University of Waterloo
Department of Electrical and Computer Engineering
Spring 2014 – ECE457A Cooperative and Adaptive Algorithms

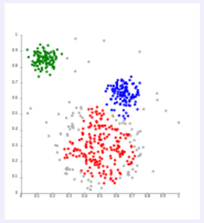
CPAMI
INTELLIGENT RESEARCH EXCELLENCE

IEEE Computational Intelligence Society

26. Solving Multiple Targets Clustering Problem using Metaheuristics Algorithms

Multiple Targets Clustering

Our project, Multiple Targets Clustering Problem, is the task of grouping a set of objects in d-dimensional space, such that objects in the same group are more similar to each other than to those in other clusters. It acts as an important role in many fields, including image analysis, machine learning, pattern recognition, information retrieval and bioinformatics.



Problem Formulation

Inputs:

- number of dimensions d
- number of clusters k
- number of points n
- A n by d position matrix $P = \begin{matrix} p_{11} & p_{12} & \cdots & p_{1d} \\ p_{21} & p_{22} & \cdots & p_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nd} \end{matrix}$ providing d-dimensional positions of all the n points

Outputs:

A group of k clusters containing n point $X = \{C_1, C_2 \ldots C_k\}$

Objectives:

Minimize the average square of distance of all points to centroid of the cluster contain the points.

The cost function:

$$f = \frac{\sum_{i=1}^{k} \sum_{j=1}^{length\ of\ Ci} \sum_{l=1}^{d} (P(C_{ij},l) - centroid(C_i,l))^2}{d * n}$$

Where C_1, C_2, C_3 ... C_k are sets of numbers, C_{ij} is the j^{th} element in the set C_i,

$C_1 \bigcap C_2 \bigcap C_3 \ldots \bigcap C_k = \{\}$

$C_1 \bigcup C_2 \bigcup C_3 \ldots \bigcup C_k = \{1,2,3 \ldots n\}$

The location of the centroid:

$$centroid(C_i,l) = \frac{\sum_{m=1}^{length\ of\ Ci} P(C_{im},l)}{length\ of\ C_i}$$

Problem Modelling:

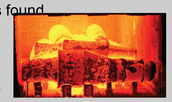
Since the Multiple Targets Clustering Problem itself is a well-studied optimization model, no further modeling is required. We can apply it to many fields, including image analysis, machine learning, pattern recognition, information retrieval and bioinformatics.

Metaheuristics

Tabu Search

- Initialization is randomly generated
- Each solution is stored as an array of cluster, and each cluster contains the index of membership
- New solutions will be randomly generated, once accepted, it will been placed in the tabu list
- Contents in tabu list will not be visited for tabu tenure iterations
- Forcing the algorithm to seek new solutions outside the local optimum
- The so far best solution will be stored and updated once a better solution is found

Simulated Annealing

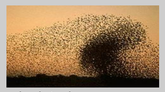


- Each solution X stores the cluster membership
- Neighbours are generated by moving one point from one cluster to another
- Geometric cooling schedule $T_{t+1} = T_t * a$, constantly updated after a number of iterations
- Calculate cost function $\Delta f = f_{new} - f_{old}$
- If $\Delta f < 0$, accept the solution
- Else, generate a random number r between 0 and 1, accept solution if $\exp(-\Delta f/T) > r$
- Update the best solution X and f

Genetic Algorithm

- Each population stores k d-dimensional centroids
- The square of distance between each points and centroid in each population is calculated
- Data points are assigned to k cluster based on the square of distance
- Geometric centroid of each cluster is determined and fitness is calculated using cost function
- If two random selected population fitness is not equal to min(fitness) perform crossover
- If random selected population is larger than mean(fitness) perform mutation

PSO



- Each particle stores k d-dimensional centroids
- The square of distance between each points and centroid in each particle is calculated
- Membership assigned based on the square of distance
- Geometric centroid of each cluster is determined
- Cost function based on the square of distance between the geometric centroid and each points within the cluster
- Update x and v

ACO

- The give points count as both food and nest.
- Pheromone are deposited near the points to attract ants.
- Ant randomly walking around trying to seek food or nest.
- Ant will pick up food and drop it off at the nest, pheromone level will update accordingly.
- The cluster number will shrink as more points are dropped off at the nest.
- The iteration will stop when the cluster number equals to the desired cluster number.

Experimental Results

Experiment Setup

Data input

	DataSet1	DataSet2	DataSet3	DataSet4	DataSet5
Data points	100	150	200	250	300
Dimension	2	2	2	2	2
Cluster	3	3	3	3	3

Experiment	Simulation Metrics
1	Set a CPU time of 10s, 15s, 20s, 25s and 30s, run each algorithm on the 300 points dataset and record the cost function
2	Set a CPU time of 10s, run algorithms on each dataset and record the cost function
3	Run algorithms on each dataset, record the time per iteration

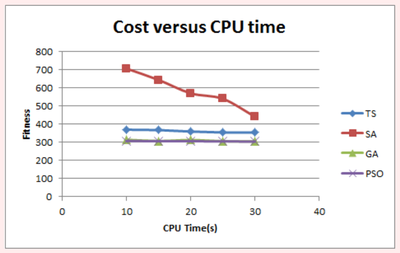
Evaluation Metrics

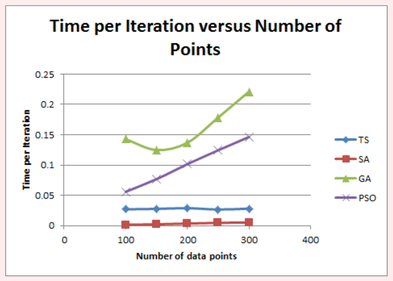
Evaluation Metrics

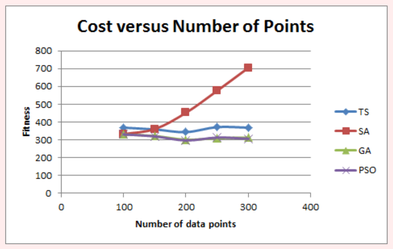
After conducting different experiments, we decided to make our evaluation metrics should the following to best evaluate the algorithm:

- Average cost for running 10s on each dataset
- Single iteration time in seconds
- Average cost for running 10s, 15s, 20s, 25s and 30s on the 300 points dataset

Comparative Study







Conclusion

The test result shown Genetic Algorithm and Particle Swarm Optimization not only produced the best result, but also converged to the optimal solution faster. Ant Colony Algorithm produces the worst run time because it simulates real world ant movement, which will take a long time for each iteration. Simulated Annealing and Tabu search produced moderate performance.

For centroid-based clustering algorithm, computation power to calculate the new centroids for each solution increase exponentially as the data set gets bigger. In general, this algorithm prefer relative small size data set as all centroids will be calculate as each iteration.
Future adaptation to specific problem can be done by tuning algorithm parameters. This can make the algorithm more efficient for the problem

In adaptive PSO, regular PSO is hybridized with Genetic Algorithm. The cost function for 300 points 2 dimensional clustering reduces by 1.404% in average. In adaptive simulated annealing, cooling takes place every time some move is accepted instead of fixed number of iterations per temperature.For the same clustering problem above, the cost function reduces by 8.55% in average.

Student Name: Zhechen Du, Zhengfang Duanmu, Lingyun Li, Yijie Liang, Tao Shen

Professor: Alaa Khamis