Unsupervised Updation Strategies for ACO Algorithms

Raghavendra G S
Faculty in Computer Science,
BITS-Pilani K.K. Birla Goa Campus, GOA

Prasanna Kumar N
Faculty in Mathematics,
BITS-Pilani K.K. Birla Goa Campus, GOA

ABSTRACT
Ant Colony Optimization (ACO) algorithms belong to class of metaheuristic algorithms, where a search is made for optimized solution rather than exact solution, based on the knowledge of the problem domain. ACO algorithms are iterative in nature. As the iteration proceeds, solution converges to the optimized solution. In this paper, we propose new updation mechanism based on clustering techniques, an unsupervised learning mechanism aimed at exploring the nearby solutions region. We also report in detail the impact on performance due to integration of cluster and ACO.

Keywords
Ant, Meta-heuristic, Optimization, unsupervised, cluster.

1. INTRODUCTION
One of the basic research objectives in the field of nature inspired algorithms is the derivation of a methodology and associated tools for modeling and measuring the dynamic behavior of the underlying process. Among many such evolutionary algorithms, Ant Colony Optimization (ACO) is a recently developed population based algorithm which has been applied to many NP-hard combinatorial optimization problems [1]. The foraging behavior of ants has fascinated many researchers, which has lead to the development of Ant algorithms. In search of food, ants leave their nest and move towards food source in a random direction. On their way towards the food source, they leave behind a chemical substance called pheromone trials. This pheromone trial will guide the subsequent ants to make a move towards the food source. Ants will also use this pheromone trial to trace back their nests. The shortest path on which ants traveled from nest to food source and back to nest will have more pheromone concentration compared to the other paths, thus making it more favorable path. Eventually all the ants follow this shortest path. Thus they find a shortest path from nest to food source. The above mentioned feature has resulted in evolution of Ant algorithms. A close observation reveals that foraging behavior can be used to attack the combinatorial problems. This new computing paradigm has a feature of positive feedback, distributed computation and use of constructive greedy heuristic approach. The positive feedback, a sort of reinforcement speaks about the quality of solution found during the search and is expressed in terms of amount of deposition, distributive computation avoids the premature convergence as group of ants are involved, which will exploit the search space effectively and greedy heuristic helps to find the acceptable solution. Many variants of Ant Algorithms have been proposed in the literature and each algorithm improvises the earlier versions [2-6]. These algorithms try to strike the balance between exploration and exploitation. An Ant which explores the search space around the optimally best solution may not reach globally best solution. Similarly, exploiting the search space will get the globally best solution, but needs more time to converge. Therefore it is necessary to strike the balance between exploration and exploitation for better performance in terms of quality of solution found and time to converge. These Ant algorithms have been successfully applied to benchmark problems like, Traveling Salesman Problem (TSP), the Job-Shop Scheduling (JSP), the Vehicle Routing Problem (VRP), Graph Coloring Problem (GRP) and Quadratic Assignment Problem (QAP). The ACO algorithms have been extended to continuous search domain. Based on the literature survey, research work related to ACO can be classified into following categories: Devising new strategies for pheromone updation [8, 11], Reward-Penalty approaches [13], Dynamic parameter adjustment [9-10], Hybridization of ant algorithms [14], Proofs for convergence [7,12,16,17] and applying the ant algorithms to multidisciplinary fields.

2. CLUSTERING
Data clustering [20] is one of the most important human activities that involve discovering groups and identifying interesting distribution and patterns in the underlying data. Clustering problem is about partitioning the data into groups/classes of objects in such a way that the objects within the group are very similar and the objects across the group are quiet different. Clustering is an unsupervised approach, where no labeled data will be available. The ultimate goal of the clustering is to assign the unlabelled data to labeled classes. The labels to the classes are categorical in nature and are purely data driven; that is, they are obtained from data. It is possible that sometime even class labels may not be defined, but still cluster process should identify the natural closeness among the data and should group them. In general, data belong to only one cluster. However, it is possible for the data to belong more than one clusters and its association with particular cluster is determined by the degree of membership. We will discuss some of the important clustering algorithms available in the literature and these algorithms will be used to come up with cluster integrated ACO algorithm.

2.1 k-Median algorithm

The k-Median algorithm is a variant of k-Mean algorithm [15]. This algorithm creates k partitions of the data points and median represents the centroid of each partition. The algorithm typically employs the absolute-error criterion that minimizes the sum of the absolute distance of each data point with respect to centroid of the cluster. The working of k-Median algorithm is as follows:

Input: The number of clusters k and the n data sets.
Output: A set of k clusters that minimizes the squared-error criterion.

Method:
1. Arbitrarily choose k data as the initial cluster centers.
2. (Re)assign each of the data to the nearest cluster C_w , i.e.,
$x_j \in C_w$, if $\| x_j - m_w \| < \|x_j - m_i\|$
for j = 1, 2, ...N , i = w, and i = 1,2,...k.
3. Recalculate the cluster center (mean) for the current partition.
4. Repeat the steps 2 and 3 until there is no change for each cluster.

2.2 DBSCAN

DBSCAN (Density Based Spatial Clustering of Applications with Noise) is a density based clustering algorithm proposed by Martin et al [21]. The density based approaches treat the "Clusters" as a set of points, such that each point can be reached from every other point within the group and "noise" as a set of unreachable points. The algorithm can be better understood with the following definitions:

Definition 1 (ϵ-neighborhood of a point). The ϵ -neighborhood of a point x is defined as

$$N_\epsilon(x) = \{y \in D: d(x, y) \leq \epsilon\}$$

where D is the set of data points, d(., .) is a certain distance function and ϵ specifies the radius of the circle. The ϵ-neighborhood of a point should contain minimum number of points say Min_pts, then the point is called core point otherwise it is a border point. The core points are present inside the cluster and border points form the boundary of the cluster.

Definition 2 (Directly density-reachable). A point *x* is said to be directly density-reachable from a point *y* (with respect to ϵ and Min_pts) if
1. $x \in N_\epsilon(y)$
2. $N_\epsilon(y) \geq$ Min pts, where $N_\epsilon(y)$ denotes the number of points (core point condition).
If two core points *x* and *y* belongs to the same cluster then *x* can be directly density-reachable from *y* and vice versa. However, if *x* is a core point and *y* is a border point then *y* is directly density-reachable from *x* but other way around is not possible.

Definition 3 (Density-reachable). A point *x* is said to be density-reachable from point y if there is a sequence of points $x = x_1, x_2, \ldots\ x_i = y$ such that x_l is directly density-reachable from x_{l+1} for l = 1, 2, ... i-1.
The density-reachability is an extension of directly density reachable. The definition suggests that all the core points in a cluster *C* can be visited as a sequence of points.

Definition 4 (Density-connected). Two points *x* and *y* are said to be density-connected w.r.t ϵ and Min_pts if there exists a point *z* such that both *x* and *y* are density reachable from *z* w.r.t ϵ and Min_pts.
The border points of cluster *C* may not be density reachable to each other; however there must exist a set of core point in *C* that is density reachable to border points. The definition 4 specifies the condition for establishing the relation between the border points of the clusters *C*.

Definition 5 (Cluster). Let D be the dataset. A Cluster *C* w.r.t ϵ and Min_pts is a nonempty subset of D satisfying the following conditions:
1. $\forall x,y \in D$, if $x \in C$ and *y* is density-reachable from *x* w.r.t ϵ and Min_pts then $y \in C$.
2. $\forall x,y \in$ C, *x* and *y* are density connected w.r.t ϵ and Min_pts.

The DBSCAN defines the cluster as set of density-connected points and noise as a set of points that does not belong to any of the clusters. DBSCAN starts with an arbitrary point *x* and finds all the point that are density-reachable from *x* w.r.t ϵ and Min_pts. If *x* is a core point, then a cluster w.r.t ϵ and Min_pts is formed. If *x* happens to be the border point, then no points are density-reachable and DBSCAN visits the next unclassified point.

The DBSCAN algorithm works as follows:
Input: Set of data points *D*, ϵ and the minimum number of points Min_pts.
Output: A Set of *m* clusters.
Method:
1. Mark all the points as unclassified.
2. Do the following until all the points in *D* are marked as classified.
  2a. Select an unclassified point *P* in *D* and mark it as classified.
  2b. Set *N* = neighbor (*P*, *ϵ*) % Identify the neighborhood set of points.
  2c. If sizeof(*N*) < MinPts
    2c1. mark *P* as Noise.
    else
    2c`1. Create a cluster *C* and add the point *P* to it.
    2c`2. Do the following for all the points in *N*.
      **2c`21.** Select the point *P*` in *N* and mark it as classified.
      **2c`22.** Set *N*` = neighbor(*P*`, ϵ)
      **2c`3.** If sizeof (*N*`) ≥ MinPts
        **2c`31.** *N* = *N* combined with *N*`.
      **2c`4.** Add the point *P*` to cluster *C*.

It should be noted that DBSCAN needs two parameters ϵ and Min_pts to work but identifying the values for parameters is not easy. A simple heuristics called k-dist graph is developed to find the values for parameters. The DBSCAN algorithm needs to compute the distance between point and the k nearest point. These distances are sorted and then k-graph is plotted. The first "valley" in the graph is identified and the corresponding point is used to set the ϵ. The value of Min pts is set to k = 4, since k-dist graph won't vary much for higher values.

3. INTEGRATION OF CLUSTERS IN ACO

The experimental simulation reveals that there exist a correlation between the quality of solution found and the distance from good or optimal solutions. In literature, several measures to access the quality of solution can be found and one such measure is fitness-distance correlation (FDC) function [19]. The FDC computes the correlation coefficient which determines the goodness of the obtained solutions with respect to global best solution. The correlation coefficient will have high positive value, if the obtained solution is near to the global best solution. Infact, for the problems like TSP [6], large number of local optimum solutions is concentrated in a small region near the global best solution. Inorder to exploit the regions near the best solution, we propose a cluster based updation strategy which reinforces the toured paths in an unconventional manner. The cluster based updation strategy has the following characteristics:

• It groups the nearby tour performances and each tour performances within the group is reinforced with the same amount of pheromone trial, thereby supports the exploitation of (best) solutions.
• It reinforces all the paths, thereby supports the exploration.

Ideally, best tour in the group will be selected and its performance will be taken as a reference for updating the rest of the paths present in the group. The following subsections will discuss the incorporation of clustering mechanism in ACO algorithm.
The general outline of the cluster integrated ACO is as follows:

Input: A dataset D, number of cities n, number of ants m, number of cluster k and parameter τ , η, ρ.
Output: The best tour length s_{bs}.
Method: Initialize the Pheromone trial, parameters and set s_{bs} to null.
while termination condition not met **do**
$\chi_{iter} \leftarrow$ null
for j = 1, · · · , n **do**
s ← ConstructSolution
s ← LocalSearch Optional
if(f (s) < f (s_{bs})) or (s_{bs} = NULL) then $s_{bs} \leftarrow$ s
$\chi_{iter} \leftarrow \chi_{iter} \cup \{s\}$
end for
Identify the clusters in χ_{iter}
Apply the Pheromone Updates for each cluster by choosing the best solution in the corresponding group.
end while

3.1 k-Median ACO (kMed-ACO)

The k-Median algorithm creates *k* partitions of the tour performances. The updation strategy involves reinforcing all the tour paths in the cluster with the same amount of pheromone trial as that of best tour path in the cluster.

3.2 Density Based Clustered ACO (DBC-ACO)

The problem with the k-Median algorithms is that the shape of the cluster will be influenced by exceptional/outlier data points. The outliers are the set of points that may not logically belong to any of the clusters, but due to logical nearness they are forced to be part of one of the clusters. The reason for assigning the outlier to one such cluster is due to fixed number of clusters that are specified as a part of parameter settings. The evolved clusters may not look natural in presence of outliers. Inorder to access the impact of natural looking clusters, we incorporated the DBSCAN algorithm in ACO algorithms. The number of clusters that evolve purely depends on the distribution of data. The basic DBSCAN algorithm was modified to suit the single dimension data space. The modified DBSCAN algorithm has a single parameter similar to ϵ and it is derived from the distribution of data. The computation of ϵ is as follows:

- Sort the tour performances in ascending order.
- Compute the sum of the difference between the tour performances.
- Calculate the average of the sum of difference.

The computed average, now onwards called as ϵ mean difference is used as distance measure for clustering process. The clustering process proceeds as follows:

1. Sort the tour performances TL_1, TL_2 …TL_m.
2. Create a new cluster *C*.
3. Select a new data point TL_i in the increasing order, assign it to newly created cluster *C* and mark it as assigned.
4. Check whether next data point TL_{i+1} is located within the mean difference distance or not i.e., $\| TL_i - TL_{i+1} \| \leq \epsilon$.
5. If it is reachable, assign it to the cluster, mark it as assigned and repeat from step 2 until all the points are marked as assigned.
6. If it is not reachable, repeat from step 3.

It should be noted that algorithm treats the outliers as a cluster that contains only single point.

4. EXPERIMENTAL STUDY

4.1 Parameter Settings

Since clustering mechanism have been incorporated in the ACO algorithms, additional parameters pertaining to cluster need to be specified as a part of parameter settings. The k-Median ACO need k number of clusters as a parameter for clustering process. We made extensive simulations by varying the parameters α, β from 1 to 5, ρ from 0.7 to 1.0 with the increment of 0.03 and number of ants m were varied in range from {10,n/2,n}, where n is the number of ants in the system. The parameter k was varied in the range of 20-80% of the number of ants and the total number of iterations was set to 1,00,000.

4.2 Result Analysis for Primary Updation

The Table 1 shows the comparative results of cluster integrated ACO for primary updation. The maximum deviation observed in k-Median is 0.46% for bays29 dataset. The DBSCAN algorithm provides better optimal solution compared to k-Median, but it constructs large number of clusters for updation purpose. Inorder to access the behavior of algorithm, graphs are plotted for varying number of ants, clusters and pheromone at different intensity level. In subsequent section, we will present the graphical analysis of the behavior of all the cluster integrated ACO algorithms.

Table 1. Comparative Results for Cluster integrated ACO algorithms for primary updation.

Dataset	Algorithm	Best (Std Dev)	Average (Std Dev)
bays29	kMed-ACO	2029.4 (0.46%)	2035.3 (0.75%)
	DBC-ACO	2025.4 (0.26%)	2028.2 (0.4%)
att48	kMed-ACO	10665.1 (0.34%)	10688.6 (0.57%)
	DBC-ACO	10651.6 (0.22%)	10673.4 (0.42%)
eil51	kMed-ACO	427.8 (0.42%)	430.5 (1.05%)
	DBC-ACO	427.1 (0.25%)	429.5 (0.82%)
st70	kMed-ACO	678.2 (0.47%)	681.8 (1.00%)
	DBC-ACO	677.4 (0.35%)	680.5 (0.81%)
eil76	kMed-ACO	540.3 (0.42%)	547.4 (1.74%)
	DBC-ACO	539.4 (0.26%)	543.6 (1.04%)
kroa100	kMed-ACO	21332.4 (0.23%)	21359.4 (0.36%)
	DBC-ACO	21313.5 (0.14%)	21340.4 (0.27%)
kroa200	kMed-ACO	29466.4 (0.33%)	29503.2 (0.46%)
	DBC-ACO	29408.1 (0.13%)	29443.5 (0.26%)
lin318	kMed-ACO	42221.2 (0.45%)	42278.2 (0.59%)
	DBC-ACO	42146.7 (0.28%)	42091.8 (0.14%)

4.3 Result Analysis for Secondary Updation

We extended the proposed approach by incorporating additional reinforcement mechanism. The additional reinforcement is done after primary updation. The additional/ secondary updation is adopted to provide the diversification for the search process. The primary updation mechanism updates the pheromone trial proportional to the quality of solution found. The secondary updation mechanism uses cluster based updation strategy to reinforce the traveled paths. The Table 2 shows the comparative results for the cluster integrated ACO for secondary updation. On comparing Table 2 with Table 1, it can be observed that secondary updation strategy improvises most of the obtained solution. The extended approach provides best solution for att48, eil76, kroa100 and kroa200 dataset for DBC-ACO algorithm.

Table 2. Comparative Results for Cluster integrated ACO algorithms for secondary updation.

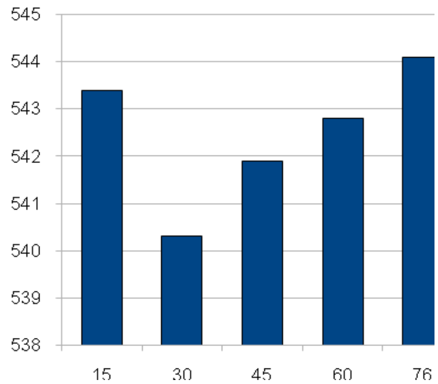
Datasets	Algorithm	Best (Std Dev)	Average (Std Dev)
bays29	kMed-ACO	2024.6 (0.22%)	2029.1 (0.45%)
	DBC-ACO	2022.3 (0.11%)	2026.7 (0.33%)
att48	kMed-ACO	10654.3 (0.24%)	10671.2 (0.40%)
	DBC-ACO	10630.5 (0.02%)	10644.3 (0.15%)
eil51	kMed-ACO	427.1 (0.25%)	431.4 (1.26%)
	DBC-ACO	428.7 (0.63%)	432.7 (1.57%)
st70	kMed-ACO	677.3 (0.34%)	682.5 (1.11%)
	DBC-ACO	675.7 (0.1%)	678.9 (0.57%)
eil76	kMed-ACO	539.4 (0.26%)	544.1 (1.13%)
	DBC-ACO	538.4 (0.07%)	541.2 (0.59%)
kroa100	kMed-ACO	21302.1 (0.09%)	21339.5 (0.27%)
	DBC-ACO	21291.6 (0.04%)	21318.3 (0.17%)
kroa200	kMed-ACO	29423.6 (0.18%)	29478.6 (0.37%)
	DBC-ACO	29378.5 (0.03%)	29406.4 (0.13%)
lin318	kMed-ACO	42115.6 (0.2%)	42171.6 (0.33%)
	DBC-ACO	42076.1 (0.11%)	42131.9 (0.24%)

4.4 Performance Analysis of Algorithms

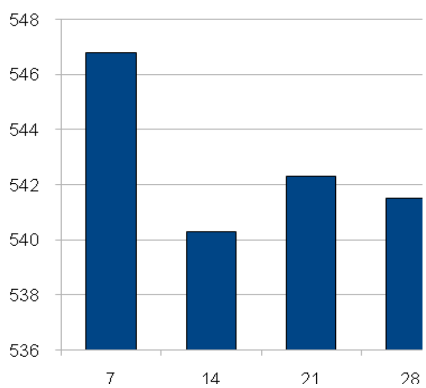
In this section, we will discuss the impact of various parameters affecting the performance of the algorithm. A comparative graph are drawn by varying number of ants, number of clusters and pheromone trials of different strength for each variant of the algorithm in order to access the performance. The execution profile graph is plotted to access the nature of convergence of algorithm.

4.4.1 k-Median ACO

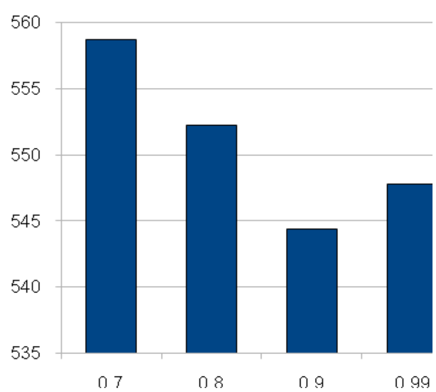
The eil76 dataset was selected to assess the behavior of the algorithm. The Figure 1 shows the comparative results of primary updation for kMed-ACO. The Fig 1a shows that algorithm exhibits comparatively less variation in obtained solutions for varying number of ants and obtains better results for smaller number of ants. The kMed-ACO provides optimal result, when the number of clusters is 50% of the number of ants as seen in Fig 1b. Similarly, it exhibits least variation in the best solution for the varying pheromone trial strength and with the increase in pheromone persistent factor, quality of solution improvises. The Figure 2 shows the comparative results of secondary updation for kMed-ACO. The kMed-ACO marginally improvises the solution across the varying number of ants and the cluster and provides good solution for higher pheromone trial.



(a) Variation in number of ants

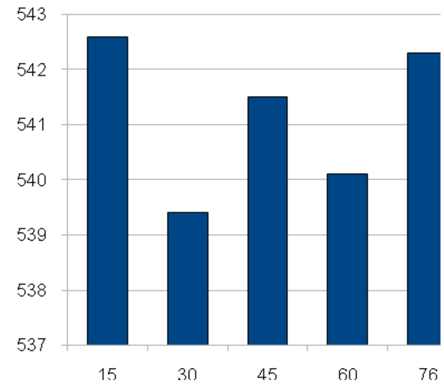


(b) Variation in number of clusters

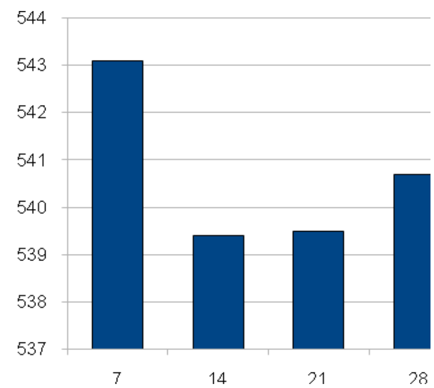


(c) Variation in pheromone trial

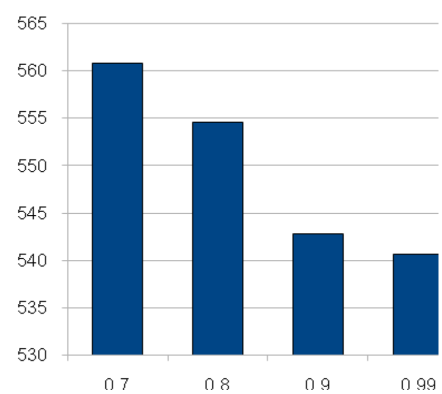
Figure 1: Comparative graph of kMed-ACO for primary updation.



(a) Variation in number of ants



(b) Variation in number of clusters

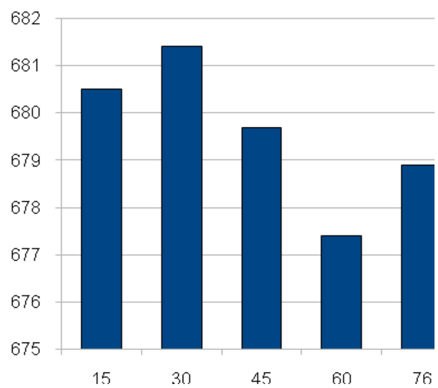


(c) Variation in pheromone trial

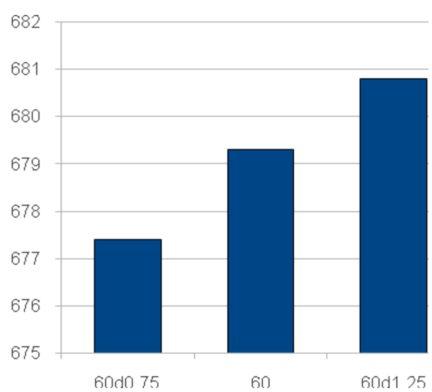
Figure 2: Comparative graph of kMed-ACO for secondary updation.

4.4.2 Density Based ACO

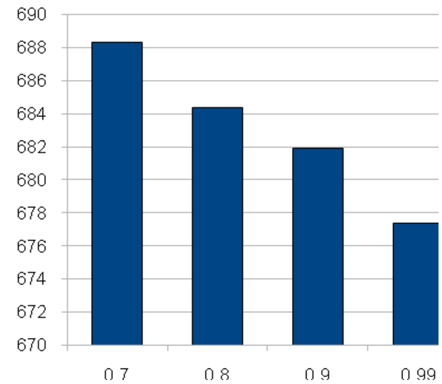
The st70 dataset was selected to assess the behavior of the algorithm. The Figure 3 shows the comparative results of primary updation for DBC-ACO. The Fig 3a shows that DBC-ACO exhibits least variation in obtained solutions compared to other strategies for varying number of ants and obtains best solution, when there are around 60 numbers of ants. The ϵ mean variance of varying size from 0.75 to 1.25 was considered for experimentation purpose. It can be observed from Fig 3b that better results were obtained for smaller size. The Fig 3c shows that quality of solution will improve by retaining most of the past experiences. The Figure 4 shows the comparative results of secondary updation for DBC-ACO strategy. The Fig 4a shows that better quality of solution will be obtained for large number of ants. The Fig 4b and 4c shows that algorithm is not so sensitive to varying ϵ mean variance and pheromone trial.



(a) Variation in number of ants

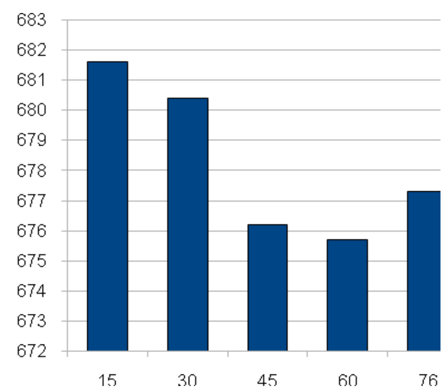


(b) Variation in ϵ mean

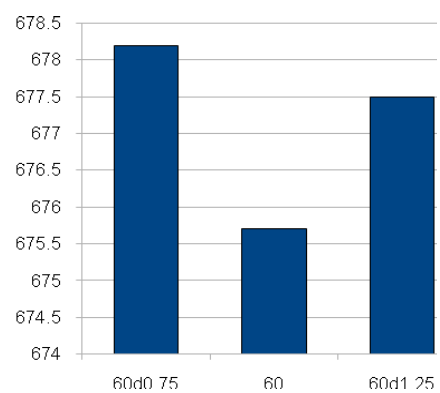


(c) Variation in pheromone trial

Figure 3: Comparative graph of DBC-ACO for primary updation.



(a) Variation in number of ants



(b) Variation in ϵ mean

c) Variation in pheromone trial

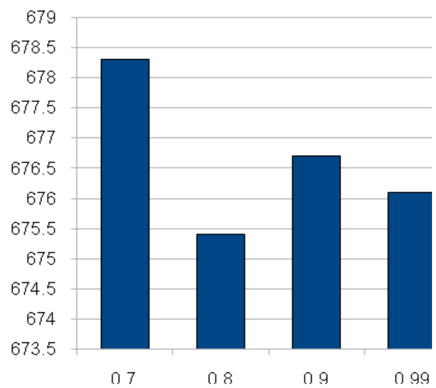


Figure 4: Comparative graph of DBC-ACO for secondary updation.

5. CONCLUSION

In this paper, we have presented some novel approaches to update the pheromone trial. The updation strategies have been incorporated with cluster mechanism that reinforces the logically nearer paths with same amount of pheromone trial. The paper discussed about the incorporation of k-median and DBSCAN strategy and the impact of incorporation on the performance of ACO algorithm. The performance analysis is done with respect to primary updation and secondary updation. For the first time, such a technique has been developed by us that can be used for any randomized search heuristics.

6. REFERENCES

[1] Dorigo,M., Maniezzo, V., and Colorni, A.(1996) Ant System: Optimization by a colony of cooperating agents, IEEE Transaction on Systems, Man and Cybernetics, Vol 26,No.1,pp.29-41.

[2] Bullnheimer, Hartl, R. F., and Strauss, C.(1999) A new rank based version of the Ant System: A computational study, Central European Journal for Operation Research and Economics, Vol 7,No.1,pp.25-38.

[3] Blum, C., Roli, A., and Dorigo, M.(2004) HC-ACO: The Hypercube framework for Ant Colony Optimization, IEEE Transaction on Systems, Man and Cybernetics, Vol 34,No.2,pp.1161-1172.

[4] Gambardella, L.M., and Dorigo, M.(1995) Ant-Q: A reinforcement learning approach to the traveling salesman problem. In Proceedings of 12th International Conference on Machine Learning.

[5] Gambardella, L.M., and Dorigo. M.(1997) Solving symmetric and asymmetric TSP by ant colonies. In Proceedings of IEEE International Conference on Evolutionary Computation. Porto,Spain.

[6] Stutzle, T., and Hoos, H.H.(2000) MAX - MIN Ant System, Future Generation Computer System, Vol 16,No 8,pp.889-914.

[7] Hong-hao, Z., and Fan-lun, X.(2006) A New Approach of Ant Colony Optimization and its Proof of Convergence. In Proceedings of 6th World Congress on Intelligent Control and Automation.

[8] Jun, S., Sheng-Wu and Fu-Ming, G.(2004) A New Pheromone Updating Strategy in Ant Colony Optimization. In Proceedings of 3rd International Conference on Machine Learning and Cybernetics.

[9] Lijie, L., Shangyou, J., and Ying, Z.(2008) Improved Ant Colony Optimization for the Traveling Salesman Problem. In Proceedings of International Conference on Intelligent Computation Technology and Automation.

[10] Masaya, Y., Masahiro, F., and Hidekazu, T.(2008) A New Pheromone Control Algorithm of Ant Colony Optimization. In Proceedings of International Conference on Smart Manufacturing Application.

[11] Naimi, H.M., and Taherinejad, N.(2009) New Robust and efficient ant colony algorithms: Using new interpretation of local updating process, Expert System with Applications, Vol 36,No.1,pp.481-488.

[12] Stutzle, T., and Dorigo, M.(2002) A Short Convergence Proof for a Class of Ant Colony Optimization, IEEE Transaction on Evolutionary Computation, Vol 20,No.3,pp.1-9.

[13] Raghavendra, G. S., and Prasanna K.N.(2008) Relative Reward Pheromone Update Strategy for ACO Algorithms. In Proceedings of MSAST2008, IMBIC,Kolkata,India.

[14] Blum, C.(2007). Ant Colony Optimization: Introduction and Hybridization. In Proceedings of 7th International Conference on Hybrid Intelligent Systems.

[15] MacQueen J (1967) Some methods for classification and analysis of multi-variate observations. In Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp.281-297.

[16] Prasanna, K., and Raghavendra, G.S.(2011) On the Evaporation Mechanism in the Ant Colony Optimization Algorithms, Annales of Computer Science Series, Vol 9,No.1,pp.51-56.

[17] Prasanna, K., and Raghavendra, G.S.(2011) A note on the Parameter of Evaporation Mechanism in the Ant Colony Optimization Algorithms, International Mathematical Forum, Vol 6,No.34,pp.1655-1659.

[18] Wang, W., Yang, J., Muntz, R (1997) STING: A Statistical information grid approach to spatial data mining. In Proceedings of International Conference on Very Large Databases, pp.186-195.

[19] Jones T, Forrest S (1995) Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In Proceedings of the 6th International Conference on Genetic Algorithms, pp.184-192.

[20] Rui X, Donald W (2005) Survey of Clustering Algorithms. IEEE Transaction On Neural Networks, Vol.16,No 3,pp.645-678.

[21] Martin, E., Kriegel, H., Sander, J. and Xiaowei X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), pp.226-231.