A Multi-Agent Based Approach To Clustering: Harnessing The Power of Agents

Santhana Chaimontree, Katie Atkinson and Frans Coenen

Department of Computer Science, University of Liverpool, Liverpool, L69 3BX, UK.
S.Chaimontree,katie,coenen}@liverpool.ac.uk

Abstract. A framework for multi-agent based clustering is described whereby individual agents represent individual clusters. A particular feature of the framework is that, after an initial cluster configuration has been generated, the agents are able to negotiate with a view to improving on this initial clustering. The framework can be used in the context of a number of clustering paradigms, two are investigated: K-means and KNN. The reported evaluation demonstrates that negotiation can serve to improve on an initial cluster configuration.

Keywords: Multi-Agent Data Mining, Clustering

1 Introduction

Data Mining and Multi-Agent Systems (MAS) are well established technologies which are finding increasing application. One of the current challenges of data mining is how to cope with the ever increasing size of the data sets that we wish to mine. A highlevel answer is to adopt and apply greater computational power. This can be achieved in a number of manners. One approach is to make use of distributed [6, 8, 10, 21, 22, 25] or parallel [13, 26, 27] processing techniques so that several processors can be applied to the problem. This of course assumes that appropriate "farms" of processors are available. However, a more specific disadvantage is that of centralised control and lack of generality. Distributed and parallel data mining techniques typically assume a "master" process that directs the data mining task, therefore control is centralised at the master process and consequently these systems lack robustness. Further, distributed and parallel approaches tend to be directed at specific data mining applications and are difficult to generalise (because of the centralised control inherent to these systems). MAS offer an alternative to handling large quantities of data by harnessing the power of numbers of processors, with the added advantages that control is not centralised and consequently such systems can be much more robust and versatile.

A MAS is essentially a collection of software entities (agents) that are intended to cooperate in some manner so as to undertake some processing task. An important aspect of this cooperation is that the agents behave in an autonomous manner; they *negotiate* with one another to complete a given task

rather than being directed to do so by some master process. The idea of adopting MAS technology for data mining is therefore an attractive one. Multi-agent Data Mining (MADM) or Agent Assisted Data Mining (AADM) also 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. MADM also supports the creation of frameworks that can be allowed to grow in an almost anarchic manner, agents can be easily incorporated into such frameworks as long as they comply with whatever protocols have been specified. The nature of these protocols remains a research issue. A number of Agent Communication Languages (ACLs) have been proposed but often these are difficult to fit to particular applications; it is therefore frequently necessary to extend or partially replace the set of performatives that are specified as part of these ACLs in order to taylor them to the specific scenario.

This paper describes an MADM framework, a set of agent communication performatives and supporting protocol, directed at unsupervised learning (clustering). The framework comprises four general categories of agent: user agents, data agents, clustering agents, validation agents (there are also house keeping agents, but these can be considered to be orthogonal to the task of data mining). Some of these agents are persistent while others are spawned as required, and have a lifetime equivalent to the duration of a given clustering task. To support the desired MADM communicates a dedicated set of performatives have been derived. A particular feature of the framework is that it supports negotiation between agents. This negotiation process allows for the enhancement of clusters once an initial cluster configuration has been established.

2 Previous Work

This section presents a review of some related work to that described in this paper. The section commences with some general background to MADM and then continues with a brief review of some parallel work on MAS clustering, highlighting the distinction between this work and the proposed MAS clustering approach.

There are two main paradigms for the interaction and integration between agent and data mining [4]: (i) data mining-driven agents which is the use of data mining to support the abilities of agents such as adaptation, coordination, learning, reasoning, etc. and (ii) agent-driven data mining, commonly known as Multi-Agent Data Mining (MADM), is the use of a collection of agents to perform data mining tasks. Surveys of agent-based distributed data mining can be found in [12, 17, 20].

PADMA [14] and PAPYRUS [2] are two of the earliest (late 1990s) reported multi-agent clustering systems. These systems aimed to achieve the integration of knowledge discovered from different sites with a minimum amount of network communication and a maximum amount of local computation. PADMA uses a facilitator (or coordinator) agent to direct interaction between the mining agents. As such, it is based on a centralised architecture. PADMA agents are

used to access local data and perform analysis. The *local clusters* are collected centrally to generate the *global clusters*. In addition, PADMA can be used to generate hierarchical clusters in the context of document categorisation. On the other hand, PAPYRUS differs from PADMA in that it is a distributed clustering system. PAPYRUS adopted a Peer-to-Peer model where both data and results can be moved between agents according to given MAS strategies. A more recent multi-agent clustering system is KDEC [16]. KDEC is a distributed density-based clustering algorithm also founded on the Peer-to-Peer model. In KDEC density estimation samples are transmitted, instead of actual data values so as to preserve data privacy and minimize communication between sites. A multi-agent clustering system directed at documents has been proposed in [24]; the objective here is to improve the accuracy and the relevancy of information retrieval processes. Kiselev et al. [15] proposed a clustering agent based system dealing with data streams in distributed and dynamic environments whereby input data sets and decision criteria can be changed at runtime (clustering results are available at anytime and are continuously revised to achieve the global clustering).

The above systems use agents to facilitate data privacy and support distributed data clustering (mining). There is little reported work on an agent based clustering systems that support intra-agent negotiation in order to refine (enhance) cluster configurations. In [1] Agogino et al. proposed an agent-based cluster ensemble approach to generate a best cluster configuration. Reinforcement learning, to maximise a utility function with respect to the original clustering results, is used to achieve the desired best clustering. However, the approach proposed in the paper operates in a different manner by using negotiation among agents to improved the quality of clustering result. It is argued that this approach harnesses the true power of agents.

3 The MADM Framework

The proposed MADM framework, as noted above, comprises four categories of agent:

1. User agents.
2. Data agents.
3. Clustering agents.
4. Validation agents.

User agents are the interface between end users and the MADM environment. The agents are responsible for obtaining the input from the user, spawning clustering agents in order to perform the clustering task and presenting the derived clustering result. To the above list of specific MADM agents we can also add a number of housekeeping agents that are utilised within the MADM framework.

Data agents are the "owners" of data sources. There is a one-to-one relationship between data agents and data sources. Data agents can be thought of as the conduit whereby clustering agents can access data.

Clustering agents are the "owners" of clusters. Groups of clustering agents can be though of as representing a clustering algorithm. With respect to this paper the K-means [18] and K-Nearest Neighbour (KNN) [7] clustering algorithms have been adopted; however, our collections of clustering agents could have been configured to perform some alternative form of clustering (for example hierarchical clustering). A number of clustering agents will be spawned, as required, by a user agent in order to perform some clustering task. Thus, each clustering agent represents a cluster and is responsible for selecting a record from a data set and determining whether that record would belong to its cluster or not. The number of clustering agents, therefore depends on the number of clusters (K). 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 KNN approach only one initial clustering agent will be spawned; then, as the KNN algorithm progresses further clustering agents may be created. Details concerning the operation of K-means and KNN with respect to the proposed MADM framework will be presented in Section 5. Clustering agents collectively have two principal functions: (i) initial generation of a "start" cluster configuration, and (ii) cluster refinement.

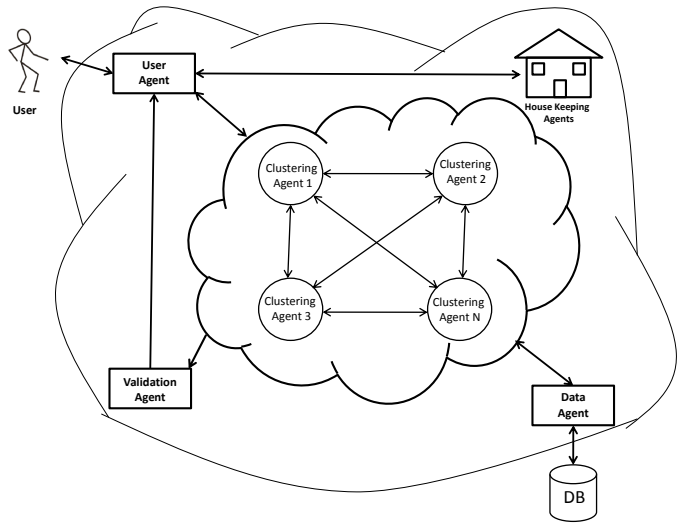

Fig. 1. Proposed MADM Framework

Validation agents are a special type of agent that performs validation operations on clustering results. Each validation agent is the "owner" of a technique for measuring the "goodness" of a given cluster configuration. In the current system validation agents consider either cluster cohesion or cluster separation or both.

A possible configuration for the proposed MADM framework, incorporating the above, is presented in Figure 1. The Figure includes a User Agent, a collection of Clustering Agents, a Data Agent, a Validation Agent and some house-

keeping agents. The directed arcs indicate communication between agents. Note that communication can be bidirectional or unidirectional and that the Data Agent directly communicates with each Clustering Agent. Intra-communication between Clustering Agents takes follows a protocol that permits negotiation about cluster exchange to take place.

The framework has been realised using the Java Agent Development Environment (JADE) [3]. The house keeping agents provided by JADE, include the AMS (Agent Management System) agent and the DF (Directory Facilitator) agent. The AMS agent is responsible for managing and controlling the lifecycle of other agents in the platform, whereas the DF agent provides a "yellow pages" service to allow agents to register their capabilities.

4 Agent Communication within the Framework

The agents within our framework need to be able to communicate to carry out their tasks. JADE provides a communication mechanism that makes use of the FIPA ACL performatives [9]. However, as has been discussed previously in [19], the FIPA ACL has limited applicability to dialogues not involving purchase negotiations. Given that we wish to permit the agents in our system to engage in dialogues about the exchange of items between cluster configurations, we need a more expressive language to support this communication. As such, we have defined and implemented a set of performatives to enable our agents to engage in negotiations about the suitability of moving items/merging between clusters. At the highest level, the performatives are categorised as follows: holding a dialogue; performing the clustering task, negotiating about the movement of items between clusters; informing others about the clustering results.

The performatives are defined axiomatically in terms of the pre-conditions that must hold for an agent to be able to use the performatives and the post-conditions that apply following the use of the performatives. The agents use the performatives as part of a protocol that governs the exchange of information between the agents. For reasons of space we do not include here the details of the semantics of the performatives, but instead describe the communication protocol that the agents follow when using the communication language. We indicate in italics the performatives used at each stage.

A dialogue opens (*mining request*) which triggers a mining request to the data, cluster and validation agents to join. Once the recipient agents have entered the dialogue (*join dialogue*), the clustering task is performed (*inform data*). On completion of the initial clustering, the agents evaluate the cohesion and seperation of their clusters and as a result may broadcast to other agents that items be moved between clusters (*propose item move*). The recipients of the proposal can then reply by accepting (*accept item move*) or rejecting the proposed move (*reject item move*). We also permit retraction of a proposed item move (*retract item move*) if it is subsequently found to yield an unsuitable configuration. When the items have been moved between clusters and the agents are happy to accept the results, the clustering agents inform the validation agent of the new config-

urations (*inform cluster*). The overall cluster configuration is then sent to the user agent (*inform cluster configuration*), after which moves are made for agents to exit the dialogue (*leave dialogue*) and subsequently end it (*close dialogue*).

The way in which the reconfiguration of clusters happens is dependant upon the clustering algorithm used, as will be described in the next section.

5 Operation

The operation of the proposed MADM clustering mechanism is described in this section. As noted in the foregoing, Clustering Agents are spawned by a User Agent according to the nature of the end user's initial "clustering request". Fundamentally there are two strategies for spawning Clustering Agents: the K-means strategy and the KNN strategy. In the K-means strategy the user pre-specifies the number of clusters, K, that are required; in which case K Clustering Agents are spawned. In the case of the KNN strategy the MADM process decides how many clusters are required, thus initially only one Clustering Agent is spawned (more may be generated later as required). The operation of the proposed MADM clustering mechanism comprises two phases: a bidding phase and a refinement phase. During the bidding phase Clustering Agents compete for records by "bidding" for them in an "auction" setting where the Data Agent acts as the auctioneer. For each record the Clustering Agent that poses the best bid wins the record and includes it in its cluster (see Sub-sections 5.1 and 5.2). During the refinement phase each Clustering Agents tries to pass unwanted records (records that no longer fit within its cluster definition) to other agents. This can also be conceptualised in terms of an auction; each Clustering Agent acts as a local auctioneer and tries to sell its unwanted records by inviting other clustering agents to bid for them. The operation of the refinement phase is entirely independent of the spawning strategy adopted. However the operation of the bidding phase differs according to the nature of the spawning strategy. The rest of this section is organised as follows. In Sub-sections 5.1 and 5.2 the operation of the biding phase with respect to the K-means and KNN spawning strategies are described. Sub-section 5.3 then describes the operation of the refinement phase.

Table 1. Bidding phase founded on K-means Spawning Strategy

Phase I: Bidding using K-means
Input: Dataset ($D = \{d_1, d_2, \cdots, d_n\}$), the desired number of clusters (K) Output: An initial clustering configuration
1. User Agent spawns K Clustering Agents ($C = \{c_1, c_2, \cdots, c_K\}$) 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. $\forall c_j \in C$ ($j = 1$ to K) 7. $bidDistance = d_i -$ *centroid* c_j 8. Allocate d_i to c_j so as to minimise $bidDistance$

Table 2. Bidding phase founded on KNN Spawning Strategy

Phase I: Bidding using KNN
Input: Dataset ($D = \{d_1, d_2, \cdots, d_n\}$), threshold t Output: An initial clustering configuration
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 $bidDistance < t$, 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

5.1 Biding Phase founded on the K-means Spawning Strategy

The operation of the bidding process with respect to the K-means strategy is presented in Table 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 centroid of each cluster (lines 2 to 4). The clustering agents then bid for the remaining records in D (lines 5 to 8). In the current implementation the *bidDistance* equates to the "distance" of d_i to the nearest neighbour of d_i in the cluster. 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, the "goodness" of this initial configuration is very much dependent on the nature of the first K records selected to define the initial clusters.

5.2 Biding Phase founded on the KNN Spawning Strategy

The biding phase founded on the KNN spawning strategy, as noted above, commences with a single Clustering Agent (C_i). The operation of this bidding process is presented in Table 2. Note that the process requires a *nearest neighbour threshold*, t, as a parameter. The threshold is used to determine the "nearest neighbour". If the *bidDistance* is less than the threshold, this record in question is allocated to the "closest" cluster (line 6). If there is no "closest" cluster a new Clustering Agent is spawned (line 7). Note that the chosen value of t can significantly affect the number of Clustering Agents that are spawned. The authors proposed a method to identify the most appropriate value for t in [5].

5.3 Refinement (Negotiation) Phase

The refinement process is presented in Table 3. The refinement process is driven using individual cluster cohesion values and an overall cluster configuration

seperation value. We wish to generate a configuration which minimises the cohesion values and maximises the separation value. The refinement phase commences (line 1) with each Clustering Agent determining its own cluster cohesion value and the set of Clustering Agents collectively determining a 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. For this latter purpose each Cluster Agent defines its cluster in terms of its centroid and these values are then transmitted to all other clustering agents so that an overall separation value can be determined. With respect to the framework described in this paper the Within Group Average Distance (WGAD) and the Between Group Average Distance (BGAD) metrics were adopted to determine cluster cohesion and separation respectively (alternative methods could equally well have been adopted). These metrics are described in Section 6. The cohesion and separation values are sufficient if the cohesion values are below a specified cohesion threshold and the separation value is above a prespecified separation threshold. If this is not the case the cluster associated with each Clustering Agent (c_i) will be be split into two sub-clusters: a *major sub-cluster* and a *minor sub-cluster*. The splitting is achieved by applying a standard K-means algorithm (with K set to 2) to the records held by c_i. The cluster with the smallest number of records is then designated to be the minor sub-cluster and these records are then put up for auction. The auction proceeds in a similar manner to that described for the bidding phase founded on the K-means strategy (see Sub-section 5.1) with the distinction that the WGAD value is used as the *bidDistance*. This process repeats until satisfactory cohesion and separation values are reached. Note (line 8) that on any given iteration, if a record cannot be allocated to a class (because its WGAD value is too high) it is allocated to an *outlier cluster*.

6 Cluster Configuration Metrics

In order for agents to bid for examples some appropriate metric must be adopted. The process for deciding when and how to move a record also requires recourse to some metric, as does deciding when to split and merge clusters. Essentially there are three ways of measuring the "goodness" of a proposed cluster configuration. We can measure intra-cluster cohesion, we can measure inter-cluster separation or we can adopt both metrics. This section presents a review of some of the metrics that may be used to measure the goodness of a cluster configuration and the metrics used in the context of the work described in this paper.

In context of the work described the authors have adopted the Within Group Average Distance (WGAD) [23] and the Between Group Average Distance (BGAD) metrics:

$$WGAD = \frac{\sum_{i=1}^{i=|C|} dist(x_i, c)}{|C|} \quad . \tag{1}$$

Table 3. Algorithm for refinement phase

Algorithm Phase II: Refinement

Input: a set of clusters
Output: an improved clustering result

1. For all clusters calculate cluster cohesion and separation values
2. DO WHILE there exists cluster cohesion value > cohesion threshold
   or cluster separation value < separation threshold
3. $\forall c_i \in C$ ($i = 1$ to$K$)
3. Split a cluster c_i into two sub-clusters, c_{major} and c_{minor} using K-means
4. $\forall d \in c_{minor}$
5. $\forall c_j \in C$ ($j = 1$ to K and $j \neq i$)
6. $bidDistance = WGAD_j$ (see Section 6)
7. IF $\exists c_j \in C$ such that $bidDistance < cohesion\ threshold$, allocate d to c_j so as to minimise $bidDistance$
8. ELSE Allocate d to "outlier cluster"
9. IF no "successful" bids end loop

$$BGAD = \sum_{i=1}^{i=K} dist(c_i, c) \ . \tag{2}$$

where $|C|$ is the number of objects in a cluster (i.e. the size of a cluster), c is the cluster centroid, and $dist(x_i, c)$ is the distance between object x_i and the cluster centroid. K is the number of clusters.

The lower the WGAD value the greater the cohesiveness of the cluster, whereas the higher the BGAD value the greater the separation of the clusters from one another. We wish to minimise the WGAD and maximise the BGAD to achieve a best cluster configuration. The target WGAD value, the cohesion threshold, is the average WGAD across the identified set of clusters 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 ($1.0 < q < 2.0$). Our experiments indicate that values of $p = 0.8$ and $q = 1.2$ tend to produce good results.

7 Evaluation

To evaluate our approach we experimented with a selection of data sets taken from the UCI machine learning repository [11]. We compared the operation of our MADM approach with the well known K-means and KNN clustering algorithms. In each case we recorded the accuracy of the clustering operation, with respect to the known (*ground truth*) clustering. For the K-means algorithm the number of desired clusters must be pre-specified in order to spawn an appropriate number of clustering agents. For this purpose we have used the number of classes given in the UCI repository. The t parameter used for KNN was selected, for evaluation

Table 4. Comparison of the result accuracy provided by K-means task distribution before and after cluster configuration improvement.

No.	Data Set	Num Classes	Accuracy Phase I	Accuracy Phase II	Cohesion threshold	Seperation threshold
1	**Iris**	3	0.89	0.97	1.41	2.03
2	**Zoo**	7	0.76	0.78	1.94	1.95
3	**Wine**	3	0.68	0.70	204.95	296.73
4	**Heart**	2	0.55	0.59	33.87	106.28
5	**Ecoli**	8	0.76	0.80	0.42	0.54
6	Blood Transfusion	2	0.76	0.76	794.16	4582.29
7	**Pima Indians**	2	0.65	0.66	65.08	290.60
8	**Breast cancer**	2	0.79	0.85	283.16	1729.93

purposes, according to the nature of the input so as to be compatible with the specified number of clusters. The results using K-means and KNN are reported in Tables 4 and 5 respectively. The columns in the tables describe: the number of identified clusters (classes), the accuracy after the initial Phase I clustering (i.e. the accuracy that would be achieved using K-means or KNN without any further negotiation), the accuracy after refinement (Phase II), and the calculated cohesion and separation thresholds. Accuracy values are calculated as follow:

$$Accuracy = \frac{\sum_{i=1}^{i=K} C_i}{m} \quad . \tag{3}$$

Where K is the number of clusters, m is the number of records and C_i is the size (in terms of the number of records) of the majority class for cluster i. Note that the ground truth is only used here to evaluate the outcomes.

Table 5. Comparison of the result accuracy provided by KNN task distribution before and after cluster configuration improvement.

No.	Data Set	Num Classes	t	Accuracy Phase I	Accuracy Phase II	Cohesion threshold	Seperation threshold
1	**Iris**	4	0.99	0.84	0.93	1.90	1.93
2	Zoo	7	2.24	0.82	0.73	2.42	2.55
3	Wine	3	164.31	0.65	0.65	281.49	282.98
4	**Heart**	2	115.29	0.55	0.64	21.96	62.51
5	**Ecoli**	7	0.42	0.64	0.67	0.38	0.26
6	Blood Transfusion	2	3500.83	0.76	0.76	1222.54	3090.16
7	Pima Indians	2	149.08	0.65	0.65	133.77	152.08
8	**Breast cancer**	2	826.75	0.63	0.84	205.51	847.98

From the tables it can be seen that in the majority of cases (shown in bold font) agent negotiation serves to enhance the initial clustering, with the K-means approach tending to outperform the KNN approach. Interestingly, in the case of the Zoo data set when using the KNN approach, negotiation had an adverse effect; the research team conjecture that this is something to do with

the large number of classes (and hence the large amount of "splitting") featured in this data set. In the remaining cases the situation remained unchanged, either because a best configuration had been identified immediately, or because the WGAD and BAGD values could not be reduced). It should also be noted that the number of class values given in Table 4 (column three) are the ground truth values; the KNN approach does not always produce the correct number of classes and hence this is why the accuracy values are not always as good as in the case of the K-means approach.

8 Conclusion

A MADM framework to achieve multi-agent based clustering has been described. A particular feature of the framework is that it enables agents to negotiate so as to improve on an initial clustering. The framework can be used in a number of ways. Two approaches were considered: K-means and KNN. Evaluation using a number of datasets taken from the UCI repository indicates that in most cases (and especially when using the K-means approach) a better clustering configuration can be obtained as a result of the negotiation process.

References

1. Agogino, A., Tumer, K.: Efficient agent-based cluster ensembles. In: Proc. 5^{th} Int. Conf. on Autonomous agents and multiagent systems. pp. 1079–1086. AAMAS '06, ACM, New York, NY, USA (2006)
2. Bailey, S., Grossman, R., Sivakumar, H., Turinsky, A.: Papyrus: A system for data mining over local and wide area clusters and super-clusters. IEEE Supercomputing (1999)
3. Bellifemine, F., Bergenti, F., Caire, G., Poggi, A.: JADE: a java agent development framework. In: Bordini, R.H. (ed.) Multi-agent programming : languages, platforms, and applications, p. 295. New York : Springer (2005)
4. Cao, L., Gorodetsky, V., Mitkas, P.A.: Guest editors' introduction: Agents and data mining. IEEE Intelligent Systems 24(3), 14–15 (2009)
5. Chaimontree, S., Atkinson, K., Coenen, F.: Best clustering configuration metrics: Towards multiagent based clustering. In: Proc 6^{th} Int. Conf. Advanced Data Mining and Applications (ADMA'10). pp. 48–59. Springer LNAI 6440 (2010)
6. Coenen, F., Leng, P., Ahmed, S.: T-trees, vertical partitioning and distributed association rule mining. In: Proc. 3^{rd} IEEE Int. Conf. on Data Mining. pp. 513–516. ICDM '03, IEEE Computer Society, Washington, DC, USA (2003)
7. Dasarathy, B.V.: Nearest neighbor (NN) norms: NN pattern classification techniques. IEEE Computer Society Press, Las Alamitos, California (1991)
8. Dasilva, J., Giannella, C., Bhargava, R., Kargupta, H., Klusch, M.: Distributed data mining and agents. Engineering Applications of Artificial Intelligence 18(7), 791–807 (2005)
9. FIPA: Communicative Act Library Specification. Tech. Rep. XC00037H, Foundation for Intelligent Physical Agents (2001), `http://www.fipa.org`
10. Forman, G., Zhang, B.: Distributed data clustering can be efficient and exact. ACM SIGKDD Explorations Newsletter 2, 34–38 (2000)

11. Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
12. Giannella, C., Bhargava, R., Kargupta, H.: Multi-agent systems and distributed data mining. In: Klusch, M., Ossowski, S., Kashyap, V., Unland, R. (eds.) Cooperative Information Agents VIII. Lecture Notes in Computer Science, vol. 3191, pp. 1–15. Springer Berlin / Heidelberg (2004)
13. Kargupta, H., Chan, P. (eds.): Advances in Distributed and Parallel Knowledge Discovery. MIT Press, Cambridge, MA, USA (2000)
14. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent based architecture. In: Proceedings the Third International Conference on the Knowledge Discovery and Data Mining. pp. 211–214. AAAI Press (1997)
15. Kiselev, I., Alhajj, R.: A self-organizing multi-agent system for online unsupervised learning in complex dynamic environments. In: Proc. 23^{rd} AAAI Conference on Artificial Intelligence. pp. 1808–1809. AAAI Press (2008)
16. Klusch, M., Lodi, S., Moro, G.: Agent-based distributed data mining: The KDEC scheme. In: Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science). vol. 2586, pp. 104–122 (2003)
17. Klusch, M., Lodi, S., Moro, G.: The role of agents in distributed data mining: Issues and benefits. In: IAT '03: Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology. p. 211. IEEE Computer Society, Washington, DC, USA (2003)
18. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proc. 5^{th} Berkeley Symposium on Mathematical Statistics and Probability. pp. 281–297 (1967)
19. McBurney, P., Parsons, S., Wooldridge, M.: Desiderata for agent argumentation protocols. In: Castelfranchi, C., Johnson, W.L. (eds.) Proc. 1^{st} Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2002). pp. 402–409. ACM Press: New York, USA, Bologna, Italy (2002)
20. Moemeng, C., Gorodetsky, V., Zuo, Z., Yang, Y., Zhang, C.: Agent-based distributed data mining: A survey. In: Cao, L. (ed.) Data Mining and Multi-agent Integration. pp. 47–58. Springer US (2009)
21. Park, B.H., Kargupta, H.: Distributed data mining: Algorithms, Systems, and Applications. In: Data Mining Handbook. pp. 341–358. IEA (2002)
22. Provost, F.: Distributed data mining: Scaling up and beyond. In: In Advances in Distributed and Parallel Knowledge Discovery. pp. 3–27. MIT Press (1999)
23. Rao, M.: Clustering analysis and mathematical programming. Journal of the American statistical association 66(345), 622–626 (1971)
24. Reed, J.W., Potok, T.E., Patton, R.M.: A multi-agent system for distributed cluster analysis. In: Proc. 3^{rd} Int. Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'04) W16L Workshop - 26th International Conference on Software Engineering. pp. 152–155. IEE, Edinburgh,Scotland, UK (2004)
25. Younis, O., Fahmy, S.: Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach. In: INFOCOM 2004. 23^{rd} Annual Joint Conf. of the IEEE Computer and Communications Societies. vol. 1, pp. 629–640 (2004)
26. Zaki, M.J., Ho, C.T. (eds.): Large-Scale Parallel Data Mining, vol. 1759. Springer Verlag (2000)
27. Zaki, M.J., Pan, Y.: Introduction: Recent developments in parallel and distributed data mining. Distributed Parallel Databases 11, 123–127 (2002)