DOI: 10.21917/ijsc.2011.0032

IMPROVING THE CLUSTER PERFORMANCE BY COMBINING PSO AND K-MEANS ALGORITHM

G. Komarasamy[1] and Amitabh Wahi[2]

[1]*Department of Computer Science and Engineering, Bannari Amman Institute of Technology, Tamil Nadu, India*
E-mail: bit_kumar4u@yahoo.co.in

[2]*Department of Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India*
E-mail: awahi@rediffmail.com

Abstract

Clustering is a technique that can divide data objects into groups based on information found in the data that describes the objects and their relationships. In this paper describe to improving the clustering performance by combine Particle Swarm Optimization (PSO) and K-means algorithm. The PSO algorithm successfully converges during the initial stages of a global search, but around global optimum, the search process will become very slow. On the contrary, K-means algorithm can achieve faster convergence to optimum solution. Unlike K-means method, new algorithm does not require a specific number of clusters given before performing the clustering process and it is able to find the local optimal number of clusters during the clustering process. In each iteration process, the inertia weight was changed based on the current iteration and best fitness. The experimental result shows that better performance of new algorithm by using different data sets.

Keywords:

Clustering, Particle Swarm Optimization, K-means, Inertia Weight

1. INTRODUCTION

Data mining is the process of extracting patterns from large data. Data mining algorithms must have good scalability in order to receive the original information from the large amount of data. The purpose of clustering is to grouping related data points, which are close to one another. Data mining is a logical process that is used to search through large amounts of information in order to find important data. The goal of this technique is to find patterns that were previously unknown. Once you have found these patterns, you can use them to solve a number of problems. In general, clustering methods can be classified into two major categories, such as hierarchical clustering and nonhierarchical clustering approaches [1].

The hierarchical clustering approaches can be further classified into two categories: agglomerative and divisive approaches. In an agglomerative clustering method, each data point represents a cluster in the beginning, and two appropriate small clusters merge each other to form a larger cluster recursively. On the contrary, a divisive clustering method starts with a large cluster containing all of the data points, and then split the cluster into two smaller but dissimilar clusters repeatedly until an appropriate clustering result presents.

The nonhierarchical clustering approach is also called the partitioning approach. Given a set of n data points, a partitioning clustering method tries to assign n data points into one of K-clusters based on the distances between a data point and each cluster center. A data point is assigned to a cluster in which the center has the shortest distance to the data point.

Data members of a cluster are similar to each other; while data points belonging to different clusters are different. Therefore, the clustering problem can be viewed as a problem of partitioning a multi-dimensional space into k subspaces. One of the famous nonhierarchical clustering approaches is K-means. The advantages of K-means are fast convergence and easy implementation. However, K-means has two main drawbacks. First, the number of clusters has to be specified a prior; second, the initial condition may affect the clustering results [2].

In order to remedy the drawbacks of K-means, this paper proposes a new clustering algorithm based on particle swarm optimization [3]. The algorithm aims to group a given set of data into a user specified number of clusters. The advantage of a new algorithm is that it does not require the number of clusters to be specified in advanced.

2. PARTICLE SWARM OPTIMIZATION

PSO was introduced by Kennedy and Eberhart [4] it was inspired by the swarming behavior of animals, human social behavior, flocks of birds and herding phenomena in vertebrates. PSO is a population-based optimization algorithm, which could be implemented and applied easily to solve various functions of optimization problems. As an algorithm, the main strength of PSO is its fast convergence.

A particle swarm is a population of particles, in which each particle is a moving object which can move through the search space and can be attracted to the better positions. PSO must have a fitness evaluation function to decide the better and best positions, the function can take the particle's position and assigns it a fitness value. Then the objective is to optimize the fitness function. In general, the fitness function is pre-defined and is depend on the problem.

Each particle has own coordinate and velocity to change the flying direction in the search space. And all particles move through the search space by following the current optimum particles. Each particle consists of a position vector 'z', which can represent the candidate solution to the problem, a velocity vector 'v', and a memory vector 'pid', which is the better candidate solution encountered by a particle.

$$Z_i = \{z_{i1}, z_{i2},\ldots z_{in}\}$$

$$v_i = \{v_{i1}, v_{i2},\ldots v_{in}\},\ i=1,2,\ldots n$$

where n is the size of swarm. The best previous experience of the ith particle is represented as:

$$pid\_i = \{ pid\_i_1, pid\_i_2,\ldots pid\_i_n\}$$

Another memory vector 'pgd', which is the best candidate solutions encountered by all particles, is used. The particles are then manipulated according to the following equations:

$$v_{id}(t+1) = \omega v_{id}(t) + \varphi_1\_rand(pid\_i - z_{id}(t)) + \varphi_2\_rand(pgd - z_{id}(t)) \quad (1)$$

$$z_{id}(t+1) = z_{id}(t) + v_{id}(t+1) \quad (2)$$

where, ω - an inertia weight, which used to control the impact of the previous history of velocities on the current velocity, and regulate the trade-off between the global and local exploration abilities of the swarm. A big inertia weight facilitates global exploration, while a small one tends to facilitate local exploration. In order to get a better global exploration, w can be gradually decreased to get a better solution.

φ_1 and φ_2 - two positive constants

rand - uniformly generated random number.

The Eq. (1) shows that in calculating the next velocity for a particle, the previous velocity of the particle, the best location in the neighborhood about the particle, the global best location all contribute some influence to the next velocity. Particle's velocities in each dimension can arrive to a maximum velocity v max, which is defined to the range of the search space in each dimension. The position of each particle is updated by Eq. (2). The velocity updating scheme has been illustrated in Fig.1 with a humanoid particle.

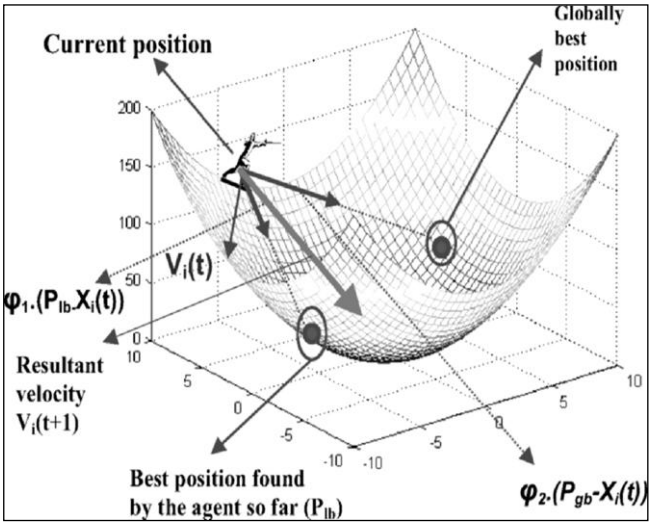

Fig.1. Illustrating the velocity updating scheme of basic PSO

Algorithm steps:

1. Initialize each particle with random position and velocity.
2. Evaluate the fitness for each particle (individual best).
3. Keep track of individual's highest position (global best).
4. Modify the velocity based on individual best and global best position by Eq. (1).
5. Update particles position by Eq. (2).
6. Terminate till the condition met.

3. K-MEANS ALOGORITHM

The K-means algorithm groups the set of data points in space into a predefined number of clusters. In this regard, the Euclidean distance is commonly used as a similarity measure [4]. K-means is a clustering algorithm that aims to partition the set of observation points into K clusters. Let R be the set of real numbers and R^d be d –dimensional vector space. Given a finite set $X \subseteq R^d$ $X=\{x_1, x_2 \ldots, x_n\}$, where n is the number of vectors. The K-means algorithm partitions the set X into subset S, whose subsets are $S=\{S_1,S_2,\ldots,S_K\}$, where K is a predefined number. Each cluster is represented by a vector c, $C=\{c_1,c_2,\ldots,c_K\}$ is the center set in the vector space. A Euclidean distance measure is used to calculate the distance between vectors and centers. The distance measure is,

$$d(x_i,c_j)=\left\{\sum_{i=1}^{dw}(x_{im}-c_{jm})\right\}^{1/2} \quad (3)$$

where, dw is the number of features of data vector. Place the vectors in the clusters. The cluster is recalculated by,

$$c_j=\frac{1}{n}\sum_{x\in s}x_j \quad (4)$$

where, n is the number of data vectors in the subset s.

Algorithm Steps:

1. Place K points into the space represented by the objects that are being clustered. These points represent the initial group is reached.
2. Calculate the distance between the cluster centre and the data vectors according to the Eq. (3).
3. Assign each object to the group that has the minimum distance.
4. When all the objects have been assigned recalculate the cluster center according to the Eq. (4).
5. Repeat the steps until the termination condition reached.

4. COMBINING PSO AND K-MEANS ALGORITHM

PSO clustering algorithm performs a global search in the entire solution space. If given enough time, the PSO clustering algorithm can generate good clustering results than the K-means algorithm, but PSO requires much more iterations to converge to the optima than the K-means algorithm [5],[8].

The K-means algorithm is a fast method due to its simple and small number of iterations. But the dependency of the algorithm on the initialization of the centers has been a major problem and it usually gets stuck in local optima though it tends to converge faster than the PSO algorithm. Using the merits of both algorithms, PSO and K-means are combined. The new algorithm does not depend on the initial clusters and can avoid being trapped in a local optimal solution.

In the new algorithm [6],[7] a single particle represents a set of cluster centers, that is, a particle represents one possible solution for clustering and the position of each particle x_i is constructed as,

$$x_i = (c_{i1}, c_{i2}, \ldots, c_{iK})$$

where, K is the number of clusters, c_{ij} is the j-th cluster centre of the i-th particle. Then the swarm represents a candidate cluster result. The fitness of each particle is measured as,

$$\text{fitness}=1/\sum_{j=1}^{K}\sum_{x\in c} d\left(x_i, c_j\right) \quad (5)$$

where $d(x_i, c_j)$ is defined in the Eq. (3) and c_j is the j-th cluster.

Algorithm Steps:

1. Initialize each particle to contain K randomly selected cluster centers, at the same time, randomly choose K cluster centers for K-means algorithm. The process of K cluster centers being chosen in PSO and K-means is realized by randomly assigning each data vector to a cluster and computing the cluster center.
2. For each particle compute the fitness value according to the Eq. (5).
3. Compare the i-th particle's fitness value with particle's best solution pid_i, the better candidate solution encountered by particle i, if current value is better than pid_i, set pid_i equal to the current particle.
4. Compare particle's fitness value with the population's overall previous best pgd, the best candidate solution encountered by all particles, if current value is better than pgd, set pgd to the current particle's value and location.
5. At the same time, sse, the sum of distance between each datavector and its cluster center in K-means algorithm is computed according to the Eq. (3).
6. Compare pgd and 1/sse; choose the big one as pgd.
7. Update the velocity and location of each particle using the new pgd according to Eqs. (1) and (2) respectively.
8. Assign each data vector to the closest cluster based on the new location of each particle, and then recalculate the cluster center according to the Eq. (4).
9. Repeat the steps until the maximum iteration exceeds.

5. EXPERIMENTAL RESULTS

The experiment is to compare the quality of the clustering. The quality is measured according to the following criteria:

bc(C) : between cluster variation,

$$\text{bc(C)}=\sum_{1<i<j<K} d\left(x_i, c_j\right) \quad (6)$$

This measures the distance between different clusters, the bigger the sum of the distances is, the higher the quality of clustering.

wc (C):within cluster variation,

$$\text{wc}(C)=\sum_{j=1}^{K}\sum_{x\in C} d\left(x_i, c_j\right) \quad (7)$$

This measures how compact or tight the clusters are, the smaller the sum of the distances is, the higher the quality of clustering is.

Precise rate (PR): the number of data vectors that placed in right class divided by the number of data vectors.

The data set is used to compare the performance of PSO, K-means and new combined algorithm. It is the real data set available in UCI machine learning repository.

Table.1. Results of Iris dataset

Criteria	PSO algorithm	K-means algorithm	A new algorithm (Combined PSO and K-means)
bc (between clusters)	6.168	9.569	10.021
wc (within cluster)	226.03	130.99	97.42
Iteration	55	5	55
PR	65%	81%	89%

From the experiment, it is known that combined PSO and K-means algorithm gives good result than PSO algorithm and K-means. In the new algorithm, it doesn't depend on the initial centers.

6. CONCLUSION

The algorithm combining PSO and K-means gives good performance than other two algorithms. The new algorithm improve the convergence speed of PSO and helps K-means independent on initial clusters. The future work is to improve the PSO alone and then combines the improved PSO and K-means algorithm to show the better performance.

REFERENCES

[1] J. Han and M. Kamber, "*Data mining: Concepts and Techniques*", San Francisco: Morgan Kaufmann Publisher, 2001.

[2] J. M. Pena, J. A. Lozano, and P. Larranaga, "An empirical comparison of four initialization methods for the K-Means algorithm", *Pattern Recognition Letters*, Vol. 20, No. 10, pp. 1027-1040, 1999.

[3] J. Kennedy, and R. Eberhart, "Particle Swarm Optimization", *Proceedings in IEEE International Conference on Neural Networks*, Vol. 4, pp. 1942–1948, 1995.

[4] Dong J and Qi M, "A New Algorithm for Clustering based on Particle Swarm Optimization and K-means", *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence*, pp. 264-268, 2009.

[5] Van der Merwe D W and Engelbrecht A P, "Data clustering using particle swarm optimization", *Proceedings of IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 215-220, 2003.

[6] X. Cui, T.E. Potok and P. Palathingal, "Document clustering using particle swarm optimization", *Proceedings of IEEE Swarm Intelligence Symposium*, pp.185-191, 2005.