Chapter 3

Cluster Integrated Updation Strategies for ACO Algorithms

Chapter 3

Cluster Integrated Updation Strategies for ACO Algorithms

3.1 Introduction

Data clustering is one of the most common and important human activities (Jain et al., 1999). It involves discovering groups and identifying interesting distribution and patterns in the underlying data. Clustering problem is about partitioning the data objects into groups/classes, such that the objects within the group are very similar and the objects across the groups are quite different. Clustering is an unsupervised approach, where no labeled data will be available. The ultimate goal of the clustering is to assign the unlabeled data to labeled classes. The label of 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 clustering process should identify the natural closeness among the data and should group them. The data can belong to only one cluster or more than one clusters. If data belongs to more than one cluster then its association with particular cluster is determined by the degree of membership.

The above discussion leads to the conclusion that there is no universal way to group the data and different clustering algorithms may provide different result to the same data set. In general, selection of clustering algorithms depends on the nature of data and the intended applications. In this chapter, an ant clustering approach which looks close to the original ACO algorithms, but explicitly considers particular clustering model will be discussed. Before we introduce new ant based clustering models, a brief explanation on cluster evolution process and the existing cluster algorithms is required.

3.1.1 Evolution of Clusters

The process of clustering typically comprises of four phases as shown in the Figure 3.1 (Xu, 2005).

1. **Feature Selection** - This phase involves selecting the appropriate features/ attributes that helps in distinguishing different groups present in the data.

2. **Cluster Algorithm Selection** - This phase involves selecting appropriate clustering algorithm that provides good clustering scheme for the given data. The clustering algorithm has two important characteristic features:

   – *Proximity Measure* - A measure that quantifies the *similarity* between two objects.

   – *A Criterion Function* - A function expressed in the form of rules or cost function that acts as a template for good partitioning scheme.

3. **Validation** - In this phase, quality of the clusters are validated based on a specified criteria. A set of test functions (Halkidi et al., 2001) are available in the form of indices to validate the cluster. The clustering algorithms are usually validated for sensitive to parameters like number of clusters, order of presentation of data to algorithms etc. Based on the validation result, an appropriate clustering scheme can be selected, that fits well to the given data.

4. **Result Interpretation** - In this phase, a meaningful insight of data is provided to the user by interpreting the evolved cluster structures. Often a subject expert help might be necessary to interpret the partitioned data.

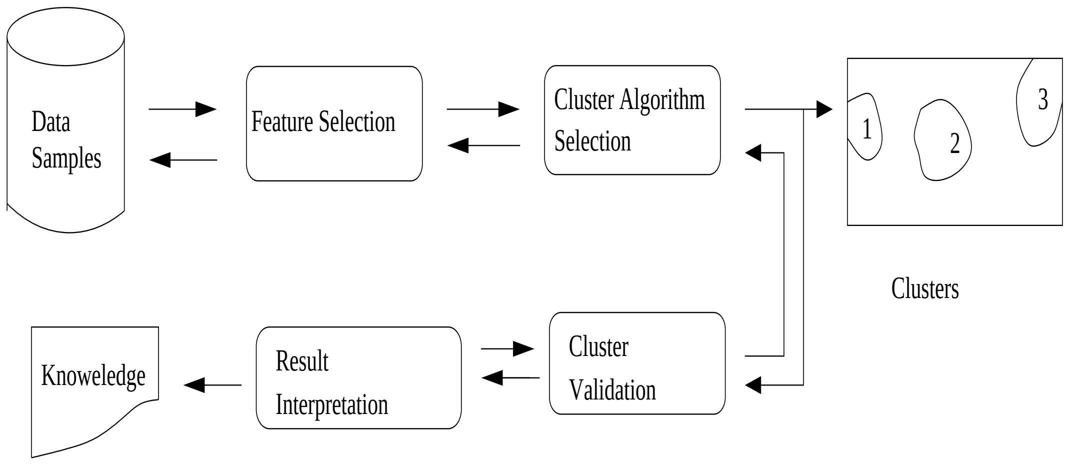


Figure 3.1: Stages involved in the clustering process.

3.1.2 A Categorization of Clustering Methods

In literature, there exists a large number of clustering algorithms and all these algorithms can be classified into the following categories (Han and Kamber, 2004):

- **Partitional Methods** - The Partitional methods create partition of original data set. Given a set of n data, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$. After partitioning, these methods use an *iterative relocation technique* that attempts to improvise the partition by moving the data from one group to another. The algorithm terminates, when there is no movement of data across the partitions. The k-means (MacQueen, 1967) and k-medians algorithms falls under this category. Given a set of data $X = \{x_1, x_2, \cdots x_n\}$ the partitional clustering creates K partition of X into C, where $C = \{C_1, C_2 \cdots C_K\}$, $K \leq N$, such that

  1) $C_i \neq \emptyset$;

  2) $\cup_{i=1}^{k} C_i = X$;

  3) $C_i \cap C_j = \emptyset$; $i, j = 1, 2, \cdots K$ and $i \leq j$

- **Hierarchical Methods** - A hierarchical method creates a hierarchical decomposition of the given set of data. The hierarchical methods can be classified into *agglomerative* or *divisive* depending upon the approach used for constructing the clusters. An agglomerative clustering approach initially considers each data as a separate cluster and then merges the data/group successively in each step depending upon the closeness, until it results in one large group. The divisive clustering approach initially considers all the data as one large group and in successive steps, cluster is split up into smaller one, until each cluster contains single data or termination condition is met. Hierarchical clustering constructs a tree-like nested structure that partitions X into H, where $H = \{H_1, H_2 \cdots H_Q\}$, $Q \leq N$, such that $C_i \in H_m, C_j \in H_l$ and $m > l$ imply $C_i \subset C_j$ or $C_i \cap C_j = \emptyset \ \forall i, j \neq i, m, l = 1, 2, \cdots m$. The BIRCH (Zhang et al., 1997) and CURE (Guha et al., 2001) algorithms fall under this category.

- **Density-Based Methods** - The density based methods work on the notion of *density* and view the clusters as dense region of data in the data space that are separated by low density. These methods start with a random data point and define a radius of a circle (in two dimension) to the selected random point to support the notion of density. The circular radius should include pre-specified number of data points and these included data points are called *neighbors* of random data point. The neighbor data points include random data point form the *neighborhood sets.* The neighborhood set is expanded by adding neighbors of each element of the set and the expansion process stops, when it is not possible to add data points to the set. The above process is repeated until, all the data points belong to one of the neighborhood sets. Thus, a collection of neighborhood sets are obtained and each one of them can be viewed as a cluster. The DBSCAN (Martin et al., 1996) and OPTICS (Ankerst et al., 1999) fall under this category.

- **Grid-Based Methods** - Grid-based methods quantize the data space into finite number of cells that form the part of the grid structure. The clusters are obtained by merging adjacent cells, if both of them contain the pre-specified number of points. The notable differences between grid methods and partition methods are:

– *Evolution of Cluster* - Grid methods follow *divide and merge* approach where as, partition methods follow *divide and distribute* approach to arrive at clusters.

– *Size of the Partitioned Cluster* - The size of the partition in grid method is fixed, but in partition method it is variable.

STING (Wang et al., 1997) is a typical example for grid-based method.

3.2 Clustering Algorithms

In this section, we will discuss some of the important clustering algorithms available in the literature and these algorithms will be incorporated in ACO algorithm.

3.2.1 Statistical Information Grid (STING)

STING (Wang et al., 1997) is a grid based clustering technique. The grid based approaches recursively divide the spatial area into rectangular cells. The partitioned rectangular cells can be represented in hierarchical structure as shown in Figure 3.2. The hierarchical structure represents layers of rectangular cells and each layer represents the data at different level of resolution. The hierarchical structure follows the parent - child relationship in which lower level cells are the partitioned cells of higher level cells. The area of leaf level cells can be controlled based on the density of the data. The parental cells store the statistical information like maximum, minimum, mean and type of distribution etc of the child cells.

The STING method is useful for query answering, where relevant cells are considered for searching data. The statistical information guides the search process and is used to compute the *Confidence Interval* (CI). The CI will reflect the cell's relevancy to the posed query. The cells with highest relevancy will be considered for further processing. The process of finding relevant cells will continue until most relevant leaf cell is found.

The advantages of STING approach lies in the grid architecture that supports parallel processing and hence it is fast in processing the queries. The STING approach goes through the data points only once and hence its time complexity is O(n), where n is the number of data points.

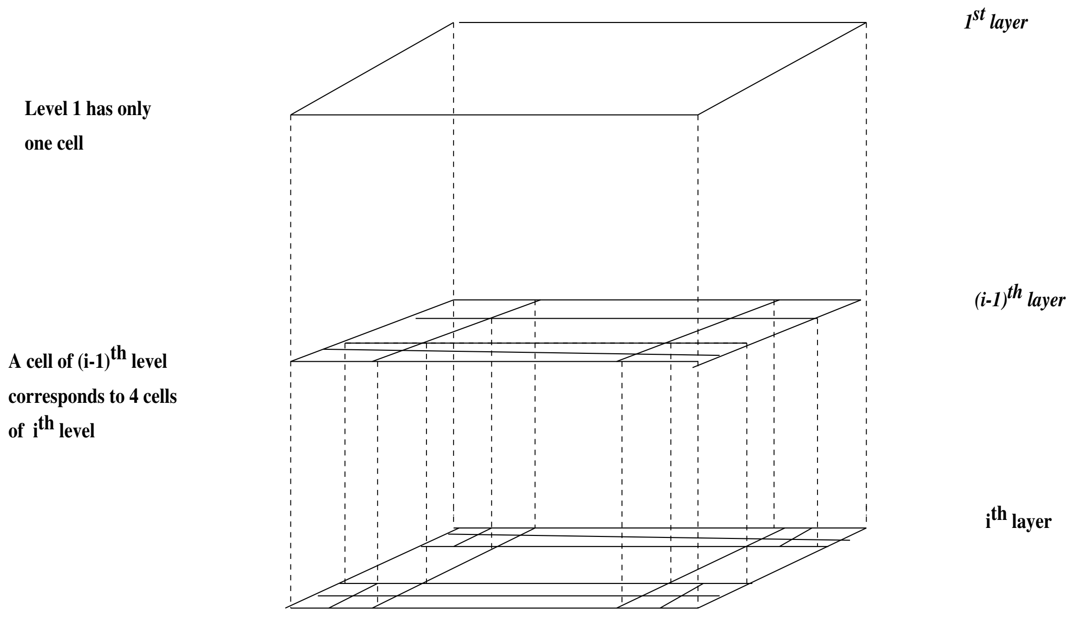

1st layer
Level 1 has only one cell
(i-1)th layer
A cell of (i-1)th level corresponds to 4 cells of ith level
ith layer

Figure 3.2: Hierarchical structure of 2-D data space

3.2.2 k-Means Algorithm

The k-means algorithm is one of the simplest and most commonly used cluster algorithms, proposed by MacQueen (1967). The algorithm creates k partitions of the given n data points. The algorithm employs the *square-error criterion* with the intension to minimize the variance within the clusters and to maximize the variance across the clusters. Typically, squared-error criterion computes the distance between the data points and the center (mean) of all the clusters and then assign the data points to the nearest cluster. The pseudocode for k-means algorithm is given by Algorithm 8:

Algorithm 8 k-Means Algorithm

```
Initialize the number of clusters $k$.
Arbitrarily choose $k$ data points as the initial cluster centers.
while there is no change for each cluster do
  for $j = 1, 2, ...N$ do
    for $i = 1, 2, ...k$ do
      if $||x_j - m_w|| < ||x_j - m_i||$ and $i \neq w$ then
        $x_j \in C_w$
      end if
    end for
  end for
  Recalculate the cluster center (mean) for the current partition.
end while
```

where $x = \{x_1, x_2, ...x_N\}$ is the set of data points. C_w and m_w represent the label of the cluster and mean of the cluster w respectively.

The k-means algorithm is simple and easy to implement. The computational complexity of the algorithm is $O(Nkt)$, where N is the number of data-points, k is the number of clusters and t is the number of iterations and normally $k << N$ and $t << N$. The advantages of k-means algorithm is that, it works well with the compact and hyperspherical clusters. It is efficient in clustering large datasets, since its computational complexity is linearly proportional to the size of the data sets. The drawback of the algorithm includes sensitiveness to the initial centroid selection for the clusters. The convergence of the algorithm depends on initial centroids values and may converge to a local minimum of the criterion function, if the initial centroids are not properly selected. The algorithm is sensitive to parameter k. Ideally, there is no universal way to select the parameter k and its selection affects the shape of the cluster. Often, many trials need to be conducted to select the appropriate value for k, so that high quality of clusters can be obtained. The k-means algorithm is also sensitive to outliers and noise data. If the data point (outlier) is located far away from the cluster centroid, it might be forced to be part of the cluster that affects the natural shape of the cluster.

3.2.3 k-Medians Algorithm

The k-medians algorithm is a variant of k-means algorithm. The k-medians algorithm creates k partitions of data points and each partition is represented by the median that acts as a centroid for set of data points. The algorithm typically employs the *absolute-error criterion* that minimizes the sum of the absolute distances of each data point with respect to centroid of the cluster. The pseudocode for k-medians algorithm is given by Algorithm 9:

Algorithm 9 k-Medians Algorithm

```
Initialize the number of clusters $k$.
Arbitrarily choose $k$ data points as the initial cluster centers.
while there is no change for each cluster do
  for $j = 1, 2, ...N$ do
    for $i = 1, 2, ...k$ do
      if $||x_j - m_w|| < ||x_j - m_i||$ and $i \neq w$ then
        $x_j \in C_w$
      end if
    end for
  end for
  Recalculate the cluster center (median) for the current partition.
end while
```

where $x = \{x_1, x_2, ..., x_N\}$ is set of data points. C_w and m_w represent the label of the cluster and median of the cluster w respectively.

3.2.4 Density Based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN (Martin et al., 1996) is a density based clustering algorithm. The density based approach treats 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 a *core point* otherwise it is a *border point.* The core points which 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 belong 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 $x = x_1, x_2, \cdots, x_i = y$ 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 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 points 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 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 a set of density-connected points and noise as an isolated point that do not belong to any of the cluster. The DBSCAN algorithm works as follows: It starts with an arbitrary point x and finds all the points 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 are formed. If x happens to be the border points, then no points are density-reachable and DBSCAN visits the next unclassified point. The pseudocode for DBSCAN algorithm is given by Algorithm 10:

Algorithm 10 DBSCAN Algorithm

```
Initialize $\epsilon$ and the minimum number of points Min_pts.
Mark all the points as unassigned.
while there are some data points $D$ need to be assigned to the cluster do
  Select a unassigned point $P$ in $D$ and mark it as assigned.
  Set $N$ = neighbor($P, \epsilon$)
  if sizeof($N$) $<$ Min_pts then
    mark $P$ as Noise.
  else
    Create a cluster $C$ and add the point $P$ to it.
    while there are some data points $N$ need to be assigned do
      Select the point $P'$ in $N$ and mark it as classified.
      Set $N'$ = neighbor($P', \epsilon$)
      if sizeof($N'$) $\geq$ Min_pts then
        $N = N \cup N'$.
      end if
      Add the point $P'$ to cluster $C$.
    end while
  end if
end while
```

It should be noted, that DBSCAN needs two parameters ϵ and Min_pts, but identifying the best values for parameters is not easy. A simple heuristics called *k-dist* graph have been 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 value is used to set ϵ. The value of Min_pts is set to k = 4, since k-dist graph will not vary much for higher values.

3.3 Integration of Clusters and ACO

The experimental simulation reveals the existence of correlation between the solution's quality and the distance from good or optimal solutions. In literature, several measures to assess the solution quality can be found and one such measure is Fitness-Distance Correlation(FDC) function. The FDC computes the correlation coefficient (Jones and Forrest, 1995) and describes the goodness of the obtained solutions with respect to global best solution. The high/ low positive value indicates the smaller/ larger distance between obtained solution and the global best solution. Infact, for the problems like TSP (Stutzle and Hoos, 2000), large number of local optimum solutions are concentrated in a small region near the global best solution. In order to exploit the regions near the global best solution, a cluster based updation strategy has been proposed here, 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 group will be reinforced with the same amount of pheromone trial. This mechanism ensures the exploitation of regions near the (best) solutions.
- The reinforcement is done for all the paths, thereby which supports the exploration.

Ideally, best tour in a group will be selected and its performance will be taken as a reference for updating the rest of the paths 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 given by Algorithm 11:

Algorithm 11 Cluster Integrated ACO Algorithm

```
Initialize the parameter $k$ number of cluster, $\tau$, $\eta$, and $\rho$.
$s_{bs} \leftarrow \emptyset$.
while termination condition not met do
  $\chi_{iter} \leftarrow \emptyset$
  for $j = 1, \cdots, n$ do
    $s \leftarrow$ ConstructSolution.
    $s \leftarrow$ LocalSearch (Optional)
    if $(f(s) < f(s_{bs})))$ or $(s_{bs} =$ NULL$)$ then
      $s_{bs} \leftarrow s$
    end if
    $\chi_{iter} \leftarrow \chi_{iter} \cup \{s\}$
  end for
  Identify the clusters $C_1, C_2, \cdots C_k$ in $\chi_{iter}$.
  for $i = 1, \cdots, k$ do
    $s_{bs_i} \leftarrow C_i$
  end for
  for $j = 1, \cdots, k$ do
    while there are some paths left in the cluster $C_j$ need to be updated do
      $TL_{ij} \leftarrow \frac{1}{s_{bs_j}}$
    end while
  end for
end while
```

where χ_{iter} represent set of solutions in current iteration, s_{bs} represent the best solution in current iteration and s_{bs_i} is the best solution in cluster i used for updating paths that belong to cluster i.

3.3.1 Grid Structured ACO (GS-ACO)

The GS-ACO sorts the tour performance in non decreasing order and place them in grid structure. The grid structure is quantized into finite number of blocks. Let $G = \{TL_1, TL_2 \cdots TL_n\}$ represents the grid structure containing n sorted tour length TL_i. The grid structure is divided into m equi-sized partition blocks $B = \{B_1, B_2 \cdots B_m\}$ and $m \leq n$. Figure 3.3 shows the grid structure containing four equi-sized blocks and eight tour performances. It can be observed that each block contains variable number of tour performances and some blocks may be empty as well. The GS-ACO treats each block as a cluster and updates the individual tour paths in the cluster with same amount of pheromone trial as that of best tour path in the block. It should be noted that algorithm degenerates to normal AS algorithm, when the number of blocks is equal to number of ants.

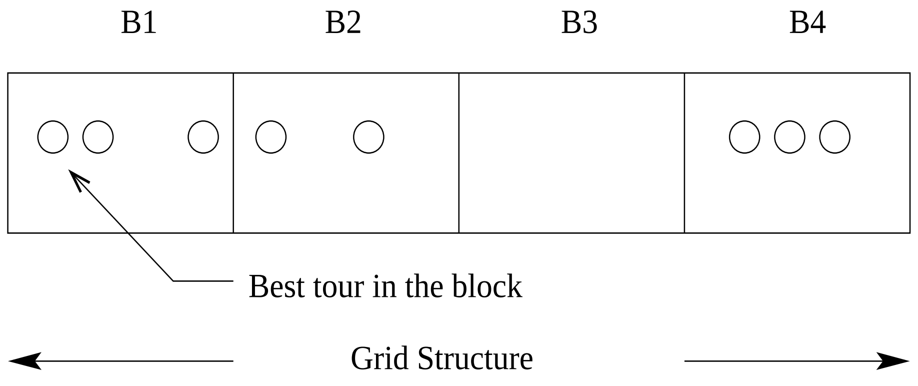


Figure 3.3: Grid structure containing tour lengths.

3.3.2 k-Means ACO (kM-ACO) and k-Medians ACO (kMed-ACO)

In previous section, we discussed about GS-ACO that treats the partitioned blocks as a cluster. Since, number of partitions are fixed as a part of parameter settings, evolved blocks do not look like natural clusters. Consider two blocks $B1$, $B2$ and two points in the Figure 3.3. The first point under consideration is the one located at right end of the block $B1$ and the second one is located at the left end of the block $B2$. The two points may be logically belong to the same group, but the rigid partition scheme puts them in two separate blocks. This necessitates to use the clustering scheme that places the logically near data into same groups. The k-means and k-medians algorithms have been employed to cluster the data. The updation strategy followed is similar to that of grid strategy, where all the tour paths in the cluster will be reinforced with the same amount of pheromone trial as that of best tour path in the cluster.

3.3.3 Density Based Clustered ACO (DBC-ACO)

The problem with the k-means and k-medians algorithms is that the quality of the cluster is 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 of the clusters 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 assess the impact of natural looking cluster, DBSCAN algorithm is integrated with the 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 called ϵ *mean difference* and it is quite similar to ϵ. The ϵ mean difference is derived from the data and it will be used as a distance measure for clustering process. The ϵ mean difference also called as Average Sum of Difference (ASOD) is given by the equation:

$$ASOD = \frac{1}{n}\sum_{i=1}^{n-1} |D_i - D_{i+1}|$$

where n is the number of data objects D_i and $D_1 \geq D_2 \geq \cdots \geq D_n$.

The modified clustering scheme places the adjacent data objects D_i in one group, if they are present within the ASOD distance. The pseudocode for modified DBSCAN algorithm is given by Algorithm 12:

The Modified DBSCAN algorithm is integrated with the ACO to have a new variant called DBC-ACO algorithm. The DBC-ACO groups the nearby tour lengths that are with ASOD distance in order to evolve a set of clusters. The number of clusters evolve will depend on the distribution of tour lengths.

Algorithm 12 Modified DBSCAN Algorithm

```
Compute the ASOD.
Mark all the tour lengths $D$ as unassigned to the cluster.
for $i = 1, 2, \cdots n$ do
    Create a cluster $C$.
    Select a unassigned tour length $D_i$ in $D$ and mark it as assigned.
    $C \leftarrow D_i$.
    while $|D_i - D_{i+1}| \leq$ ASOD or $i \neq n$ do
        $C \leftarrow D_{i+1}$.
        $D_i \leftarrow D_{i+1}$.
        Select the next unassigned tour length $D_{i+1}$ in $D$ and mark it as assigned.
    end while
end for
```

3.4 Experimental Study

3.4.1 Parameter Settings

The cluster incorporated ACO algorithms have additional parameters pertaining to cluster that need to be specified as a part of parameter settings. The GS-ACO, kM-ACO and kMed-ACO need k, the number of clusters as a parameter for clustering process. An extensive simulations were carried out 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.

3.4.2 Primary Updation

In ACO, ants update the traveled path with the trials proportional to the quality of solution after the completion of tours and this process is termed as primary updation. Since clustering mechanism is incorporated in ACO, nearby tour performance will be clustered and then pheromone trial will be updated on the traveled paths with the best performance within the cluster. Table 3.1 shows the comparative results of cluster integrated ACO for primary updation. The general observation is that GS-ACO exhibits larger deviation from optimal solution compared to other variants. In GS-ACO, most of the obtained solutions deviate by more than 1%. The solution obtained for kMed-ACO is slightly better than the kM-ACO algorithm. The DBC-ACO algorithm provides better optimal solution compared to other variants, but it constructs large number of clusters for updation purpose.

Table 3.1: Performance comparison of cluster integrated ACO algorithms for primary updation.

Datasets	Algorithms	Best (Std Dev)	Average (Std Dev)
bays29	GS-ACO	2046.4 (1.30%)	2061.2 (2.03%)
	kM-ACO	2033.5 (0.66%)	2038.5 (0.91%)
	kMed-ACO	2029.4 (0.46%)	2035.3 (0.75%)
	DBC-ACO	2025.4 (0.26%)	2028.2 (0.4%)
att48	GS-ACO	10745.8 (1.10%)	10807.5 (1.68%)
	kM-ACO	10685.6 (0.54%)	10704.4 (0.71%)
	kMed-ACO	10665.1 (0.34%)	10688.6 (0.57%)
	DBC-ACO	10651.6 (0.22%)	10673.4 (0.42%)
eil51	GS-ACO	432.7 (1.57%)	435.1 (2.13%)
	kM-ACO	428.3 (0.53%)	432.9 (1.61%)
	kMed-ACO	427.8 (0.42%)	430.5 (1.05%)
	DBC-ACO	427.1 (0.25%)	429.5 (0.82%)
st70	GS-ACO	684.3 (1.37%)	688.2 (1.95%)
	kM-ACO	681.6 (0.97%)	684.4 (1.39%)
	kMed-ACO	678.2 (0.47%)	681.8 (1.00%)
	DBC-ACO	677.4 (0.35%)	680.5 (0.81%)
eil76	GS-ACO	547.4 (1.74%)	552.3 (2.65%)
	kM-ACO	543.7 (1.05%)	546.7 (1.61%)
	kMed-ACO	540.3 (0.42%)	547.4 (1.74%)
	DBC-ACO	539.4 (0.26%)	543.6 (1.04%)

Datasets	Algorithms	Best (Std Dev)	Average (Std Dev)
Kroa100	GS-ACO	21397.4 (0.54%)	21439.9 (0.74%)
	kM-ACO	21348.5 (0.31%)	21376.2 (0.44%)
	kMed-ACO	21332.4 (0.23%)	21359.4 (0.36%)
	DBC-ACO	21313.5 (0.14%)	21340.4 (0.27%)
kroa200	GS-ACO	29740.5 (1.26%)	29793.7 (1.44%)
	kM-ACO	29545.6 (0.60%)	29580.4 (0.72%)
	kMed-ACO	29466.4 (0.33%)	29503.2 (0.46%)
	DBC-ACO	29408.1 (0.13%)	29443.5 (0.26%)
lin318	GS-ACO	42778.7 (1.78%)	42838.6 (1.92%)
	kM-ACO	42434.2 (0.96%)	42516.9 (1.16%)
	kMed-ACO	42221.2 (0.45%)	42278.2 (0.59%)
	DBC-ACO	42146.7 (0.28%)	42091.8 (0.14%)

3.4.3 Secondary Updation

The proposed approach is extended by incorporating additional reinforcement mechanism. The additional reinforcement is done after primary updation. The additional/ secondary updation provides 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. Table 3.2 shows the comparative results of cluster integrated ACO for secondary updation. On comparing Table 3.2 with Table 3.1, secondary updation strategy improvises most of the solutions. The extended approach provides best solution for att48, eil76, kroa100 and kroa200 dataset for the DBC-ACO algorithm.

Table 3.2: Performance comparision of cluster integrated ACO algorithms for secondary updation.

Datasets	Algorithms	Best (Std Dev)	Average (Std Dev)
bays29	GS-ACO	2040.4 (1.00%)	2055.1 (1.73%)
	kM-ACO	2028.4 (0.41%)	2034.4 (0.71%)
	kMed-ACO	2024.6 (0.22%)	2029.1 (0.45%)
	DBC-ACO	2022.3 (0.11%)	2026.7 (0.33%)
att48	GS-ACO	10711.7 (0.78%)	10756.3 (1.20%)
	kM-ACO	10672.6 (0.42%)	10680.1 (0.49%)
	kMed-ACO	10654.3 (0.24%)	10671.2 (0.40%)
	DBC-ACO	10630.5 (0.02%)	10644.3 (0.15%)
eil51	GS-ACO	430.3 (1.00%)	433.4 (1.73%)
	kM-ACO	429.1 (0.72%)	431.6 (1.31%)
	kMed-ACO	427.1 (0.25%)	431.4 (1.26%)
	DBC-ACO	428.7 (0.63%)	432.7 (1.57%)
st70	GS-ACO	680.1 (0.75%)	685.5 (1.55%)
	kM-ACO	679.4 (0.65%)	682.9 (1.17%)
	kMed-ACO	677.3 (0.34%)	682.5 (1.11%)
	DBC-ACO	675.7 (0.1%)	678.9 (0.57%)
eil76	GS-ACO	544.9 (1.28%)	548.8 (2.00%)
	kM-ACO	541.4 (0.57%)	545.3 (1.35%)
	kMed-ACO	539.4 (0.26%)	544.1 (1.13%)
	DBC-ACO	538.4 (0.07%)	541.2 (0.59%)
Kroa100	GS-ACO	21355.7 (0.34%)	21381.3 (0.46%)
	kM-ACO	21338.4 (0.26%)	21355.8 (0.34%)
	kMed-ACO	21302.1 (0.09%)	21339.5 (0.27%)
	DBC-ACO	21291.6 (0.04%)	21318.3 (0.17%)
kroa200	GS-ACO	29518.8 (0.51%)	29572.4 (0.69%)

Datasets	Algorithms	Best (Std Dev)	Average (Std Dev)
	kM-ACO	29478.1 (0.37%)	29508.8 (0.47%)
	kMed-ACO	29423.6 (0.18%)	29478.6 (0.37%)
	DBC-ACO	29378.5 (0.03%)	29406.4 (0.13%)
lin318	GS-ACO	42481.2 (1.07%)	42570.6 (1.28%)
	kM-ACO	42360.6 (0.75%)	42394.9 (0.87%)
	kMed-ACO	42115.6 (0.2%)	42171.6 (0.33%)
	DBC-ACO	42076.1 (0.11%)	42131.9 (0.24%)

3.4.4 Performance Analysis of Algorithms

In this section, we will discuss the impact of various parameters affecting the performance of the algorithms. A comparative graphs are drawn by varying number of ants, number of clusters and pheromone trial of different strength for each variant.

Performance Analysis of GS-ACO

The Kroa100 dataset was used to assess the behavior of algorithm, since it provides better result compared to other datasets. Figure 3.4 shows the comparative results of primary updation for GS-ACO. It can be observed from the Figure 3.4(a) that, better results are obtained, when number of ants are around 50-70% of the total number of cities. Figure 3.4(b) shows the results for varying number of clusters. The number of ants was set to 50 and the number of the clusters was set to {10, 23, 36, 47}. It can be observed that, as the number of clusters increases, quality of solution improves and better solution was obtained, when number of clusters was around 70% of the number of ants. The pheromone trial is responsible for remembering the past experiences of the ants. In GS-ACO, lower persistent factor leads to poor quality of solution and search stagnation occurs, when pheromone trial strength was 0.7. Figure 3.5 shows the comparative results of secondary updation for grid strategy. One can observe from comparison of Figure 3.4 and 3.5 that quality of solution has improvised across the variation in the number of ants, pheromone trials and for different cluster size. The secondary updation provides better result for the same parameter values as that of primary updation. It can be observed that secondary updation provides poor quality of solution for lower pheromone strength and search stagnation did not occur.

Performance Analysis of kM-ACO

The eil51 dataset was selected to assess the behavior of the algorithm. Figure 3.6 shows the comparative results of primary updation for kM-ACO. Figure 3.6(a) shows that kM-ACO provides better results, when there are around 30 ants. In-order to assess the impact of varying number of clusters as shown in Figure 3.6(b), the number of ants was set to 30 and number of clusters was set to {7, 14, 21, 28}. The better results were obtained, when the number of clusters is 50% of the number of ants. Figure 3.6(c) shows that pheromone persistence has a greater effect on the solution quality and solution quality sharply improvises with the increase in pheromone trial strength and best result was obtained for ρ=0.99. Figure 3.7 shows the comparative results of secondary updation for kM-ACO. The best solution obtained by secondary updation is relatively inferior to the primary updation. However, the Figure 3.7(a) reveals that obtained solutions exhibit least variation for the varying number of ants and provide best solution 30 ants. Figures 3.7(b) and 3.7(c) reveal that better result was obtained, when the number of clusters was 70% of the number of ants and for ρ=0.9.

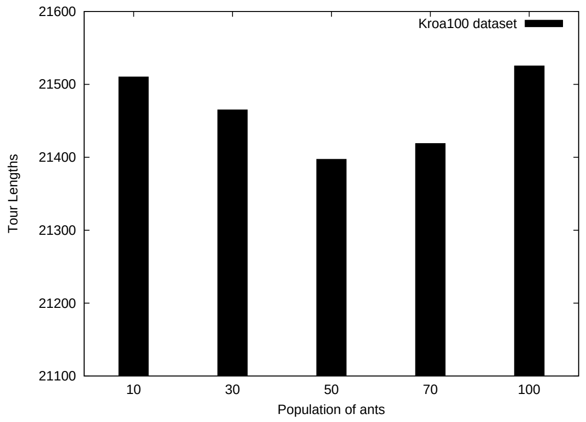
Performance Analysis of kMed-ACO

The eil76 dataset was selected to study the behavior of the algorithm. Figure 3.8 shows the comparative results of primary updation for kMed-ACO. Figure 3.8(a) shows that algorithm exhibits comparatively less variation in solutions for varying number of ants and provides 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 Figure 3.8(b). Similarly, Figure 3.8(c) reveals that quality of solution improvises with the increase in pheromone persistent factor and best result was obtained for ρ=0.9. Figure 3.9 shows the comparative results of secondary updation for kMed-ACO. The kMed-ACO strategy marginally improvises the solution across the varying number of ants and clusters, and provides good solution for higher pheromone trial. The best solution was obtained when the number of ants n=30, number of clusters is 14 and trail strength ρ=0.99.

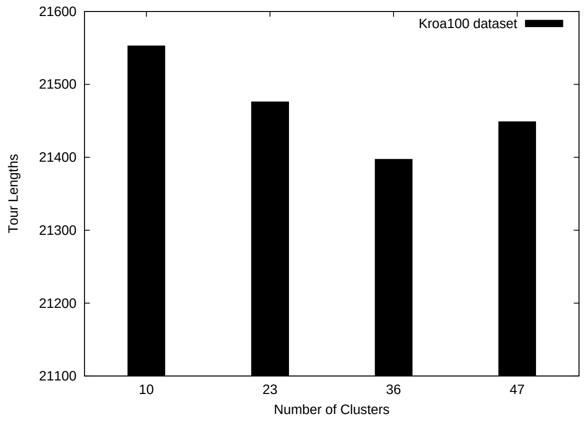
Performance Analysis of DBC-ACO

The st70 dataset was selected to assess the behavior of the algorithm. Figure 3.10 shows the comparative results of primary updation for DBC-ACO strategy. Figure 3.10(a) 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 ants. The ASOD was varied from 0.75 to 1.25 with an increment of 0.25. It can be observed from Figure 3.10(b) that, better results were obtained for smaller value of ASOD i.e., ϵ mean difference = 0.75. Figure 3.10(c) shows that, quality of solution will improve by retaining most of the past experiences and provides best result for ρ=0.99. Figure 3.11 shows the comparative results of secondary updation for DBC-ACO. Figure 3.11(a) shows that better solutions were obtained for large ants population compared to smaller ant population and best result was obtained for n=45. Figures 3.11(b) and 3.11(c) show that algorithm is not so sensitive to varying ASOD and pheromone trial. The best result was obtained for ϵ mean difference = 1 and ρ=0.8.

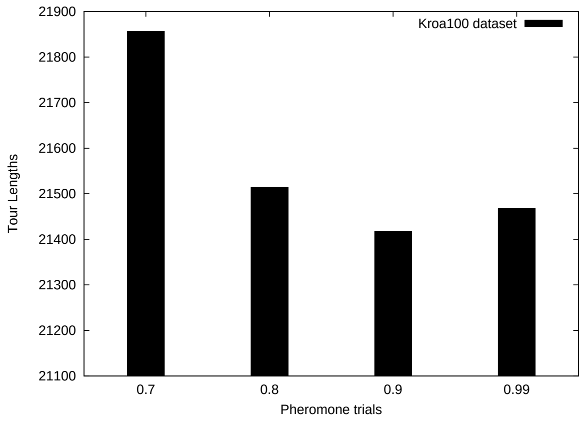
Thus, we analyzed the performance of all the four variants and found that number of clusters, number of ants will affect the performance of algorithm given the higher values of trial strength. However, convergence time could be reduced in special cases when the number of ants is larger like in DBC-ACO or when number of ants is smaller like in kMed-ACO. In a broader sense, the usage or not of clusters have strong impact on parameter settings for ACO algorithms.



(a) Variation in number of Ants

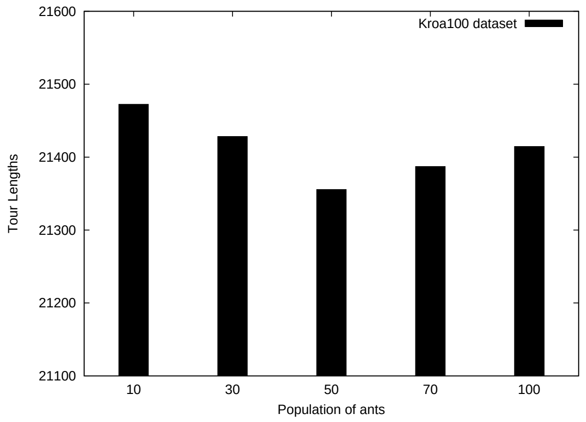


(b) Variation in number of Clusters

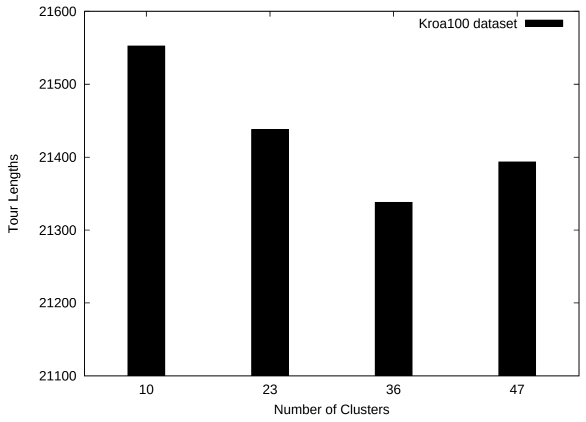


(c) Variation in Pheromone trials

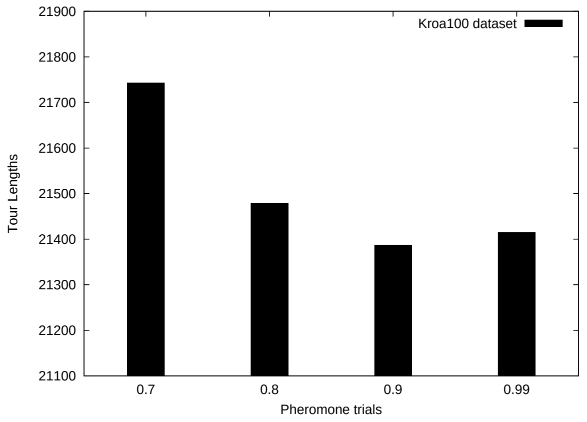
Figure 3.4: GS-ACO algorithm sensitivity to parameters for primary updation.



(a) Variation in number of Ants

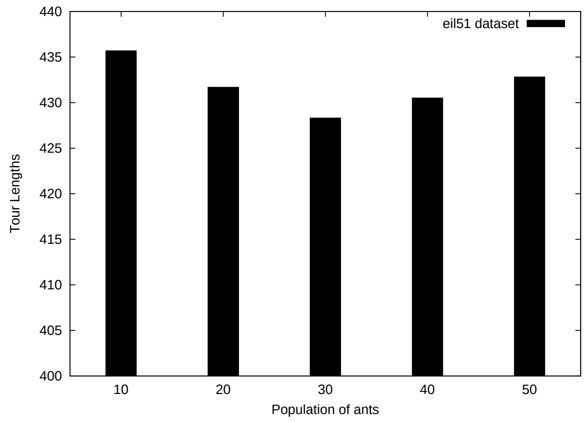


(b) Variation in number of Clusters

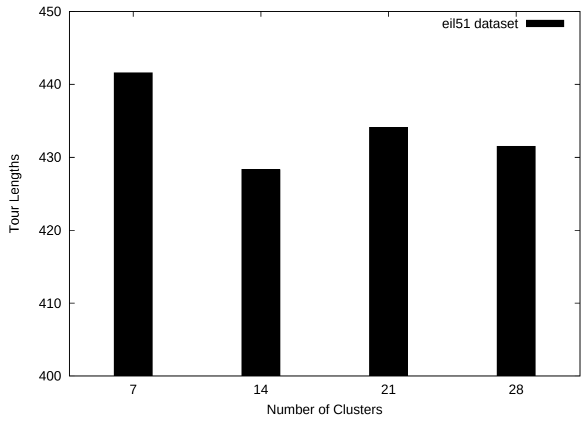


(c) Variation in Pheromone trials

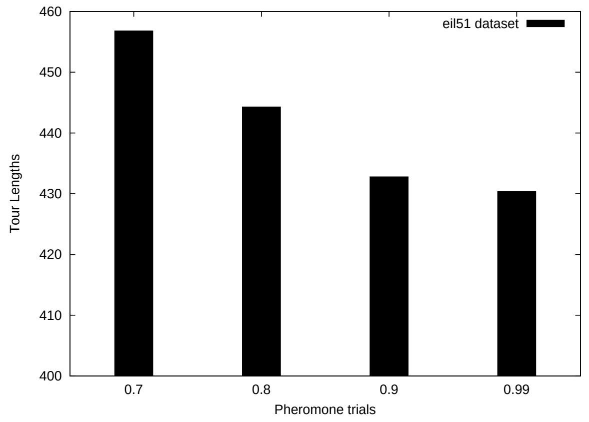
Figure 3.5: GS-ACO algorithm sensitivity to parameters for secondary updation.



(a) Variation in number of Ants

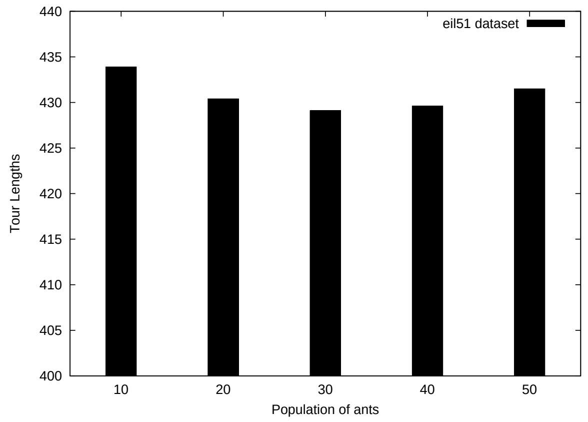


(b) Variation in number of Clusters

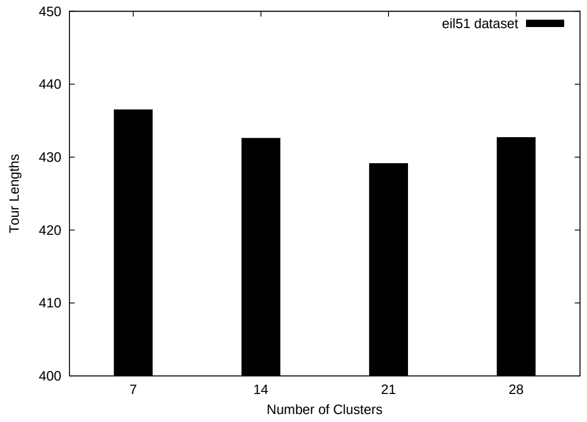


(c) Variation in Pheromone trials

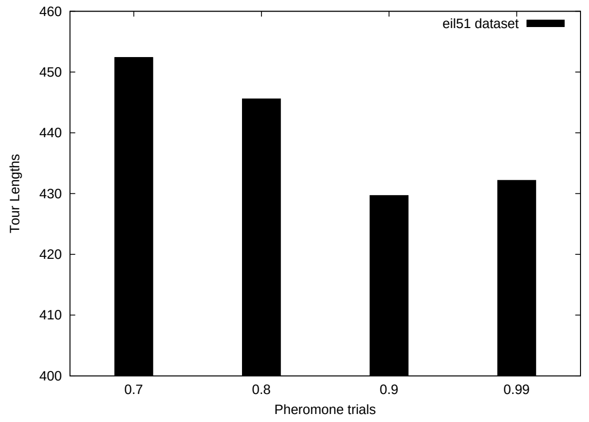
Figure 3.6: kM-ACO algorithm sensitivity to parameters for primary updation.



(a) Variation in number of Ants

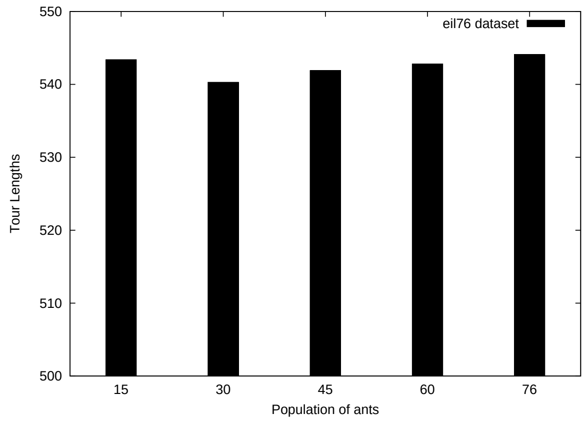


(b) Variation in number of Clusters

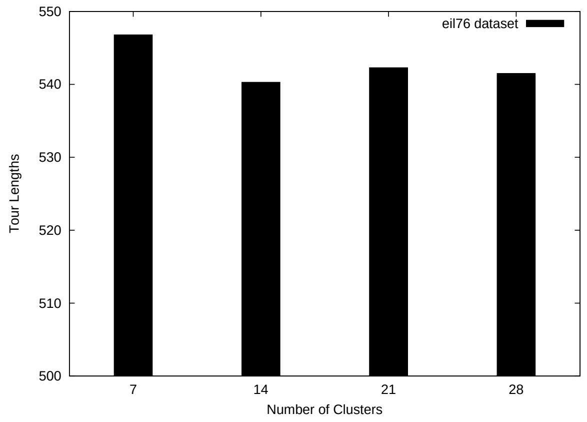


(c) Variation in Pheromone trials

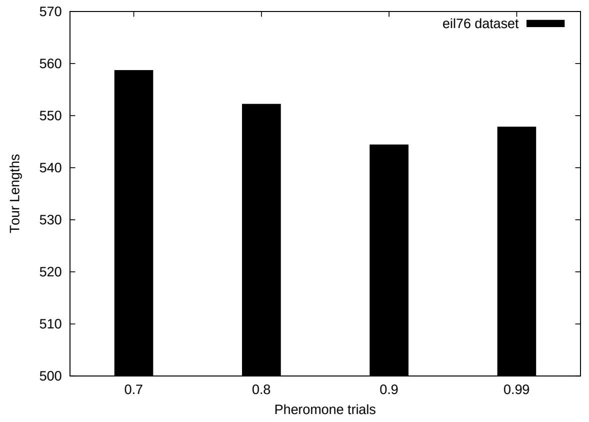
Figure 3.7: kM-ACO algorithm sensitivity to parameters for secondary updation.



(a) Variation in number of Ants

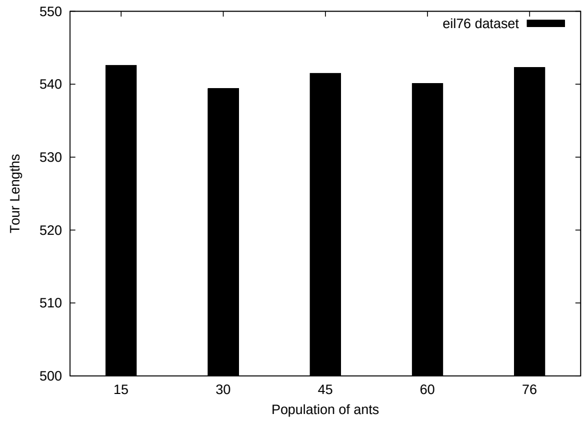


(b) Variation in number of Clusters

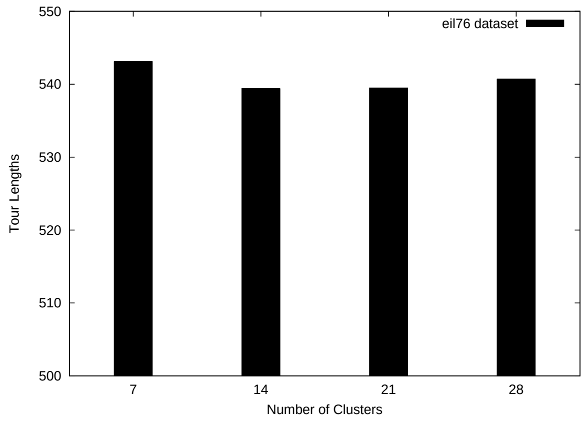


(c) Variation in Pheromone trials

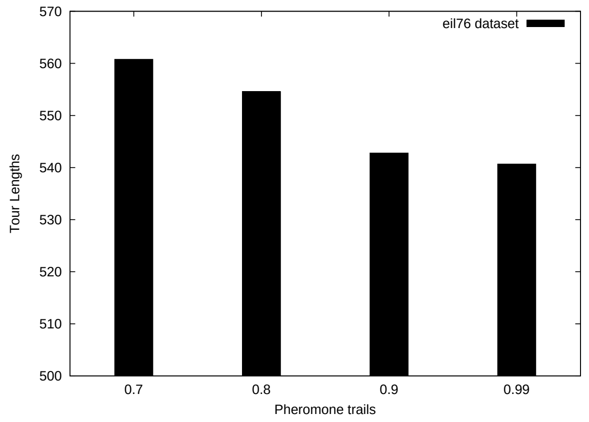
Figure 3.8: kMed-ACO algorithm sensitivity to parameters for primary updation.



(a) Variation in number of Ants

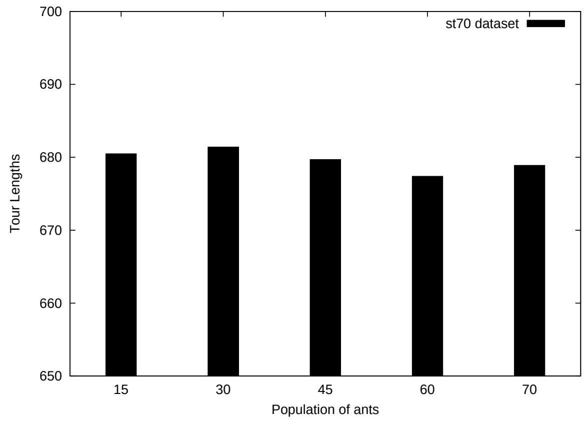


(b) Variation in number of Clusters

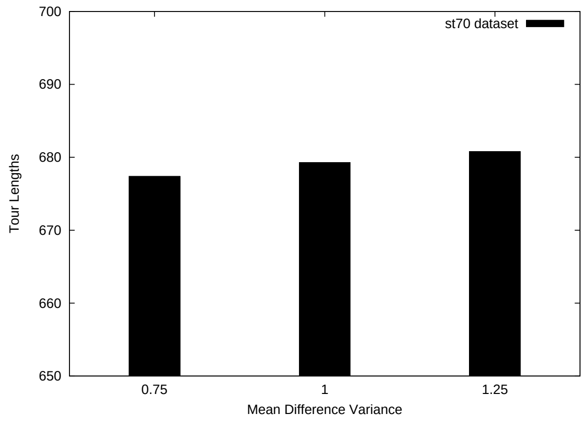


(c) Variation in Pheromone trials

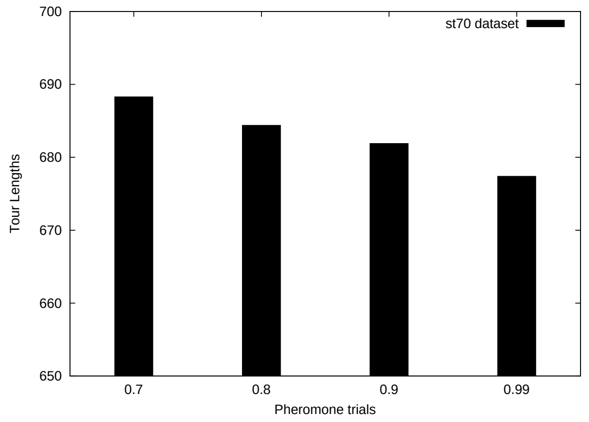
Figure 3.9: kMed-ACO algorithm sensitivity to parameters for secondary updation.



(a) Variation in number of Ants

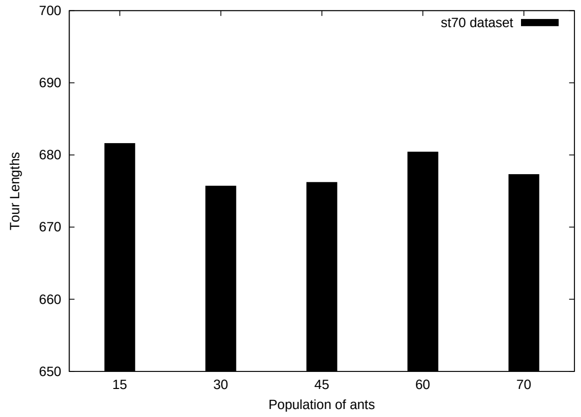


(b) Variation in Mean Difference

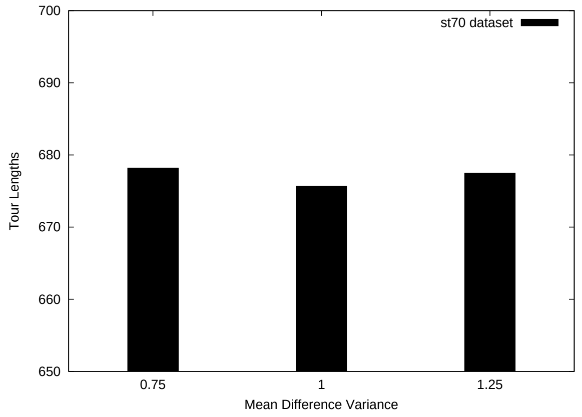


(c) Variation in Pheromone trials

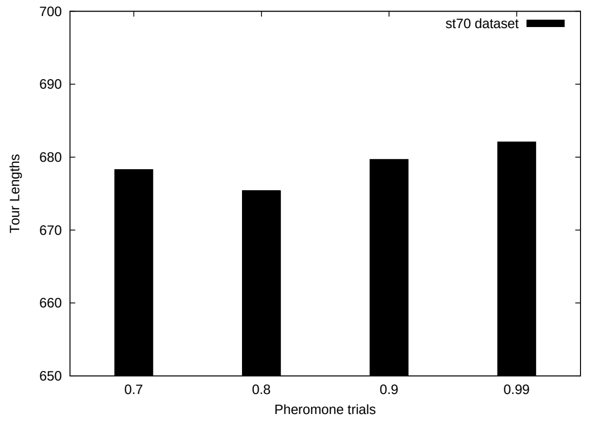
Figure 3.10: DBC-ACO algorithm sensitivity to parameters for primary updation.



(a) Variation in number of Ants



(b) Variation in Mean Difference



(c) Variation in Pheromone trials

Figure 3.11: DBC-ACO algorithm sensitivity to parameters for secondary updation.

3.5 Concluding Remarks

In this chapter, a different framework by coupling unsupervised learning and the ACO algorithms has been established effectively. This technique is introduced first time in the ACO literature. The unsupervised learning approach facilitates the knowledge for ants that will be used for the path updation. The incorporated knowledge guides the ants towards a region that contains the optimal solution and better results were obtained, when it was used along with the normal updation process.

The method exploits the ability of ants to explore the search space, which is evident from the analysis done in the later sections of this chapter. Having heuristics and cluster based procedures, this approach may be easily applied to many combinatorial optimization problems. The computational experiments supporting these variants will be done on train scheduling problem in chapter 5.