Applied Soft Computing 12 (2012) 2123–2131

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Distributed clustering for group formation and task allocation in multiagent systems: A swarm intelligence approach

Daniela Scherer dos Santos[a], Ana L. C. Bazzan[b,*]

[a] Universidade Luterana do Brasil, Rua Martinho Lutero, 301, Cachoeira do Sul, RS, Brazil
[b] PPGC/Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS), Caixa Postal 15064, 91.501-970 Porto Alegre, RS, Brazil

ARTICLE INFO

Article history:
Received 8 July 2010
Received in revised form 27 February 2012
Accepted 4 March 2012
Available online 16 March 2012

Keywords:
Multiagent systems
Swarm intelligence
Distributed clustering
RoboCup Rescue

ABSTRACT

Most clustering methods rely on central data structures and/or cannot cope with dynamically changing settings. Besides, these methods need some hints about the target clustering. 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 according to some features of the environment in order to have agents accomplishing the available tasks in an efficient way. In this paper we discuss the application of a clustering algorithm that is inspired by swarm intelligence techniques such as organization of bee colonies and task allocation among social insects. This application involves a complex task allocation scenario, the RoboCup Rescue, where tasks with different characteristics must be allocated to agents with different capabilities. Our results have shown that clustering agents is effective in this scenario as agents act in a more coordinated way.

1. Introduction

Agents in a multiagent environment may need to cooperate in order to perform tasks. 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 efficiently. Various groups of agents may have different degrees of efficiency when performing different tasks due to each member's suitability to execute a particular task.

In multiagent systems, this issue is even more complicated given that agents do not have full knowledge about others' capabilities and about the demands other tasks pose. Thus forming groups of agents according to some criteria to execute those tasks in an efficient way is a non-trivial problem. To illustrate this, let us think about a disaster scenario composed of many agents (fire brigades that may extinguish fires, police forces that may clear roads, ambulance teams that may carry civilians to safe places). These agents must be able to group together on their own in a distributed fashion in order to perform their activities in teams. This way they may get the best possible result regarding, for instance, rescuing victims in a disaster scenario. 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 clustering method, the *k*-means algorithm, requires information about the number of groups in the data and is known to converge to local minima. Moreover, the performance of this algorithm depends strongly on the information one has about the data regarding the possible number of groups, which poses a problem in applications where this information is not known a priori or changes dynamically.

Other classical as well as ACO-based methods have been usually developed in a centralized fashion, requiring that data be located at a single place. This means that these algorithms cannot be applied in the case of distributed data sets. However, due to geographical, time, or space constraints, data is increasingly distributed among many locations. Also, those algorithms rely on data structures that must be accessed and modified at each step of the operation, thus creating a single point of failure. Furthermore, issues related to the current use of Internet resources (dispersion of data, privacy, etc.) require new ways of dealing with data clustering.

Finally, another possible clustering approach includes the agglomerative hierarchical clustering algorithm based on the linkage metric average link. The algorithm starts with the finest partitioning possible and, in each iteration, merges the two least distant clusters. The distance between two clusters is computed as the average dissimilarity between all possible pairs of data elements within these two clusters. Hierarchical clustering methods are thought to give higher quality solutions than partitioning methods. However, their runtime increases quadratically and results heavily depend on the linkage metric used. Also, the derivation of appropriate stopping criteria can be difficult, if the correct number of clusters is not known [1].

In short, as we review in the next section, existing methods have drawbacks that prevent their use in distributed and dynamic

* Corresponding author.
E-mail address: daniela.scherer@gmail.com (D.S. dos Santos).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2012.03.016

applications. Thus the main contribution of the present paper is the swarm-based approach that solves the clustering problem in a decentralized fashion. In multiagent systems this may be useful since agents lack full knowledge. Also, our approach does not require initial information about number of classes, number or size of partitions.

Our algorithm, *bee clustering*, relies on recruitment observed among honey bees. It was proposed [2] and tested using datasets such as Iris and Yeast (from the University of California, Irvine, repository). In the present work, we want to test this algorithm in a quite different type of problem, namely group formation for task allocation in environments that have two main characteristics: data is distributed and attributes change dynamically.

The basis of the algorithm is foraging. In nature, bees travel far away from the hive to collect nectar. They return to the hive with nectar and information about the nectar source to recruit other bees to this food source. This recruitment is performed by dancing, during which a bee communicates to other bees the directions, distance, and quality of the food source. This metaphor is used for clustering and task allocation.

We empirically test our algorithm in a well-known testbed for task allocation in multiagent systems, namely the RoboCup Rescue scenario [3]. The aim is to form clusters or groups of agents with similar or complementary expertise and abilities. Given a set of agents and a set of tasks perceived by these agents, they try to recruit other agents to group and perform the same task. This is done taking into account the characteristics of the agents regarding tasks features. In the rescue scenario the main features are: the seriousness/urgency of the problem/task to be solved, and distance between agents and the tasks that need to be performed. Three main kinds of agents – fire brigades, police forces, and ambulance teams – need to fulfill tasks such as extinguishing fires, unblocking streets, and rescuing civilians, respectively. However most of the tasks are better accomplished if allocated to groups, instead of to single agents (e.g., a fire cannot be controlled by a single fire brigade and risks propagating to a point that not even all fire brigades can cope with it). Assigning agents to tasks is of course a non-trivial problem that has received a great deal of attention; in this paper we restrict ourselves to clustering and classical approaches.

The rest of the paper is organized as follows: Section 2 presents clustering approaches that are related to ours; Section 3 presents the RoboCup Rescue scenario. In Section 4 the details of the *bee clustering* algorithm are given. Experimental results are presented in Section 5, while Section 6 discusses the conclusions and future work.

2. Related work

In the next subsections we present some works that are related to clustering (basis of our approach) as well as to task allocation (our target scenario).

Several approaches exist, which deal with the clustering problem and are inspired by social insects. Some are [4–8]; see also [9] for a brief review of the literature. Most of them are inspired by two main behaviors observed in ant colonies: ant-foraging and corpse clustering. Next, we explain these behaviors and give some examples of clustering algorithms that are inspired by them, and are related to the *bee clustering* algorithm. We start with ant colony optimization (ACO) based approaches, then present others that are based on cemetery organization, some algorithms for distributed clustering, and finally other algorithms that are based on other facets of swarm intelligence. We notice that ACO is one of the subareas of swarm-intelligence but our approach itself follows task allocation in colonies of social insects rather than ACO, once this relates to ant foraging behavior, i.e., ants depositing pheromone and foragers following trails.

2.1. Clustering approaches based on ant foraging behavior

In the algorithm presented by Shelokar et al. [6], ants visit data objects one by one and select clusters for data objects by considering pheromone information. Ants use a pheromone matrix which guides other ants toward the optimal clustering solution. After generating a population of R trial solutions, a local search is performed to further improve the fitness of these solutions. The pheromone matrix is then updated according to the quality of solutions produced by the agents.

It is easy to deduce that this and other ACO-based algorithms depend on a global pheromone matrix that is a single point of failure. Besides, it does not have a multiagent flavor.

2.2. Approaches based on corpse clustering

Another class of clustering algorithms is inspired in the way real ants clean their nests and organize dead bodies in their colonies. Here, in contrast to ACO, pheromones are not used. Rather, the environment itself provides the stigmergic component. Lumer and Faieta [5] proposed an ant-based data clustering algorithm that associates a position in a toroidal grid with each one of the data items to be clustered. The positions of these items, as well as those of the agents moving them around, are initialized randomly. These agents have a sorting behavior based on local rules. The number of moves an agent can perform is defined a priori. Agents try to pick up or drop objects on the two-dimensional board according to a local measure of similarity.

The approach proposed by Yand and Kamel [7] uses a hypergraph to combine clustering produced by three colonies. Each ant colony randomly projects data objects onto a plane and the clustering process is done by ants picking up or dropping down objects with different probabilities. The same authors have also developed an extended version, where they have added a *centralized* element to compute the clustering: a queen ant agent. This agent receives the results produced by all colonies, calculates a new similarity matrix, and broadcasts to all other colonies. Each colony re-clusters the data using the new information received.

The main problem of these approaches regards the nature of the algorithm's output. They do not generate an explicit partitioning, but a spatial distribution of the data elements. While this may contain clusters that are obvious to a human observer, an evaluation of the clustering performance requires the retrieval of these clusters, and it is not trivial to do this without human interaction.

Apart from techniques inspired on ant foraging and corpse clustering, a number of other swarm-intelligence based behaviors have been used for clustering.

A probabilistic ant-based clustering algorithm (PACE) for distributed databases was proposed, which uses chemical recognition, a behavior in which an ant identifies another group of ants from the same colony using a distinctive odor that is unique to each colony. This is used to identify and form a group of ants that carry related data. Its main characteristic is the formation of numerous zones in various distributed sites based on the user's query to a distributed database. An extended version is presented in [10] (I-PACE), where the authors propose the introduction of weights for individual or groups of data items in each zone according to their relevance to the queries and a familial pheromone trail as part of an ant odor identification model. The aim is to reduce the convergence time and to improve the quality of clustering. PACE and I-PACE were developed in a distributed way; however they rely on information that needs to be given a priori.