Distributed Clustering for Group Formation and Task Allocation (Extended Abstract)

Daniela Scherer dos Santos and Ana L. C. Bazzan
PPGC – Universidade Federal do Rio Grande do Sul
Caixa Postal 15064, CEP 91501-970, Porto Alegre, RS, Brazil
bazzan@inf.ufrgs.br

ABSTRACT

Most clustering methods rely on central data structures and/or cannot cope with dynamically changing settings. However, issues related to the current use of Internet resources (distribution of data, privacy, etc.) require new ways of dealing with data clustering. In multiagent systems this is also becoming an issue as one wishes to group agents in an efficien way and according to some features of the environment. In this paper we briefl discuss how a distributed clustering algorithm that is inspired by swarm intelligence techniques is used in problems of task allocation.

Categories and Subject Descriptors

I.2.11 [Artificial Intelligence]: Distributed Artificia Intelligence—*Multiagent systems*

General Terms

Algorithms

Keywords

Multiagent Task Allocation, Distributed Clustering, Biologically-inspired approaches and methods, Swarm-Intelligence

1. INTRODUCTION

The allocation of tasks to groups of agents is necessary when tasks cannot be performed by a single agent or when an agent cannot perform them efficientl . Various groups of agents may have different degrees of efficien y regarding task performance due to each member's suitability to execute a particular task. Moreover, agents may not have full knowledge of others' capabilities nor of tasks' demands. Thus forming groups of agents according to some criteria to execute those tasks in an efficien way is a non-trivial problem. Grouping agents based on similar or complementary characteristics can be viewed as a clustering problem and is the main focus of our work. The simplest method for clustering, the k-means algorithm, requires information about the number of groups in the data. This poses a problem in applications where this information is not known a priori or changes dynamically. Other classical as well as ACO-based methods rely on data structures that must be accessed and modifie at each step of the operation, thus creating a single point of failure.

Cite as: Distributed Clustering for Group Formation and Task Allocation (Extended Abstract), Daniela Scherer dos Santos and Ana L. C. Bazzan, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 1429-1430

Our algorithm, *bee clustering* [4], is based on recruitment observed among honey bees. This recruitment is performed by dancing, during which a bee communicates to other bees the directions, distance, and quality of the food source.

2. SWARM INTELLIGENCE-BASED APPROACH

Honey bees collectively select the best nectar source available using simple behavioral rules. In the process of foraging, bees have three possible behaviors: to dance to communicate the quality of the food source to other bees, trying to recruit them to that food source; to continue foraging without recruiting other bees; to abandon the food source and go to the area inside the hive called dance floo to observe dancing bees and select another food source.

Using this as a metaphor, in *bee clustering*, each bee represents an agent (which has a set of attributes) that needs to be grouped. These agents have only a limited knowledge: they only know about the agents that are placed in their groups and they cannot remember their past groups, i.e. they have no memory.

In *bee clustering* agents may be in one of the following states: *v*, *w*, and *d*. When in state d an agent is dancing to recruit other agents to join its group. State v means that an agent is visiting the agent which is inviting it, while state w means that an agent is watching the dancers to randomly choose one to visit. At the beginning the state of all agents (bees) is v.

During the clustering process agents need to make a couple of decisions. Let us assume that agent i starts the clustering process and agent j is the agent that i is visiting. The state of i is v and a decision needs to be made about whether or not to abandon j. If i abandons j, its state changes to w and it will watch those now dancing for different groups. Then i randomly chooses a bee that is dancing and visit it. On the other hand, if i decides not to abandon j, then i must decide if it changes to the group of j, or continues in its current group. In both cases, the state of i changes to d and its next action will be to dance to recruit for its current group or for j's group. Following this, i needs to decide if it continues dancing or not. If not, the process restarts and i visits j.

Summarizing, the possible decisions of the agents are: to abandon an agent or not; to change or not to the group of the visited agent; to continue or not dancing to recruit other agents for a group; and to visit a dancer or not.

To ground these decisions we use three well known mathematical models. The intensity of the dance and the rate of abandonment are computed by Camazine and Sneyd's mathematical model [3]. The response threshold model described in [2] is used mainly to compute the probabilities of abandoning an agent, of visiting a dancer, and of continuing dancing. Finally, the difference utilities approach is used to let agents compute the utility of a group with and without itself, so that an agent can decide whether or not it changes groups. We now explain these methods in more details.

In the *bee clustering* algorithm agents visit each other and form groups thanks to invitations made by other agents that are dancing. The time an agent remains dancing is key. If it is too long, the model might not work because all agents will be dancing at the same time; if agents dance for too short a time, the algorithm converges to a clustering with a high number of small groups. To control this time, agents use the response threshold model (details in [2]). Agents in *bee clustering* use this model to decide about whether or not to continue dancing, depending on the quality of the agent's group. If the group has a good quality, then the agent has a tendency to continue dancing for its group. If, in contrast, the group has low quality then the agent stimulus decreases potentially leading it to stop dancing.

Another issue is that agents must decide whether they change groups. This is an essential decision for a good clustering result. Because we do the clustering in a distributed way, no one is in charge of maximizing the global utility. Rather, each agent is acting locally. In [1] agents need to maximize a global utility computed over the difference between the initial and the fina clustering. For this to be done in a distributed way, the computation of the global utility involves agents broadcasting what they believe the fina clustering will be. We remark that in [1], agents do broadcasting (while in *bee clustering* agents only know about the cluster they belong) and that the authors use this technique to compute an ensemble of clusters, which is a different problem than the one we deal with here.

Bee clustering uses the utility difference approach to help agents make a decision about whether or not to change groups. If the utility of agent i's group is better without i, then i abandons its group and changes to the group of the agent it is visiting. Otherwise it remains in its group. Notice that the agent is able to calculate only the utility of the group it is currently in.

The main steps of *bee clustering* are roughly as described next. Let us assume that agent i is in state v. This means that i needs to decide if it will abandon the agent j it is visiting. This probabilistic decision is based on the similarity between i and j. Similarity is domain-dependent quantity; thus it must be tailored for the specifi multiagent task allocation problem one is addressing. If i abandons j, i changes its state to w indicating that the next action is to observe other agents whose state is d. On the other hand, if i does not abandon j, then it needs to decide if it leaves its group C_i to join the group of agent j, C_j. To decide this, i calculates the utility of group C_i with its participation ($U(C_i)$) and without it, $U(C_{-i})$.

Next, agent i compares both utilities. If $U(C_i)$ is higher than $U(C_{-i})$, indicating that the group utility of i is better with its participation, i remains in its group C_i; otherwise i changes to C_j of agent j. In both cases i starts dancing to recruit other agents to its group. Thus, i changes its state to d.

Then, if the state of i is d, this means that i needs to verify if it continues dancing or not. With a probability given by the response threshold model, the state of i remains d. Otherwise i stops the dance and changes to state v, indicating that it will visit agent j again.

If the state of agent i is w, this means that i is observing those agents in state d and will then randomly choose an agent j. Next, i decides whether or not to accept the invitation of j. If the invitation is not accepted, i will choose another agent that is dancing and decide whether or not to accept the invitation. When an invitation is accepted, the state of i changes to v and the loop restarts.

3. GROUP FORMATION VIA CLUSTERING IN MULTIAGENT SYSTEMS

Bee clustering was already employed in clustering problems using standard datasets from UCI (e.g. Iris) [4]. In the present paper we discuss the main steps to use this algorithm in a task allocation problem. Be a set $\mathcal{D}$ of agents and a set τ of tasks. Each $\tau_j \in \tau$ has an attribute vector, whose values may change over time. Each agent $i \in \mathcal{D}$ perceives a set of tasks and computes the Euclidean distance between its capabilities and each perceived task. For instance, the Euclidean distance is zero if both i and the perceived task are in the same location, and i is fully capable of performing this task.

Each i starts in state v thus they start visiting each other computing the probabilities discussed in the previous section, as well as the Euclidean distances, utilities, etc. until groups of similar agents are formed to deal with the perceived tasks. Agents then start executing the tasks.

In a scenario in which the attribute values of task and/or the capabilities of the agents change, the task allocation must be reviewed from time to time or upon given events. Such events can be for instance the utility of the group dropping below a given threshold, which could triggers a re-clustering.

4. CONCLUSIONS

In this paper we have briefl discussed how a swarm intelligence-based approach to distributed clustering (*bee clustering*) can be used in problems of task allocation. Here, agents need only a limited knowledge about other agents placed in their groups. Also, the clustering is performed in a distributed way, as it allows agents to group and regroup only based on the set of tasks they perceive locally, and on the characteristics of the agents in the same group. In the full version of this paper, a detailed example is given in which the 3 kinds of agents in the robocup rescue simulator (Kobe map, with 12 ambulance teams, 20 fir brigades, and 16 police forces) are grouped and assigned to hundreds of interrelated tasks (e.g. fir fighting civilian rescue).

Acknowledgments

This research was partially supported by the Air Force Offic of Scientifi Research (AFOSR) (grant number FA9550-06-1-0517) and by the Brazilian National Council for Scientifi and Technological Development (CNPq).

5. REFERENCES

[1] A. Agogino and K. Tumer. Efficien agent-based cluster ensembles. In P. Stone and G. Weiss, editors, *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, AAMAS '06*, pages 1079–1086, New York, NY, USA, 2006. ACM.

[2] E. Bonabeau, G. Theraulaz, and M. Dorigo. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York, USA, 1999.

[3] S. Camazine and J. Sneyd. A model of collective nectar source selection by honey bees: Self-organization through simple rules. *Journal of Theoretical Biology*, 149(4):547–571, April 1991.

[4] D. S. d. Santos and A. L. C. Bazzan. A biologically-inspired distributed clustering algorithm. In *Proc. of the 2009 IEEE Swarm Intelligence Symposium*, pages 160–167, Nashville, 2009. IEEE.