Ant Clustering Algorithm by Cased-based Reasoning

Yen-Liang Liu* and Ying-Tung Hsiao
Department of Computer Science, Nation Taipei University of Education
glenzoo@hotmail.com

Abstract

This paper discusses the ant clustering algorithm by case-based reasoning to explore the behavior model on the corporation between ants. Every ant has a case base which is updated iteratively by the process of the case-based reasoning. The process considers past case and arrives at decision on comparison between the current and old cases. By the process, an ant can learn the knowledge for dropping an item and picking up items. Thus, these ants are capable of cooperating to share their knowledge for better clustering.

Keyword: case base reason, ant clustering, behavior model, expert ant

1. Introduction

Clustering is an important technology for application in wild area. It was applied in many cases, for example, image process, marking, data mining, and information retrieval. The object of the clustering is to classify a large set of data into many groups. In these applications, a workable method for a user to classify these kinds of data sets automatically is therefore necessary. The degree of similarity between two objects is measured by some distance functions (indexes) between their feature vectors. As an index, the search is performed by returning the objects that are nearest to the query object in vast spaces. The similarity or dissimilarity is utilized as measurement for the distance among data.

There are many intelligence approaches to be applied for clustering, e.g. swarm intelligence algorithm and ant colony optimization. The swarm intelligence algorithm is with many advantages such as self organization, robustness and decentralization. As to the ant colony optimization, it is a new research area and belongs to a branch of the intelligence method [1]. There are many concerns for the algorithm based on the ant colony optimization because data can be clustered without the shape of data set and the information of the number of the data.

The following will introduce a clustering algorithm based on the expert ant. An ant can learn the knowledge for dropping an item and picking up items. Thus, these ants are capable of cooperating to share their knowledge for better clustering [1]. At the beginning, the cases of all ants are empty. We divide the ants into two groups, the normal and expert ants. For the normal ants, all the empty cases cannot drop any object. As to the expert ants, it picks up a loading to search its case at any time to meet the conditions. If we can find a case like that, the case will be retrieved and the loads will be dropped. The case base of all expert ants updates each time they drop down an object. Also, expert ants are able to cooperate with each other to find the best place for dropping their load.

2. Clustering Analysis

Clustering technical is to find the
distribution of data and the hiding information. Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity. For example, it is usual to analysis and process numbers in great number of data without known itself meaning when the manager try to find out the useful decision information among the considering data [2]. Data sets with many dimensions or enormous size and with high similarity pose very challenging problems in recognizing the data sets for efficient processing. The developing of recognizing with high efficiency and accuracy to support vast data access has become an active research area. We can cluster the large data sets into subgroups by the clustering analysis data mining methods. The complexity and reuse can be reduced for advance analysis the clustering data. In the application of data mining, it is usually to classify or divide a large data set to realize the types of data by data mining method. For example, the great number of mail in the post office can be grouped according their post number, emergent, register or not, size of mail, etc.

2.1 What Is Clustering?

Clustering deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters. Figure 1 shows the distribution of the original data before and after clustering. There are many algorithms and methods for application in clustering analysis. The object of these algorithms is to optimal clusters to let each group with the most dissimilarity between clusters as shown in Fig. 2. The characteristics on clustering analysis.

2.2 K-Means

The $k$-means clustering is a method of cluster analysis which aims to partition $n$ observations into $k$ clusters in which each observation belongs to the cluster with the nearest mean. Given an initial set of $k$ means, which may be specified randomly or by some heuristic, the algorithm proceeds by alternating between two steps: Assign each observation to the cluster with the closest mean (i.e. partition the observations according to the cluster with the closest mean (i.e. partition the observations according to the standard square error generated by the means).

$$E = \sum_{i=1}^{k} \sum_{p \in c_i} |p - m_i|^2$$

(1)

where $E$ denotes the sum of standard square error the object data, $p$ is the clustering object, $m$ denotes the mean of the type (pattern).
where \(|C_i|\) denotes the number of the clustering object. There is no guarantee that the K-means method will converge to the global optimum, and the result may depend on the initial clusters. As the algorithm is usually very fast, it is common to run it multiple times with different starting conditions. It has been shown that there exist certain point sets on which \(k\)-means takes super polynomial time to converge, but these point sets do not seem to arise in practice.

### 2.3 Density-Based Clustering Method

The density-based clustering method (DBSCAN) is a data clustering algorithm. It is a density based clustering algorithm because it finds a number of clusters starting from the estimated density distribution of corresponding nodes. DBSCAN is one of the most common clustering algorithms and also most cited in scientific literature. DBSCAN's definition of a cluster is based on the notion of density reach ability. Basically, a point \(q\) is directly density-reachable from a point \(p\) if it is not farther away than a given distance \(\varepsilon\) (i.e., is part of its \(\varepsilon\)-neighborhood), and if \(p\) is surrounded by sufficiently many points such that one may consider \(p\) and \(q\) be part of a cluster. \(q\) is called density-reachable from \(p\) if there is a sequence \(p_1, p_2, \ldots, p_i\) of points with and \(p_1 = p\) and \(p_i = q\) where each \(p_{i+1}\) is directly density-reachable from \(p_i\). Note that the relation of density-reachable is not symmetric (since \(q\) might lie on the edge of a cluster, having insufficiently many neighbors to count as a genuine cluster element), so the notion of density-connected is introduced: two points \(p\) and \(q\) are density-connected if there is a point \(o\) such that \(o\) and \(p\) as well as \(o\) and \(q\) are density-reachable. Figure 3 shows the density reach ability.

![Density Reach Ability](https://via.placeholder.com/150)

**Fig.3. The density reach ability**

DBSCAN requires two parameters: \(\varepsilon\) (eps) and the minimum number of points required to form a cluster (minPts). It starts with an arbitrary starting point that has not been visited. This point's \(\varepsilon\)-neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as noise. Note that this point might later be found in a sufficiently sized...
ε-environment of a different point and hence is made part of a cluster.

If a point is found to be part of a cluster, its ε-neighborhood is also part of that cluster. Hence, all points that are found within the ε-neighborhood are added, as is their own ε-neighborhood. This process continues until the cluster is completely found. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster of noise.

Pseudocode

\[
\text{DBSCAN}(D, \text{eps}, \text{MinPts})
\]

\[
\text{C} = 0 \\
\text{for each unvisited point } P \text{ in dataset } D \\
\text{mark } P \text{ as visited} \\
\text{N} = \text{getNeighbors}(P, \text{eps}) \\
\text{if sizeof(N) < MinPts} \\
\text{mark } P \text{ as NOISE} \\
\text{else} \\
\text{C} = \text{next cluster} \\
\text{expandCluster}(P, N, C, \text{eps}, \text{MinPts})
\]

\[
\text{expandCluster}(P, N, C, \text{eps}, \text{MinPts})
\]

add P to cluster C

\[
\text{for each point } P' \text{ in N} \\
\text{if } P' \text{ is not visited} \\
\text{mark } P' \text{ as visited} \\
\text{N'} = \text{getNeighbors}(P', \text{eps}) \\
\text{if sizeof(N') >= MinPts} \\
\text{N} = N \text{ joined with } N' \\
\text{if } P' \text{ is not yet member of any cluster} \\
\text{add } P' \text{ to cluster } C
\]

2.4 The Principle Of The Ant Clustering

Ant colony optimization (ACO) is an evolutionary meta-heuristic algorithm based on a graph representation that has been applied successfully to solve various hard combinatorial optimization problems. The main idea of ACO is to model the problem as the search for a minimum cost path in a graph. Artificial ants walk through this graph, looking for good paths. Each ant has a rather simple behavior so that it will typically only find rather poor-quality paths on its own. Better paths are found as the emergent result of the global cooperation among ants in the colony.

The behavior of artificial ants is inspired from real ants. They lay pheromone trails on the graph edges and choose their path with respect to probabilities that depend on pheromone trails and these pheromone trails progressively decrease by evaporation. Ants prefer to move to nodes, which are connected by short edges with a high amount of pheromone. In addition, artificial ants have some extra features that do not find their counterpart in real ants. In particular, they live in a discrete world and their moves consist of transitions from nodes to nodes. Also, they are usually associated with data structures that contain the memory of their previous action. In most cases, pheromone trails are updated only after having constructed a complete path and not during the walk, and the amount of pheromone deposited is usually a function of the quality of the path. Finally, the probability for an artificial ant to choose an edge often depends not only on pheromone, but also on some problem-specific local heuristics.

At each generation, each ant generates a complete tour by choosing the nodes according to a probabilistic state transition rule. Every ant selects the nodes in the order in which they will appear in the permutation. For the selection of a node, ant uses heuristic factor as well as pheromone factor. The heuristic factor, denoted by $\eta_{ij}$, and the pheromone factor, denoted by $\tau_{ij}$, are indicators of how good it seems to have node $j$ at node $i$ of the permutation. The heuristic value is generated by some problem dependent heuristic whereas the pheromone factor stems from former ants that have found good solutions.
The next node is chosen by an ant according to the following rule that has been called Pseudo-Random-Proportional Action Choice Rule. With probability \( q_0 \), where \( 0 \leq q_0 < 1 \) is a parameter of the algorithm, the ant chooses a node \( j \) from the set \( S \) of nodes that have not been selected so far which maximizes

\[
[f_j] \left[ \eta_j \right]^\beta
\]

where \( \alpha \geq 0 \) and \( \beta \geq 0 \) are constants that determine the relative influence of the pheromone values and the heuristic values on the decision of the ant. With probability \( (1 - q_0) \) the next node is chosen from the set \( S \) according to the probability distribution that is determined by

\[
p_{ij} = \frac{f_j \eta_j^\beta}{\sum_{h \in S} f_h \eta_h^\beta}
\]

Therefore the transition probability is a trade-off between the heuristic and pheromone factor. For the heuristic factor, the close nodes (low cost of path) should be chosen with high probability, thus implementing greedy constructive heuristic. As to the pheromone factor, if on edge \((i, j)\) there has been a lot of traffic then it is highly desirable, thus implementing the autocatalytic process. If the selection results a \( P_{ij} \), carry out the rule (9). In (9), the heuristic factor \( \eta_j \) is computed according to the following rule

\[
\eta_j = \frac{1}{f(X_j)}, \quad j \in S
\]

where \( f(X) \) represents the cost function of \( X \). In (9), it is favor that the choice of edges which are shorter (with low cost) and which have a greater amount of pheromone. Along the path from \( i \) to \( j \) the ant lays a trail substance so defined

\[
\Delta \tau_{ij}^k = \begin{cases} \frac{Q}{L_k} & \text{if } k\text{th ant uses edge } (i, j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}
\]

where \( Q \) is a constant related to the quality of pheromone trail laid by ants and \( L_k \) is the cost of the tour performed by the \( k \)th ant. In other word, pheromone updating is intended to allocate a greater amount of pheromone with low cost (shorter tours). This value is evaluated when the ant has completed a tour and consisting of a cycle of \( n \) iterations (generations). Then, it is used to update the amount of substance previously laid on the trail, on the following rules

\[
\tau_{ij}(t + n) = \rho \cdot \tau_{ij}(t) + \Delta \tau_{ij}(t)
\]

\[
\Delta \tau_{ij}(t) = \sum_{k=1}^{m} \tau_{ij}(t)
\]

where \( m \) denotes the number of ants, \( \rho \), \( \rho \in (0,1) \), is a coefficient of persistence of the trail during a cycle such that \((1-\rho)\) represents the evaporation of trail between generation \( n_g \) and \( n_g+1 \). The pheromone-updating rule was meant to simulate the change in the amount of pheromone due to both the addition of new pheromone deposited by ants on the visited edges and to pheromone evaporation. The algorithm stops iterating either when an ant found a solution or when a maximum number of generations have been performed.

\section{3. Case Based Reasoning}

Case-based problem-solving solves problems by retrieving and applying solutions to previous problems. The CBR paradigm covers a range of different methods for organizing, retrieving, utilizing and indexing the knowledge retained in past cases. Cases may be kept as concrete experiences, or a set of similar cases may form a generalized case. Cases may be stored as separate knowledge.
units, or split up into subunits and distributed within the knowledge structure. Cases may be indexed by a prefixed or open vocabulary, and within a flat or hierarchical index structure. The solution from a previous case may be directly applied to the present problem, or modified according to differences between the two cases. The matching of cases, adaptation of solutions, and learning from an experience may be guided and supported by a deep model of general domain knowledge, by more shallow and compiled knowledge, or be based on an apparent, syntactic similarity only. CBR methods may be purely self-contained and automatic, or they may interact heavily with the user for support and guidance of its choices. Some CBR method assume a rather large amount of widely distributed cases in its case base, while others are based on a more limited set of typical ones. Past cases may be retrieved and evaluated sequentially or in parallel.

3.1 The CBR cycle

At the highest level of generality, a general CBR cycle may be described by the following four processes[4]:

1. RETRIEVE the most similar case or cases
2. REUSE the information and knowledge in that case to solve the problem
3. REVISE the proposed solution
4. RETAIN the parts of this experience likely to be useful for future problem solving

A new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing knowledge-base (case-base). The four processes each involve a number of more specific steps, which will be described in the task model. In figure 4, this cycle is illustrated.

4. The Algorithm Of Clustering By Expert Ants

The details of our algorithm are as follows:

1. Spread the data and ants randomly on the grid and give all ants a random load.
2. While iter < max_iter
3. For i=1 to #ants
4. a = Choose a random ant without replacement
5. step = Generate a random speed from interval (1,max_speed)
6. new_location=Move (a, step)
7. if ant a is unladen and there is an object 0 in location 1 then pick_done=try_pick(0)
8. if pick_done=O then try to find a laden expert ant in the neighborhood of ant a and take the load of it with probability .5
9. end if
10. end if

Fig.4. The CBR cycle
(From: Zahra Sadeghi and Mohammad Tesnelab ,”Ant Colony Clustering by Expert Ants”, P.96)
10. if ant a is laden then search its case base for a case which has the similarity of more than .9 with the current load,
11. if such a case was found then
12. If retrieve
Drop the load in a neighborhood of 2 from the place of that case
13. end if
14. if a case was found with the same value with the current load then
15. If revise
Find a place for dropping it and change the place part of the case with the new found location
16. end if
17. if no similar case was found and the current location of it is empty then
18. drop_done=try_drop(l)
19. end if
20. if drop_done=1
21. If retain
Make a new case and save it in case base
22. end if
23. if drop_done=O
24. find an unladen expert ant in the neighborhood of 2 and give the load of ant a to it with probability of .5
25. if no unladen expert ant was found, find a laden expert ant in the neighborhood of 2 and exchange the load of ant a with the load of the neighboring ant
26. end if
27. End For
28. End while
try_pick(o)
1. r= a random number from interval (0,1)
2. p=compute probability function (1)
3. if p>r then
4. pick up the object 0
5. pick_done=1
6. else
7. pick_done=O
8. end if
try_drop(l)
1. r= a random number from interval (0,1)
2. d=compute probability function (3)
3. if d>r then
4. drop down load 1
5. drop_done=1
6. else
7. drop_done=0
8. end if

5. Conclusion

This paper introduces the novel method for clustering analysis. The case-based reasoning method explores the behavior model on the corporation between ants is present in this paper. An ant can learn the knowledge for dropping an item and picking up items. Thus, these ants are capable of cooperating to share their knowledge for better clustering. There are many concerns for the algorithm based on the ant colony optimization because data can be clustered without the shape of data set and the information of the number of the data.

References

[1] Zahra Sadeghi and Mohammad Tesnelab ,”Ant Colony Clustering by Expert Ants”
[3] Yun-Tao Chang and Lin Gong,”Data mining- principle and technology”,P.70