STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVI**, Number 2, 2011

A CONTEXT-AWARE ASM-BASED CLUSTERING ALGORITHM

RADU D. GĂCEANU AND HORIA F. POP

Abstract. We present a context-aware algorithm based on ASM (Ants Sleeping Model) in order to resolve the clustering problem. In the ASM model data is represented by agents placed in a two dimensional grid. The agents will group themselves into clusters by making simple moves in their environment according to some local information the parameters being selected and adjusted adaptively. In order to avoid the agents to be trapped in local minima, they are also able to directly communicate with each other. Moreover, the agent moves are expressed by fuzzy IF-THEN rules and hence hybridization with a classical clustering algorithm is needless. Being aware of the context the agents can easily adapt when the environment changes.

1. Introduction

Several clustering algorithms exist each with its own strengths and weaknesses. Some algorithms need an initial estimation of the number of clusters (k-means, fuzzy c-means); others could often be too slow (agglomerative hierarchical clustering algorithms). Ant-based clustering algorithms often require hybridization with a classical clustering algorithm such as k-means. We propose an algorithm based on ASM (Ants Sleeping Model) [1, 3] in order to resolve the clustering problem. In order to avoid the agents to be trapped in local minima, they are able to directly communicate [2] with each other. Furthermore, the agent moves are expressed by fuzzy IF-THEN rules [5] and hence hybridization with a classical clustering algorithm is needless. Being aware of the context, the agents can adapt when changes in the environment occur; so the items from the dataset can change at runtime and the agents are able to spot these changes leading to a result based on the updated dataset. Dealing with changes in the environment becomes a necessity in data streams, real-time systems, wireless sensor networks. The rest of the paper is structured as follows. Section 2 presents a motivation of this paper outlining the

Received by the editors: March 14, 2011.

2010 *Mathematics Subject Classification.* 68T05, 68T10.

1998 *CR Categories and Descriptors.* I.5.3 [**Pattern recognition**]: Clustering – *Algorithms*.

Key words and phrases. fuzzy, clustering, ant.

relevance of the idea together with the related work. The proposed model is described in Section 3 and Section 4 presents a case study. The closing Section 5 contains the conclusions and future work.

2. Motivation and related work

Context-aware systems could greatly change the way we interact with the world — they could anticipate our needs and advice us when taking some decisions. In a changing environment context-awareness is undoubtedly beneficial. In this section we present some papers which we consider relevant for our clustering approach. In [1] an ant-based clustering algorithm is presented. It is based on the ASM (Ants Sleeping Model) approach. In ASM, an ant has two states on a two-dimensional grid: active state and sleeping state. When the artificial ant's fitness is low, it has a higher probability to wake up and stay in active state otherwise it would sleep. However, by using local information only the risk of getting trapped into local optimum solutions exists. In [2] a Stigmergic Agent System (SAS) combining the strengths of Ant Colony Systems and Multi-Agent Systems concepts is proposed. The agents from the SAS are using both direct and indirect communication. However, as showed in [5], most ant-based algorithms can be used only in a first phase of the clustering process because of the high number of clusters that are usually produced. In a second phase a k-means-like algorithm is often used. In [5], an algorithm in which the behaviour of the artificial ants is governed by fuzzy IF-THEN rules is presented. Like all ant-based clustering algorithms, no initial partitioning of the data is needed, nor should the number of clusters be known in advance. The ants are capable to make their own decisions about picking up items. Hence the two phases of the classical ant-based clustering algorithm are merged into one, and k-means becomes superfluous. The algorithm from [3] is extended in this paper by the idea of context-awareness, the agents being here able to detect changes in the environment and adjust their moves accordingly.

3. Proposed model

The skeleton of our approach is based on the ASM-like algorithm from [1] embellished with features from [2, 5, 3]. In the ASM model each data item is represented by an agent and due to the need for security they ants are constantly choosing a more comfortable environment to sleep in. The ants feel comfortable among individuals having similar characteristics. While it doesn't find a suitable position to have a rest, it will actively move around to search for it and stop when he finds one; when it is not satisfied with his current position, he becomes active again. The definitions 1-5 related to the grid, the neighbourhood, agent fitness, agent activation probability etc are taken from [1] and will not be repeated here due to space limitations. At the beginning of the algorithm, the agents are randomly scattered on the grid in active state. In each loop, after the agent moves to a new position, it will recalculate its current fitness f and the activation probability p_a so as to decide whether it needs to continue moving. If the current p_a is small, the agent has a lower probability of continuing moving and higher probability of taking a rest at its current position. Otherwise the agent will stay in active state and continue moving. In the end, similar agents will be grouped together in small areas while different types of agents will be located in separate areas.

Definition 1. We use the following definition for the fitness in this paper:
$f(agent_i) = \frac{1}{(2s_x+1)(2s_y+1)} \sum_{agent_j \epsilon N(agent_i)} \frac{\alpha^2}{\alpha^2 + disim(agent_i, agent_j) de(agent_i, agent_j)}$
$de(agent_i, agent_j)$ represents the euclidian distance between the agents on the grid
$disim(agent_i, agent_j)$ denotes the disimilarity between the two agents.

```
Algorithm Clustering is
    initialize parameters $\alpha, \lambda, t, s_x, s_y$
    for each agent do
        place agent at randomly selected site on the grid
    endFor
    while (not termination)
        for each agent do
            compute agents fitness and activate probability $p_a$ according
                to definitions 5, 6 and 7
            r ← random (0,1)
            if ($r < p_a$) then activate agent and adaptively move based
                on the context to a site in the neighbourhood
                using fuzzy IF-THEN rules
            else stay at current site and sleep
            endif
        endFor
        adaptively update parameters $\alpha, \lambda, t, s_x, s_y$
    endWhile
endAlgorithm
```

The agents decide upon the way they move on the grid according to their similarity with the neighbours, using fuzzy IF-THEN rules. Thus two agents can be similar (S), different (D), very different (VD). If two agents are similar they would get closer to each other. If they are different or very different they will get away from each other. The number of steps they do each time they move depend on the similarity level. So if the agents are VD they would jump many steps away from each other; if they are D they would jump less steps away from each other. In the end the ants which are S will be in the same cluster. The similarity computation is taking into account the actual structure of the data or the data density from the agent's neighbourhood; a bigger change from one agent to another translates into a certain similarity which then affects the agent's movement on the grid. The parameter α is the average distance between agents and this changes at each step further influencing the fitness function. The parameter λ influences the agents' activation pressure and it may decrease over time. The parameter t is used for the termination condition which could be something like $t < t_{max}$. The parameters s_x, s_y, the agent's vision limits may also be updated in some situations.

We outline that the result of the algorithm is not a fuzzy partition. However, in order to perform a deeper analysis, the membership degree of each item to the obtained clusters will be considered and a representative for each cluster will be chosen. So the problem of computing the similarity degree between the item and the cluster is reduced to considering the similarity degree between the item and the chosen representative. Other fuzzy clustering approaches could perform similar operations at each step of the clustering process; our approach does this only once at the end of the clustering process so we consider our approach an improvement from this point of view. For finding these cluster representatives we try to simulate the real-life process in which the data analyst would point such representatives with the mouse. Of course that if he deals with a high density cluster then he normally can only make a rough approximation. We can refine his choice by proposing an item in the neighbourhood which has the highest fitness. So we randomly choose a candidate representative from each cluster and then replace it with the best fitted agent from a certain radius.

4. Case study

In order to test the algorithm in a real-world scenario, the Iris dataset [6] was considered. The data set contains 3 classes of 50 instances each, each class refering to a type of iris plant. There are 4 attributes plus the class: sepal length in cm, sepal width in cm, petal length in cm, petal width in cm, class (Iris Setosa, Iris Versicolour, Iris Virginica). The last 2 attributes (petal length in cm and petal width in cm) are highly correlated according to [6]. However we do not dismiss any of these attributres because we would like to keep as much of the data unchanged. We do however scale the data to the interval $[0, 1]$. This dataset is appropriate for rather testing classification, but it was prefered for clustering too because the class attribute is given and hence there is a way to evaluate the algorithm. According to the Iris dataset [6], items ranging from 0 to 49 belong to the first class, items ranging from 50 to 99 belong to the second class and items ranging from 100 to 149 belong to the third class. Comparing the final grid configuration of all agents (not listed here due to space limitations) with the information from [6], it appears that the following clusters contain some misclassifications:

- $Cluster1$ (items $0 - 49$): no misclassifications
- $Cluster2$ (items $50 - 99$): 106, 119, 23, 43
- $Cluster3$ (items $100 - 149$): 86, 70, 83, 52, 56

So it appears that the algorithm has misclassified nine items. However, it is unclear why should items 106 and 119 from Figure 1a be considered misclassifications. According to our similarity measures they have a 0.20 and a 0.18 similarity with the representative item 90. This makes them S ($Similar$) with this item. The membership degree with $Cluster2$ suggests that these items belong to this cluster. However the membership degree with $Cluster3$ is also high. The highest membership degree is with $Cluster2$ though and because of this it could be claimed that the items are actually correctly classified with respect to the considered metric. However we believe that items 106 and 119 cannot be considered to strictly belong either to $Cluster2$ or to $Cluster3$ as they are clearly at the border of the two clusters so they belong to both. In this case we also believe that they should not be regarded as misclassifications. After a similar reasoning is applied to the items from Table 1b, it turns out that only items 23 and 43 are really classification errors.

Cluster2 — RepresentativeId (90)				
MisclassificationId	Similarity	C1	C2	C3
106	0.20	0.0	1.0	0.9
119	0.18	0.0	1.0	0.91
23	0.50	1.0	0.0	0.0
43	0.51	1.0	0.0	0.0

(A) Cluster2, RepresentativeId (90)

Cluster3 — RepresentativeId (120)				
MisclassificationId	Similarity	C1	C2	C3
86	0.34	0.0	0.98	0.91
70	0.26	0.0	0.91	1.0
83	0.32	0.0	1.0	0.98
52	0.32	0.0	0.95	0.92
56	0.31	0.0	0.96	0.94

(B) Cluster3, RepresentativeId (120)

FIGURE 1. Miclassifications

For benchmarking reference purposes, the k-means algorithm from [4] was evaluated on both datasets, with three misclassifications reported on the custom dataset and 17 misclassifications on the Iris dataset. Compared to the approach from [3], the dataset can be changed at any time and the agents will react on this change, they will operate on the updated dataset.

The ability to handle the dataset changes at run-time is an important feature in dynamic environments where changes occur over time independent form the agent's actions. An agent from such a system is iteratively making a decision based on the context without the knowledge of the future changes in the environment. Planning systems in general need to deal with changes in the environment. For example a portfolio management system clearly needs to handle changes, the stock market being very dynamic. Also, in large healthcare systems, when an update in medical analysis occurs that perhaps corrects previous entries, it could be impractical to recompute the entire model. One could be tempted to judge the quality of algorithms operating in a static environment with the quality of the algorithms operating in a dynamic environment. When such a comparison is done it should be clear that in a static environment all information is available from the beginning and the problem of adapting to changes in the environment is a completely different problem.

5. Conclusions and future work

The algorithm we have presented is based on the adaptive ASM approach from [1]. The major improvement is that, instead to moving the agents at a randomly selected site, we are letting the agents choose the best location. Agents can directly communicate with each other — similar to the approach from [2]. In [5], the fuzzy IF-THEN rules are used for deciding if the agents are picking up or dropping an item. In our model we are using the fuzzy rules for deciding upon the direction and length of the movement. Compared to [3] the agents are able to adapt their movements if changes in the environment would occur. More experiments with other clustering methods using larger, real-world data sets are on-going.

Acknowledgement

The authors wish to thank for the financial support provided from programs co-financed by The Sectorial Operational Programme Human Resources Development, Contract POSDRU 6/1.5/S/3 "Doctoral studies: through science towards society".

References

[1] L. Chen, X. H. Xu, and Y. X. Chen. An adaptive ant colony clustering algorithm. In *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on, Vol. 3*, pages 1387–1392, 2004.

[2] C. Chira, D. Dumitrescu, and R. D. Găceanu. Stigmergic agent systems for solving NP-hard problems. *Studia Informatica*, Special Issue KEPT-2007: Knowledge Engineering: Principles and Techniques (June 2007):177–184, June 2007.

[3] R. D. Găceanu and H. F. Pop. An adaptive fuzzy agent clustering algorithm for search engines. In *MACS2010: Proceedings of the 8th Joint Conference on Mathematics and Computer Science*. Komarno, Slovakia, 2010.

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[5] S. Schockaert, M. D. Cock, C. Cornelis, and E. E. Kerre. Fuzzy ant based clustering. In *Ant Colony Optimization and Swarm Intelligence, 4th International Workshop (ANTS 2004), LNCS 3172*, pages 342–349, 2004.

[6] http://archive.ics.uci.edu/ml/datasets/iris.

Babeş-Bolyai University, Department of Computer Science, 1 M. Kogalniceanu Street, 400084, Cluj-Napoca, Romania

E-mail address: {rgaceanu,hfpop}@cs.ubbcluj.ro