
1	Task Agent	Clustering Agent	"Database" URL "All" $< NAME\_OF\_ALGO >$ $< CLUSTERING\_PARAM >$
2	Clustering Agent	Data Agent	"All"
3	Data Agent	Clustering Agent	array_of_data
4	Clustering Agent	Validation Agent	$< APPROACH\_NAME >$ $< CLUSTER\_LIST\_OF\_LISTS >$
5	Validation Agent	Task Agent	$< CLUSTER\_LIST\_OF\_LISTS >$ $< ACCURACY >$
6	Task Agent	User Agent	$< CLUSTER\_LIST\_OF\_LISTS >$ $< ACCURACY >$

The above scenario has been realised, and experimented with using a JADE implementation of the proposed MADM framework. Three Clustering Agents were included, each with a distinct clustering algorithm: (i) K-means, (ii) K-NN and (iii) Divisive hierarchical clustering. Note that more clustering algorithms can be easily be added to the framework (provided they subscribe to OntoDM).

3.3 Evaluation

For evaluation purposes, a number of experiments were conducted which were designed to analyse the operation of the proposed framework in relation to clustering activities. More specifically, the experiments were designed to demonstrate that the proposed MADM approach can identify a best set of clusters by using clustering metrics to evaluate the quality of cluster configurations, and to compare a number of clustering metrics that may be used for identifying a best cluster configuration (namely the WGAD cluster cohesion metric and the BGAD cluster separation metric).

A sequence of pre-labelled ("classification") data sets taken from the UCI machine learning data repository [38] were used. The data sets were all pre-labelled with class values. Each class was considered to present a cluster. Thus the results produced using the proposed MADM framework can be compared with the known cluster configuration. Note that the MADM framework makes no use of these class labels when undertaking clustering operations, they are only used here to evaluate the outcomes. To determine the "accuracy" of any clustering operation the F-measure and the WGAD and BGAD metrics were used. A description of the F-measure, and the WGAD and BGAD metrics, was presented in Chapter 2 Subsection 2.2.2.

For the experiments three clustering Data Mining Agents were used, each equipped with a different clustering algorithm: (i) K-means, (ii) K-NN or (iii) divisive hierarchical clustering.

3.3.1 Performance Comparison of the MADM Framework When Used For MABC

This subsection describes the results obtained from experiments used to evaluate the performance of the proposed MADM framework when applied to MABC. With respect to the K-means and divisive hierarchical clustering algorithms, so that the number of desired clusters may be specified in advance, the number of classes given in the UCI repository were used as the input (K-NN determines it own most appropriate number of clusters). The parameter used for K-NN was a threshold, τ, used to determine the nearest neighbour. Note that the most appropriate τ value was derived using an "alternative approach to identifying best parameters" process as described later in this thesis in Chapter 6.

Table 3.3 presents some statistical information regarding the data sets used for the reported evaluation. The table gives the size of the data set (in terms of the number of data records and attributes), the number of predefined classes and the nature of the attributes (numerics, real, ordered, binary, nominal, integer and/or boolean).

Table 3.4 lists the results obtained using the proposed MADM approach to MABC when applied to the ten selected data sets. The table gives the number of clusters produced and the associated F-measure (F1), for each of the three clustering algorithms

Table 3.3: Statistical information for the data sets used in the evaluation.

No.	Data Set	Num Records(N)	Num Attr.	Num Classes	Attribute Description
1	Breast Tissue	106	9	6	9 Numeric
2	Iris	150	4	3	4 Numeric
3	Wine	178	13	3	13 Numeric
4	Heart	270	13	2	6 Real, 1 Ordered, 3 Binary, 3 Nominal
5	Ecoli	336	7	8	7 Real
6	Blood Trans.	748	4	2	4 Integer
7	Pima Indians	768	8	2	8 Numeric
8	Ionosphere	351	34	2	2 Boolean, 32 Numeric
9	Breast Cancer	569	30	2	30 Numeric
10	Yeast	1484	8	10	8 Numeric

and the values for the required parameters (K and τ). The clustering configurations generated by each Clustering Agent were communicated to the Validation Agent for evaluation using the F-measure. The clustering result with the highest F1 value was then selected by the Validation Agent and sent back to the User Agent (via the Task Agent).

Table 3.4: The clustering results as produced by the MABC framework.

No.	Data sets	K-means		K-NN			Divisive	
		Num Classes	F1	τ	Num Classes	F1	Num Classes	F1
1	Breast Tissue	6	0.43	3692.43	6	0.30	6	0.35
2	Iris	3	0.88	0.99	3	0.78	3	0.83
3	Wine	3	0.71	164.31	3	0.68	3	0.66
4	Heart	2	0.61	58.99	2	0.59	2	0.52
5	Ecoli	8	0.63	0.42	7	0.61	8	0.71
6	Blood Trans.	2	0.71	3500.83	2	0.75	2	0.67
7	Pima Indians.	2	0.64	149.08	2	0.70	2	0.58
8	Ionosphere	2	0.71	5.31	2	0.69	2	0.74
9	Breast Cancer	2	0.84	826.75	2	0.77	2	0.91
10	Yeast	10	0.42	0.34	10	0.41	10	0.43

Table 3.5 gives the best result, returned to the end user in each case (as identified by the Validation Agent). Table 3.5 supports the observation that there is no single best clustering algorithm, consequently supporting the motivation for the proposed mechanism and the research described in this thesis. From the table it can also be seen that there is no obvious link between particular clustering algorithms and the features associated with individual data sets. The data sets presented in bold font in Table 3.5 indicate the cases where the MABC, using the proposed MADM, identified the most appropriate clustering algorithm which provided a best set of clusters. From Table 3.5

it can be seen that in eight of ten cases, the most appropriate clustering configuration was identified.

Thus it can be argued that the proposed approach worked well in the majority of cases. It is also worth noting that, in many cases, K-means and Divisive hierarchical clustering algorithms tended to out perform K-NN.

Table 3.5: The "best" cluster result identified using the MADM framework.

No.	Data sets	F-Measure	Best clustering algo.
1	**Breast Tissue**	0.43	K-means
2	**Iris**	0.88	K-means
3	**Wine**	0.71	K-means
4	**Heart**	0.61	K-means
5	Ecoli	0.71	Divisive
6	**Blood Trans.**	0.75	K-NN
7	**Pima Indians.**	0.70	K-NN
8	**Ionosphere**	0.74	Divisive
9	**Breast Cancer**	0.91	Divisive
10	Yeast	0.43	Divisive

Although good results were obtained using the external F1 measure, of course in "real life" we would not know in advance what the "right" number of clusters might be. Thus some alternative measure of cluster configuration correctness is required. With respect to the work described in this thesis use of the internal WGAD and BGAD measures is proposed. A comparison of these two measures is presented in the following subsection.

3.3.2 Comparison of The WGAD and BGAD Metrics

In order to compare the WGAD and BGAD measures, so as to determine which was the best suited to the identification of a best cluster configuration, a number of experiments were conducted:

- Identification of the WGAD and BGAD values generated using K-means where K is set to the known number of clusters.
- Identification of the WGAD and BGAD values generated using K-NN where τ is set according to the known number of clusters.
- Identification of the WGAD and BGAD values generated using divisive hierarchical clustering and the known K value to control the decomposition.
- A comparison between WGAD and BGAD to determine which metric served to identify the best cluster configuration in the majority of cases.

Each is discussed in the remainder of this subsection. For the purpose of the comparison (the fourth set of experiments), the same parameter values for K and τ used in the earlier experiments were used. The values were dependent on the nature of the data set. The results of the experiments, using K-means, K-NN and Divisive hierarchical clustering, are reported in Tables 3.6 to 3.8 respectively. Note that the number of generated classes is equal to K for K-means and divisive hierarchical clustering while the number of generated classes for K-NN is dependent on the τ value used. Hence, the number of generated classes is included in Table 3.7. Each row in each table includes the clustering parameter for each clustering paradigm and the WGAD and BGAD values. Recall that the Validation Agent uses a clustering metric to evaluate clustering configurations generated by Clustering Agents using different clustering algorithms and hence identify a best cluster configuration. In the case of the Validation Agent using the WGAD metric the clustering result with the lowest WGAD value would be selected to be sent back to the User Agent. On the other hand, the highest BGAD value is required when using the BGAD metric. The results are reported in Table 3.9.

Table 3.6: WGAD and BGAD values using K-means.

No.	Data Set	K-means		
		K	normalised WGAD	normalised BGAD
1	Breast Tissue	6	1.00	1.00
2	Iris	3	0.92	0.99
3	Wine	3	0.95	0.98
4	Heart	2	0.98	0.99
5	Ecoli	8	0.92	0.86
6	Blood Trans.	2	0.94	0.58
7	Pima Indians	2	0.90	0.67
8	Ionosphere	2	0.99	0.72
9	Breast Cancer	2	0.99	0.66
10	Yeast	10	0.78	0.67

Figure 3.2 presents the comparison of the WGAD and BGAD values produced by K-means, K-NN and divisive hierarchical clustering using the ten UCI data sets. Table 3.9 gives a summary of the operation of MABC using the proposed framework when the best cluster configuration is chosen by either: (i) minimising the WGAD measure, or (ii) maximising the BGAD measure. The table lists the best performing WGAD and BGAD values with respect to each of the considered data sets as presented in Tables 3.6 to 3.8. In each case the associated clustering algorithm that produced the best performance according to each clustering metric is also listed. It is interesting to note, from the table, that there is little agreement (except in the case of the $Ecoli$, the $Yeast$ and the $Ionosphere$ data sets) between the two metrics regarding the most appropriate clustering algorithm given a specific data set.

Table 3.7: WGAD and BGAD values using K-NN.

No.	Data Set	K-NN			
		Num Classes	τ	normalised WGAD	normalised BGAD
1	Breast Tissue	6	3692.43	0.90	0.98
2	Iris	3	0.99	1.00	1.00
3	Wine	3	164.31	0.96	1.00
4	Heart	2	58.99	0.99	1.00
5	Ecoli	7	0.42	0.72	1.00
6	Blood Trans.	2	3500.83	1.00	1.00
7	Pima Indians	2	149.08	1.00	1.00
8	Ionosphere	2	5.31	0.76	1.00
9	Breast Cancer	2	826.75	1.00	1.00
10	Yeast	10	0.34	0.76	1.00

Table 3.8: WGAD and BGAD values using Divisive hierarchical clustering.

No.	Data Set	Divisive Hierarchical Clustering		
		K	normalised WGAD	normalised BGAD
1	Breast Tissue	6	0.89	0.96
2	Iris	3	0.98	0.96
3	Wine	3	1.00	0.86
4	Heart	2	1.00	0.92
5	Ecoli	8	1.00	0.94
6	Blood Trans.	2	0.83	0.48
7	Pima Indians	2	0.83	0.59
8	Ionosphere	2	1.00	0.68
9	Breast Cancer	2	0.99	0.61
10	Yeast	10	1.00	0.86

Table 3.9: Summary of the best WGAD and BGAD values obtained with respect to Tables 3.6 to 3.8 and the best F-measure values obtain with respect to Table 3.4.

No.	Data Set	Best clustering algo.(WGAD)	Best clustering algo.(BGAD)	Best clustering algo.(F1)
1	Breast Tissue	Divisive	**K-means**	**K-means**
2	Iris	**K-means**	K-NN	**K-means**
3	Wine	**K-means**	K-NN	**K-means**
4	Heart	**K-means**	K-NN	**K-means**
5	Ecoli	K-NN	K-NN	Divisive
6	Blood Trans.	**Divisive**	**K-NN**	**K-NN**
7	Pima Indians	**Divisive**	**K-NN**	**K-NN**
8	Ionosphere	K-NN	K-NN	**Divisive**
9	Breast Cancer	**Divisive**	K-NN	**Divisive**
10	Yeast	K-NN	K-NN	Divisive

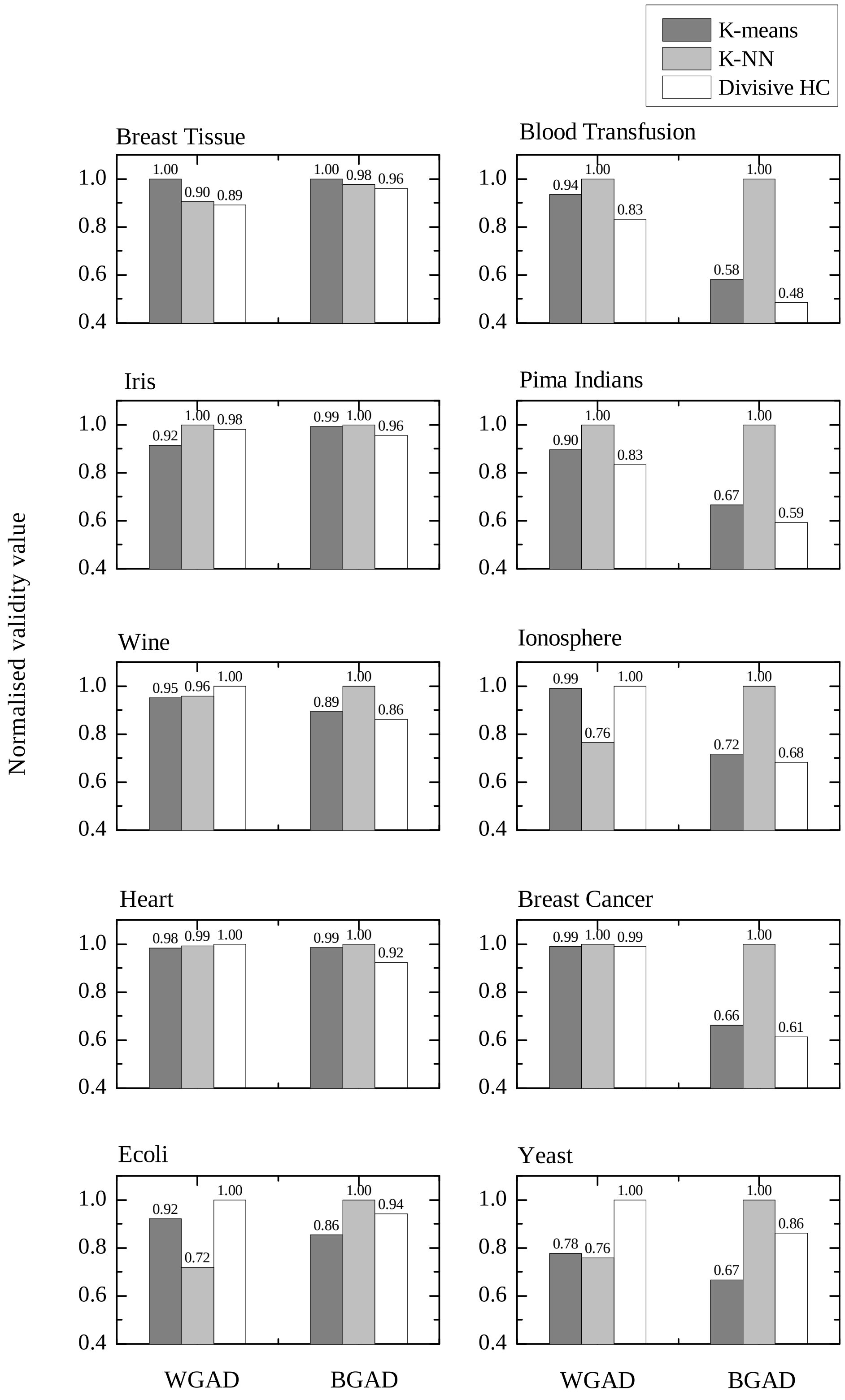


Figure 3.2: Histograms of WGAD and BGAD produced using K-means, K-NN and divisive hierarchical clustering algorithms.

Table 3.10 gives the "clustering accuracy", associated with the experiments described in Tables 3.6 to 3.8, with respect to a "ground truth" configuration, the accuracy is calculated as follows:

$$Accuracy = \frac{\sum_{i=1}^{i=K} C_i}{m} \quad . \tag{3.1}$$

Where K is the number of clusters, m is the number of total records and C_i is the size (number of records) of the majority class for cluster i. Not that the accuracy produced is not dependent on the "known" number of clusters (classes). Thus if a given input data set is known to have three classes (x, y and z) and the framework produces four clusters, the first two of which contain only examples of class x, the third contains only examples of class y and the fourth contains only examples of class z; then an accuracy value of 100% will be returned. Of course the "ground truth" accuracy presented in Table 3.10 may not represent the "best" cluster configuration, however the table suffices as a guide to what may be the most appropriate configuration. Table 3.10 also (last column) lists the clustering algorithm used to generated the best result.

The clustering algorithms presented in bold font in Table 3.9 indicate the cases where the MABC, using the proposed MADM approach, identified the best performing clustering algorithm which generated a best cluster configuration. By cross referencing between Tables 3.9 and 3.10 it can be seen that, with respect to the *Blood Transfusion* and *Pima Indians* data sets, both WGAD and BGAD identify the best performing clustering algorithm. For these two cases all three clustering algorithms produced a best set of clusters according to the best accuracy result presented in Table 3.10. In three cases (*Ecoli*, $Yeast$ and $Ionosphere$) neither of the two approaches finds the most appropriate algorithm. From the remaining five data sets WGAD operates most successfully in four of the five cases. An argument can therefore be made in favour of WGAD as the most appropriate mechanism for identifying best cluster configurations.

Table 3.10: Best accuracy values produced using K-means, K-NN, and Divisive hierarchical clustering algorithms from Tables 3.6 to 3.8.

No.	Data Set	K-means Acc.(%)	K-NN Acc.(%)	Divisive HC Acc.(%)	Best clustering algo.
1	Breast Tissue	43	30	32	K-means
2	Iris	89	84	83	K-means
3	Wine	70	65	69	K-means
4	Heart	61	59	55	K-means
5	Ecoli	83	65	72	K-means
6	Blood Trans.	76	76	76	K-means,K-NN,Divisive
7	Pima Indians.	66	66	66	K-means,K-NN,Divisive
8	Ionosphere	71	64	74	Divisive
9	Breast Cancer	85	79	91	Divisive
10	Yeast	54	34	40	K-means

3.4 Critique and Conclusion

In this chapter, a proposed MADM framework has been described. This was evaluated by using it for MABC. The motivation was two fold. The first was to provide the solution to the "best cluster configuration" problem. The second was to illustrate the operation of MABC within the context of a generic MADM approach. To identify a best clustering configuration the use of WGAD and BGAD was explored and compared. The evaluation, with respect to MABC, clearly demonstrates that: (i) an MADM framework can successfully be used to find a best cluster configuration given a number of competing clustering algorithms and (ii) that clustering metrics (such as WGAD and BGAD) can be used to identify a best cluster configuration. However, it may be appropriate to consider both WGAD and BGAD together as will be demonstrated in the next chapter.

The MADM framework proposed in this chapter represents the first attempt at providing a generic approach to MABC. Criticism of the proposed approach can be considered under the following two headings:

- Decentralised control
- Capabilities of agent

Decentralised control is, arguably, one of the most significant advantages offered by MAS with respect to the data mining process where it is often necessary to process distributed resources. In MASs, each agent is designed to achieve its delegated objectives in an independent manner (an agent does something because it wants to, not because it has to). One criticism of the proposed MADM framework described in this chapter is that the framework seems not to feature what might be referred to as "real decentralised control". Task Agents act as coordinators or facilitators with respect to other agents in the system, the other agents have to communicate via the Task Agent. To make agents more independent the role of the Task Agent needs to be reconsidered.

The second criticism refers to the fact that true capabilities of agents have not been adopted, such as cooperation or negotiation. These social capabilities may offer benefits with respect to MABC. In the set-up described in this chapter the proposed MADM framework allows a number of clustering Data Mining Agents to perform a clustering task using different clustering algorithms applied to the same data set. To identify a best cluster configuration, each clustering result is evaluated by the Validation Agent. This means that the same task (albeit in a different manner) is performed by different agents. It would be more in the spirit of MAS if a collection of agents collaborated to produce a best cluster configuration. These agents would interact with each other so as to produce a best cluster configuration using a process of cooperation and negotiation.

A criticism of the use of ontologies, such as the proposed OntoDM, is that all interested parties need to agree on the ontology to be used for a specific domain. There are also problems when a system is intended for use across two or more domains. Although the operation of OntoDM was perfectly adequate with respect to the proposed framework it can be argued that the use of such ontologies should be avoided and replaced with a much simpler set of "utterances" that all agents that may be incorporated into the proposed framework subscribe to. In the following chapter a revised version of the proposed framework is presented. The revisions are designed to address the criticisms of control and the lack of utilisation of the full capabilities of agents featured in the current framework, and the effectiveness of the use of ontologies such as OntoDM.

Chapter 4

Agent Negotiation for Multi-Agent Based Clustering

From the previous chapter, and as noted by many other practitioners, there is no single best clustering method for all possible data sets and that the selection of appropriate clustering parameters greatly influences clustering results. This chapter describes the extension of the proposed Multi-Agent Data Mining (MADM) framework introduced in the previous chapter. The most significant feature of the proposed extended MADM framework, in the context of MABC, is that it allows agents to negotiate so as to improve an initial cluster configuration. The revisions are designed to address the criticisms of control and lack of utilisation of the full capabilities of agents featured by the previous proposed framework. Thus, the extended framework allows agents to be more independent and provides for greater agent interaction. In the revised framework there are no coordinators or facilitators that are responsible for managing and controlling the overall clustering process.

The extended framework encourages a two phase approach to clustering. The first (Phase 1) comprises the application of one of a number of standard clustering techniques, except in a decentralised manner. This allows a collection of agents to cooperate to perform clustering tasks and then generate an initial cluster configuration. How this is achieved depends on the nature of the clustering paradigm adopted. The second (Phase 2) comprises a negotiation phase where agents seek to improve the initial cluster configuration produced in Phase 1; during this phase agents interact with one another with the aim of “swapping” records so as to improve their initial cluster configuration. It is this second phase that is the most novel feature of the proposed extended framework. The negotiation process can of course be “mimicked” within a centralised setting, however it is much more suited to a MAS infrastructure which readily supports the concept of negotiation. Three clustering paradigms are used to support Phase 1 to produce an initial clustering result: K-means, K-NN and divisive hierarchical clustering. The rest of this chapter describes the extended MABC approach directed at harnessing the true potential of MASs, including the negotiation mechanism used to improve cluster configurations. The main contributions of the work described in this chapter can be itemised as follows:

- A more versatile MADM framework (than that described in the foregoing chapter).
- A set of communication performatives to specifically support negotiation with respect to MABC.
- A negotiation protocol whereby agents can interact with one another with the express aim of exchanging records to improve cluster configurations.

It is this last contribution, the negotiation ability, that sets the proposed extension to the MADM framework apart from other clustering mechanisms such as distributed/parallel mechanisms. It can be argued that the proposed intra-agent negotiation allows the framework to harness the true potential of Multi-Agent Systems (MASs) rather than simply using the concept of a MAS to achieve a distributed clustering.

The remainder of this chapter is organised as follows. Section 4.1 gives an overview of the proposed extended MADM framework. Section 4.2 describes the adopted communication mechanism and details the performatives used to achieve the desired clustering, including the operation of the extended MABC framework in terms of these performatives. Section 4.3 describes the operation of the proposed extension to the extended MADM framework from an algorithmic perspective. To facilitate the complete understanding of the extended MABC approach Section 4.4 illustrates its operation using a simple example. Section 4.5 concludes the chapter.

4.1 The MABC framework

The extended MADM framework architecture is similar to the architecture of the proposed MADM framework introduced in the previous chapter. The extended framework again comprises the four principle categories of agent identified previously: (i) User Agents, (ii) Data Agents, (iii) Validation Agents and (iv) Data Mining Agents. However, the architecture of the extended MADM framework described in this section differs from that previously described in that there are no coordinators or facilitators (Task Agents) to manage and control (clustering) tasks because of the criticism of control as discussed in Chapter 3. Thus, the agents are more independent to carry out their tasks.

The extended framework allows a collection of agents to cooperate to perform clustering tasks and improve their cluster configurations. Hence, the Clustering Agent[1] functionality is revised to achieved the desired extended framework functionality. Two

[1]Recall that Clustering Agents are the type of Data Mining Agent.

main new functions of the Clustering Agents are: (i) generation of an "initial" cluster configuration and (ii) cluster configuration refinement through a negotiation mechanism.

Each Clustering Agent represents a cluster, thus a group of Clustering Agents can be thought of as representing a clustering algorithm and a set of clusters (a clustering result). Note that this Clustering Agent concept significantly differs from the Clustering Agent concept presented in the previous framework in which each Clustering Agent was equipped with a clustering algorithm and held a complete set of clusters. The Clustering Agents are able to cooperate to generate an initial cluster configuration and spawn additional Clustering Agents as required. The number of Clustering Agents used for a particular clustering task depends on the nature of the adopted clustering paradigm and the nature of the input data set.

The extended framework is designed to be used with a number of different clustering paradigms. With respect to the thesis the framework is illustrated using the well known K-means, K-NN and divisive hierarchical clustering paradigms. However, in each case, these algorithms were revised so that the Clustering Agents could perform the clustering task in a decentralised manner and cooperate to produce the final clustering result. At the start of the process the User Agent spawns one or more Clustering Agents depending on the adopted clustering paradigm. The Clustering Agents represent clusters. In the case of the K-means algorithm the number of clusters is predefined; thus, by extension, the number of Clustering Agents that will be spawned will also be predefined. In the case of the K-NN and divisive hierarchical clustering algorithms the User Agent will only spawn a single Clustering Agent, however Clustering Agents may spawn further Clustering Agents as the clustering process progresses. The framework also allows for two Clustering Agents to merge to form a single cluster (in which case one of the Clustering Agents will "disappear"), for use in (say) conglomerative hierarchical clustering.

To achieve the generation of an initial cluster configuration using cooperation between Clustering Agents in Phase 1, each Clustering Agent has to determine whether each given record fits well with its cluster or not. If the record is similar to other records in the cluster, the Clustering Agent will adopt the record and merge the record into its cluster. The Data Agent is responsible for offering up records for adoption by Clustering Agents thus the Data Agent also operates in a slightly different manner so that described with respect to the original MADM framework. Note that in the examples presented in this chapter it is assumed that a single Data Agent exists, it is of course possible for the framework to operate using more than one Data Agent (thus multiple data sources). The Clustering Agents then place a "bid" for the record; the record will be assigned to the winning Clustering Agent. To win the bid, with respect to K-means, each Clustering Agent determines the distance between the record and its cluster centroid. The winning Clustering Agent is then the agent which holds the shortest distance. With respect to K-NN the bid is the nearest neighbour distance. The record adoption process in Phase 1, using divisive hierarchical clustering differs from the record adoption process for K-means and K-NN in that the splitting Clustering Agent will offer up records for a new spawned Clustering Agent. Details concerning the operation of K-means, K-NN and divisive hierarchical clustering with respect to the extended MABC framework are presented in Section 4.3.

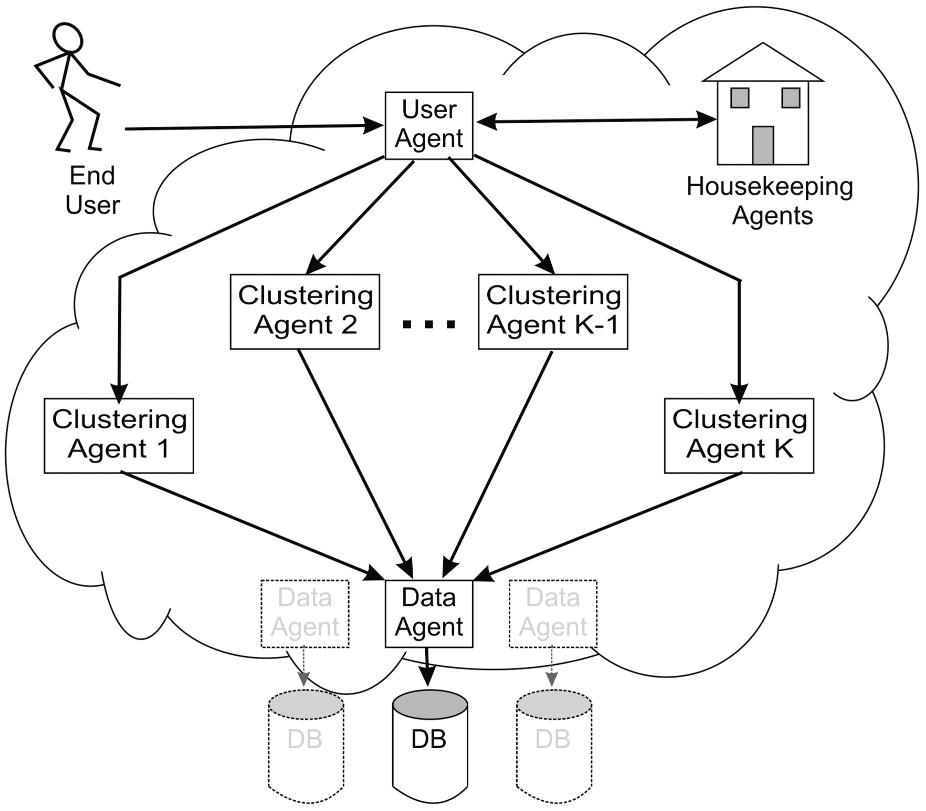


Figure 4.1: Schematic of initial cluster configuration using K-means.

The Clustering Agent is also able to evaluate its cluster configuration to determine whether its cluster configuration should be refined (improved) or not. The cluster configuration is assessed in terms of cluster cohesion and cluster separation according to the cohesion threshold and the separation threshold. The WGAD and the BGAD metrics are suggested for this purpose. If the initial cluster configuration is satisfactory the process ends. Otherwise the negotiation stage is entered (Phase 2). If a Clustering Agent finds that its cluster configuration should be refined, the Clustering Agent will identify a set of unwanted records that do not fit well with its cluster and put them up for adoption by other Clustering Agents. The idea of adoption in Phase 2 is independent from the idea of adoption in Phase 1 in terms of the “bid value”. In Phase 2, the bid value is a cluster cohesion value of a cluster belonging to the Clustering Agent which wishes to adopt the record. The bid is calculated assuming that the record is already merged into its cluster. Any "unwanted" records are identified as "outliers". As already mention above, the framework allows two Clustering Agents to merge into a single cluster; in this case one of Clustering Agents puts all its records up for adoption. Thus this second Clustering Agent will have no records within its cluster and will therefore disappear. Detail concerning the operation of the negotiation element of the process is presented in Section 4.3.

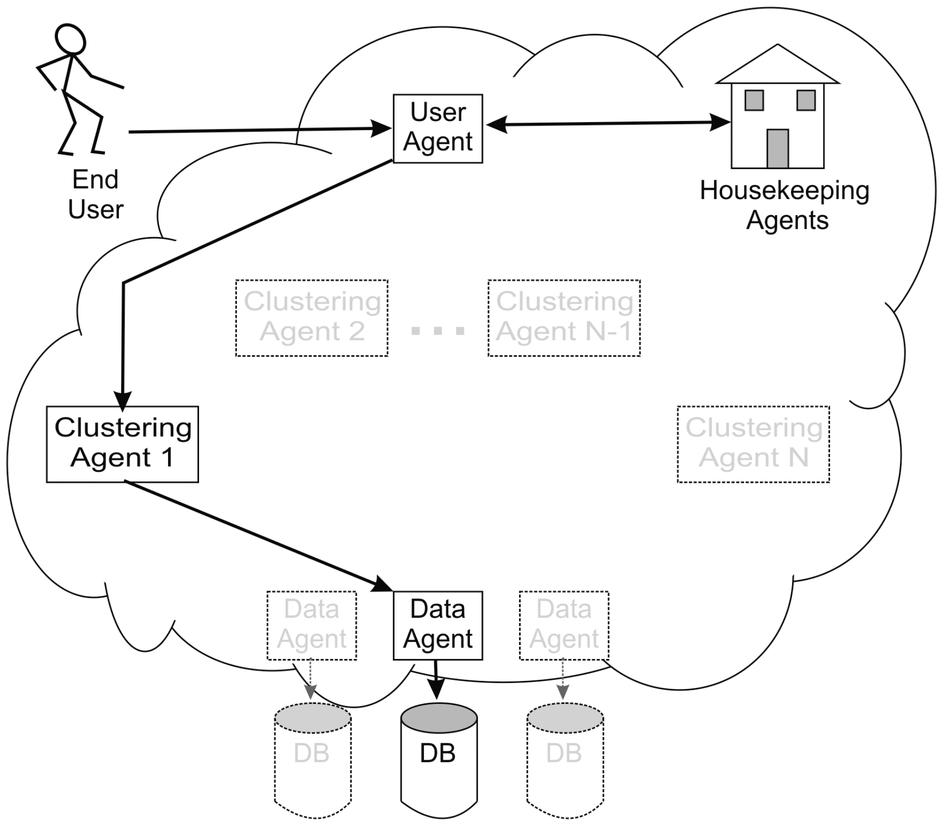


Figure 4.2: Schematic of initial cluster configuration using either K-NN or divisive hierarchical clustering.

Figures 4.1 and 4.2 present two possible MABC agent configurations (to facilitate ease of understanding a Validation Agent has not been included in the figures). Figure 4.1 presents a generic K-means configuration where K Clustering Agents are spawned at the outset. Figure 4.2 presents a generic K-NN or divisive hierarchical clustering configuration where only one Clustering Agent is spawned at the outset (although further agents may be spawned later in the process). The directed arcs indicate communication between agents; communication can be bidirectional or unidirectional. Note that the desired clustering may be conducted using either a single data source or a distributed data source. Similarly, only one Clustering Agent is spawned with respect to the generic divisive hierarchical clustering configuration (although further agents may be spawned later during process). Both figures assume a single Data Agent; but, as indicated by the "grayed out" Data Agents, multiple Data Agents can be included. Note also that there are a number of "house keeping" agents in addition to the specific MABC agents listed above. In Chapter 2 it was noted that, with respect to the implementations conducted to support the work presented in this thesis, the house keeping agents are provided by the JADE framework. These house keeping agents have no direct connection with MABC, but provide various facilities to maintain the broad operation of the MAS.

Validation Agents in the extended MABC framework are responsible for measuring the quality of the final clustering result and sending the cluster validation value to the User Agent. As mention above, a Clustering Agent can measure the "goodness" of its cluster in terms of the cluster cohesion and separation using WGAD and BGAD respectively. The overall cluster cohesion and separation can be calculated, without reference to the Validation Agent, using communication between Clustering Agents. For this purpose, each Clustering Agent has to communicate its cluster centroid to all other agents. The Validation Agents are equipped with clustering metrics. Thus after finishing an initial cluster configuration phase or during the negotiation phase, the Clustering Agent can request the Validation Agent to evaluate its cluster configuration using clustering metrics such as the Silhouette Coefficient (Sil. Coef.), the Davies-Bouldin (DB) index and WGAD-BGAD. Note that for the evaluation of MABC using the extended MADM framework, only the WGAD and the BGAD were used (this evaluation is described in Chapter 5).

4.2 Agent Communication within the Framework

As noted in the review of agent communication presented in Chapter 2; the FIPA ACL, that JADE comes equipped with, is rather broad and not universally applicable to all types of dialogue that can take place within different domains. The extended MADM framework was specifically designed for use in MABC, hence the agents in the system should be able to communicate explicitly about their associated data clustering activities. Whilst it may be possible to exchange the required information within the content of FIPA ACL messages, it is desirable to make the syntax of the performatives used by agents in the system more specific and descriptive of the particular task that the agents are communicating about within the clustering domain. For this reason, a set of performatives was defined and implemented in order to enable the agents to negotiate about the suitability of moving records or merging records between clusters. The defined set of performatives follows some of the desiderata given in [61] that the FIPA ACL does not meet, for example "enablement of self-transformation"; the process whereby agents can change their beliefs, preferences or intentions as a result of their interactions with one another. To facilitate the expression of self-transformation, a "retract" performative is included. The details of the communication language are

Table 4.1: Performatives to support Multi-Agent Clustering System (I).

Performatives	**Pre-conditions**	**Post-conditions**
Simple dialogue		
Mining request (start dialogue)	An agent wishes to initiate a dialogue. No dialogue about this *mining request* is open.	Receivers (Data Agents, a collection of Clustering Agents, Validation Agents, etc.) obtain mining request. Subsequently, they can either accept or refuse the request to join the dialogue.
Join dialogue	The agent sending the request is not yet a participate in the dialogue. The agent wants to join the dialogue to cooperate on a clustering problem.	Sender maintains its intention to achieve its goal. The Sender agent is now a participant in the dialogue.
Refuse join dialogue	Sender has received a request to join the dialogue to cooperate on a clustering problem and is not a participant in the dialogue.	The Sender agent has not joined the dialogue. A reason for the refusal is sent to the initiator of the dialogue (e.g. "I am busy right now").
Leave dialogue	Sender is currently participating in the dialogue.	Sender has left the dialogue (regardless of whether its goal has been achieve), leaving it ready to join other dialogues.
Close dialogue	The User Agent has received the cluster configuration.	The agents have all left the dialogue.

presented below. The performatives used at each stage are indicated in italics.

The performatives are classified into four categories (at a high conceptual level) as follows:

1. Holding a dialogue.
2. Performing a clustering task.
3. Negotiating about the movement of records between the clusters.
4. Informing about clustering results.

Tables 4.1, 4.2, 4.3 and 4.4 illustrate the semantics of the defined performatives, classified into the previous categories (the performatives are split over four tables solely for ease of presentation). The tables give the names of the performatives, the pre-conditions and the post-conditions. The pre-conditions indicate the necessary "state" for an agent to send a given performative. The post-conditions describe the state of the

Table 4.2: Performatives to support Multi-Agent Clustering System (II).

Performatives	**Pre-conditions**	**Post-conditions**
Clustering task dialogue		
request data	Receivers and Sender are in the same dialogue. Receivers (i.e. Data Agents) have held a data source.	Receiver has information, such as Receiver agent name. Hence, the Receiver knows who will obtain its data.
Refuse sending data	Receiver and Sender are in the same dialogue. Sender currently has obtained the *request data* performative. Sender wishes not to send the data to the Receiver.	The reason for the refusal has been sent to the Receiver.
Inform data	Receivers and Sender are in the same dialogue. Receiver does not currently have a data.	Receivers (i.e. Clustering Agents) have obtained a set of items and started to perform the clustering task.
Refuse clustering task performing	Receiver and Sender are in the same dialogue. Sender currently has a data for the clustering task. Sender wishes not to perform clustering task.	The reason for the refusal has been sent to the Receiver.
Inform cluster centroid	Sender holds an initial cluster configuration and knows its cluster centroid. Receiver does not currently have this cluster centroid information.	Receiver keeps the cluster centroid for cluster separation evaluation. Receiver checks the number of cluster centroids. If the Receiver obtained all of other cluster centroids the Receiver performs cluster separation evaluation.
Refuse cluster centroid	Receiver and Sender are in the same dialogue. Sender currently has knowledge of the sender's cluster centroid. Sender wishes not to perform cluster separation evaluation.	The reason for the refusal has been sent to the Receiver.

sender after the successful utterance of a performative, and the state of the receiver after the receiving and processing of a message. The process can be illustrated using state transition diagrams. At the beginning of the clustering process a dialogue is opened by an agent who wishes to initiate a dialogue. Other agents are invited to join the dialogue (*join dialogue*). Meanwhile, a request for data is sent to the agent (or agents) who holds the data source (*request data*). The agent holding the data source then sends the data using the *inform* performative to a Clustering Agent (or a collection of Clustering Agents) designated to perform the clustering task (*inform data*). The clustering task is thus performed and the resulting initial cluster configuration generated and evaluated for "goodness". Each Clustering Agent transmits its cluster centroid to the other Clustering Agents, using an *inform cluster centroid* performative, so that the Clustering Agent can collectively calculate and evaluate the overall cluster separation. The *propose item move* performative is then used if a Cluster Agent finds that its cluster configuration can be improved. The recipients of an item move proposal can either refuse or accept (*refuse/accept item move*). The sender agent then determines who should receive the record (*confirm item move*). However, the sender agent can "change its mind" and retract a proposed item move using *retract item move* performative, the receiver has to remove the item from its cluster and the item will belong to the sender again. This may happened given the situation where a proposing agent, after suggesting an item move, adopts records from some other agent as a result of which its cluster centroid is altered and consequently the initial suggested move is no longer valid. At the end of the communication concerning cluster refinement, the Clustering Agents inform the clustering result to the User Agent who made the initial mining request (*inform cluster*). So that all eventualities are covered the set of performatives include the option for agents refuse requests. For example, although unlikely, there may be a situation where an agent might not send data when a *data request* performative has been received. In this case, the agent has to send the reason for the refusal to the sender agent who has requested the data using the *refuse sending data* performative.

The agents use the above performatives as part of a protocol that governs the exchange of information between the agents. To illustrate the use of the protocol, Figures 4.3 to 4.6 provide some simple state transition diagrams that show the dialogue moves (i.e. performatives that can be uttered) by the different agents and the subsequent choice of moves which are then available in the new state.

An User Agent establishes a dialogue by sending a *mining (clustering) request* as shown in Figure 4.3. The user agent also sends the request asking agents to join the dialogue (*join dialogue*), such as Data Agents, Validation Agents or Clustering Agents. Thereafter, the agents in the same dialogue know with whom and how they will interact and cooperate to perform some clustering task. The *request data* performative is sent by the Data Agent in order to commence a clustering process. The User Agent waits

Table 4.3: Performatives to support Multi-Agent Clustering System (III).

Performatives	**Pre-conditions**	**Post-conditions**
Cluster Ensemble dialogue		
Propose item move	Sender has found that its cluster and cluster cohesion and/or cluster separation could be improved.	Receiver has determined whether a given item set would belong to its cluster or not. After that, receiver can either accept or refuse the proposed item move.
Refuse item move	Sender has received *propose item move* and established that cluster cohesion could not be improved.	Receiver updated its information that the Receiver is not a candidate for item (record) adoption.
Accept item move	Sender has received *propose item move* and found that its cluster and cluster cohesion could be improved.	Receiver has updated its information that the Sender is a candidate for item (record) adoption and determined which agent would obtain the item set.
Confirm item move	Sender has received either *accept item move* or *refuse item move.*	Receiver has combined a given item set with existing items, in the case of an "accept" (Sender agreed that the Receiver is the item bidding winner); or does not perform any actions, in the case of a "refusal".
Retract item move	Sender has confirmed item move to the Receiver and received a *propose item move* from the other agent. The receiving item fits its cluster and the given item also fits to its cluster.	Receiver has removed the item from its cluster and sent reply message for notification.

Table 4.4: Performatives to support Multi-Agent Clustering System (IV).

Performatives	**Pre-conditions**	**Post-conditions**
Clustering result dialogue		
Inform cluster	The User Agent does not know the result of the cluster refinement. The Sender has no further refinements to make.	Receiver has performed a cluster evaluation process. Cluster validation value is sent to the sender. If all clusters have been arrived Receiver performs overall cluster evaluation process and replies the overall cluster configuration value to the User Agent, who initially sent the *mining request*.
Inform cluster configuration	The User Agent does not know the result of the cluster refinement. The Sender has no further refinements to make.	Receiver (a User Agent) obtains an improved cluster configuration.

until the final clustering result is obtained and then the dialogue will be closed.

After receiving the *request data* performative, the Data Agent(s) will allocate the data to associated Clustering Agents using the *inform data* performative and wait until an initial cluster configuration is produced and then leave the dialogue as shown in Figure 4.4.

The Validation Agent (Figure 4.5) will wait until a Clustering Agent sends a cluster for determination of the "goodness" of the cluster (*inform cluster*). The Validation Agent measures the cluster configuration and sends the result to the Clustering Agent who has sent the request for cluster configuration validation. The Validation Agent waits for another *inform cluster* and decides to leave the dialogue when the User Agent receives the final clustering result as presented in Figure 4.5.

When a Clustering Agent enters a dialogue (see Figure 4.6), each Clustering Agent waits for a record to be offered by the Data Agent, again using *inform data*, and then bids for the record in order to adopt the record to its cluster. Once all the records have been assigned, each Clustering Agent will hold an initial cluster. Each Clustering Agent then evaluates its cluster and determines whether its cluster could be refined or not. In the case where the cluster configuration is satisfactory, the Clustering Agent sends the cluster to the User Agent using *inform cluster*. Otherwise, where the cluster could be refined, unwanted records are identified and proposed for adoption by the other Clustering Agents using the *propose item move* performative. The proposing Clustering Agent then has to wait until an *accept item move*, *refuse item move* or *retract item move* performative arrives. The proposing Clustering Agents will wait until all responses arrive and then make a decision as to which agent to give the unwanted records to using *confirm item move*. While it is waiting, the proposing Clustering Agent may also receive *propose item move* performatives from other Clustering Agents, thus the Clustering Agent has to also consider whether any proposed record should be adopted or not. If a record "fits" its cluster the agent uses an *accept item move* performative, otherwise the *refuse item move* performative is used. When the Clustering Agent receives the *confirm item move* performative, that allows the Clustering Agent to adopt the record, the record will be merged into its cluster. The Clustering Agent can leave the dialogue after its cluster has been sent to the User Agent.

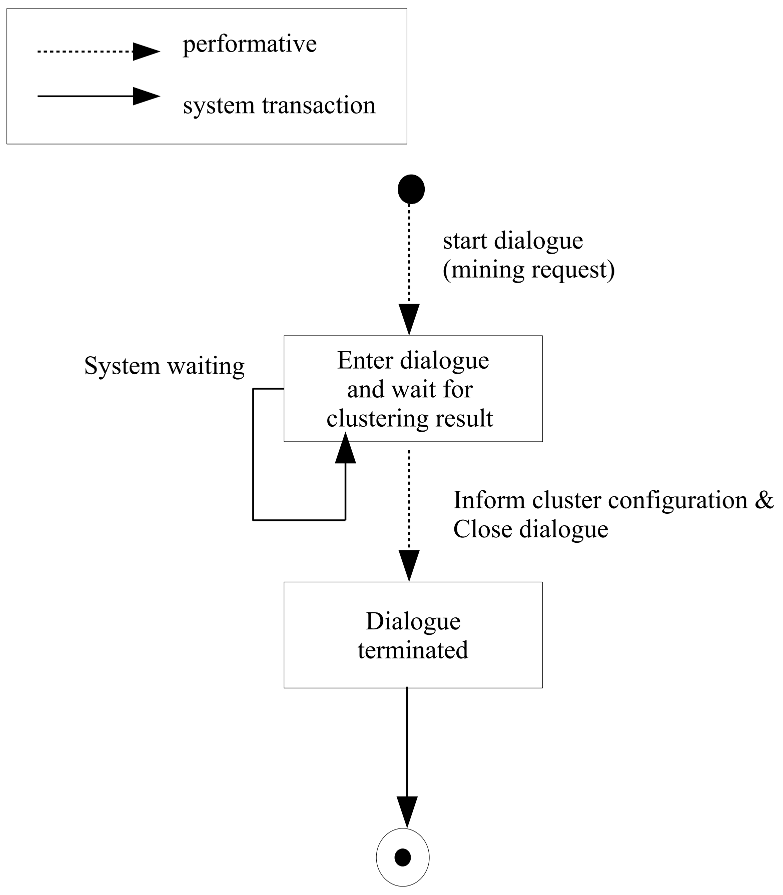


Figure 4.3: State transition diagram for User Agent.

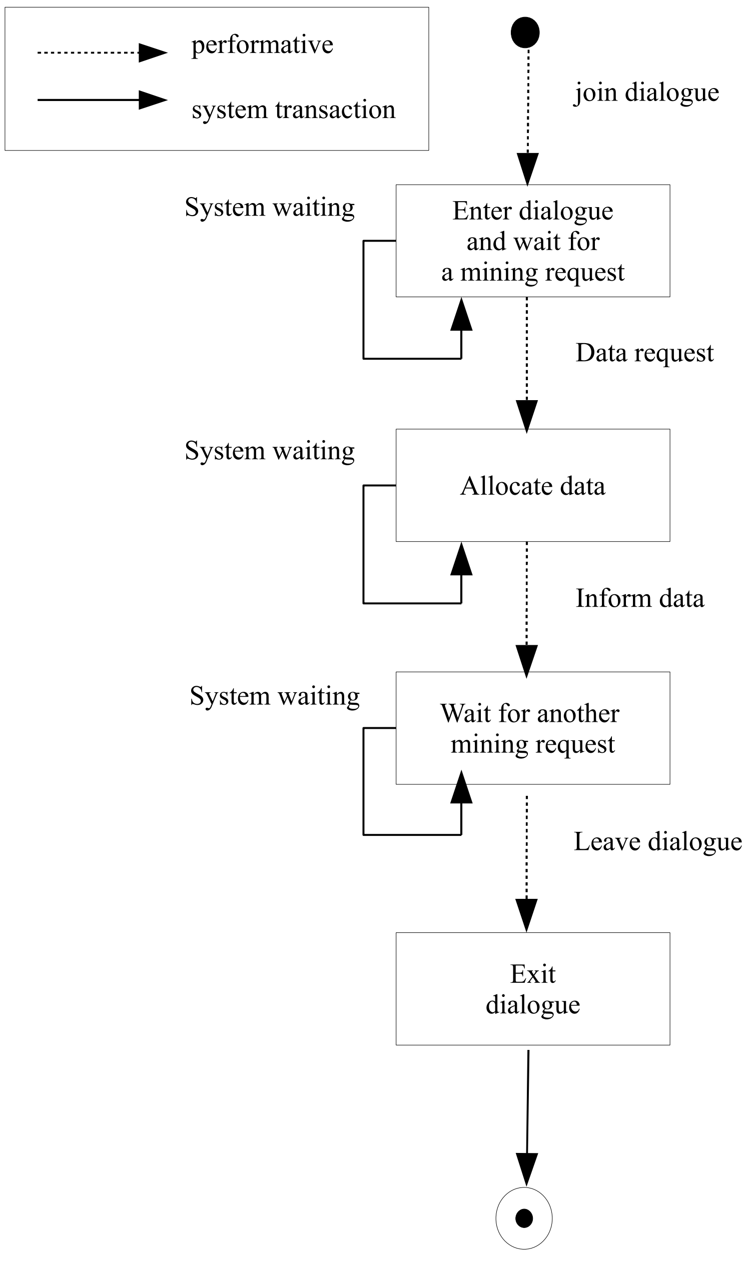


Figure 4.4: State transition diagram for Data Agent.

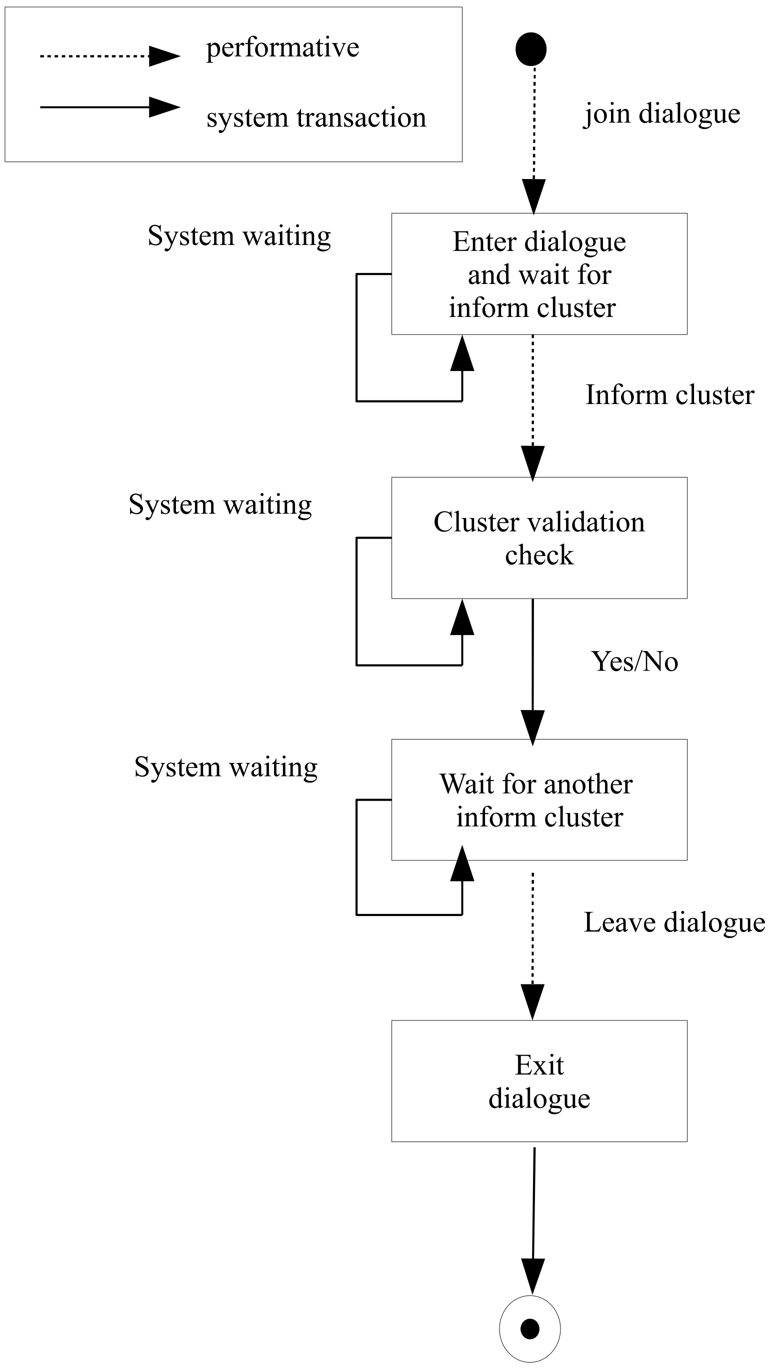


Figure 4.5: State transition diagram for Validation Agent.

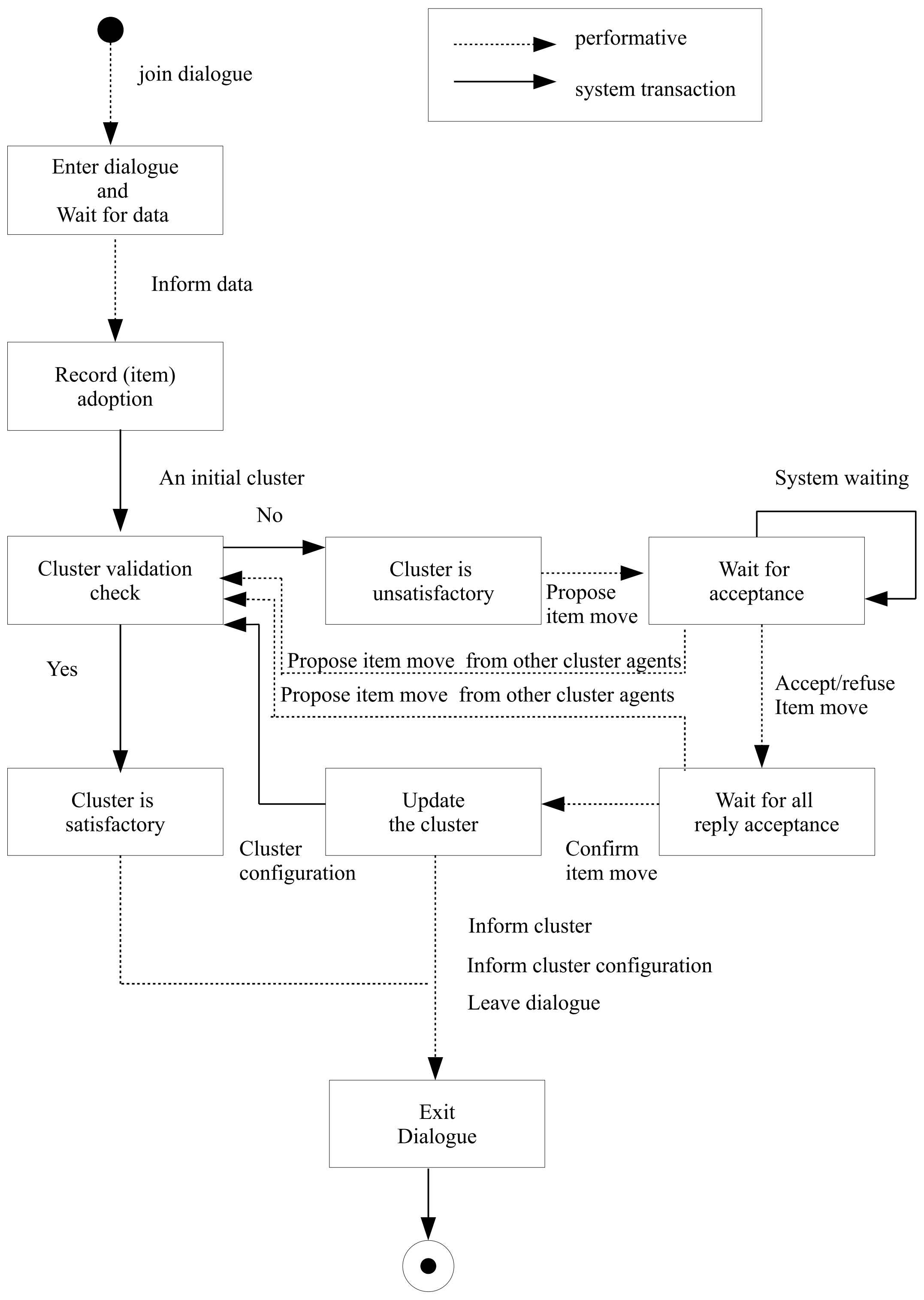


Figure 4.6: State transition diagram for Clustering Agent.

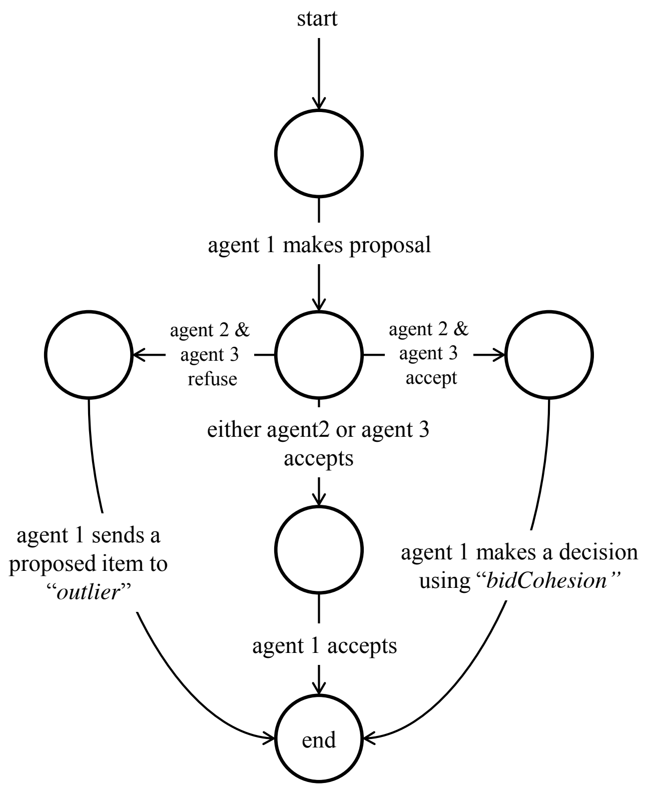

start
agent 1 makes proposal
agent 2 & agent 3 refuse
agent 2 & agent 3 accept
either agent2 or agent 3 accepts
agent 1 sends a proposed item to "*outlier*"
agent 1 makes a decision using "*bidCohesion*"
agent 1 accepts
end

Figure 4.7: A simple alternating offer protocol for negotiation within MABC.

A simple alternating offer protocol for negotiation was adopted with respect to the proposed MABC approach as shown in Figure 4.7. The negotiation participants interact in terms of a many-to-one negotiation: that is, a single agent (agent 1) negotiates with other agents to improve its cluster configuration. In the example given in Figure 4.7 three Clustering Agents are engaged in negotiation: agent 1, agent 2 and agent 3. The negotiation commences when agent 1 makes a proposal to agent 2 and agent 3. Agent 2 and agent 3 receive the *propose item move* performative and determine, from their clusters, whether an item move can improve their cluster configuration or not. The *accept item move* performative is used when the record move can improve the agent's cluster configuration, while the *refuse item move* performative is used for refusing the record, because if the agent merges this record into its cluster the cluster configuration will get worse. There are three possible cases: (i) both receiving agents accept the item move, thus agent 1 determines who will obtain the item move using "bidCohesion", which is the cohesion bid value used for record bidding in the negotiation phase, (ii) only one of agents 2 and 3 accept, thus an agreement is reached amongst agents, or (iii) both receiving agents refuse the item move (in which case the record will be designated to be an outlier).

4.3 Operation

The operation of the proposed extended MADM framework, in the context of MABC, is described in this section from an algorithmic perspective (the previous section described the operation in terms of the communication performatives used). As already noted, the proposed MABC features a two phase mode of operation: an *initial clustering* phase (Phase 1) and a *negotiation* phase (Phase 2). The objective of Phase 1 is to generate an initial cluster configuration in a decentralised manner. For Phase 1, three distinct clustering paradigms are currently supported: (i) K-means, (ii) K-NN and (iii) divisive hierarchical clustering. Recall that the clustering paradigms were revised to support the concept of a collection of agents that cooperates to produce an initial cluster configuration. Note that the initial cluster configuration will be the same as that produced using a “traditional” centralised approach. The objective of Phase 2 is then to refine this initial cluster configuration using an intra-agent negotiation process.

4.3.1 Initial Cluster Configuration founded on the K-means Paradigm

The operation of the initial cluster configuration process with respect to the K-means paradigm is detailed in the algorithm presented in Table 4.5. Referring to Table 4.5 and Figure 4.1, K Clustering Agents are spawned to represent the clusters (line 1). Each Clustering Agent is then allocated a single record, by the identified Data Agent, and this record is used to represent the initial centroid of each cluster (lines 2 to 4). In Table 4.5, to facilitate ease of understanding, a single data source is assumed; the process will not be that different given multiple data sources. The Data Agent(s) will offer up records for “adoption” by Clustering Agents one-by-one. Each Clustering Agent places a “bid” for records (lines 5 to 8). In the implementation used to evaluate the process the $bidDistance$ equates to the “distance” of d_i ($d_i \in D$) from the centroid of the cluster represented by the Clustering Agent in question. The Clustering Agent with the lowest $bidDistance$ “wins” the record. Note that the centroid for the cluster winning the current record is recalculated (line 9). At the end of the process the K Clustering Agents will collectively hold an initial cluster configuration. Note that, in common with standard K-means clustering, as the process proceeds the centroid will tend to “move”, and thus a best cluster configuration may not be arrived at. The “goodness” of this initial configuration will thus be very much influenced by the nature of the first K records selected to define the initial clusters.

4.3.2 Initial Cluster Configuration founded on the K-NN Paradigm

The initial clustering phase founded on the K-NN paradigm, as noted above, commences with a single Clustering Agent (C_i) (see Figure 4.2). The operation of this process is presented, from an algorithmic perspective, in Table 4.6. For ease of understanding

Table 4.5: Initial cluster configuration phase using the K-means paradigm.

```
Phase I: An initial cluster configuration phase using K-means
Input: Dataset ($D = \{d_1, d_2, ..., d_n\}$), the desired number of clusters ($K$)
Output: An initial clustering configuration ($C = \{c_1, c_2, ..., c_K\}$)
1. User Agent spawns $K$ Clustering Agents ($C = \{c_1, c_2, ..., c_K\}$)
2. Each Clustering Agent sends a data request to the indicated Data Agent
3. Data Agent sends first $K$ records ($\{d_1, d_2, \cdots, d_K\}$) to the $K$ Clustering Agents;
                    $d_1$ to $c_1$, $d_2$ to $c_2$, and so on.
4. Each Clustering Agent calculates its cluster centroid
5. $\forall d_i \in D$ ($i = K + 1$ to $n$)
6.     $\forall c_j \in K$ ($j = 1$ to $K$)
7.         $bidDistance = d_i -$ $centroid$ $c_j$
8.         Allocate $d_i$ to $c_j$ so as to minimise $bidDistance$
9.         Recalculate centroid for $c_j$
```

Table 4.6 again assumes a single data source, however the process can easily be extended to operate with multiple data sources (as suggested in Figure 4.2). The Data Agent will put records up for adoption by Clustering Agents. In this case, the winning agent is the agent that holds the "nearest neighbour" to the current record provided that the closest distance is less than a predefined *nearest neighbour threshold* parameter, τ; this is to prevent "outliers" from being associated with their "closest" cluster. If the *bidDistance* is less than the threshold, the record in question is allocated to the "closest" cluster (line 6). If there is no "closest" cluster a new Cluster Agent is spawned (line 7). The record is assigned to the newly created Clustering Agent which will continue to operate in an entirely independent manner to the Clustering Agent that created it.

Note that the chosen value of τ can significantly affect the number of Clustering Agents that are spawned. The author proposes in Chapter 6 a method, in the context of MABC, to identify the most appropriate value for τ where a number experiments were conducted with respect to a number of cluster validation techniques to identify the optimum parameters for clustering algorithms. In Chapter 6 the desired final configuration was generated using a sequence of parameter values to produce a collection of cluster configurations from which the most appropriate was selected. These reported experiments clearly demonstrated that there was no single "best" value for τ. From the experiments it was also clear there were many factors that influence the choice of the best value for τ, such as the nature of the data set size, distribution of values, number of potential clusters and so on; consequently it was not possible to identify specific correlations between τ values and the nature of the data set. However, for evaluation purposes (see Chapter 5) values for τ were selected so as to provide correlations with the "ground truth" number of clusters.

Table 4.6: Initial cluster configuration phase using the K-NN Clustering Paradigm

```
Phase I: An initial cluster configuration phase using K-NN
Input: Dataset ($D = \{d_1, d_2, ..., d_n\}$), threshold $\tau$
Output: An initial clustering configuration ($C = \{c_1, c_2, ..., c_K\}$)
1. User Agent spawns a single Clustering Agent ($c_1$)
2. $K = 1$
3. $\forall d_i \in D$ ($i = 2$ to $n$)
4.   $\forall c_j \in K$ ($j = 1$ to $K$)
5.     $bidDistance$ = nearest neighbour to $d_i$
6.     IF $\exists c_j \in C$ such that $bidDistance < \tau$, allocate $d_i$ to $c_j$ so as to minimise
                 $bidDistance$
7.     ELSE $K = K + 1$, spawn Clustering Agent $c_K$, allocate $d_i$ to $c_K$
```

4.3.3 Initial Cluster Configuration founded on the divisive hierarchical clustering Paradigm

In the case of the divisive hierarchical clustering paradigm, the Divisive Analysis (DIANA) algorithm was adopted to generate the initial cluster configuration. The DIANA algorithm is one of the most well-know divisive hierarchical clustering algorithms [50]. The algorithm as extended for use in the context of MABC is presented in Table 4.7. As noted above, the splitting Clustering Agent will offer up a set of records to be adopted by the newly spawned Cluster Agent. A "bid" system is not used in this case. The process commences with the entire data set loaded into a single Clustering Agent (line 4 and 5). The dividing process repeats until each cluster cohesion value is less than a given cohesion threshold. In each round, Clustering Agents calculate their cluster cohesion values. The Clustering Agent that holds the cluster containing the longest distance between two records is selected to be divided (split); the longest distance between any two of its records represents the largest dissimilarity (line 7 and line 8). To divide the selected cluster, the Cluster Agent looks for its most disparate record which has the largest average dissimilarity to the other records of the selected cluster (line 10). This record is used to initiate a new sub-cluster (line 12). In subsequent steps, the algorithm reassigns (offers up for adoption) records that are closer to the new sub-cluster than the old cluster (line 14 to 24). The result is a division of the selected cluster into two sub-clusters (A and B). In order to assign a record to the new sub-cluster (B), the Clustering Agent determines the difference between the average intra-distance, which is the average distance between the record and the other records in the same cluster, and the average inter-distance which is the average distance that the record is far from records in the other clusters (line 16). If the difference value is more than 0 then the record is closer to the outer records in other clusters than the records in the same cluster, thus the record is assigned to the new sub-cluster B (line 17-23). The Clustering Agent then keeps the largest sub-cluster containing the majority of records and spawns a second

Clustering Agent to which the smaller sub-cluster is allocated. The process continues until all cluster cohesion values are less than the cohesion threshold value.

There is no standard cohesion threshold value, the most appropriate threshold for a particular data set is usually derived experimentally by the user. For the conducted experiments described in Chapter 5, cohesion thresholds were used and calculated to give results according to the expected number of clusters (K).

Note that the algorithm presented in Table 4.7 uses the distance between a record (data object) x_i and a cluster C_k to determine similarity. The distance is defined as [16]:

$$\overline{D}(x_i, C_k) = \begin{cases} \frac{1}{|C_k|-1} \sum_{x_j \in C_k, j \neq i} dist(x_i, x_j) & , x_i \in C_k \\ \frac{1}{|C_k|} \sum_{x_j \in C_k} dist(x_i, x_j) & , x_i \notin C_k \end{cases} \tag{4.1}$$

$|C_k|$ is the number of records in cluster C_k.

4.3.4 Negotiation (Refinement) Phase

The negotiation process is presented in Table 4.8 and illustrated in Figure 4.11. At the end of Phase 1 the extended MABC framework will have produced an initial cluster configuration. Each Clustering Agent must then evaluate its cluster cohesion and calculate the overall cluster separation so as to determine whether its cluster, in its current form, should be improved or not. If the cluster should be improve, the Clustering Agent will engage in the negotiation process (Phase 2).

During the negotiation process individual Clustering Agents identify records that do not fit well with their cluster definition and put them up for "adoption" by other Clustering Agents. At the same time individual Clustering Agents will "bid" for records put up for adoption by other Clustering Agents. It is this negotiation phase that sets the proposed extended MADM framework apart from the basic MADM framework described in Chapter 3. Clustering Agents select records to be put up for adoption according to their cluster cohesion and the overall cluster separation. The overall aim is to minimise the cluster cohesion value and maximise the cluster separation value.

Cluster cohesion (the compactness of a cluster) can be determined, by each Clustering Agent, simply by considering the distance between the members of its cluster. In order to determine the degree of separation (the distinctiveness of each cluster with respect to each other cluster) agents must communicate with one another to obtain cluster centroids. With respect to the framework described in this Chapter, the Within Group Average Distance (WGAD) and the Between Group Average Distance (BGAD) metrics were adopted to determine cluster cohesion and separation respectively, as described in Chapter 2.

There are three main activities in the process: splitting, moving and merging. With reference to Table 4.8 the negotiation phase commences (lines 1 and 2) with each

Table 4.7: Initial cluster configuration phase using the divisive hierarchical clustering paradigm.

Phase I: Initial cluster configuration phase using divisive hierarchical clustering

```
Input: Dataset ($D = \{d_1, d_2, ..., d_n\}$), the cohesion threshold
Output: An initial clustering configuration ($C = \{c_1, c_2, ..., c_K\}$)

1. User Agent spawns a single Clustering Agent ($c_1$)
2. $K = 1$
3. $C = \{c_1\}$
4. $\forall d_i \in D$ ($i = 1$ to $n$)
5.    Allocate $d_i$ to $c_1$
6. DO WHILE there exists cluster cohesion value > cohesion threshold
7.    $\forall c_i \in C$ ($i = 1$ to $K$)
8.       A = the cluster containing two records with the longest distance.
9.    B = $\emptyset$
10.   $x_i$ = the record in A with maximum $\overline{D}(x_i, A)$
11.    A = A$-\{x_i\}$
12.    B = B$\cup\{x_i\}$
13.    split = true
14.    DO WHILE split = true
15.     $\forall x_j \in A$ ($j = 1$ to $|A|$)
16.         $e(j) = \overline{D}(x_j, A) - \overline{D}(x_j, B)$
17.     IF $\exists e(j) > 0$
18.       $x_k$ = the record in A with maximum e(j)
19.       A = A$-\{x_k\}$
20.       B = B$\cup\{x_k\}$
21.       split = true
22.     ELSE
23.       split = false
24.    END DO WHILE
25. $K = K + 1$
26. A Clustering Agent is spawned. ($C = C \cup \{c_K\}$)
27. IF $|A| > |B|$
27. $\forall x_i \in B$ ($i = 1$ to $|B|$)
28.   Allocate $x_i$ to $c_K$
29. ELSE
30. $\forall x_i \in A$ ($i = 1$ to $|A|$)
31.   Allocate $x_i$ to $c_K$
32. $\forall c_i \in C$ ($i = 1$ to $|C|$)
33.   Calculate the cluster cohesion
34. END DO WHILE
```

Clustering Agent, c_i, determining its own cluster cohesion value, $WGAD_i$. The complete set of Clustering Agents then collectively determine an average cohesion value, $AveWGAD$, and an overall BGAD separation value (lines 3 and 4 respectively). Two threshold values were also used, a cohesion threshold ($WGADthold$) and a separation threshold ($BGADthold$), to determine whether the current cluster configuration is satisfactory or not. The threshold values are calculated on line 5 and 6. The factors, p and q, are used to calculate the target WGAD (the cohesion threshold) and the target $BGAD$ value (the separation threshold). A loop is then entered where the process repeatedly attempts to move unwanted records until either the overall cluster configuration cohesion and separation is within the identified thresholds or no more records can be moved. For each Clustering Agent the process first determines whether its local WGAD value, $WGAD_i$, is below the WGAD threshold, if so there is no need to attempt to move any records. Otherwise (line 10) the Clustering Agent will split the cluster into two sub-clusters, C_{major} and C_{minor} using the K-means algorithm (with K set to 2). The C_{major} sub-cluster contains the majority of records while the C_{minor} sub-cluster holds the smaller number of records. Note that this splitting process is entirely independent from the K-means algorithm that may be adopted in Phase 1. Records in C_{minor} are then put up for adoption. So as to limit the communication overhead, Clustering Agents identify themselves as potential "adopters" using individual $BGAD$ values (distance between their cluster centroids). Clustering Agents whose $BGAD$ values are less than the $BGADthold$ threshold value then bid for the records. The Clustering Agents determine whether records should belong to their clusters or not using the distance between their centroids and the individual records put up for adoption, $distance_{jk}$ (lines 15 and 16). The cluster cohesion value ($bidCohesion$) is used for the record bidding. The Clustering Agents consider the records that have been placed up for adoption one-by-one, the $bidCohesion$ value is calculated by assuming that the record is merged to their cluster. Thus, the magnitude of the $bidCohesion$ value depends on the foregoing records. The Clustering Agents are then ranked according to their individual $distance_{jk}$ to determine the "winning" Clustering Agent. Then $bidCohesion$ value is calculated in lines 17 to 20. Each Clustering Agent will bid for the records where $bidCohesion < WGADthold$, otherwise the records are refused. Again the Clustering Agent with the lowest $bidCohesion$ wins the record. In cases where two (or more) Clustering Agents make the same offer, the "winner" is randomly selected. If there are no Clustering Agents willing to bid, or no $bidCohesion$ is received below the threshold, the record is moved to an *outlier cluster* (line 18). This process repeats until either satisfactory cohesion and separation values are reached or no more records can be moved. A "timeout" has been included to avoid extensive processing and potential deadlocks occurring during the process.

Table 4.8: Algorithm for negotiation phase: moving, splitting, and merging

```
Algorithm Phase II: Negotiation
Input: a set of clusters ($C = \{c_1, c_2, ..., c_K\}$)
Output: an improved clustering result

1. $\forall c_i \in C$ ($i = 1$ to $K$)
2.     Calculate $WGAD_i$
3. Calculate overall $AveWGAD$ value
4. Calculate overall $BGAD$ value
5. Global $WGADthold = AveWGAD \times p$
6. Global $BGADthold = BGAD \times q$
7. DO WHILE there exists cluster cohesion value > cohesion threshold
   or cluster separation value < separation threshold
8.     $\forall c_i \in C$ ($i = 1$ to $K$)
9.         IF $WGAD_i > WGADthold$
10.            Split a cluster $c_i$ into two sub-clusters, $c_{major}$ and $c_{minor}$ using $K$-means
                   using $K = 2$
11.             $\forall c_j \in C$ ($j = 1$ to $K$ and $j \neq i$)
13.                Calculate $BGAD_{ij}$
14.                IF $BGAD_{ij} < BGADthold$
15.                    $\forall d \in c_{minor}$ ($k = 1$ to $|c_{minor}|$)
16.                        Calculate $distance_{jk}$
17.                    $\forall d \in c_{minor}$ are listed in ascending order of distance
18.                    $\forall d \in c_{minor}$ ($k = 1$ to $|c_{minor}|$)
19.                        Calculate $WGAD_{jk}$
20.                        $bidCohesion = WGAD_jk$
21.                       IF $bidCohesion < WGADthold$ and $bidCohesion$ is minimum
22.                           add $d$ to cluster $j$
23.                        ELSE Allocate $d$ to "outlier cluster"
24.     IF no records have been reallocated, EXIT
25. END DO WHILE
```

4.4 Multi-Agent Clustering Example

To illustrate the operation of the proposed extended MABC framework this section presents a simple worked example. The example is designed to illustrate the complete functionality of the proposed extended MABC framework. Recall that, with respect to the implementation of the extended framework, in Phase 1, three clustering paradigms: (i) K-means, (ii) K-NN and (iii) divisive hierarchical clustering may be used to generate an initial cluster configuration. The generation of the initial cluster configuration, using each of these example paradigms, is therefore discussed in Subsections 4.4.1, 4.4.2 and 4.4.3 below. The negotiation phase is the same regardless of which paradigm is used, and is considered in Subsection 4.4.4.

4.4.1 Multi-Agent Clustering Example using K-means

With respect to the K-means paradigm, the data set as held by the Data Agent, is presented in Figure 4.8(a). It is assumed that $K = 3$, thus the process commences with the spawning of three Clustering Agents (Figure 4.8(b)), one per cluster. Each Clustering Agent obtains a record from the Data Agent and uses this record to define its initial cluster centroid. The agents then bid for the remaining nine records (Figure 4.8(c)). At the end of the process each clustering agent holds a cluster, collectively the agents hold an *initial cluster configuration* (Figure 4.8(d)).

4.4.2 Multi-Agent Clustering Example using K-NN

The operation of the initial cluster configuration process with respect to the K-NN paradigm is illustrated in Figure 4.9. Recall that the process requires a threshold, τ, to determine the nearest neighbour. The data set given in Figure 4.8(a) is again used. A single Clustering Agent is spawned in the first step (Figure 4.9(a)) and the first record held by the Data Agent is assigned to this Clustering Agent. The Clustering Agent then "bids" for the next record, d_2. The *bidDistance* is defined in terms of the nearest neighbour value connecting record d_1 to d_2. In the example it is assumed that the *bidDistance* is greater than τ, thus a new Clustering Agent is spawned (Figure 4.9(b)). Both Clustering Agents then bid for the record d_3. The example again assumes that both *bidDistances* are greater than τ, and thus a third Clustering Agent is spawned (Figure 4.9(c)). The agents then continue to bid for the remaining eight records (Figure 4.9(d)). In the example no more Clustering Agents are spawned, all the remaining records are assigned to one of the existing Clustering Agents so far. The initial clustering configuration arrived at is then as shown in Figure 4.9(c). Note that for illustrative purposes, it was assumed that a slightly different configuration is arrived at than when using the K-means paradigm (Figure 4.9(d)).

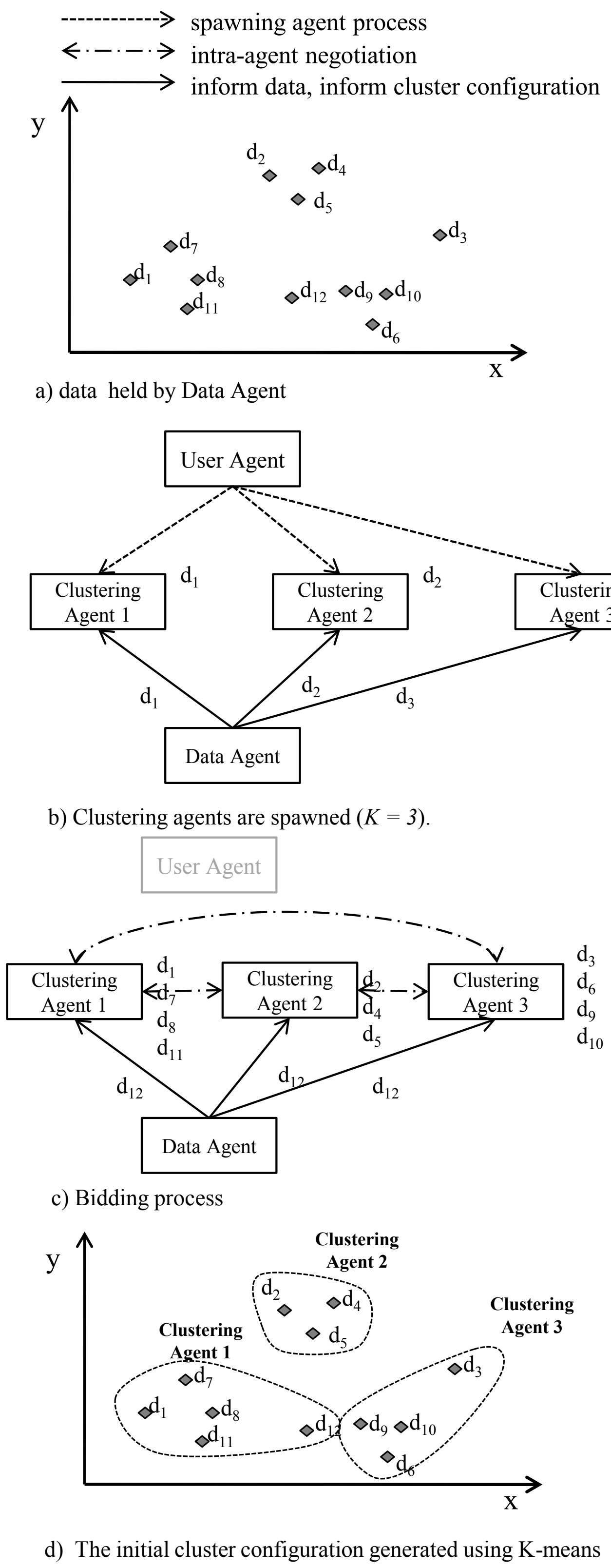

spawning agent process
intra-agent negotiation
inform data, inform cluster configuration
a) data held by Data Agent
User Agent
Clustering Agent 1
Clustering Agent 2
Clustering Agent 3
Data Agent
b) Clustering agents are spawned (K = 3).
c) Bidding process
d) The initial cluster configuration generated using K-means

Figure 4.8: Example of initial cluster configuration generation using K-means.

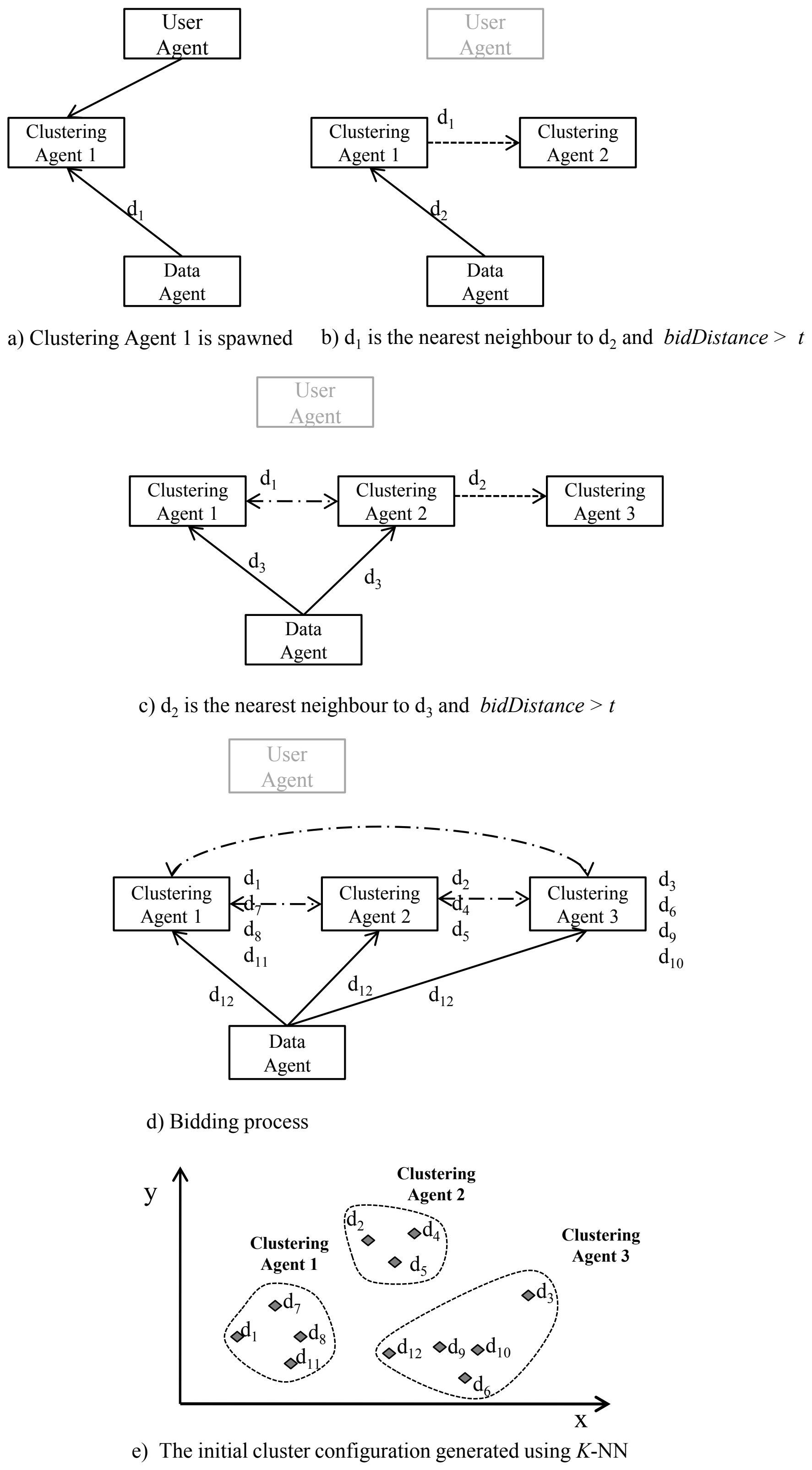

User Agent
Clustering Agent 1
Clustering Agent 2
Clustering Agent 3
Data Agent
d_1
d_2
d_3
d_4
d_5
d_6
d_7
d_8
d_9
d_{10}
d_{11}
d_{12}
a) Clustering Agent 1 is spawned
b) d_1 is the nearest neighbour to d_2 and *bidDistance* > *t*
c) d_2 is the nearest neighbour to d_3 and *bidDistance* > *t*
d) Bidding process
y
x
e) The initial cluster configuration generated using *K*-NN

Figure 4.9: Example of initial cluster configuration generation using K-NN.

4.4.3 Multi-Agent Clustering Example using divisive hierarchical clustering

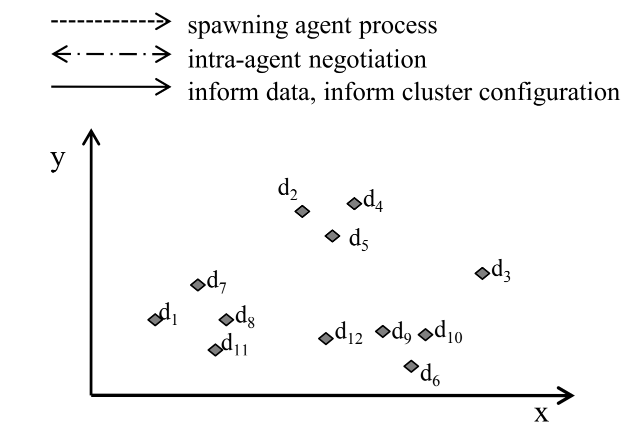
The operation of the initial cluster configuration process with respect to the divisive hierarchical clustering paradigm is illustrated in Figure 4.10. Recall that the process requires a cohesion threshold to determine a cluster configuration (alternatively, as noted previously and as used with respect to the evaluation reported later in this thesis, a K value can be used). If the cluster cohesion of each cluster belonging to each Clustering Agent is less than the cohesion threshold, the splitting process is terminated. The data set, as held by the Data Agent, is presented in Figure 4.10(a). A single Clustering Agent is spawned in the first step, Figure 4.10(b) and the entire data set held by the Data Agent is assigned to this Clustering Agent.

The Clustering Agent determines the longest distance between a pair of records in its cluster and broadcasts this to other Clustering Agents in order to identify the agent whose cluster is to be split. In this step, Figure 4.10(b), there is only Clustering Agent 1, thus Clustering Agent 1 holds the longest distance (the distance between d_1 and d_3). Clustering Agent 1 identifies its most disparate record (d_1 has the maximum average distance from other records in the same cluster) to be the seed for the new sub-cluster. Two temporary sub-clusters are therefore formed $\{d_3, d_4, d_5, d_6, d_9, d_{12}\}$ and $\{d_1, d_2, d_7, d_8, d_{10}, d_{11}\}$. Clustering Agent 1 then spawns a new Clustering Agent and assigns the smaller of the temporary sub-clusters to this new Clustering Agent, keeping the largest sub-set for itself.

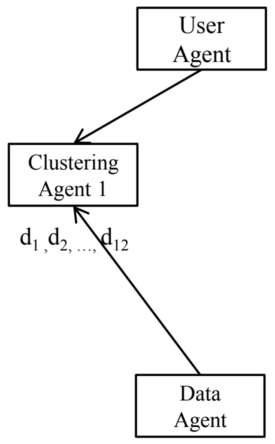
Again, each Clustering Agent determines its cluster cohesion. If the cluster cohesion is not less than the cohesion threshold, the process repeats. Note that each Clustering Agent has to inform its longest distance between any two of its records to the other agents in order to determine who will perform the splitting. In the example, Clustering Agent 2 now holds the longest distance between d_4 and d_6. Thus, a new sub-cluster is created holding d_4. Record d_5 is placed into the new sub-cluster because it is closer to d_4 than other records in the same cluster. No other records are close to d_4, thus a new Clustering Agent is spawned and the new sub-cluster is assigned to the new Clustering Agent, Clustering Agent 3 (Figure 4.10(d)). In Figure 4.10(e), Clustering Agent 1 holds the longest distance and thus record d_2 is selected as its most separate record. There is no record close to record d_2. The agent spawns Cluster Agent 4 and assigns the sub-cluster to Clustering Agent 4. The process is now complete because it is assumed that each cluster cohesion is less than the cluster threshold. The initial cluster configuration is then as in Figure 4.10(f).

4.4.4 Multi-Agent Clustering Example: Negotiation Phase

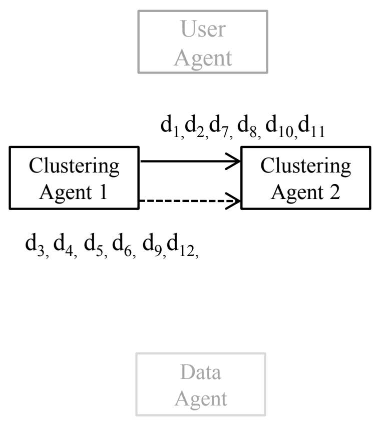
Figure 4.11 illustrates the refinement (negotiation) process using the initial cluster configurations generated using the K-means paradigm (Figure 4.11(a)). The process



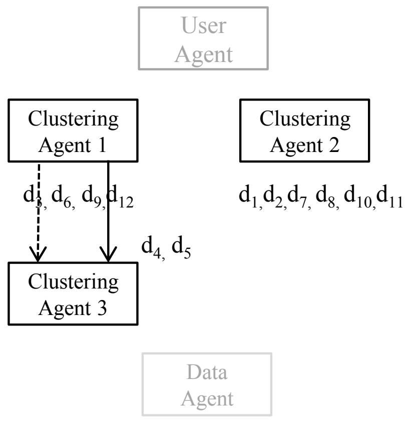
a) data held by the Data Agent.



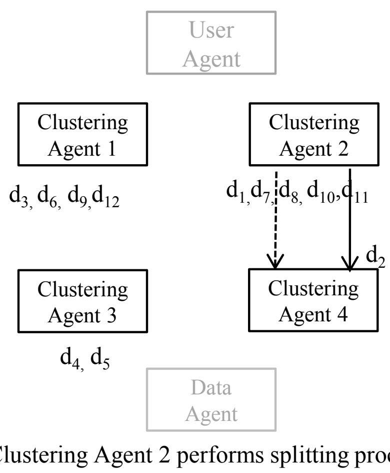
b) Clustering Agent 1 is spawned. The entire data set is loaded to Clustering Agent1.


User Agent
$d_1, d_2, d_7, d_8, d_{10}, d_{11}$
Clustering Agent 1
Clustering Agent 2
$d_3, d_4, d_5, d_6, d_9, d_{12},$
Data Agent

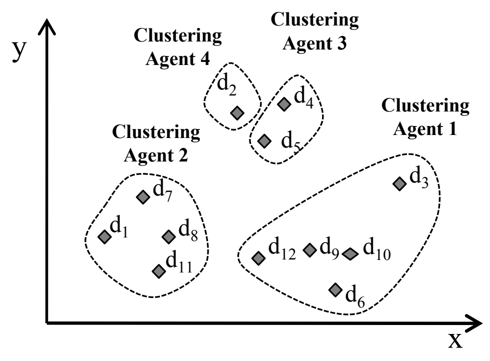
c) The longest distance is between d_1 and d_3. Thus, d_1 is placed to a new sub-cluster. $d_4, d_5, d_6, d_9, d_{12}$ are displaced in the new sub-cluster. Clustering Agent 2 is spawned and the new sub-cluster is assigned to Clustering Agent 2
.



d) Clustering Agent 1 performs splitting process.



e) Clustering Agent 2 performs splitting process. The process is now complete. Each cluster cohesion < cohesion threshold.



f) The initial clusters generated using divisive hierarchical clustering.

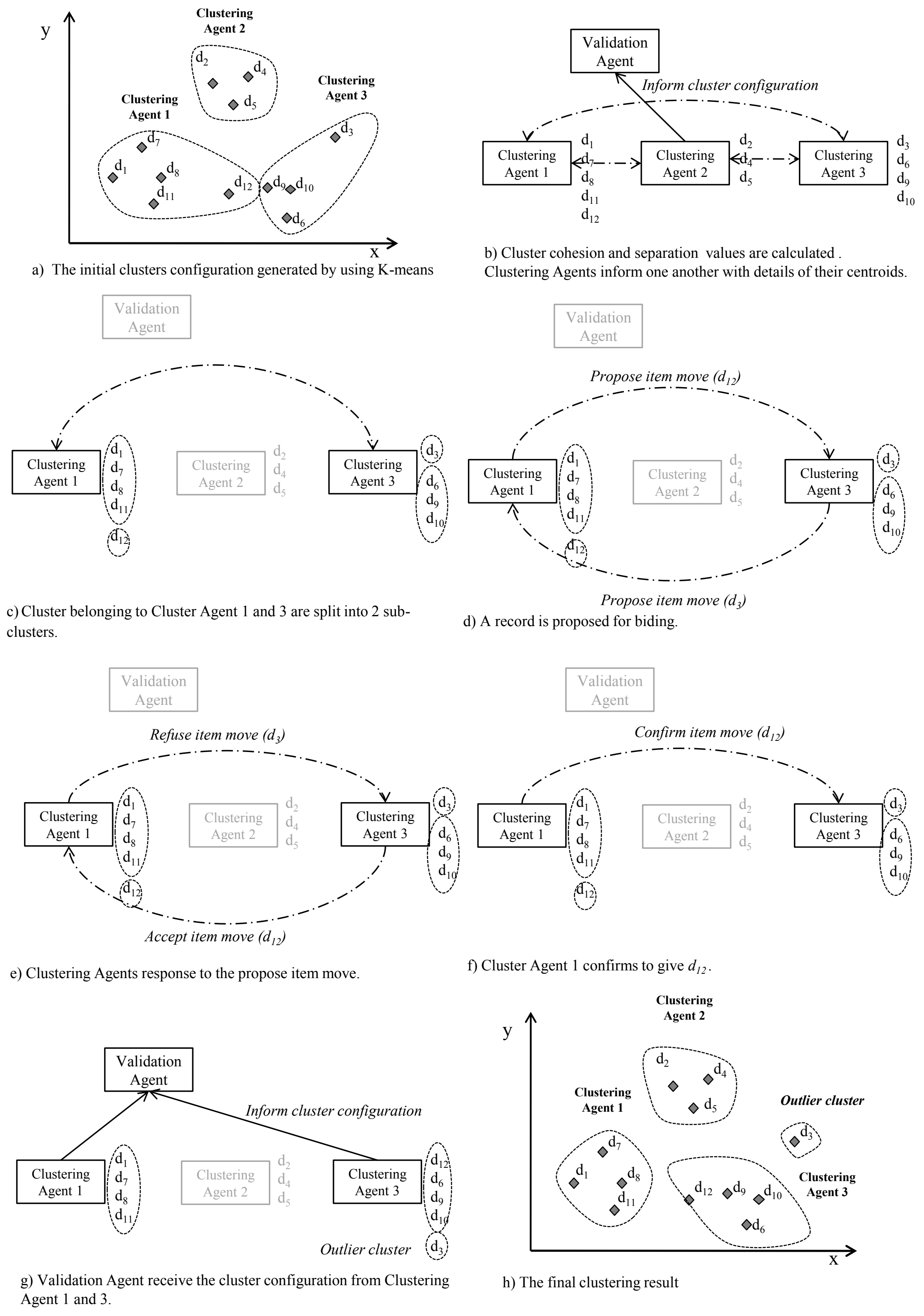
Figure 4.10: Example of initial cluster configuration generation using divisive hierarchical clustering.

commences with each agent determining its cluster cohesion (WGAD) and cluster separation (BGAD) values. Next the overall cohesion threshold (*WGADthold*) and the desired separation threshold (*BGADthold*) are determined and used to assess the initial cluster configuration. In the example, the cluster held by Clustering Agent 2 meets the cohesion and separation threshold requirements and is sent to the Validation Agent (Figure 4.11(b)). Clustering Agent 2 therefore plays no part in the refinement process. On the other hand, in the example, the clusters held by Clustering Agents 1 and 3 do not meet the cohesion and separation requirements. The agents therefore each split their clusters into two sub-clusters: a major sub-cluster and a minor sub-cluster (Figure 4.11(c)). The minor cluster associated with Clustering Agent 1 holds d_{12}, and that associated with Clustering Agent 3 holds d_3. These two records are then put up for adoption (Figure 4.11(d)). In the example the bid for d_3 is rejected, whilst the bid for d_{12} is accepted (Figure 4.11(e)). Record d_{12} is therefore moved from Clustering Agent 1 to Clustering Agent 2 (Figure 4.11(f)); and record d_3 is placed in the outlier class. The process is now complete and Clustering Agents 1 and 3 now also pass their clusters to the Validation Agent (Figure 4.11(g)). Note that the final configuration is then as shown in Figure 4.11(h).

4.5 Conclusions

The extension of the proposed MADM framework introduced in Chapter 3 has been described. The main features of this extension, in the context of MABC, can be summarised as follows:

- The extended framework supports a two phase approach to MABC; an *initial clustering* phase and a *negotiation* phase.
- Clustering Agent functionality was revised so that each Clustering Agent represented a cluster, thus a collection of Clustering Agents can be thought of as representing a clustering algorithm and a set of clusters. The Clustering Agents were designed to cooperate to produce an initial cluster configuration. Consequently, the adopted clustering paradigms founded on the K-means, K-NN and divisive hierarchical clustering were revised in order to produce the desired initial cluster configuration (in a decentralised manner).
- Agents are more independent to perform their tasks, for example agents are free to decide whether they want to take part in the negotiation process or not. In addition, there were no coordinators or facilitators for managing and controlling clustering tasks.
- The negotiation mechanism whereby agents were able to improve on an initial cluster configuration, is the most significant feature. The negotiation process



y
x
Clustering Agent 1
Clustering Agent 2
Clustering Agent 3
d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11} d_{12}
a) The initial clusters configuration generated by using K-means
Validation Agent
Inform cluster configuration
b) Cluster cohesion and separation values are calculated . Clustering Agents inform one another with details of their centroids.
c) Cluster belonging to Cluster Agent 1 and 3 are split into 2 sub-clusters.
Propose item move (d_{12})
Propose item move (d_3)
d) A record is proposed for biding.
Refuse item move (d_3)
Accept item move (d_{12})
e) Clustering Agents response to the propose item move.
Confirm item move (d_{12})
f) Cluster Agent 1 confirms to give d_{12}.
Inform cluster configuration
Outlier cluster
g) Validation Agent receive the cluster configuration from Clustering Agent 1 and 3.
Outlier cluster
h) The final clustering result

Figure 4.11: The refinement process using the initial clusters configuration generated using the K-means paradigm.

allowed agents to negotiate regarding the exchange of records so that the initial cluster configuration can be enhanced.

- A set of communicative performatives, as a part of a negotiation protocol, has been proposed to encourage the negotiation mechanism by allowing agents to interact with one another with the express aim of exchanging records to improve an initial cluster configuration.

The following chapter provides a detailed assessment, founded on a number of experiments, of the negotiation mechanism that has been incorporated into the extended MADM framework with respect to MABC.

Chapter 5

Evaluation and Analysis

The foregoing Chapter 4 described an extension to the initial MADM framework introduced in Chapter 3 (recall that it was conjectured that the initial framework did not feature the full potential of MASs). The significant feature of the extended framework was that it adopted a two phase approach to clustering. Phase one was similar to established centralised clustering approaches. The only difference resided in the fact that Phase 1 was conducted in a decentralised manner by means of agent messaging. Each Clustering Agent represented a cluster and cooperated with other Clustering Agents to produce an initial cluster configuration. Phase two comprised the most significant element of the proposed framework, featuring a negotiation phase whereby agents interacted with one another and swapped unwanted records to improve an initial cluster configuration produced in Phase 1.

This chapter presents the evaluation of the extended MADM framework in context of MABC, and especially the negotiation mechanism that is incorporated into this framework. The reported evaluation is founded on a number of experiments. The experiments were designed to demonstrate that a multi-agent based approach to clustering, and more specifically the extended MADM approach, can significantly improve the quality of cluster configurations. The reported analysis was again conducted using benchmark UCI data sets and the welfare benefits data used as an exemplar application. The welfare benefits monitoring application was selected because the vision of MABC proposed in this thesis typically operates using geographically distributed data sets which, for a variety of reasons, cannot be readily combined into a single "data warehouse". As will become clear, adoption of the extended MABC facilitates decentralised benefits monitoring in that it allows for the identification of anomalies and "outliers" without requiring data to be combined into a single warehouse (thereby infringing data protection requirements).

5.1 MABC Accuracy and Runtime Comparison

To evaluate the proposed approach described in Chapter 4, a number of experiments were conducted and designed to analyse the operation of both phases of the proposed extended MADM framework. Recall that during Phase 1, *an initial cluster configuration*, was determined, whereby a collection of agents cooperate to generate an initial cluster configuration (in a decentralised manner). How this is achieved depends on the nature of the clustering approach adopted. Three clustering approaches (founded on the K-means, K-NN and divisive hierarchical clustering) were considered for the proposed extended MABC framework. Note that the clustering paradigms were revised to support clustering in a decentralise manner. Recall also that during Phase 2, *negotiation*, the collection of agents negotiate to improve the initial cluster configuration produced in Phase 1 by swapping unwanted records that do not fit well with their clusters definition (records are put up for "adoption" by other agents).

As already noted in the introduction to this chapter, a number of experiments were conducted using a selection of data sets taken from the UCI machine learning repository, and a fictional housing benefits data set. Some statistics concerning the UCI data were presented in Table 3.3 in Chapter 3. The benefits data set was used because it represented an exemplar domain for the application of the MABC framework (as discussed in Section 1.1). The benefits data set comprised 8000 benefits records detailing applications for an artificial Retired Persons Housing Allowance (RPHA). RPHA is payable to persons who are of retirement age, whose housing costs exceed one fifth of their available income and whose capital is inadequate to meet their housing costs. Such persons should also be resident in the UK or absent only by virtue of "service to the nation" (for example, the armed forces), and should have an established connection with the UK labour force. The data comprised four classes: (i) *Entitled*, applicant is eligible to full RPHA allowance; (ii) *Entitled with Priority*, applicant is entitled to full RPHA allowance with priority; (iii) *Partial Entitled*, applicant is entitled but with a lower rate of benefits, and (iv) *Not Entitled.* Although fictional, the Housing Benefits data set uses conditions typically found in welfare benefits legislation. It has previously been used with respect to several AI and Law applications. Its usage was first reported in [15] and subsequently in [14, 48, 64, 86, 87]. The data was originally generated using a LISP program and comprised just two clusters: *entitled* and *not entitled.* The work reported in [88] later grouped the data into the four clusters listed above and used with respect to the evaluation reported in this chapter.

The operation of the extended MABC framework was evaluated by comparing its operation with and without the inclusion of Phase 2 (the *negotiation* phase) and with equivalent centralised clustering techniques. With respect to the K-means paradigm, the number of desired clusters, K, must be pre-specified in order that MABC can spawn the appropriate number of Clustering Agents. The number of known clusters was used for this purpose. The K-NN paradigm required a threshold, τ, to determine "nearness", a τ value providing the closest number of clusters to the known number of clusters was therefore selected. This value was determined using the method described in Chapter 6. With respect to divisive hierarchical clustering, a cohesion threshold value was required to control the splitting process (controlling the number of generated clusters). However, there is no standard cohesion threshold value, the most appropriate threshold for a particular data set is usually derived experimentally by the user. For the conducted experiments, cohesion thresholds were used to give results according to the expected number of clusters. For the reported experiments, as described in this Section, a threshold value was adopted equivalent to the cohesion threshold generated as a result of the application of Phase 1 when using the decentralised K-means paradigm. The cohesion value for the cluster that featured the least cohesion (compactness) was selected as the cohesion threshold value to be used for the divisive hierarchical clustering.

In the context of the extended MABC framework, the Within Group Average Distance (WGAD) metric has been adopted to measure intra-cluster cohesion (compactness), and the Between Group Average Distance (BGAD) metric to measure inter-cluster separation. These metrics were described in detail in Chapter 2. To obtain a good cluster configuration WGAD should be minimised and BGAD should be maximised. With respect to the operation of the mechanism supported by the proposed extended framework, the target WGAD value (the cohesion threshold), is the average WGAD across the identified set of clusters, in the initial cluster configuration (Phase 1), multiplied by a factor p ($0 < p < 1.0$). The target BGAD value (the separation threshold), is the BGAD value for the initial cluster configuration multiplied by a factor q ($1.0 < q < 2.0$). Experiments, described in this chapter, indicated that values of $p = 0.8$ and $q = 1.2$ tend to produce the most effective results. The reason for this was that it is desirable to choose a factor p value that makes the cohesion threshold smaller than the existing overall cluster cohesion, and likewise a factor q value that makes the separation threshold larger than the existing overall cluster separation. In each case the cluster configuration accuracy was calculated using the same equation as the accuracy measure given earlier in equation 3.1.

Tables 5.1, 5.2 and 5.3 give the clustering results obtained using the UCI data sets, with and without the application of the negotiation phase (Phase 2), with respect to the three clustering paradigms (K-means, K-NN and divisive hierarchical clustering) supported by the extended MABC framework. Recall that the "goodness" of a cluster configuration using standard K-means clustering is much influenced by the nature of the records which are assigned as the initial cluster centroids. With respect to the K-means algorithm used in the MABC framework presented in Chapter 4, the centroids of clusters was fixed using K selected records as the initial centroids, hence accuracy values obtained by using one agent to perform the clustering task and manage all clusters

Table 5.1: Accuracies obtained using the K-means paradigm with and without application of the negotiation phase (UCI data).

No.	Data Set	Num Classes	% Acc. using one agent	% Acc. Phase I	% Acc. Phase II	WGAD thold.	BGAD thold.
1	**Breast tissue**	6	43	45	48	2694.15	16852.20
2	**Iris**	3	89	89	97	1.41	2.03
3	**Wine**	3	70	68	70	204.95	296.73
4	**Heart**	2	59	55	59	33.87	106.28
5	**Ecoli**	8	76	76	80	0.42	0.54
6	Blood Trans.	2	76	76	76	934.54	3363.15
7	**Pima Indians**	2	66	65	66	65.08	290.60
8	**Ionosphere**	2	71	64	75	5.36	3.68
9	**Breast cancer**	3	85	79	85	283.16	1729.93
10	**Yeast**	10	46	46	53	0.38	0.51

(accuracy recorded in column 4 of Table 5.1), were calculated using the first K records as the initial centroids rather than using K random as presented in Table 3.10 (column 3). In the tables the values in bold indicate where the cluster configurations were improved. From the tables, it can be seen that in the majority of cases the application of agent negotiation, Phase 2, serves to enhance the initial cluster configuration.

Table 5.2: Accuracies obtained using the K-NN paradigm with and without application of the negotiation phase (UCI data).

No.	Data Set	Num Classes	% Acc. using one agent	% Acc. Phase I	% Acc. Phase II	WGAD thold.	BGAD thold.
1	**Breast tissue**	6	30	30	37	2301.66	76376.58
2	**Iris**	4	84	84	92	1.90	1.93
3	**Wine**	3	65	65	66	281.49	282.98
4	**Heart**	2	59	55	61	21.96	62.51
5	**Ecoli**	7	65	64	67	0.38	0.26
6	Blood Trans.	2	76	76	76	1222.54	3090.16
7	Pima Indians	2	66	65	65	133.77	152.08
8	Ionosphere	2	64	64	64	3.16	1.65
9	**Breast cancer**	2	79	63	84	205.51	847.98
10	**Yeast**	10	34	34	39	0.47	0.58

For the Housing Benefits data set (Table 5.4), which closely replicates a real world application, it was found that agent negotiation clearly improved the accuracy of the cluster configuration. The suggested reason why the agent negotiation tends to improve the initial clustering is that there is no generic clustering algorithm appropriate to all data sets. Different clustering algorithms provide different clustering results depending on the characteristics of the input data set and the input parameters used to define the nature of the desired clusters (even the ordering of records can affect the outcome). Thus, the extended MABC approach can enhance the cluster configuration when a negotiation phase is appended to the clustering process. As can be seen from the tables, in some cases there is no accuracy improvement because the selected paradigm was well suited to the data and/or a "best" configuration was immediately arrived at. Consequently, no records could be moved during the negotiation phase. Closer inspection of the data sets (*Blood Transfusion*, *Pima Indian* and *Ionosphere*) indicated that the number of classes was 2. Thus in this case, if one of the Clustering Agents considers that its cluster is "satisfactory", the other Clustering Agent could not be able to move its unwanted records to this cluster. Thus, there was no negotiation process. With respect to the *Breast Cancer* data set, a small reduction in accuracy was recorded with respect to the the divisive hierarchical clustering paradigm (Table 5.3). Further investigation of this case indicated that the mechanism (using K-means) whereby each initial cluster is divided into two sub clusters, C_{major} and C_{minor} (where C_{minor} contains unwanted records), resulted in some records that should have been included in C_{major} being included in C_{minor} and consequently being moved to another cluster. There was no clear explanation for this anomaly. The results demonstrated that a huge improvement was obtained using K-NN when applied to *Breast Cancer* data set; closer inspection of the generated result, consequent to Phase 1 revealed that the clustering result contained one cluster ($C1$) with only one record and the other cluster ($C2$) contained all the remaining records, thus when Phase 2 was applied the records moved from $C2$ to the $C1$ and the accuracy was substantially increased (the number of clusters contained in the *Breast Cancer* dataset was 2).

Table 5.3: Accuracies obtained using the divisive hierarchical clustering paradigm with and without application of the negotiation phase (UCI data).

No.	Data Set	Num Classes	% Acc. using one agent	% Acc. Phase I	% Acc. Phase II	WGAD thold.	BGAD thold.
1	**Breast tissue**	5	32	32	37	2872.18	89472.91
2	**Iris**	3	83	83	87	1.20	3.70
3	Wine	2	69	69	69	311.35	649.15
4	**Heart**	2	55	60	62	93.93	84.16
5	**Ecoli**	7	72	73	76	0.51	0.71
6	Blood Trans.	2	76	76	76	934.54	3363.15
7	Pima Indians	2	66	65	65	155.09	213.11
8	**Ionosphere**	2	74	74	77	3.42	4.02
9	Breast cancer	3	91	91	90	604.44	1338.51
10	**Yeast**	2	40	35	37	0.41	0.29

The work described in [88] was directed at supervised learning, more specifically, classification; while the work described in this thesis is directed at unsupervised learning, more specifically, clustering. Two very different branches of data mining; the first aimed at the construction of classifiers, the second at the segmentation of data. However, it can be argued that the "prototypes" for individual clusters can be used for classification purposes. Whatever the case, for the record, the work described in [88] generated a best classification accuracy of 98.48% while the best accuracy presented in Table 5.4 was 55.73%. This is to be expected as classifier generation, as already noted, is a supervised technique, benefitting from knowledge as to which record belongs to which class; while clustering does not make use of this prior knowledge.

Table 5.4: Accuracies obtained using the Housing Benefits data set with and without application of the negotiation phase.

No.	Data Set	% Acc. Phase I	% Acc. Phase II	WGAD thold.	BGAD thold.
1	*K*-means	41.91	55.73	3.99	3.01
2	*K*-NN	32.25	44.74	2.81	2.93
3	Hierarchical clustering	35.14	38.27	3.02	2.14

Runtime results using the UCI data sets are presented in Table 5.5, 5.6 and 5.7. Comparisons are made between: (i) running *K*-means, *K*-NN and divisive hierarchical clustering in a centralised manner, (ii) using only Phase 1 of the proposed MABC framework and (iii) using both Phase 1 and 2. The results showed that the runtime using a centralised approach is of course less than when using the proposed MABC; this is to be expected as there is no "messaging overhead". However, it is suggested here that the additional runtime did not constitute a prohibitive overhead. The execution time for Phase 1 was dependent on the size of the input data set in terms of number of records and number of attributes (see Table 3.3 in Chapter 3), for example the $Yeast$ data set had significantly more records than the $Iris$ data set. Phase 2 was dependent on the amount of negotiation undertaken (which will also be influenced by the size of the input data set as there is likely to be more negotiation given a large data set than a small data set). The runtime values obtained using the housing benefits data set are given in Table 5.8.

It is also worth noting that the runtime associated with divisive hierarchical clustering (regardless of whether the MABC framework was used or not) is always greater when using *K*-means or *K*-NN. This is because divisive hierarchical clustering has a time complexity of at least $O(n^2)$ where n is the number of data records, while *K*-NN has a complexity of approximately $O(n)$ and *K*-means a complexity of approximately $O(n \times r)$ where r is the number of iterations required to "fix" the cluster configuration.

Table 5.5: Comparison of processing time required by K-means using both a centralised and a MABC approach.

No.	Data Set	Cen. app. time(sec)	Phase I time(sec)	Phase II time(sec)	Total time(sec)
1	Breast tissue	0.016	0.171	0.280	0.451
2	Iris	0.000	0.118	0.187	0.305
3	Wine	0.016	0.346	0.078	0.424
4	Heart	0.015	0.874	0.109	0.983
5	Ecoli	0.015	0.327	0.453	0.780
6	Blood Trans.	0.015	1.732	0.015	1.747
7	Pima Indians	0.016	2.451	0.828	3.279
8	Ionosphere	0.015	3.612	1.939	5.551
9	Breast cancer	0.935	10.003	1.376	11.379
10	Yeast	0.125	1.986	0.673	2.659

Table 5.6: Comparison of processing time required by K-NN using both a centralised and a MABC approach.

No.	Data Set	Cen. app. time(sec)	Phase I time(sec)	Phase II time(sec)	Total time(sec)
1	Breast tissue	0.020	0.250	0.829	1.079
2	Iris	0.032	0.201	0.200	0.401
3	Wine	0.015	0.344	0.065	0.409
4	Heart	0.032	0.141	0.973	1.114
5	Ecoli	0.031	0.344	0.812	1.156
6	Blood Trans.	0.078	0.400	0.791	1.191
7	Pima Indians	0.125	0.301	1.371	1.672
8	Ionosphere	0.020	0.312	4.076	4.388
9	Breast cancer	0.234	0.609	10.132	10.741
10	Yeast	0.437	0.921	48.571	49.492

Table 5.7: Comparison of processing time required by divisive hierarchical clustering using both a centralised and a MABC approach.

No.	Data Set	Cen. app. time(sec)	Phase I time(sec)	Phase II time(sec)	Total time(sec)
1	Breast tissue	0.067	0.201	1.048	1.249
2	Iris	0.207	0.261	0.079	0.340
3	Wine	0.515	0.516	0.204	0.720
4	Heart	1.686	1.733	0.179	2.452
5	Ecoli	2.091	2.295	1.380	3.675
6	Blood Trans.	11.078	11.078	0.327	11.405
7	Pima Indians	23.277	23.277	0.671	23.948
8	Ionosphere	9.408	8.615	0.836	9.451
9	Breast cancer	27.316	27.677	3.340	31.017
10	Yeast	198.138	201.256	4.026	205.282

Table 5.8: Processing times recorded using benefits data set with and without the application of the negotiation phase.

No.	Data Set	Phase I time(sec)	Phase II time(sec)	Total time(sec)
1	K-means	190.717	45.564	236.281
2	K-NN	19.252	1193.080	1212.332
3	Hierarchical clustering	37445.143	2611.480	40056.623

5.2 Accuracy Comparison Using Different Parameter Values

To evaluate the operation of the framework using different clustering parameter values with respect to the number of generated clusters, a number of experiments were conducted. The recorded outcomes from these experiments are presented here. Five data sets (*Iris*, *Wine*, *Heart*, *Breast Tissue* and *Ecoli*) were used with respect to the three adopted clustering paradigms: K-means, K-NN and divisive hierarchical clustering. The reason to select these five data sets was that they served to illustrate the use of a variety different parameter values settings. The *Blood Transion*, *Pima Indian*, *Ionosphere* and *Breast Tissue* data sets features a known number of clusters of 2, as in the case of the *Iris*, *Wine* and *Heart* data sets; hence these four data sets were not used for comparison purpose. The *Breast Tissue* and *Ecoli* data sets were considered using higher parameter value ($5 \leq K \leq 8$ for the *Breast Tissue* data set and $7 \leq K \leq 10$ for the *Ecoli* data set).

The line graphs presented in Figures 5.1 to 5.5 indicate the accuracies obtained by the extended MADM framework using K-means with different numbers of clusters (K values). The horizontal axis represents "the desired number of clusters". The vertical axis represents the "accuracy" (in terms of the overall percentage) of the cluster configuration generated by the extended MABC framework in Phase 1 and Phase 2.

In the case of the *Iris* data set, Figure 5.1, it can be seen that there is an accuracy improvement when the negotiation phase was appended to the clustering process. The exception was that when K was set to 2 there would be no accuracy improvement; this was because the known number of clusters was 3, consequently some records that should have been in a third cluster were included in the two clusters. For the same reason (namely that the *Iris* data set features three cluster) the best record accuracy was when $K = 3$. In the case of the *Wine* data set, which also has a known number of clusters equal to 3, there is a slight accuracy improvement up until $K = 4$, after which accuracy tends to "drop off". In the case of *Heart* data set, there is a slight accuracy improvement up until $K = 4$ after which accuracy again "drops off". Note that the known number of clusters with respect to the *Heart* data set is 2. This means that if the number of clusters is more than 2 the known clusters are distributed over

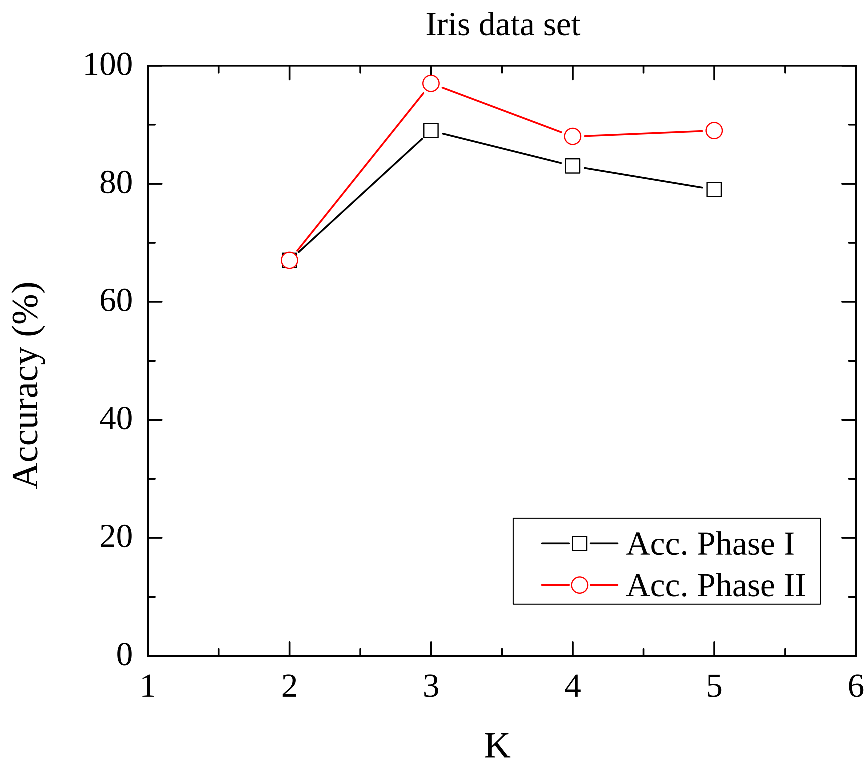


Figure 5.1: MABC using K-means with the different number of clusters, K ($Iris$ data set).

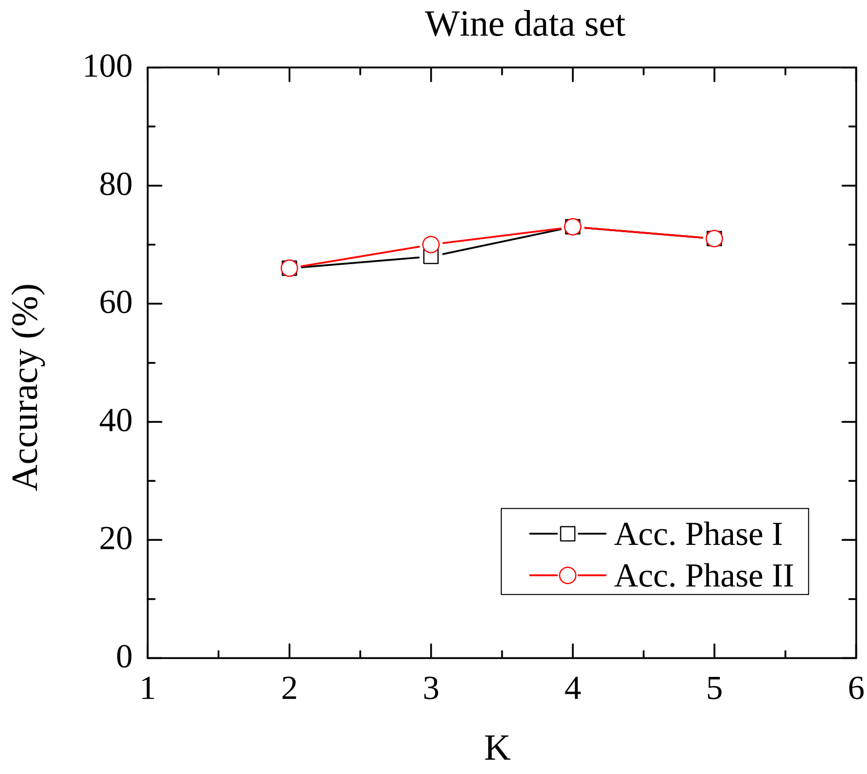


Figure 5.2: MABC using K-means with the different number of clusters, K ($Wine$ data set).

several clusters. In the case of the *Breast Tissue* data set, Figure 5.4, it is interesting to note that the two phases converge at $K = 8$. The reason for this is that the agents are no longer able to identify any records that can be moved to "better" clusters. In the case of the *Ecoli* data set, it is interesting to note that for $K = 9$, although Phase 2 clearly produces an improved accuracy, the recorded accuracy "dips". There is no clear explanation for this other that this, can be attributed to the vagaries of the data, and further evidence of the challenge associated with producing effective clustering mechanism.

The comparison of the framework's operation, in terms of accuracy, using different numbers of clusters and K-means, demonstrates that the selection of the correct value of K is important. However, in some cases, if K is close to or more than the known number of clusters, this can also produce a good quality cluster configuration. Note that the selection of the initial cluster centroids can affect the cluster configuration accuracy because the operation of the K-means algorithm is dependent on the order in which records are presented to it. Recall that the K-means algorithm used in the conducted experiments, in the same manner as in the centralised case, initialised the cluster centroid associated with each Clustering Agent according to the first record presented to it.

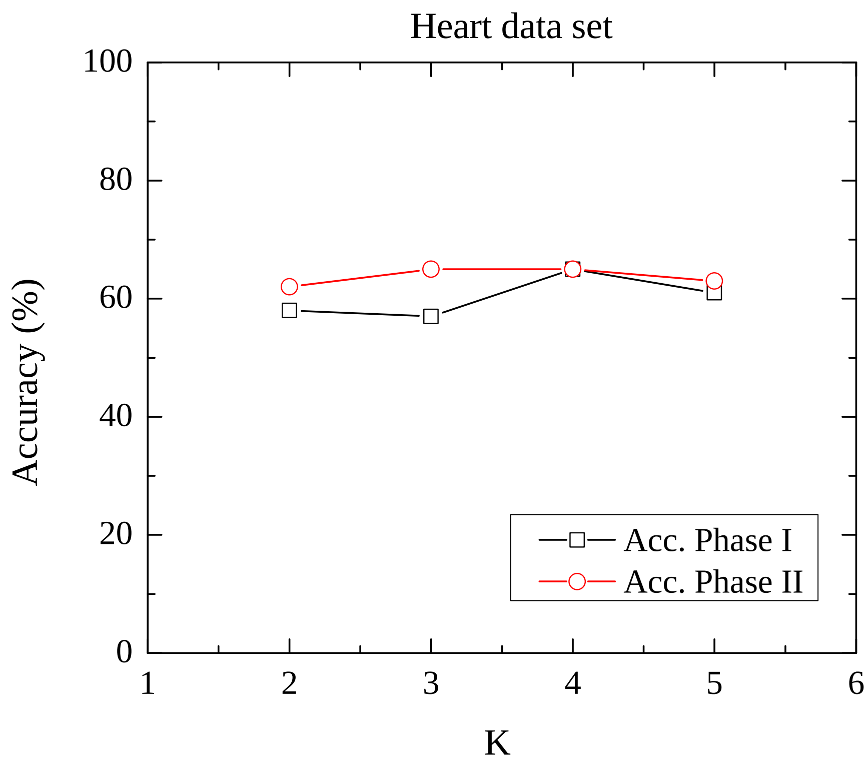


Figure 5.3: MABC using K-means with the different number of clusters, K (*Heart* data set).

Figures 5.6 to 5.10 give a comparison of accuracies obtained using MABC when

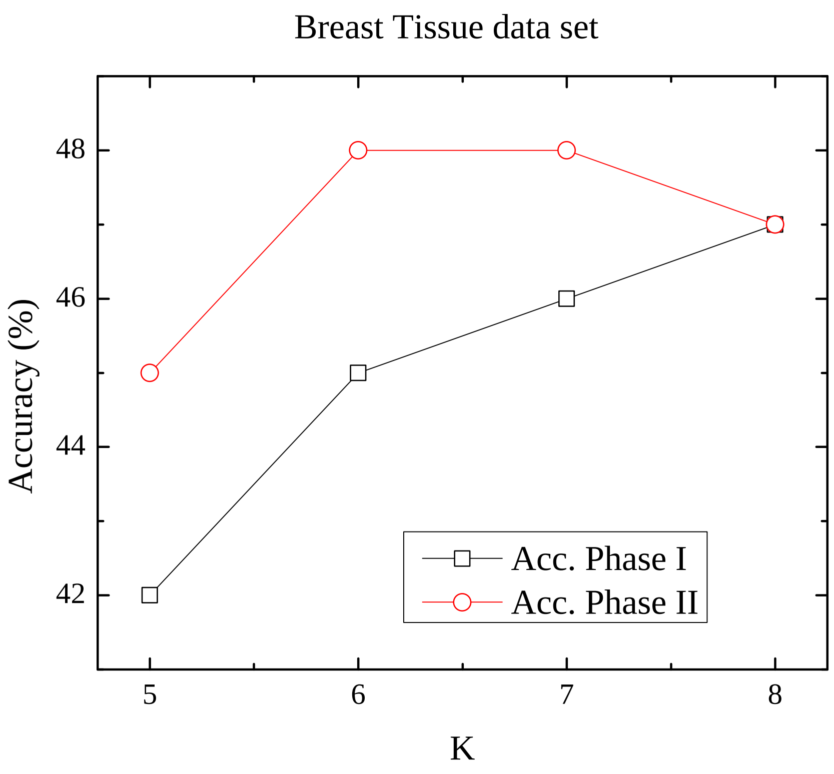


Figure 5.4: MABC using K-means with the different number of clusters, K (*Breast Tissue* data set).

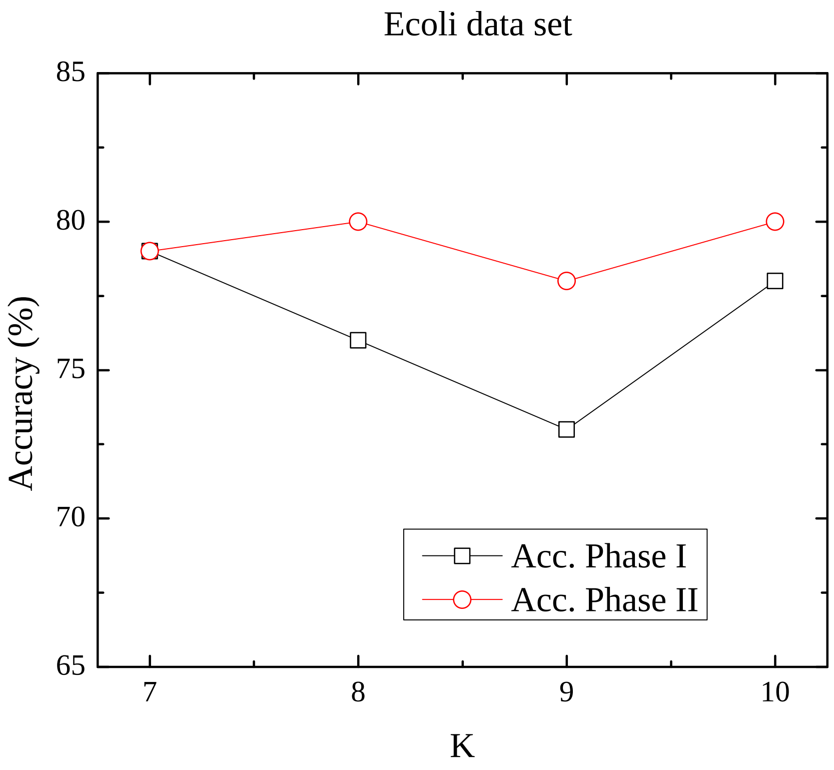


Figure 5.5: MABC using K-means with the different number of clusters, K (*Ecoli* data set).

K-NN was used for Phase 1 but with different τ thresholds. The thresholds used in the conducted experiments were selected with respect to the generated number of clusters. Typically, there is a range of thresholds that can generate the same clustering results when using the K-NN algorithm. However, the thresholds were selected using the threshold generating method described in Chapter 6 to identify the appropriate thresholds. The threshold generating method runs the K-NN algorithm multiple times with the possible threshold values which were generated from the distance between records in a given data set. The first value in the range that can generate the same clustering result was selected. The line graphs are presented in Figures 5.6 to 5.8, Figure 5.9 and Figure 5.10 compare the accuracies of the clustering result using different threshold values, with the number of generated clusters, ranging from: (i) 2 to 5 clusters, (ii) 5 to 8 clusters and (iii) 7 to 10 clusters respectively.

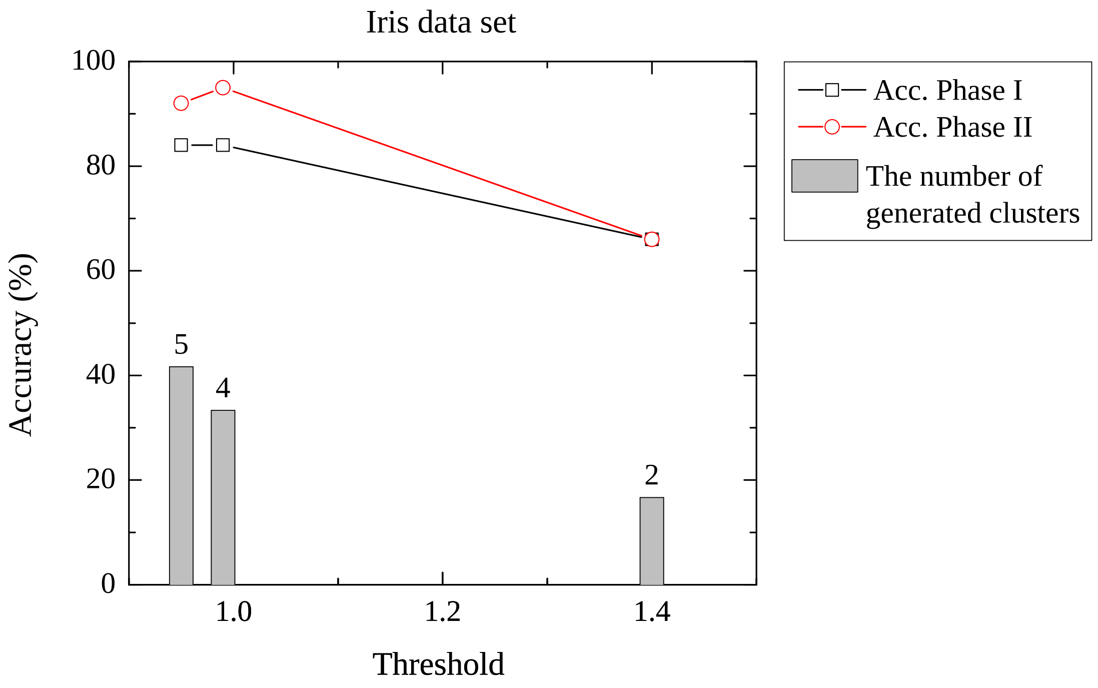


Figure 5.6: MABC using K-NN with different threshold values on *Iris* data set.

The line graphs presented in Figure 5.6 to 5.10 include histograms illustrating the number of clusters, because the chosen value of τ greatly influences the number of generated clusters. With respect to the conducted experiments described in Section 5.1, a τ value which provides the closest number of clusters to the known number of clusters was used. Hence, to compare the obtained accuracies when using different τ values the number of generated clusters was considered. Recall that the K-NN algorithm does not require the number of clusters to generate a cluster configuration but uses the τ parameter for the creation of new clusters.

In the case of the *Iris* data set presented in Figure 5.6, no threshold value could be

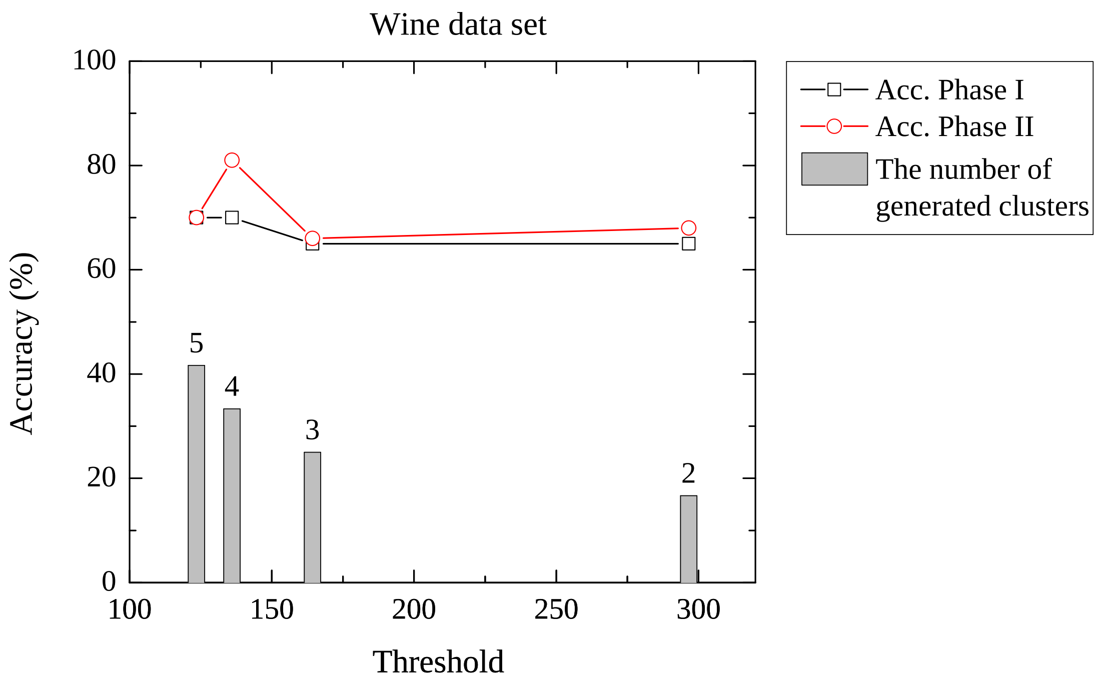


Figure 5.7: MABC using K-NN with different threshold values on *Wine* data set.

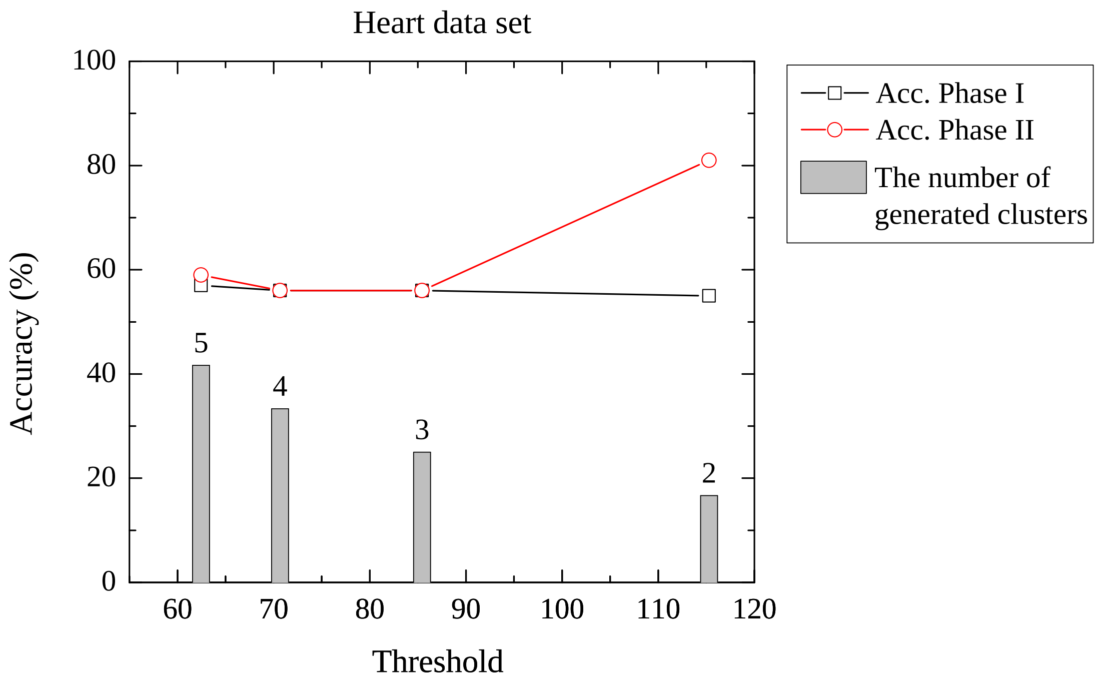


Figure 5.8: MABC using K-NN with different threshold values on *Heart* data set.

identified that resulted in the generation of precisely 3 clusters. Using the range of τ values considered, either 2, 4 or 5 clusters were produced. Best accuracy was obtained when τ resulted in 4 clusters. Recall that the known number of clusters in this case is equal to 3. Inspection of the results indicated that some records were assigned to the wrong clusters. Consequently, the accuracy of the cluster configuration, when the number of generated clusters was 2, was much less than the accuracy of the cluster configuration when the number of generated clusters was 4 or 5. Note that there is no difference in accuracy, when $\tau = 1.4$, between the application of Phase 1 on its own, and Phase 1 and 2 sequentially (for reasons that were presented above).

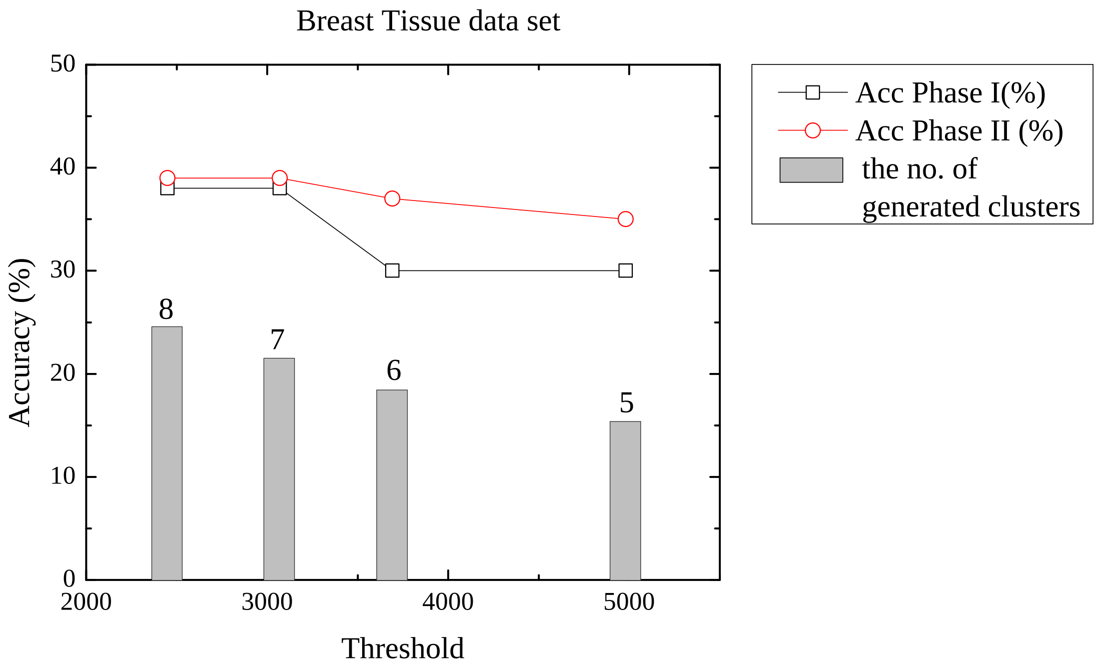


Figure 5.9: MABC using K-NN with different threshold values on *Breast Tissue* data set.

In the case of the *Wine* data set, the known number of clusters is 3. By varying the τ value it was found that configurations resulting in 2, 3, 4 and 5 clusters could be produced. However, using the threshold that generates 3 clusters (the "correct" number of clusters) does not, as might be expected, produce the best accuracy. The best accuracy obtained was recorded when τ was set so that 4 clusters were produced. From the figure it can be seen that, except when the value of τ results in a five cluster configuration, there is a small improvement in accuracy when Phase 2 is applied (and not just Phase 1 on its own).

In the case of the *Heart* data set the range of τ values again produced configurations of between 2, 3, 4 and 5 clusters. In this case there was a significant difference between the accuracy obtained using Phase 1 only and that obtained when Phase 2 was also applied. Note that when τ values of less than 100 are used there is no significant

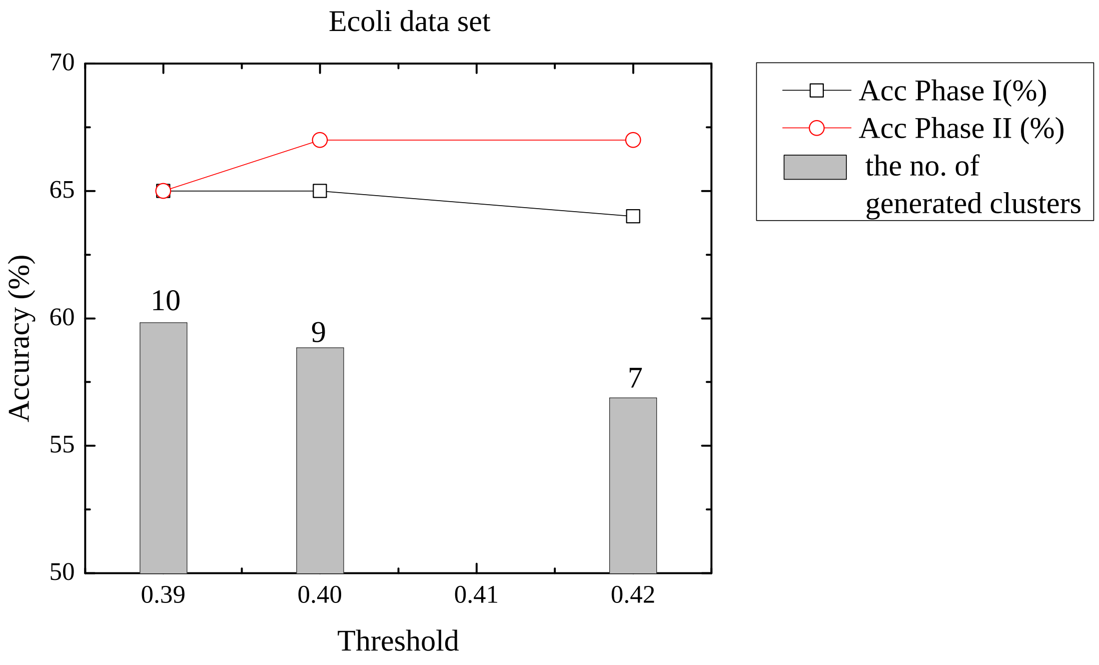


Figure 5.10: MABC using K-NN with different threshold values on *Ecoli* data set.

difference between the application of Phase 1 only and Phase 1 and 2 sequentially.

In the case of the *Breast Tissue* data set, comparing Figure 5.4 and Figure 5.9 the accuracy recorded using K-NN was not as good as when using K-means, although as τ is reduced slightly better accuracy is recorded, but not significantly so. With respect to Figure 5.10 it was not possible to identify a τ threshold that generated eight clusters (the known number of clusters). However from 5.10 it can be seen that the best accuracy lies somewhere between $\tau = 0.40$ (9 clusters) and $\tau = 0.42$ (7 clusters).

Table 5.9: Individual cluster cohesion values for a range of K values from 2 to 5 (*Iris* data set).

K	Acc. (%)	WGAD C_1	WGAD C_2	WGAD C_3	WGAD C_4	WGAD C_5
2	67	0.88	0.95	-	-	-
3	83	0.88	0.59	0.77	-	-
4	83	0.88	0.59	0.49	0.83	-
5	83	0.88	0.59	0.49	0.54	0.85

In the case of divisive hierarchical clustering, as already mentioned above, a cohesion threshold was required for cluster configuration generation purposes; the number of generated clusters depends on the nature of the cohesion threshold used. There are no established guidelines to support the selection of the most appropriate cohesion thresholds. However, the cohesion threshold values used to evaluate the proposed MABC approach were selected so as to be equivalent to the least cohesion within the cluster

Table 5.10: The number of records in each cluster using $Iris$ data set.

K	Acc. (%)	no. of rec. C_1	no. of rec. C_2	no. of rec. C_3	no. of rec. C_4	no. of rec. C_5	Total
2	67	60	90	-	-	-	150
3	83	60	49	41	-	-	150
4	83	60	49	24	17	-	150
5	83	60	49	24	9	8	150

configuration generated as a result of the application of Phase 1 using K-means. To investigate the accuracy associated with the divisive hierarchical clustering paradigm using comparison of different cohesion threshold values for divisive hierarchical clustering, the experiments were set up in a different manner from the conducted experiments in the cases of K-means and K-NN. To determine the cohesion threshold, the centralised divisive hierarchical clustering was run with a different number of clusters in order to investigate each cluster cohesion value and then the cluster cohesion value was determined to identify an appropriate cohesion threshold value with respect to the number of generated clusters.

Tables 5.9, 5.11, 5.13, 5.15 and 5.17 present the accuracies obtained using centralised divisive hierarchical clustering and the cluster cohesion of each cluster using a different number of clusters. From tables 5.9, 5.11 and 5.13 it can be seen that, again regardless of which of the three data sets we are considering, the number of records in C_1 stays constant, the remaining number of records are distributed over the remaining number of available clusters. When $K = 3$, $K = 4$ and $K = 5$ the number of records in cluster C_2 is also constant. Generally speaking, it is only the final K^{th} cluster that is split when moving from K to $K + 1$ clusters. Consequently, the accuracy is not significantly affected when the majority of records are grouped in the same clusters. With respect to Tables 5.15 and 5.17, A WGAD value of 0.00 for a cluster occurs when there is only one record in the cluster.

Table 5.11: Individual cluster cohesion values for a range of K values from 2 to 5 ($Wine$ data set).

K	Acc. (%)	WGAD C_1	WGAD C_2	WGAD C_3	WGAD C_4	WGAD C_5
2	69	103.20	183.64	-	-	-
3	69	103.20	98.31	123.22	-	-
4	69	103.20	98.31	70.17	154.18	-
5	69	103.20	98.31	70.17	68.60	250.65

To determine an appropriate threshold value with respect to the number of generated clusters, in the case of the $Iris$ data set with $K = 2$, the smallest cluster cohesion is 0.95 and hence the threshold used for generating two clusters should be more than

Table 5.12: The number of records in each cluster using *Wine* data set.

K	Acc. (%)	no. of rec. C_1	no. of rec. C_2	no. of rec. C_3	no. of rec. C_4	no. of rec. C_5	Total
2	69	107	71	-	-	-	178
3	69	107	39	32	-	-	178
4	69	107	39	20	12	-	178
5	69	107	39	20	7	5	178

0.95. Similarly, to generate three clusters the threshold should be more than 0.88 and less than 0.95 because if the threshold value is more than 0.95 the algorithm will generate two clusters. However, in the case of $K = 4$ and $K = 5$, the smallest cluster cohesion is 0.88, therefore the range of cohesion threshold values to generate four or five clusters is more than 0.88. There is an overlap value for the cohesion threshold and as a consequence there is the limitation of the divisive hierarchical clustering algorithm using a cohesion threshold as an input parameter. The algorithm will never split C_3 into C_3 and C_4 when $K = 4$ because all cluster cohesion values are less than the cohesion threshold; in this case the cohesion threshold value will be between 0.89 and 0.95. Thus, 3 initial clusters will be generated after Phase 1 and before commencing Phase 2. Typically, for a given clustering task, the chosen threshold defines implicitly the number of clusters that will be formed. A large threshold leads to a smaller number of large clusters and vice-versa. In the case when using the *Wine*, *Heart*, *Breast Tissue* and *Ecoli* data sets presented in Tables 5.11 to 5.18 an appropriate threshold value, with respect to the number of generated clusters, can be determined in a similar manner to that used for the *Iris* data set.

Table 5.13: Individual cluster cohesion values for a range of K values from 2 to 5 (*Heart* data set).

K	Acc. (%)	WGAD C_1	WGAD C_2	WGAD C_3	WGAD C_4	WGAD C_5
2	60	34.32	48.52	-	-	-
3	62	34.32	29.37	69.62	-	-
4	64	34.32	29.37	37.05	99.92	-
5	64	34.32	29.37	37.05	37.81	110.14

Table 5.14: The number of records in each cluster using *Heart* data set.

K	Acc.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	Total
	(%)	C_1	C_2	C_3	C_4	C_5	
2	60	146	124	-	-	-	270
3	62	146	73	51	-	-	270
4	64	146	73	29	22	-	270
5	64	146	73	29	13	9	270

Table 5.15: Individual cluster cohesion values for a range of K values from 5 to 8 (*Breast Tissue* data set).

K	Acc.	WGAD	WGAD	WGAD	WGAD	WGAD	WGAD	WGAD	WGAD
	(%)	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
5	32	1759.17	2430.85	4847.79	0.00	0.00	-	-	-
6	32	1759.17	2430.85	2185.02	0.00	0.00	1474.23	-	-
7	35	1759.17	1366.19	2185.02	0.00	0.00	1474.23	1612.20	-
8	45	443.33	1366.19	2185.02	0.00	0.00	1474.23	1612.20	1650.98

Table 5.16: The number of records in each cluster using *Breast Tissue* data set.

K	Acc.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	no. of rec.	Total
	(%)	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	
5	32	78	20	6	1	1	-	-	-	106
6	32	78	20	3	1	1	3	-	-	106
7	35	78	10	3	1	1	3	10	-	106
8	45	51	10	3	1	1	3	10	27	106

Table 5.17: Individual cluster cohesion values for a range of K values from 7 to 10 (*Ecoli* data set).

K	Acc. (%)	WGAD C_1	WGAD C_2	WGAD C_3	WGAD C_4	WGAD C_5	WGAD C_6	WGAD C_7	WGAD C_8	WGAD C_9	WGAD C_{10}
7	72	0.25	0.22	0.23	0.36	0.25	0.37	0.48	-	-	-
8	72	0.25	0.22	0.23	0.36	0.25	0.37	0.00	0.24	-	-
9	72	0.25	0.22	0.23	0.26	0.25	0.37	0.00	0.24	0.33	-
9	72	0.17	0.22	0.23	0.26	0.25	0.37	0.00	0.24	0.33	0.26

Table 5.18: The number of records in each cluster using *Ecoli* data set.

K	Acc. (%)	no.of rec. C_1	no.of rec. C_2	no.of rec. C_3	no.of rec. C_4	no.of rec. C_5	no.of rec. C_6	no.of rec. C_7	no.of rec. C_8	no.of rec. C_9	no.of rec. C_{10}	Total
7	72	181	93	37	12	7	3	3	-	-	-	336
8	72	181	93	37	12	7	3	1	2	-	-	336
9	72	181	93	37	6	7	3	1	2	6	-	336
10	72	102	93	37	6	7	3	1	2	6	79	336

The limitation of use of the cohesion threshold as the parameter for divisive hierarchical clustering is that the algorithm will generate the clustering result with a small number of clusters because the algorithm finds that the cohesion of all clusters is less than the cohesion threshold. However, an alternative is to use the number of clusters, K, as the input parameter which can be identified. Further investigation is required concerning a mechanism for appropriate clustering parameter identification (see Chapter 6).

5.3 Summary

The proposed extended MADM framework founded on intra-agent negotiation to support enhanced data clustering has been evaluated in this chapter in context of MABC. The framework espouses a two phase approach; *an initial cluster configuration* phase followed by a *negotiation* phase. Three distinct clustering paradigms, K-means, K-NN and divisive hierarchical clustering, were revised and adopted for generation of an initial cluster configuration in a decentralised manner. The reported evaluation was conducted using ten data sets taken from the UCI repository and a welfare benefits data set to illustrate an exemplar application. The operation of the MABC framework was evaluated by comparing its operation with and without the inclusion of the negotiation phase and with equivalent centralised clustering techniques. An extensive evaluation of the framework has been conducted, which clearly demonstrates that in the majority of cases negotiation (Phase 2) facilitates a much improved configuration. This is because there is no single "one size fits all" best clustering algorithm. In a small number of cases no improvement was recorded because, by chance, the initial cluster configuration was the "best" configuration. However, overall the reported evaluation clearly demonstrates the advantage that can be obtained from the extended MADM framework. Of course, negotiation is suited to autonomous processes, such as those featured in MASs, where the processes (agents) are free to decide whether they want to take part in the negotiation process or not.

Chapter 6

Experiments to Automatically Identify Clustering Parameters

A criticism that might be levelled at work described so far in this thesis is that parameters are not automatically identified. With respect to the three clustering paradigms adopted to demonstrate the proposed MABC framework: K-means, K-NN and divisive hierarchical clustering, all require users to provide input parameters; namely the desired number of clusters (K) for K-means, the threshold (τ) used to identify the nearest neighbour for K-NN and the cohesion threshold used to control the number of clusters for divisive hierarchical clustering. In the case of the latter, the number of clusters (K) can be used as an alternative parameter to the cohesion threshold used in divisive hierarchical clustering. Thus with respect to the work presented in this chapter the K parameter has been used in the context of the divisive hierarchical clustering algorithm (see Chapter 5, Section 5.2). Whatever the case, the algorithms require users to provide input parameters. Consequently, the resulting cluster configurations may be sensitive to such parameters. Identification of the most appropriate clustering parameters to produce a cluster configuration that "best" fits the underlying data is difficult when prior knowledge is unavailable, and is normally achieved through a "trial and error" process conducted by the user.

In the experiments described in the foregoing chapters the evaluation was conducted using the best parameter setting in each case, the fundamental idea was that no single technique should be disadvantaged in any way. It would of course be desirable for the system to select optimum parameters, but as it turns out there is no obvious mechanism for doing this. This chapter reports on ideas concerning mechanisms whereby this might be achieved. Two approaches to determine an appropriate clustering parameter are described. The first proposed approach for automatically identifying appropriate clustering parameters adopts a *generate and test* process, whereby a cluster configuration is generated using a particular parameter value and then tested using some clustering metric, as a result the parameter value is adjusted. Finally, the parameter generating the best cluster configuration will be arrived at. The desired cluster configu-

Table 6.1: Algorithm to identify the most appropriate parameter for generating the best cluster configuration using a *heuristic generate and test* method.

```
Algorithm: Generate and Test
Input: a data set $D = \{x_1, ..., x_N\}$, a parameter value ($p$)
Output: the most appropriate parameter
1. $p' = 0$, $e' = 0$
2. $c$ = Cluster configuration generated using $p$
3. $e$ = Evaluation of $c$
4. LOOP
5.     IF ($e > e'$)
6.         $e' = e$, $p' = p$
7.         IF ($p$ cannot be incremented further) Exit with $p'$
8.         ELSE
9.             $p$ = Increment $p$
10.            $c$ = Cluster configuration generated using $p$
11.            $e$ = Evaluation of $c$
12.        END IF ELSE
13.    ELSE Exit with $p'$
14.    END IF ELSE
15.END LOOP
```

ration is defined in terms of some clustering evaluation metric that must be optimised. The second approach is founded on the idea of a running a sequence of clustering runs with different parameter values, and then selecting the best parameter value according to some criterion. Again, some clustering metric must be adopted so as to identify a best cluster configuration. The associated parameter can then be identified. Three clustering metrics were considered for the purpose of identifying a best cluster configuration: the Silhouette Coefficient (Sil. Coef.), the Davies-Bouldin (DB) index and a combination of the WGAD and BGAD metrics (WGAD-BGAD) (see Chapter 2, Section 2.2 for further details). Recall also that the last was derived by the author.

The rest of this chapter is organised as follows. Sections 6.1 and 6.2 describe the two mechanisms considered to identify the most appropriate clustering parameters. Some evaluation and comparison is then presented in Sections 6.3 and 6.4, followed by some conclusions at end of this chapter in Section 6.5.

6.1 Generate and Test Identification of Parameters

As mentioned above, the adopted clustering paradigms in the proposed MABC approach require the user to identify the input clustering parameter, such as K or τ. It is well known that this parameter significantly affects the quality of a cluster configuration. To obtain the best cluster configuration, one mechanism for identifying appropriate parameters is to adopt a *heuristic generate and test* method comprising a "generate and test" loop whereby a cluster configuration is "generated" using a particular parameter value which is then "tested" (evaluated) using some clustering metric (such as those discussed in Chapter 2, Subsection 2.2.2). Subsequently, the parameter value may be adjusted. The process repeats until a best cluster configuration is arrived at. As a result, the best performing associated parameter can be identified. A number of clustering metrics can be applied to identify a best cluster configuration, however with respect to the evaluation presented in Section 6.3 and as already noted, three clustering metrics (Silhouette Coefficient, the DB index and WGAD-BGAD) are adopted. The proposed *generate and test* algorithm is presented in Table 6.1. The main disadvantage associated with generate and test procedures is that they are very much susceptible to local maxima and minima.

6.2 Alternative Approach to Identifying Best Parameters

An alternative approach to identifying the parameter setting associated with a best cluster configuration to that described in Section 6.1 above is to simply generate a sequence of cluster configurations using a range of parameter values and, on completion, select the parameter value that produced the best cluster configuration. The idea is to avoid the issue of local maxima and minima associated with the generate and test method. The range of parameter values will depend on the nature of the adopted clustering paradigm.

In the case of the K-means algorithm and the divisive hierarchical clustering algorithm a sequence of values for K ranging from 2 to $\lceil\sqrt{\frac{N}{2}}\rceil$, where N is the number of records in the given data set, was used. The results were then automatically analysed and the best parameter setting identified. "Best" in this case was again defined using the Silhouette Coefficient, the DB index and WGAD-BGAD. The algorithm is presented in Table 6.2.

In the case of the K-NN algorithm a simple range of τ values was applied. Any τ value that did not generate a number of clusters of between 2 and $\lceil\sqrt{\frac{N}{2}}\rceil$ was ignored. As a result, the best identified τ setting could be determined. The algorithm is presented in Table 6.3.

An issue with K-means is that the initial points (records/objects) used to define the initial centroids of the clusters are randomly selected. The selection of the start points can greatly influence the operation of K-means, to the extent that different best values of K can be produced depending on where in the data set the K-means process starts. Therefore the above approach to identify a most appropriate value for K would be very much dependent on the ordering of the input data. However, the τ parameter selection process can be used to determine the most appropriate value for K.

A general issue with respect to the above approach is that as a result of applying the process for identifying a best value for τ (and by extension K) using K-NN we have

Table 6.2: Algorithm to identify the most appropriate K generating the best cluster configuration using K-means.

```
Algorithm: KmeansIdentifiesK
Input: a data set $D = \{x_1, ..., x_N\}$
Output: $K$

1. For $K = 2$ to $K = \lceil\sqrt{\frac{N}{2}}\rceil$ Do
2.     Do $K$-means clustering using $K$
3.     Evaluate the clustering result (a set of clusters) using a chosen clustering
metric
4.     Keep $K$ and the goodness of cluster configuration value
5. Return $K$ which provides the "best" cluster configuration
```

Table 6.3: Algorithm to identify the most appropriate K and τ, generating the best cluster configuration using K-NN.

```
Algorithm KNNIdentifiesK
Input: a data set $D = \{x_1, ..., x_N\}$
Output: $\tau$, $K$

1. Calculate distance between each record in $D$ and every other record in $D$
and place in $Dist$
2. Sort $Dist$ to remove duplicates and order according to ascending magnitude
3. $Dist$ = {distinct distances in ascending order}
4. $\forall dist_i \in Dist$ ($i = 1$ to $|Dist|$)
5.     $\tau = dist_i$
6.     Do $K$-NN clustering using $\tau$
7.     If the result generated from step (8) is different from the last result
         and the no. of clusters < $\lceil\sqrt{\frac{N}{2}}\rceil$
8.         Evaluate a clustering result using a chosen clustering metric
9.    Keep $\tau$, $K$ and the goodness of cluster configuration value
10. Return $\tau$ which provides the "best" cluster configuration
```

also identified a best cluster configuration. However, it may be the case that, once we have an appropriate parameter setting, a better configuration can be produced using K-means or divisive hierarchical clustering instead.

6.3 Evaluation of the generate and test approach

To evaluate the generate and test method three sets of experiments were conducted. The first used the Silhouette Coefficient (Sil. Coef.) as the reference parameter, the second the Davies-Bouldin (DB) index and the third the WGAD-BGAD. Figures 6.1 to 6.6 show the results obtained using the *Iris* data set taken from the UCI data repository [38] (similar results were obtained using other data sets).

In Figure 6.1 the X-axis gives the K parameter used by K-means. Note that the results obtained for the first and second set of experiments have been combined in the same figures. From the figure it can clearly be seen that there are local maxima and minima; this means that a generate and test procedure, where either the Silhouette Coefficient was maximised or the DB index was minimised, is unlikely to prove successful. In Figure 6.2 the X-axis gives the τ parameter used by the K-NN, while in Figure 6.3 the X-axis gives the K parameter used in the case of divisive hierarchical clustering. Figures 6.2 and 6.3 show similar trend line graphs including local maxima and minima when using K-NN and divisive hierarchical clustering. However, as shown in Figure 6.3 using the DB index as the reference parameter, the best K value that can be identified is 2.

Figure 6.4 shows the result using K-means and WGAD-BGAD as the reference index. Recall that in this case the WGAD-BGAD value should be minimised. From the figure it can be seen that the identified best K value is 2, likewise using the divisive hierarchical clustering (in Figure 6.6). Figure 6.5 shows the result using the K-NN algorithm and the WGAD-BGAD measure as the reference parameter. In this case it is apparent from the figure that a range of "best" τ values were produced, making it difficult to select one. Thus the use of the WGAD-BGAD measure as the reference parameter cannot serve to provide the best clustering parameter in all cases when different clustering algorithms were applied. Therefore it was concluded that a generate and test process would not find the most appropriate parameters in the majority of cases and was thus unsuitable.

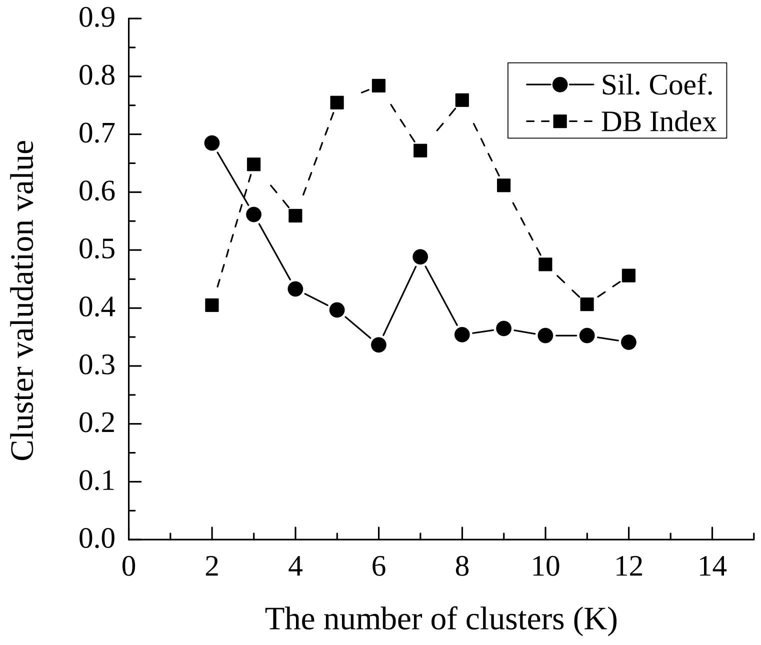


Figure 6.1: K-means clustering K parameter plotted against a range of K values using the $Iris$ data set (Silhouette Coefficient and DB index).

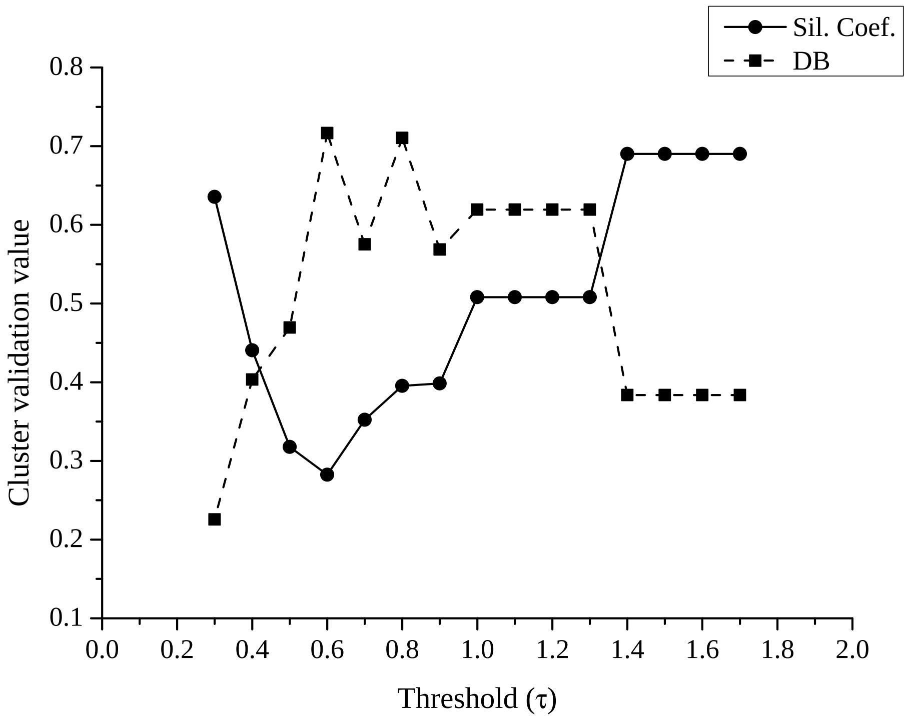


Figure 6.2: K-NN clustering τ parameter plotted against a range of τ values using the *Iris* data set (Silhouette Coefficient and DB index).

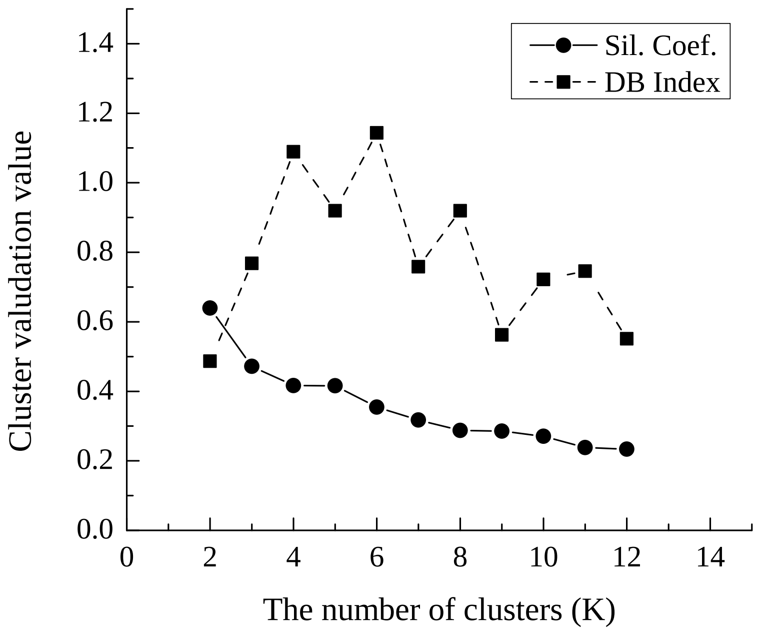


Figure 6.3: Divisive hierarchical clustering K parameter plotted against a range of K values using the *Iris* data set (Silhouette Coefficient and DB index).

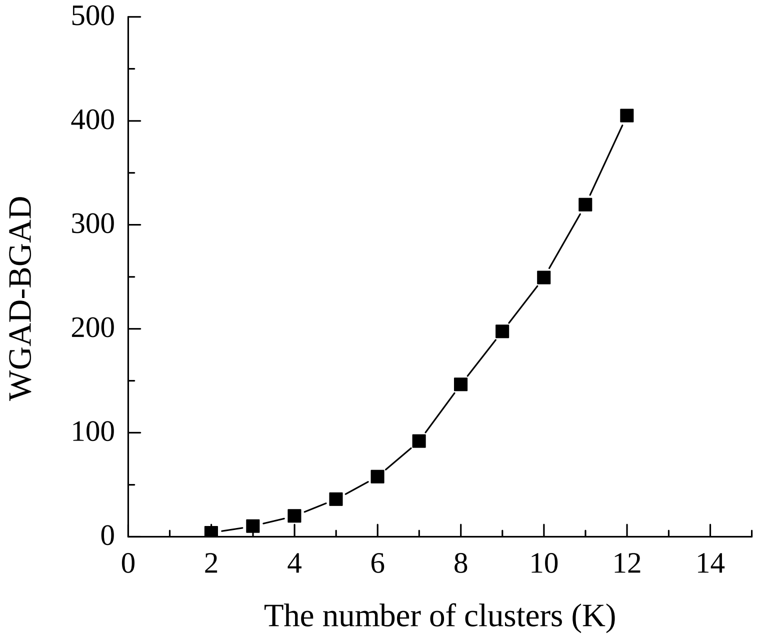


Figure 6.4: K-means clustering K parameter plotted against a range of K values using the $Iris$ data set (WGAD-BGAD).

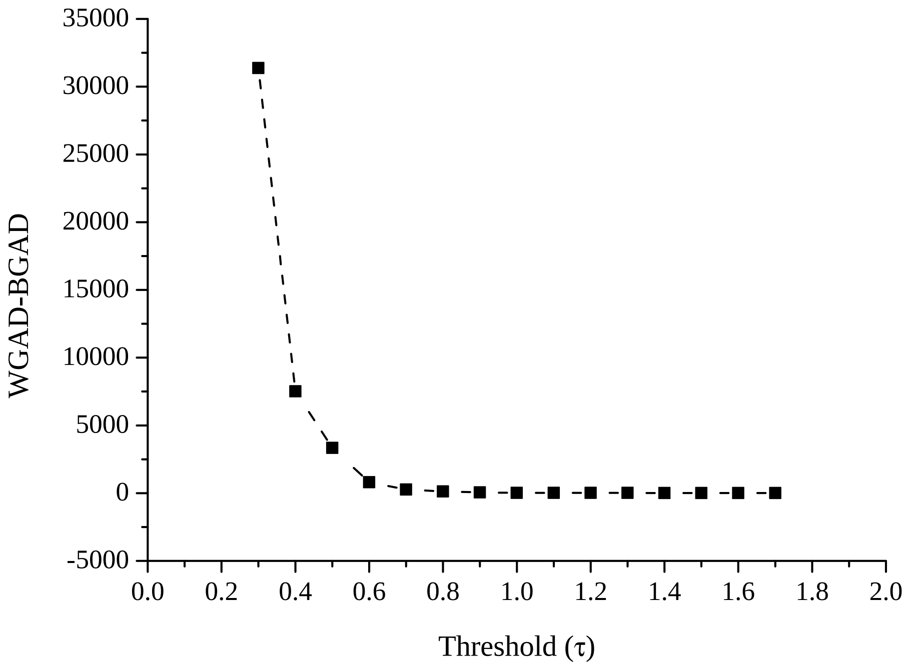


Figure 6.5: K-NN clustering τ parameter plotted against a range of τ values using the $Iris$ data set (WGAD-BGAD) .

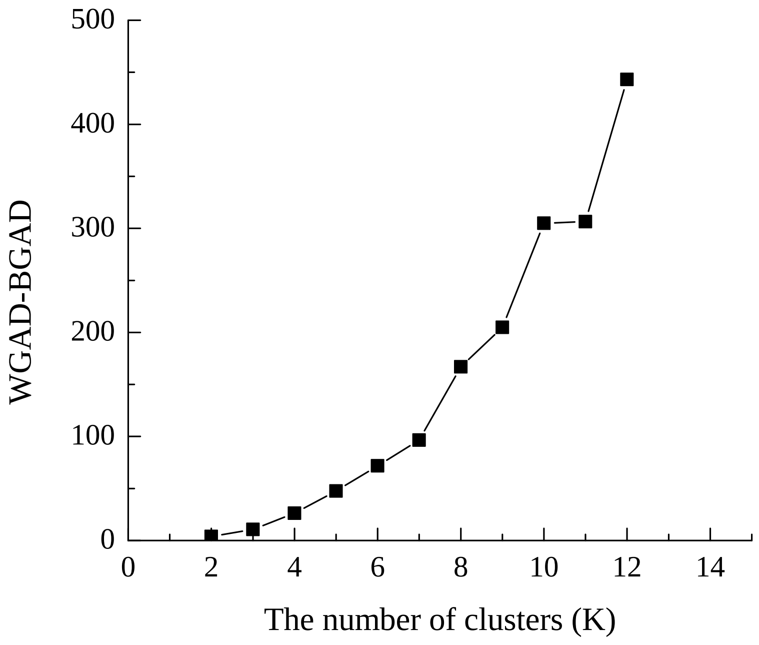


Figure 6.6: Divisive hierarchical clustering K parameter plotted against a range of K values using the *Iris* data set (WGAD-BGAD).

6.4 Evaluation of Alternative Approach

To evaluate the alternative approach used to identifying clustering parameters, a number of experiments were conducted using a collection of data sets taken from the UCI repository. Some statistical information concerning these data sets was presented in Table 3.3. The desired number of clusters, K, was identified by repeated runs of K-means and divisive hierarchical clustering. K-NN was used to identify the most appropriate value for τ. However, it should be recalled that the K-NN approach could equally well be used to identify a best setting for K. Three sets of experiments were conducted using K-means, K-NN and divisive hierarchical clustering. The results are presented in Tables 6.4 to 6.8. A summary is presented in Table 6.9.

Table 6.4 shows the evaluation results produced with respect to the K parameter when applied to the selected UCI data sets. The comparison was conducted by considering the number of clusters (K) associated with the best cluster configuration and the known value for K. In the table the values in bold indicate where the identified number of clusters (K) exactly matches the "known" value of K. Tables 6.5 to 6.7 present a similar comparison using the K-NN algorithm. K in Tables 6.5 to 6.7 is the number of generated clusters using the best τ values. The results produced using divisive hierarchical clustering are presented in Table 6.8.

Table 6.9 summarises and compares the results obtained using the K-means, K-NN and divisive hierarchical clustering algorithms in terms of the identified number of clusters (K) in comparison to the known K value taken from the UCI data repository

Table 6.4: Results using K-means to identify the best K parameter value.

No.	Data Set	Known K	K	Sil. Coef.	K	DB Index	K	WGAD-BGAD
1	Breast Tissue	6	2	0.86	4	0.18	2	61204.25
2	Iris	3	2	0.68	4	0.31	2	3.93
3	Wine	3	2	0.66	12	0.21	2	583.02
4	Heart	2	**2**	0.38	12	0.67	**2**	81.75
5	Ecoli	8	4	0.43	12	0.67	2	0.57
6	Blood Trans.	2	**2**	0.70	15	0.20	**2**	3044.25
7	Pima Indians	2	**2**	0.57	3	0.51	**2**	223.53
8	Breast cancer	2	**2**	0.70	20	0.29	**2**	1331.33
9	Yeast	10	3	0.27	26	0.78	2	0.29
10	Ionosphere	2	5	0.31	18	0.90	**2**	3.07

Table 6.5: Results using Sil. Coef. with K-NN to identify the best τ parameter value.

No.	Data Set	τ	K	Sil. Coef.
1	Breast Tissue	23904.43	2	0.95
2	Iris	1.49	2	0.69
3	Wine	195.07	**3**	0.61
4	Heart	81.63	**2**	0.76
5	Ecoli	0.53	2	0.39
6	Blood Trans.	2000.02	**2**	0.83
7	Pima Indians	193.36	**2**	0.79
8	Breast cancer	992.39	**2**	0.80
9	Yeast	0.60	2	0.74
10	Ionosphere	5.30	**2**	0.41

Table 6.6: Results using DB Index with K-NN to identify the best τ parameter value.

No.	Data Set	τ	K	DB Index
1	Breast Tissue	23904.43	2	0.03
2	Iris	1.49	2	0.38
3	Wine	75.78	14	0.33
4	Heart	81.63	**2**	0.16
5	Ecoli	0.28	10	0.39
6	Blood Trans.	250.27	18	0.12
7	Pima Indians	193.36	**2**	0.23
8	Breast cancer	992.39	**2**	0.13
9	Yeast	0.26	21	0.31
10	Ionosphere	5.30	**2**	0.46

Table 6.7: Results using WGAD-BGAD with K-NN to identify the best τ parameter value.

No.	Data Set	τ	K	WGAD-BGAD
1	Breast Tissue	23904.43	2	168762.21
2	Iris	1.49	2	3.97
3	Wine	206.37	2	810.82
4	Heart	81.63	**2**	316.67
5	Ecoli	0.53	2	0.79
6	Blood Trans.	2000.02	**2**	7404.08
7	Pima Indians	193.36	**2**	682.60
8	Breast cancer	992.39	**2**	3888.91
9	Yeast	0.60	2	0.74
10	Ionosphere	5.30	**2**	5.98

(again values in bold indicate best results). Note that to identify a best parameter setting, the selected clustering metric must be optimised. From Table 6.9 it can be observed that the DB index does not perform well, particularly when used in conjunction with K-means and divisive hierarchical clustering. The use of the DB index produced higher values of K than the known values. The Silhouette Coefficient when used in conjunction with K-NN was found to be the most effective mechanism for finding the most appropriate τ and K parameters. The WGAD-BGAD measure produced $K = 2$ in all cases. Inspection of the nature of the data sets where the identified K matched the known value (see Table 3.3) indicated that the proposed alternative approach operates best where data sets feature a small number of clusters (classes).

Table 6.8: Results using divisive hierarchical clustering to identify the best K parameter value.

No.	Data Set	Known K	K	Sil. Coef.	K	DB Index	K	WGAD-BGAD
1	Breast Tissue	6	**6**	0.69	5	0.26	2	20995.64
2	Iris	3	2	0.64	2	0.49	2	3.77
3	Wine	3	2	0.62	11	0.42	2	540.96
4	Heart	2	**2**	0.33	9	0.70	**2**	70.13
5	Ecoli	8	3	0.36	8	0.87	2	0.49
6	Blood Trans.	2	19	0.63	19	0.25	**2**	2113.82
7	Pima Indians	2	**2**	0.49	7	0.71	**2**	177.59
8	Breast cancer	2	**2**	0.62	9	0.45	**2**	1140.02
9	Yeast	10	8	0.26	17	0.95	2	0.24
10	Ionosphere	2	15	0.32	**2**	1.68	**2**	2.81

Table 6.9: A comparison of identified value of K with the known value for K using K-means, K-NN and divisive hierarchical clustering.

			K-means			K-NN			HC		
No.	Data Set	K UCI	K Sil. Coef.	K DB index	K WGAD-BGD	K Sil. Coef.	K DB index	K WGAD-BGAD	K Sil. Coef.	K DB Index	K WGAD-BGAD
1	Breast Tis.	6	2	4	2	2	2	2	**6**	5	2
2	Iris Plants	3	2	4	2	2	2	2	2	2	2
3	Wine	3	2	12	2	**3**	14	2	2	11	2
4	Heart	2	**2**	12	**2**	**2**	**2**	**2**	**2**	9	**2**
5	Ecoli	8	4	12	2	2	10	2	3	8	2
6	Blood Trans.	2	**2**	15	**2**	**2**	18	**2**	19	19	**2**
7	Pima Indians	2	**2**	3	**2**	**2**	**2**	**2**	**2**	7	**2**
8	Breast Cancer	2	**2**	20	**2**	**2**	**2**	**2**	**2**	9	**2**
9	Yeast	10	3	26	2	3	21	2	8	17	2
10	Ionosphere	2	5	18	**2**	**2**	**2**	**2**	15	**2**	**2**
	Totals		4	0	5	6	4	5	4	1	5

6.5 Conclusion

It is desirable, with respect to the proposed MADM framework described in this thesis, that clustering parameter identification be conducted automatically so as to make agents more autonomous. In this chapter two clustering parameter identification approaches have been described. The approaches can be summarised as follows:

- *A heuristic generate and test method.* Here cluster configurations were iteratively generated, each using a particular clustering parameter value, and then tested using some clustering metric. Consequently, the clustering parameter value was adjusted in an attempt to improve the cluster configuration. However, it was demonstrated that the generate and test process would tend not to find the most appropriate parameters values because of the presence of local maxima and minima.

- *Alternative approach to identify best clustering parameters.* Here a chosen clustering algorithm was run repeatedly with different parameter values and the appropriate parameter value which produced the best cluster configuration chosen. With respect to the clustering paradigms used in the proposed MABC framework, it was observed that the best τ parameter setting could be used to identify a most appropriate value for K (thus avoiding the issue associated with the ordering of records in the context of the K-means algorithm). It was also observed that the difficulties in identifying a cohesion threshold to be used in conjunction with divisive hierarchical clustering can be circumvented by using K instead.

The main finding of the work presented in this chapter are as follows:

- The overall best cluster configuration metric to identify parameter settings is the Silhouette Coefficient.

- K-NN is a much more reliable technique to find the best value for K when using the the Silhouette Coefficient than K-means and divisive hierarchical clustering (and can be used to discover the K value required by K-means).

- A *generate and test* process does not necessarily achieve the desired result.

The findings were applied and incorporated into the evaluation of the proposed MABC framework described in Chapters 3 and 4 to identify the best τ value. For example, the method described in this chapter was used with respect to the evaluation reported in Chapter 5. Overall the work reported in this chapter did not serve to identify satisfactory techniques for automatically identifying clustering parameters in all cases. The two considered techniques were either subject to local maxima and minima or entailed generating a large number of configurations. The quest for an approach to support the automatic identification of best clustering parameter values therefore remains a subject for ongoing research.

Chapter 7

Conclusion and Future Work

This chapter concludes this thesis. The chapter presents an overall summary of the research presented, the main findings, the research contributions and possible future work. The overall summary of the research is presented in Section 7.1. The main findings of the research and the contribution made are then discussed in Sections 7.2 and 7.3 respectively. Section 7.4 presents some considerations for future work.

7.1 Summary

Multi-Agent Data Mining (MADM) seeks to harness the general advantages offered by Multi-Agent Systems (MASs) with respect to the domain of data mining. The research described in this thesis was directed at Multi-Agent Based Clustering (MABC), thus MADM to support clustering. The proposed MABC approach makes use of the advantages offered by MAS, such as decentralised control, agent autonomy, agent social capabilities, communication, coordination, cooperation and negotiation, to obtain a best cluster configuration.

To obtain a best cluster configuration using the full power of MAS to support clustering, two approaches were proposed in this thesis. The first approach was a multi-agent based approach to clustering using a generic MADM framework whereby a collection of agents with different capabilities were allowed to collaborate to produce a "best" set of clusters. Five principle categories of agent were identified: (i) User Agents, (ii) Task Agents, (iii) Data Agents, (iv) Data Mining Agents and (v) Validation Agents. The initial proposed framework allowed a number of Clustering Agents to perform a clustering task using different clustering algorithms applied to the same data set. Hence, a number of clustering results were produced and a best cluster configuration chosen by the Validation Agent according to a number of clustering metrics, namely the F-measure, Within Group Average Distance (WGAD) and Between Group Average Distance (BGAD). A number of conducted experiments were used to evaluate the performance of the proposed MADM framework when applied to MABC using benchmark data sets. As a result, the experiments showed that the MADM framework could successfully be used to find a best cluster configuration. Thus, the advantage offered by this first framework is that it allows for the identification of a best cluster configuration; it is generally acknowledged that there is no best clustering algorithm to fit to every data set. With respect to the purpose of the research described in the thesis, a criticism of the proposed MADM framework was that it did not incorporate some of the most significant features typically associated with MAS such as decentralised control and intra-agent negotiation.

Hence, an extension of the proposed MABC framework was proposed. The most significant feature of the proposed extended MABC framework was that it allowed a collection of agents to negotiate so as to improve an initial cluster configuration. The revisions were designed to address the criticisms directed at the initially proposed framework. In the revised framework there was no "coordinator agent" (Task Agent) who was responsible for controlling and managing clustering tasks. Clustering Agent capabilities were extended so as to enable them to cooperate with each other when producing a clustering result. The extended framework supported a two phase approach to clustering. Phase one was similar to established centralised clustering approaches (except that it was conducted in a decentralised manner). Phase two comprised a negotiation phase where agents "swap" unwanted records. A set of communication performatives was proposed as a part of a negotiation protocol in order to facilitate intra-agent negotiation. An extensive evaluation of the framework was conducted using: (i) benchmark UCI data sets and (ii) a welfare benefits data set that provided an exemplar application. The evaluation results clearly demonstrated that, in the majority of cases, negotiation facilitates the generation of a much improved cluster configuration. It was thus concluded that the extended MADM framework could successfully be used to find a best cluster configuration using a process of cooperation and negotiation. It was thus argued that the proposed extended MABC framework, that featured negotiation, harnesses the true full potential of MASs rather than simply using the concept of a MAS to achieve a distributed clustering.

7.2 Main Findings

The aim of this thesis, as stated in the introductory chapter, was to investigate the use of MAS technology with respect to clustering, broadly so as to establish some of the advantages that MAS can offer clustering. More specifically, the research was designed to address the following research question:

> "How do we obtain a best cluster configuration using the full power of MAS to support unsupervised learning and specifically clustering?"

The research question had a number of research issues associated with it. These were considered throughout the preceding chapters. These research issues are revisited in this section. The research issues, and how the research attempted to address each issue, may be summarised as follows:

1. **How can a MAS identify the most appropriate number of clusters with respect to a given data set?** Automatically determining the number of clusters (K) has been one of the most difficult problems within the field of data clustering. As described in Chapter 6, two approaches were proposed to automatically identify clustering parameters and consequently identify the "best" number of clusters (K). The first approach was founded on a *heuristic generate and test* method whereby a cluster configuration was generated using a particular parameter value which was then evaluated using an adopted clustering metric as a result of which the parameter was adjusted. The process repeats until a best cluster configuration was arrived at. However, the evaluation of the method presented in Chapter 6 Section 6.3 illustrated that the method did not prove successful in finding the most appropriate parameter. Consequently, a second alternative approach was proposed whereby a number of clustering algorithm runs were conducted using a range of different parameter values and the best performing parameter value selected according to some criterion describing the "goodness" of the cluster configuration. However, as discussed in Chapter 6, these approaches did not serve to find the most appropriate parameter value in all cases, and further investigation is thus required.

2. **What is the nature of a MAS framework to support data mining and clustering in particular?** The work presented in this thesis clearly demonstrates the nature of MAS frameworks to support MABC. The design of the proposed frameworks encompassed a number of considerations. The first is the structure of the MAS environment comprising a collection of agents that collaborate to perform some data mining task with some degree of autonomy. As described in Chapter 3, five basic types of agents were initially identified: User Agents, Task Agents, Data Agents, Data Mining Agents (Clustering Agents) and Validation Agents. However, as discussed in Section 3.4 the role of the Task Agent was reconsidered later in this thesis where it was concluded that a coordinator or facilitator agent was not essential for a generic approach to multi-agent based clustering. The second consideration was the agent interaction capabilities required to support MABC. As presented with respect to the basic MADM framework in Chapter 3, individual agents in a MADM were equipped with different data mining algorithms which produced separate results from which the best result could be selected. The data mining task was achieved by using coordination and communication mechanisms. The proposed agent cooperation and negotiation capabilities allowed agents to be more independent and provide for a much

greater degree of agent interaction. The proposed extended MADM framework presented in Chapter 4 whereby a collection of agents were allowed to cooperate to generate an initial cluster configuration, and subsequently improve on this initial configuration, using intra-agent negotiation, demonstrated that agent negotiation can be used effectively to enhance clustering.

3. **How do a group of agents know when they have generated a "best" cluster configuration?** As described with respect to the proposed MABC frameworks presented in Chapters 3 and 4 a "best" cluster configuration can be identified using a cluster validity agent that has recourse to a set of clustering metrics. The basic MABC framework presented in Chapter 3 used Validation Agents which were equipped with different clustering metrics to identify a "best" cluster configuration because the framework allowed individual Clustering Agents to generate different clustering results. However, with respect to the extended MABC framework described in Chapter 4, each Clustering Agent was able to evaluate its own cluster configuration in terms of cluster cohesion and cluster separation and make a decision as to whether to engage in the negotiation process to improve its cluster configuration or not. Individual Clustering Agents can "know" that their clusters are satisfactory within the context of an overall "best" cluster configuration by determining whether the overall cluster configuration has met the desired cluster cohesion and separation thresholds. The author also experimented with devising an alternative best cluster configuration metric, experiments that resulted in the proposed WGAD-BGAD measure.

4. **What specific MAS mechanism can usefully be adopted to support MADM and MABC in particular?** It is well established that there is no clustering algorithm that is suited to all type of data and the selection of appropriate clustering parameters greatly influences clustering results. Thus the capability incorporated into the initial framework whereby a number of agents could derive a number of different cluster configurations was seen as a significant way in which MAS technology could be used to support MABC.

5. **What are an most appropriate cooperation and negotiation protocols for MABC?** As described in Chapter 4, agents in the proposed extended MADM framework were able to communicate explicitly about their associated clustering activities. Cooperation and negotiation protocols (sets of rules that govern agent interaction) were presented in terms of a set of performatives and agent state transition diagrams. A set of performatives was defined and implemented in order to enable the agents to engage in discussions about the suitability of moving "unwanted" records between clusters. The agents used the performatives as part of a protocol that governs the exchange of information between the agents. The

performatives were described in terms of preconditions and postconditions.

7.3 Contributions

The main contributions of the research work were initially presented in Chapter 1. For completeness these are reiterated here:

- A generic approach to multi-agent based clustering to identify a best set of clusters using a collection of "clustering agents".
- The design of a proposed MABC framework using a generic MADM environment to clustering.
- The use of clustering metrics to evaluate the quality of cluster configurations and hence identifying the most appropriate.
- A comparison of two measures, Within Group Average Distance(WGAD) and Between Group Average Distance(BGAD), which are based on cluster cohesion and cluster separation measures respectively, to identify the the most appropriate cluster configuration for a given clustering problem. The comparison showed that WGAD can be argued as the most appropriate mechanism for identifying best cluster configuration.
- An attempt at establishing mechanisms to determine the most appropriate parameters for a given clustering algorithm using clustering metrics.
- A versatile multi-agent based clustering framework.
- A set of communication performatives specifically to support negotiation within multi-agent based clustering.
- A negotiation protocol whereby agents can interact with one another with the express aim of exchanging records to improve an initial cluster configuration.

7.4 Future Work

The research described in this thesis has demonstrated that MABC can usefully be employed to cluster data, in many cases outperforming centralised approaches. However, the reported research has also served to raise a number of promising directions for future research.

- **Automated identification of the most appropriate clustering parameters**. A criticism that might be levelled at work to date described in this thesis is that parameters are not automatically identified. The parameter identification techniques investigated and reported in this thesis did not serve to find the most appropriate clustering parameters in all cases. Further investigation and experimentation into automatic identification of the most appropriate clustering parameter is thus strongly recommended.

- **Unwanted record identification**. The reported evaluation highlighted an anomalous case where a slight reduction in accuracy was recorded. As discussed in Chapter 5, it was conjectured that the mechanism to identify unwanted records using K-means resulted in some records that should have been included in the same cluster but were actually identified as unwanted records and consequently moved to another cluster. Although there was no clear explanation for this anomaly it would be interesting to investigate alternative mechanisms to identify unwanted records for record adoption.

- **Application demonstration for the MABC framework**. Evaluation of the framework clearly demonstrated that, in the majority of cases, the negotiation phase serves to produce a better cluster configuration (in terms of cohesion and separation) than that produced using a simple centralised approach. The proposed MABC framework was applied to a fictional housing benefits application using a benefits data set; the intention being to demonstrate a genuine application of the proposed MABC framework. It would be interesting to demonstrate the wider applicability of the proposed framework, and the potential benefits that can be gained, in the context of other "real" domains, such as cross border policing and academic adjudication. Such applications operate using distributed data sets which, for a variety of reasons, cannot be readily combine into a single data warehouse because of practical and/or security reasons.

Appendix A

DBSCAN Algorithm

As the proposed initial MADM framework, described in Chapter 3, comprises Data Mining Agents who are responsible for performing data mining tasks. The Data Mining Agents are equipped with data mining algorithms. With respect to clustering tasks, K-means, K-NN and Divisive hierarchical clustering were used for identifying a best cluster configuration. DBSCAN (Density-Based Spacial Clustering of Application with Noise) was used in [22, 23] and was incorporated into a Data Mining Agent to evaluate the proposed initial MADM framework. However, the conducted experiment using DBSCAN is included here because the appropriate parameters, $minPts$ and ϵ, are difficult to determine. The most appropriate parameters were identified by using a "trial and error" process conducted by users. With respect to K-means and K-NN, the most appropriate K and τ were derived using a "generate and test" process as described in Chapter 6.

Table A.1: DBSCAN Algorithm [34]

DBSCAN algorithm:

```
Input: a data set (D={$t_1, t_2, ... t_n$})
        a minimum size ($minPts$) and a density threshold ($\epsilon$)
Output: a set of clusters (C={$c_1, c_2, ... c_k$})

1. $k = 0$;
2. FOR ($i = 1$) to $n$ DO
3. IF $t_i$ is not in a cluster THEN
4.   $X = \{t_j | \text{distance } t_j \text{ to } t_i < \epsilon\}$
5.   IF $X$ is a valid cluster THEN
6.     $k = k + 1$;
7.     $c_k = X$;
8.   END IF
9. END IF
```

DBSCAN is a density-based clustering algorithm that generates clusters with a given minimum size ($minPts$) and density threshold (ϵ). This feature allows the algorithm to handle the "outlier problem" by ensuring individual outliers will not be included in a cluster. The overall number of clusters is determined by the algorithm. DBSCAN does not perform well with data sets that feature large differences in cluster densities because it is difficult to establish the most appropriate values of $minPts$ and ϵ. Table A.1 presents the DBSCAN algorithm.

Table A.2 lists the results obtained using the initial MADM framework (described in Chapter 3) to identify a best cluster configuration. The F-measure (F1) was used to evaluate the goodness of the clusters and hence to identify a best configuration. For the experiment ten data sets were taken from the UCI machine learning data repository [38]. Table A.2 gives the number of the clusters produced and the associated F-measure (F1) for each of three clustering algorithms used, i.e. K-means, K-NN and DBSCAN, and the required parameters (K, τ, $minPts$ and ϵ). Recall that, the Task Agent selects the cluster configuration with highest F1 value to be returned to the user.

Table A.2: The clustering results as produced by the MADM framework

No.	Data sets	K-means		K-NN			DBSCAN			
		Num Cls.	F1	t	Num Cls.	F1	$Min Pts$	ϵ	Num Cls.	F1
1	Lenses	3	0.50	1.00	1	0.60	1	1.0	4	0.62
2	Iris	3	0.89	1.00	4	0.78	1	5.0	4	0.87
3	Zoo	7	0.58	2.00	9	0.73	2	4.0	7	0.77
4	Wine	3	0.71	135.00	4	0.69	10	2700.0	7	0.33
5	Heart	2	0.59	94.00	3	0.67	3	55.0	3	0.02
6	Ecoli	8	0.70	0.45	6	0.46	1	0.4	7	0.71
7	Blood Trans.	2	0.71	1100.00	6	0.69	15	120.0	8	0.19
8	Pima Indians.	2	0.64	135.00	4	0.67	10	300.0	5	0.03
9	Yeast	10	0.42	0.35	9	0.41	2	0.5	9	0.54
10	Car	4	0.48	1.45	5	0.58	2	35.0	5	0.66

Table A.3 gives the best result in each case. The results were generated by a Validation Agent. From the table it can be seen that there is no obvious link between particular clustering algorithms and the features associate with individual data sets. The only thing that can be said is that DBSCAN and K-means tend (in many cases) to outperform K-NN.

Tables A.4, A.5 and A.6 present the results using K-means, K-NN and DBSCAN respectively. Each row in the table includes the number of classes generated, WGAD and BGAD values. Note that the separation (BGAD) for the *Lense* data set, using K-NN, is 0.00 because K-NN allocated all records to a single cluster!

Table A.7 gives a comparison of the best cluster configurations when selection made using: (i) minimising the WGAD or (ii) maximising the BGAD.

Table A.8 gives the actual accuracy, using all three algorithms. The overall accuracy is calculated as described in Chapter 3, Section 3.3. By cross referencing Table A.7

Table A.3: The “best” cluster result provided by the MADM framework

No.	Data sets	Overall F-Measure	Best clustering algo.
1	Lenses	0.62	DBSCAN
2	Iris	0.89	K-means
3	Zoo	0.77	DBSCAN
4	Wine	0.71	K-means
5	Heart	0.67	K-NN
6	Ecoli	0.71	DBSCAN
7	Blood Transfusion	0.71	K-means
8	Pima Indians.	0.67	K-NN
9	Yeast	0.54	DBSCAN
10	Car	0.66	DBSCAN

Table A.4: Muti-Agent Based Clustering Results using the K-means algorithm.

No.	Data Set	K-means		
		Num Cls.	WGAD	BGAD
1	Lenses	3	2.41	1.41
2	Iris Plants	3	1.95	3.62
3	Zoo	7	7.32	5.99
4	Wine	3	21.14	24.81
5	Heart	2	10.71	10.78
6	Ecoli	8	1.67	1.07
7	Blood Trans.	2	30.62	31.96
8	Pima Indians	2	44.06	21.18
9	Yeast	10	1.67	0.75
10	Car	4	8.52	3.31

Table A.5: Muti-Agent Based Clustering Results using the K-NN algorithm.

No.	Data Set	K-NN		
		Num Cls.	WGAD	BGAD
1	Lenses	1	1.17	0.00
2	Iris Plants	4	2.34	4.26
3	Zoo	9	6.76	9.88
4	Wine	3	22.97	41.80
5	Heart	3	13.84	19.09
6	Ecoli	13	1.47	1.78
7	Blood Trans.	2	31.08	35.03
8	Pima Indians	3	38.93	96.64
9	Yeast	9	1.46	1.62
10	Car	5	10.40	3.76

Table A.6: Muti-Agent Based Clustering Results using the DBSCAN algorithm.

No.	Data Set	DBSCAN		
		Num Cls.	WGAD	BGAD
1	Lenses	2	1.65	0.94
2	Iris Plants	3	3.22	2.16
3	Zoo	7	6.69	6.80
4	Wine	4	10.63	16.75
5	Heart	4	4.56	15.14
6	Ecoli	10	2.41	1.16
7	Blood Trans.	5	33.93	45.05
8	Pima Indians	9	53.26	40.19
9	Yeast	9	2.11	0.57
10	Car	5	6.69	3.48

Table A.7: A comparison of WGAD and BGD measures.

No.	Data Set	Num Classes	WGAD	Best clustering algo.	Num Classes	BGD	Best clustering algo.
1	Lenses	1	1.17	K-NN	3	1.41	K-means
2	Iris Plants	3	1.95	K-means	4	4.26	K-NN
3	Zoo	7	6.69	DBSCAN	9	9.88	K-NN
4	Wine	4	10.63	DBSCAN	3	41.80	K-NN
5	Heart	4	4.56	DBSCAN	3	19.09	K-NN
6	Ecoli	13	1.47	K-NN	13	1.78	K-NN
7	Blood Trans.	2	30.62	K-means	5	45.05	DBSCAN
8	Pima Indians	3	38.93	K-NN	3	96.64	K-NN
9	Yeast	9	1.46	K-NN	9	1.62	K-NN
10	Car	5	6.69	DBSCAN	5	3.76	K-NN

and Table A.8 it can be seen that with respect to the *Lenses*, *Pima* and *Car* data sets both metrics identify the most appropriate clustering configuration (although not necessarily using the same algorithm). In three cases (*Wine*, *Ecoli* and *Yeast*) neither approach finds the most appropriate cluster configuration.

Table A.8: Comparison of the accuracy values obtained using the K-means, K-NN, and DBSCAN algorithms.

No.	Data Set	K-means Accuracy	K-NN Accuracy	DBSCAN Accuracy	Best clustering algo.
1	Lenses	0.62	0.62	0.62	K-means,K-NN,DBSCAN
2	Iris	0.89	0.84	0.67	K-means
3	Zoo	0.78	0.83	0.85	DBSCAN
4	Wine	0.54	0.41	0.16	K-means
5	Heart	0.62	0.67	0.09	K-NN
6	Ecoli	0.83	0.71	0.87	DBSCAN
7	Blood	0.76	0.76	0.25	K-means, K-NN
8	Pima	0.65	0.65	0.12	K-means,K-NN
9	Yeast	0.53	0.34	0.64	DBSCAN
10	Car	0.70	0.70	0.70	K-means,K-NN,DBSCAN

A criticism of DBSCAN is the requirement for user-supplied parameters. It is difficult to identify the most appropriate ϵ. Another issue is that DBSCAN does not work well on small data sets because DBSCAN relies on a density-based notion of clusters. If DBSCAN is used to cluster a small data set, some genuine data objects are categorised as noisy data and therefore not included in any clusters. Most of clustering algorithms are designed for numeric data by using distance measure for a similarity function. The all include non-numeric attributes *Lense*, *Zoo* and *Car* data sets. *Lense* is a small data set containing 24 records and including 4 nominal data attributes, whereas *Zoo* comprises 15 boolean attributes and 1 numeric attribute. *Car* comprises 6 attributes of nominal. These ideally data sets require clustering approach that are designed to operate specifically with data types such as boolean or nominal.

Bibliography

[1] A. Agogino and K. Tumer. Efficient agent-based cluster ensembles. In *Proceedings of the 5th International Conference on Autonomous agents and multiagent systems*, AAMAS '06, pages 1079–1086, New York, USA, 2006. ACM.

[2] K. Albashiri, F. Coenen, and P. Leng. EMADS: An extendible multi-agent data miner. *Knowledge-Based Systems*, 22(7):523–528, 2009.

[3] L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In E. Durfee, editor, *Proceedings of the 4th International Conference on Multi-Agent Systems(ICMAS-2000)*, pages 31–38, Boston, MA, 2000. IEEE Press.

[4] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. In *Proceeding of 14^{t}h European Conference on Artificial Intelligence, ECAI'2000*, volume 23, pages 338–342, 2000.

[5] J. Austin. *How to do Things with Words.* Oxford University Press, Oxford, UK, 1962.

[6] H. Baazaoui Zghal, S. Faiz, and H. Ben Ghezala. A framework for data mining based multi-agent: An application to spatial data. In *Proceedings of the WEC'05: 3rd World Enformatika Conference*, volume 5, pages 22–26, 2005.

[7] T. Babu, M. Murty, and S. Subrahmanya. Multiagent based large data clustering scheme for data mining applications. In *Proceedings of the 6th International Conference on Active Media Technology (ACT'10)*, pages 116–127. Springer LNAI, 2010.

[8] S. Bailey, R. Grossman, H. Sivakumar, and A. Turinsky. Papyrus: A system for data mining over local and wide area clusters and super-clusters. *In Proceedings of Supercomputing. IEEE*, 1999.

[9] M. Barbuceanu and M. Fox. The design of a coordination language for multi-agent systems. In Jörg Müller, Michael Wooldridge, and Nicholas Jennings, editors, *Intelligent Agents III Agent Theories, Architectures, and Languages*, volume 1193 of *Lecture Notes in Computer Science*, pages 341–355. Springer Berlin / Heidelberg, 1996.

[10] Jr. Bayardo, W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Halal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. The infosleuth project. *SIGMOD Rec.*, 26(2):543–545, 1997.

[11] F. Bellifemine, F. Bergenti, G. Caire, and A. Poggi. JADE a java agent development framework. In Rafael H. Bordini, M. Dastani, J. Dix, and A.E. Fallah-Seghrouchni, editors, *Multi-agent programming : languages, platforms, and applications*, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 125–147. Springer US, 2005.

[12] F. Bellifemine, G. Caire, and D. Greenwood. *Developing Multi-Agent Systems with JADE*. Wiley, 2007.

[13] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa. JADE: A software framework for developing multi-agent applications. Lessons learned. *Information and Software Technology*, 50(1-2):10–21, January 2008.

[14] T. Bench-Capon, F. Coenen, and P. Leng. An experiment in discovery association rules in the legal domain. In *DEXA Workshop 2000*, pages 1056–1060, 200.

[15] Trevor Bench-Capon. Neural networks and open texture. In *Proceedings of the 4th international conference on Artificial intelligence and law*, ICAIL '93, pages 292–297, New York, NY, USA, 1993. ACM.

[16] M. Berry and M. Browne. *Lecture notes in data mining*. World Scientific, 2006.

[17] L. Cao. Introduction to agent mining interaction and integration. In Longbing Cao, editor, *Data Mining and Multi-agent Integration*, pages 3–36. Springer US, 2009.

[18] L. Cao, V. Gorodetsky, and P. Mitkas. Agent mining: The synergy of agents and data mining. *Intelligent Systems, IEEE*, 24(3):64–72, May-June 2009.

[19] L. Cao, C. Luo, and C. Zhang. Agent-mining interaction: An emerging area. In Vladimir Gorodetsky, Chengqi Zhang, Victor Skormin, and Longbing Cao, editors, *Autonomous Intelligent Systems: Multi-Agents and Data Mining*, volume 4476 of *Lecture Notes in Computer Science*, pages 60–73. Springer Berlin / Heidelberg, 2007.

[20] Lo. Cao, G. Weiss, and P. Yu. A brief introduction to agent mining. *Autonomous Agents and Multi-Agent Systems*, 25:419–424, 2012.

[21] B. Chaib-draa and F. Dignum. Trends in Agent Communication Language. *Computational Intelligence*, 18(2):89–101, 2002.

[22] S. Chaimontree, K. Atkinson, and F. Coenen. Clustering in a multi-agent data mining environment. In *Proceedings of the 6th international conference on Agents and data mining interaction*, ADMI'10, pages 103–114, Berlin, Heidelberg, 2010. Springer-Verlag LNAI 5980.

[23] S. Chaimontree, K. Atkinson, and F. Coenen. Multi-agent based clustering: Towards generic multi-agent data mining. In *Proceedings of the 10th industrial conference on Advances in data mining: applications and theoretical aspects*, pages 115–127. Springer LNAI 6171, 2010.

[24] S. Chaimontree, K. Atkinson, and F. Coenen. A multi-agent based approach to clustering: Harnessing the power of agents. In *Proceedings of the 7th international conference on Agents and data mining interaction*, ADMI'11, pages 12–29, Berlin, Heidelberg, 2011. Springer-Verlag LNAI 7103.

[25] I. Czarnowski and P. Jędrzejowicz. Agent-based non-distributed and distributed clustering. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5632 LNAI:347–360, 2009.

[26] J. da Silva, C. Giannella, R. Bhargava, H. Kargupta, and M. Klusch. Distributed data mining and agents. *Engineering Applications of Artificial Intelligence*, 18(7):791–807, 2005.

[27] B. Dasarathy. *Nearest neighbor (NN) norms: NN pattern classification techniques.* IEEE Computer Society Press, Las Alamitos, California, 1991.

[28] D. Davies and D. Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-1(2):224–227, April 1979.

[29] W. Davies. Distributed learning: An agent-based approach to data-mining. In *ML95 Workshop on Agents that Learn from Other Agents, Tahoe*, 1995.

[30] W. Davies and P. Edwards. Agent-based knowledge discovery. In *Working Notes of the AAAI Spring Symp. Information Gathering from Heterogeneous, Distributed Environments*, pages 34–37. AAAI Press, 1995.

[31] I. Dhillon and D. Modha. A data-clustering algorithm on distributed memory multiprocessors. In *Proceedings Workshop on Large-Scale Parallel KDD System, SIGKDD*, pages 245–260. Springer LNCS 1759, 2000.

[32] G. Di Fatta and G. Fortino. A customizable multi-agent system for distributed data mining. *Proceedings of the ACM Symposium on Applied Computing*, pages 42–47, 2007.

[33] C. dos Santos and A. C. Bazzan. Integrating knowledge through cooperative negotiation - a case study in bioinformatics. In Vladimir Gorodetsky, Jiming Liu, and Victor Skormin, editors, *Autonomous Intelligent Systems: Agents and Data Mining*, volume 3505 of *Lecture Notes in Computer Science*, pages 81–115. Springer Berlin / Heidelberg, 2005.

[34] M. Dunham. *Data Mining: Introductory And Advanced Topics.* Pearson Education, 2006.

[35] R. Elmasri and S. Navathe. *Fundamentals of Database Systems.* Addison-Wesley Publishing Company, USA, 6^{th} edition, 2010.

[36] A. Farquhar, R. Fikes, and J. Rice. The ontolingua server: a tool for collaborative ontology construction. *International Journal of Human-Computer Studies*, 46(6):707–727, June 1997.

[37] FIPA. Communicative Act Library Specification. Technical Report XC00037H, Foundation for Intelligent Physical Agents, 2001. Available from the website *http://www.fipa.org.* Accessed 23 January 2011.

[38] A. Frank and A. Asuncion. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2010. Accessed 10 January 2011.

[39] C. Giannella, R. Bhargava, and H. Kargupta. Multi-agent systems and distributed data mining. In Matthias Klusch, Sascha Ossowski, Vipul Kashyap, and Rainer Unland, editors, *Cooperative Information Agents VIII*, volume 3191 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2004.

[40] T. Gruber. The role of common ontology in achieving sharable, reusable knowledge base. In *Proceeding of the 2^{nd} International Conference on Principles of Knowledge Representation and Reasoning*, pages 601–602, San Mateo, CA, 1991.

[41] T. Gruber. Ontology progress - what is an ontology. *Software World*, 34(1):6, 2003.

[42] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster validity methods: part I. *SIGMOD Record*, 31(2):40–45, 2002.

[43] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Clustering validity checking methods: part II. *SIGMOD Record*, 31(3):19–27, 2002.

[44] J. Han, M. Kamber, and J. Pei. *Data mining Concepts and Techniques.* Morgan Kaufmann, Waltham, MA, USA, 3^{rd} edition, 2012.

[45] A. Jain, M. Murty, and P. Flynn. Data clustering: a review. *ACM Computing Survey*, 31:264–323, September 1999.

[46] E. Januzaj, H. Kriegel, and M. Pfeifle. Scalable density based distributed clustering. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD '04, pages 231–244, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[47] N. Jennings, P. Faratin, A. Lomuscio, S. Parsons, M. Wooldridge, and C. Sierra. Automated negotiation : prospects, methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.

[48] B. Johnston and G. Governatori. Induction of defeasible logic theories in the legal domain. In *Proceedings of the 9th international conference on Artificial intelligence and law*, ICAIL '03, pages 204–213, New York, NY, USA, 2003. ACM.

[49] H. Kargupta, I. Hamzaoglu, and B. Staffordan. Scalable, distributed data mining using an agent based architecture. In *Proceedings of the 3rd International Conference on the Knowledge Discovery and Data Mining*, pages 211–214. AAAI Press, 1997.

[50] L. Kaufman and P. Rousseeuw. *Finding Groups in Data An Introduction to Cluster Analysis.* Wiley Interscience, New York, 1990.

[51] M. Klusch, S. Lodi, and G. Moro. Agent-based distributed data mining: The KDEC scheme. In *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, volume 2586, pages 104–122, 2003.

[52] M. Klusch, S. Lodi, and G. Moro. The role of agents in distributed data mining: Issues and benefits. In *Proceedings IEEE/WIC International Conference on Intelligent Agent Technology (IAT '03)*, pages 211–218, 2003.

[53] Y. Labrou. Standardizing agent communication. In *Selected Tutorial Papers from the 9th ECCAI Advanced Course ACAI 2001 and Agent Link's 3rd European Agent Systems Summer School on Multi-Agent Systems and Applications*, EASSS '01, pages 74–97, London, UK, 2001. Springer-Verlag.

[54] Y. Labrou and T. Finin. Readings in agents. In Michael N. Huhns and Munindar P. Singh, editors, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, chapter Semantics and conversations for an agent communication language, pages 235–242. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[55] Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: the current landscape. *Intelligent Systems and their Applications, IEEE*, 14(2):45–52, March/April 1999.

[56] D. Mackenzie. Question-begging in non-cumulative systems. *Journal of Philosophical Logic*, 8:117–133, 1979.

[57] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[58] L. Mashayekhy, M. Nematbakhsh, and B. Ladani. E-negotiation model based on data mining. In *IADIS International Conference e-Commerce 2007*, pages 369–373, 2006.

[59] P. Mcburney and S. Parsons. Dialogue games in multi-agent systems. *Informal Logic*, 22(3):257–274, 2002.

[60] P. McBurney and S. Parsons. Dialogue game protocols. In Marc-Philippe Huget, editor, *Communication in Multiagent Systems*, volume 2650 of *Lecture Notes in Computer Science*, pages 269–283. Springer Berlin / Heidelberg, 2003.

[61] P. McBurney, S. Parsons, and M. Wooldridge. Desiderata for agent argumentation protocols. In C. Castelfranchi and W. L. Johnson, editors, *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, pages 402–409, Bologna, Italy, 2002. ACM Press: New York, USA.

[62] P. Mitkas, D. Kehagias, A. Symeonidis, and I. Athanasiadis. A framework for constructing multi-agent applications and training intelligent agents. In Paolo Giorgini, Jörg Müller, and James Odell, editors, *Agent-Oriented Software Engineering IV*, volume 2935 of *Lecture Notes in Computer Science*, pages 255–290. Springer Berlin / Heidelberg, 2004.

[63] C. Moemeng, V. Gorodetsky, Z. Zuo, Y. Yang, and C. Zhang. Agent-based distributed data mining: A survey. In Longbing Cao, editor, *Data Mining and Multi-agent Integration*, pages 47–58. Springer US, 2009.

[64] M. Mozina, J. Zabkar, T. Bench-Capon, and I. Bratko. Argument based machine learning applied to law. *Artificial Intelligence and Law*, 13(1):53–73, 2005.

[65] R. Nkambou, P. Fournier-Viger, and E. Nguifo. Improving the behaviour of intelligent tutoring agents with data mining. *Intelligent Systems, IEEE*, 24(3):46–53, May-June 2009.

[66] P. O'Brien and R. Nicol. Fipa-towards a standard for software agents. *BT Technology Journal*, 16(3):51–59, July 1998.

[67] V. Olman, F. Mao, H. Wu, and Y. Xu. Parallel clustering algorithm for large data sets with applications in bioinformatics. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 6(2):311–352, 2009.

[68] A. P. Canuto, A. C. Campos, V. S. Bezerra, and M. da C. Abreu. Investigating the use of a multi-agent system for knowledge discovery in databases. *International Journal of Hybrid Intelligent Systems*, 4(1):27–38, 2007.

[69] R. Patil, R. Fikes, P. Patel-Schneider, D. Mckay, T. Finin, T. Gruber, and R. Neches. The DARPA knowledge sharing effort: Progress report. In Bernhard Nebel, Charles Rich, and William Swartout, editors, *KR'92. Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference*, pages 777–788. Morgan Kaufman, 1992.

[70] F. Pereira and D. Warren. Readings in natural language processing. chapter Definite clause grammars for language analysis, pages 101–124. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1986.

[71] D. Pop, V. Negru, and C. Sandru. Multi-agent architecture for knowledge discovery. In *Proceedings of the 8th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2006*, pages 217–223, 2007.

[72] S. Ramchurn, C. Sierra, L. Godo, and N. Jennings. Negotiating using rewards. *Artif. Intell.*, 171(10-15):805–837, July 2007.

[73] M. Rao. Cluster analysis and mathematical programming. *Journal of the American Statistical Association*, 66(335):622–626, 1971.

[74] V. Rao. Multi agent-based distributed data mining: An overview. *International Journal of Reviews in Computing*, pages 83–92, 2009.

[75] J. Reed, T. Potok, and R. Patton. A multi-agent system for distributed cluster analysis. In *Proceedings of the 3rd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'04) W16L Workshop - 26th International Conference on Software Engineering*, pages 152–155, Edinburgh, Scotland, UK, 2004. IEE.

[76] P. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, November 1987.

[77] F. Sadri, F. Toni, and P. Torroni. Dialogues for negotiation: Agent varieties and dialogue sequences. In *Revised Papers from the 8th International Workshop on Intelligent Agents VIII*, ATAL '01, pages 405–421, London, UK, UK, 2002.

[78] J. Searle. *Speech Acts: An Essay in the Philosophy of Language.* Cambridge University Press, Cambridge, UK, 1969.

[79] C. Sierra, N. Jennings, P. Noriega, and S. Parsons. A framework for argumentation-based negotiation. In *4th Int. Workshop on Agent Theories, Architectures and Languages (ATAL 97)*, pages 167–182, 1997.

[80] C. Stern. Clustering of environmental Data using a Multi-agent System. In *the 13th AGILE International Conference on Geographic Information Science*, 2010.

[81] K. Sycara. Multiagent systems. *AI Magazine*, 19(2):79–92, 1998.

[82] V. Tamma, S. Phelps, I. Dickinson, and M. Wooldridge. Ontologies for supporting negotiation in e-commerce. *Engineering Applications of Artificial Intelligence*, 18(2):223–236, 2005.

[83] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining.* Addison-Wesley, 2005.

[84] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Fourth Edition.* Academic Press, 4th edition, 2008.

[85] C. van Rijsbergen. *Information Retrieval.* Butterworths, London, 2nd edition, 1979.

[86] M. Wardeh, T. Bench-Capon, and F. Coenen. Argument based moderation of benefit assessment. In *Proceedings of the 2008 conference on Legal Knowledge and Information Systems: JURIX 2008: The Twenty-First Annual Conference*, pages 128–137, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.

[87] M. Wardeh, T. Bench-Capon, and F. Coenen. Padua: a protocol for argumentation dialogue using association rules. *Artificial Intelligence and Law*, 17:183–215, 2009.

[88] M. Wardeh, F. Coenen, and T. Bench-Capon. Multi-agent based classification using argumentation from experience. *Autonomous Agents and Multi-Agent Systems*, 25:447–474, 2012.

[89] M. Wooldridge and N. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[90] R. Xu and D. C. Wunsch. *Clustering.* Wiley/IEEE Press, 2009.

[91] Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making (IJITDM)*, 5(04):597–604, 2006.