Applied Soft Computing 9 (2009) 632–640

Contents lists available at ScienceDirect

Applied Soft Computing

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

Hyperbox clustering with Ant Colony Optimization (HACO) method and its application to medical risk profile recognition

G.N. Ramos [a,*], Y. Hatakeyama [b], F. Dong [a], K. Hirota [a]

[a] Department of Computational Intelligence and Systems Science, Tokyo Institute of Technology, Yokohama, Japan
[b] Center of Medical Information Science, Medical School, Kochi University, Nankoku city, Japan

ARTICLE INFO

Article history:
Received 24 August 2006
Received in revised form 31 August 2008
Accepted 11 September 2008
Available online 26 September 2008

Keywords:
Clustering
Ant colony
Hyperbox
Optimization
Pattern recognition

ABSTRACT

A clustering method, called HACO (Hyperbox clustering with Ant Colony Optimization), is proposed for classifying unlabeled data using hyperboxes and an ant colony meta-heuristic. It acknowledges the topological information (inherently associated to classification) of the data while looking in a small search space, providing results with high precision in a short time. It is validated using artificial 2D data sets and then applied to a real medical data set, automatically extracting medical risk profiles, a laborious operation for doctors. Clustering results show an improvement of 36% in accuracy and 7 times faster processing time when compared to the usual ant colony optimization approach. It can be further extended to hyperbox shape optimization (fine tune accuracy), automatic parameter setting (improve usability), and applied to diagnosis decision support systems.

1. Introduction

The Ant Colony Optimization (ACO) meta-heuristic [4,5] uses a population of agents (ants) guided by an autocatalytic process directed by a greedy force for discrete combinatorial optimization problems. Previous studies applied optimization techniques to clustering problems, for example [11,12,15,18], aim to minimize a fitness function, usually a distance measure, relating the data to a cluster centroid.

Minimizing the distance between data points and cluster centroids is a logical approach, yet it does not necessarily provide topological information of the data. The mentioned methods have provided reasonable minimum distance results, though they lack the information that is, in many cases, essential for extracting intuitive knowledge from the data. When used in pattern recognition or classification applications, despite the fact that the fitness criterion is satisfied, result accuracy (misclassification) may be an issue.

The Hyperbox clustering with Ant Colony Optimization method (HACO) is proposed for clustering unlabeled data by placing hyperboxes in the feature space optimized by the ACO. It applies an optimization technique combined to a well-known local search algorithm to the clustering problem, acknowledging the topological information of the data, if available.

Hyperboxes are placed in the search space using the ACO meta-heuristic and then clustered using the Nearest-Neighbor (NN) method. The number of hyperboxes to perform the search with is usually smaller than (or, in the worst case, equal to) the number of samples, which means that the search can be done in less iterations, making HACO a fast classifier.

HACO is applied to three computer-generated 2D data sets which have significant topological information for validation as a classification method, considering speed and accuracy. It is then applied to a Human Papillomavirus (HPV) data set in order to identify probable infection profiles that may be used as a basis for preventive medical check-ups. It is compared to two well-established clustering methods and the usual ACO approach and the results show that the HACO method can be more effective than the others.

A brief description of ACO and definition of hyperboxes are presented in Section 2; Section 3 proposes and details the HACO method; clustering experiments and the results are analyzed in Section 4.

2. Description of ant colony optimization and hyperboxes

2.1. A review on ant colony optimization

The ACO meta-heuristic is population-based and can be readily applied to discrete combinatorial optimization problems. It makes an analogy of the way real ant colonies work to optimize combinatorial problems [4,5]. The basic idea is the synergy of

* Corresponding author.
E-mail addresses: ramos@hrt.dis.titech.ac.jp, ramos.at.titech@gmail.com (G.N. Ramos).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2008.09.004

applying multiple communicating agents to build a solution. Real ants communicate with each other by depositing pheromone on the trail between the food source and the nest [3]. The shorter the trail, the faster the ants will go through it and thus more pheromone will be deposited. Since ants have a high probability of following trails with higher pheromone deposition, the process reinforces itself [4,5].

This is a distinctive feature of ACO: the pheromone matrix works as dynamic memory, indicating how desirable a data object is to the solution [4], and thus mediates how one ant's behavior is determined by the previous ants [3–5]. The values are updated according to the quality of the solutions, so the process "remembers" good solutions and "forgets" bad ones. Similar agent-based applications have been used for data clustering, but such algorithms usually follow the Ant Cemetery approach [12], which provides no global control over the agents. This approach has been combined with Fuzzy C-Means [11] and K-Means [15] algorithms in order to improve the quality of results.

The main characteristics of ACO approach are positive feedback (improves speed of finding good solutions), distributed computation (avoids early convergence) and greedy heuristic (finds reasonable a solution early in the process) [4,5,7]. Due to such characteristics, however, it may be outperformed by specialized algorithms [4,5].

The ACO can be simplified in three basic procedures per iteration [4,5]: build solutions, local optimization (an optional step) and pheromone update. When applied to finding suitable data set partitions for clustering [18], i.e., dividing the data set into distinct classes represented by the clusters, ACO aims to minimize distances between the samples and the centroids.

Due to its inherent characteristics (flexibility and fast convergence), the ACO algorithm is a good approach for partition clustering. In this case, since the objective is to minimize the distance between data objects and the cluster centroids, it attempts to cluster the closest data objects. This is done by assigning clusters to each data object, and then calculating the distances. It may not produce the best results; however, depending on how the data is distributed on the feature space. This approach does not consider the topology of the feature space, which is inherently associated to classification processes [6,17]. For example, consider the data set shown in Fig. 1.

Intuitively, it is clear that the classes are distributed as one long curved-shaped cluster (the points with positive vertical coordinates) and one oval cluster (points with negative vertical coordinates), as in Fig. 1a. The ACO algorithm, however, defines a partition of the data in such way that it is divided into one elongated cluster and one larger cluster, as in Fig. 1b. The fitness of the solution in Fig. 1b is indeed better than the fitness of Fig. 1a; nevertheless it is clear that the solution lacks the topological information.

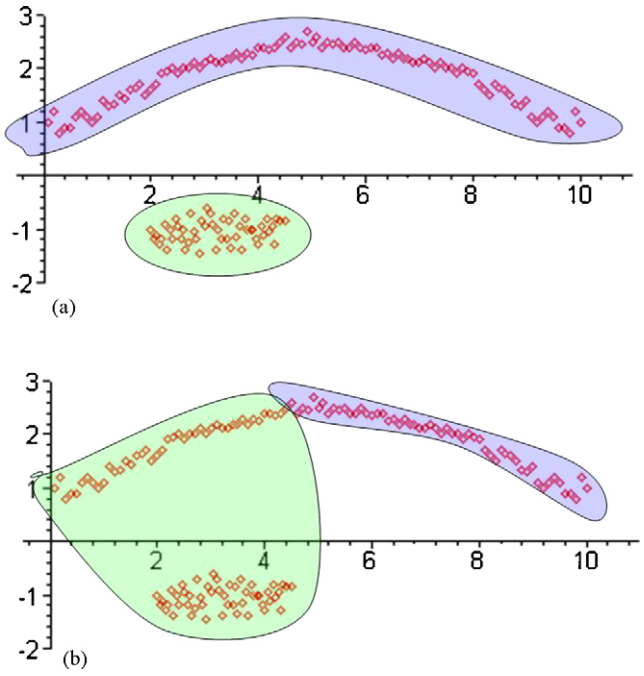

Fig. 1. Data partition examples: (a) intuitive partition; (b) ACO partition.

2.2. Describing hyperboxes

A hyperbox defines a region in an n-dimensional space [17,19,20] and is fully described by two vectors, usually its two extreme points: a_l which is the lower bound and b_l, the upper bound. Assuming an n-dimensional space of real numbers ($\Re^n$) and a hyperbox $H_l = (a_l, b_l)$, where $a_l \leq b_l$, a point y is said to be in H_l if

$$\begin{aligned} &H = \{H_1, H_2, \ldots, H_l, \ldots, H_C\}, \\ &H_l \subset \Re^n, \\ &y = \{y_1, y_2, \ldots, y_j, \ldots, y_n\}, \\ &y \in H_l \Rightarrow a_{lj} \leq y_j \leq b_{lj}, \quad a_l, b_l \in \Re^n, \end{aligned} \tag{1}$$

where C is the total number of hyperboxes and y_j is the jth attribute of y.

Using this definition it is necessary to have two points for a hyperbox; however, the objective of HACO is to group data objects that are near each other, which accounts to some problems on choosing this points. It is, therefore, more convenient to use a slightly different approach, which maintains the useful characteristics of a hyperbox. It can be defined by one point (in HACO, one data object x_i) and an n-dimensional vector D which defines the edge lengths for each attribute, as in $H_l = (x_i, D)$. Therefore, each hyperbox will define a region in the space around such data point, as follows:

$$\begin{aligned} &X = \{x_1, x_2, \ldots, x_i, \ldots, x_N\} \subset \Re^n, \\ &D = \{D_1, D_2, \ldots, D_j, \ldots, D_n\}, \\ &\forall y \in \Re^n, \\ &y \in H_l \Rightarrow x_{ij} - \frac{D_j}{2} \leq y_j \leq x_{ij} + \frac{D_j}{2}, \end{aligned} \tag{2}$$

where X is the set of data points with cardinality N, and D_k is the edge length for the jth hyperbox dimension.

Hyperbox classifiers can give straightforward interpretation for classification rules [17], such as "if $y \in [a_l, b_l]$ then y belongs to the class defined by H_l'', without calculating any distances. Also, if associated with a fuzzy membership function, they can be used as inputs for fuzzy min–max neural networks to be applied in classification [19] or clustering [20]. In these applications, data is assumed to be labeled and part of it is used for training.

It is possible to automatically determine shape patterns by grouping hyperboxes, and then define a class according to the specific characteristics. Since overlapping may occur but data objects are not allowed to belong to different classes (crisp clustering [10]), the proposed method requires that overlapping hyperboxes represent the same class. In other words, hyperboxes representing different classes must be disjoint.