CAPACITATED ARC ROUTING PROBLEM:

COLLECTION OF SOLID WASTE AT KWADASO ESTATE, KUMASI

A THESIS SUBMITTED TO THE GRADUATE SCHOOL BOARD

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY,

KUMASI, GHANA.

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

MASTER OF PHILOSOPHY (MPHIL) IN APPLIED MATHEMATICS

BY

DOMINIC OTOO

(BSC MATHEMATICS)

MARCH, 2012

DECLARATION

I hereby declare that this submission is my own work towards Master of Philosophy degree and that to the best of my knowledge, it contains no materials previously published by another person nor materials which has been accepted for an award of any other degree of a university, except where due acknowledgement has been made in the text.

Dominic Otoo	....	....
(PG5071810)	Signature	Date
Student's Name and ID		

Certified by:

Dr. S. K. Amponsah	....	....
Supervisor's Name	Signature	Date

Certified by:

Mr. K. F. Darkwah	....	....
Head of Department's Name	Signature	Date

DEDICATION

I dedicate this work to all students in Ghana. I encourage then to contribute their quota, however small it may be to move the nation (Ghana) forward.

ACKNOWLEDGEMENTS

May the name of Almighty God be praised forever and ever; it is his bountiful endowment of grace through Christ Jesus that kept me through. I would like to acknowledge the support and guidance received from my supervisor, Dr. S. K. Amponsah. I am grateful to Mr. F. K. Darkwah, head of Mathematics department for his encouragement. I would also want to thank operations manager of Zoom Lion Ghana Limited, Kumasi branch and the branch manager of State Housing Company, Kumasi branch for their assistance. I am very grateful to all members of Mathematics department for their assistance. My special thanks go to my wife Angeles Adacorsah Otoo, and children, Abraham, Sylvia, Frederick, Wycleff and Gabriella, for their patience and encouragement throughout my work.

ABSTRACT

Solid waste management in most cities in Africa has become more challenging in which Ghana is not an exception. This is as a result of industrialization and urban migration, it has therefore become necessary to formulate a model which can be used by city authorities, Governments and waste management groups alike to use a minimal distance to collect more waste in an area. It is upon this basis why this work was done to come out with a solution that can be used to minimize the tour in a collection area and also give some sort of flow chart for the collection. In this study, we selected an area in Kumasi called Kwadaso estate which have 157 collection points and 588 240 litre bins. We first, found all pair shortest path and partitioned the entire collection points into smaller clusters based on the capacity of the vehicle and then used Ant Colony Optimization (ACO) to find the minimum tour in each cluster, which will also serve as a flow chart to guide collection in a cluster. Our study has improved the total distance by about 40% as compelled to the existing figures given by the waste management group in the area (Zoom Lion Ghana Limited).

CONTENT **PAGE**

CHAPTER ONE: INTRODUCTION

CHAPTER TWO: LITERATURE REVIEW

CHAPTER THREE: METHODOLOGY

CHAPTER FOUR: DATA COLLECTION AND ANALYSIS

CHAPTER FIVE: CONCLUSIONS AND RECOMMENDATIONS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

1.0 INTRODUCTION

Waste (solid) has been in existence since the creation of Adam. During the early periods of civilisation, solid waste was conveniently disposed of without any problem, since the population was low and there was a large open space.

With the advent of urbanisation and industrialisation, waste generation increased and hence an increase in waste disposal. This increase in waste began to pose health challenges and so urban planners were tasked to find how to dispose of waste from our communities. The collection, transportation and disposal of solid waste involve a large expenditure. Three of the aspects of waste management are the design of efficient route, efficient and economic collection of waste and the location of dumpsites.

The management of urban waste is now undergoing a strong change and innovation phase as required by the growing environmental concern from citizens and governments. As a result, in recent years there has been an increasing number of directives from waste management groups and government. In particular, in 2010 the government of Ghana gave an open invitation to the general public in waste management as to how to effectively collect the ever increasing waste on our streets and homes. Government of Ghana has tasked the various municipal assemblies to collect and dispose of waste in their various municipalities. Kumasi have ten (10) sub metropolitan assemblies, and waste collection in these sub metropolitan assemblies has been assigned to private waste management companies to manage the waste. However, new and interesting management problems arise in each of the sub metros:

What collection system should be applied, where to locate the collection points, how many bins and of what type should be assigned to each house, which are the most appropriate collection routes, which frequency of collection should be applied in each section, how big the fleet of vehicles should be, and so on. In Kwadaso sub metropolitan alone, there are about one hundred and eighty thousand (180,000) inhabitants and the form of garbage collection is of two main types; pay as you dump (community collection) and the bin collection (house to house collection), the enormous challenges outlined above motivated me to write a thesis as to how to collect garbage in one part of Kwadaso sub metro called Kwadaso estate, which have about forty-seven thousand (12,000) inhabitants

(Waste management agency, KMA, 2010)

1.1 BACKGROUND OF THE STUDY

1.1.1 SCOPE OF THE STUDY AREA

Ashanti region is one of the ten regions in Ghana, which is the second highest populated regions in Ghana after greater Accra region. Ashanti region is centrally located in the middle belt of Ghana, it lies between longitudes 0.15W and 2.25W and latitude 5.50N and 7.46N. The region shares boundaries with four of the ten regions, Brong Ahafo to the north, Eastern region to the east, Central region to the south and Western region to the south west. The population of the region is concentrated in a few districts, Kumasi metropolis alone accounts for nearly one-third of the regions population. The high level of urbanization in the region is due mainly to the high level of concentration of the population in the Kumasi metropolis. The region occupies a land area of 24,389 square kilometres representing 10.2 percent of the total land area of Ghana in which Kumasi alone is 250 square kilometers. It is the third largest region after Northern and Central regions. The region has a population density of 148.1 persons per square kilometre and Kumasi has a population of about 1.5 million people. The people of the region are into farming, mining and trading. Tradition is held very high in the region and blends well with modernity. Residential land use in Kumasi forms about 60% of the total land use in the metropolitan area and they are categorised into three zones namely; the low income, middle income and the high income zone. The municipal area has one teaching hospital, 9 hospitals and some few private hospitals and clinics, two public universities and six private universities. To help improve collection and disposal of waste, the metropolitan assembly have divided the metropolis into ten sub metropolitan assemblies and assigned to private waste management groups to manage the waste. (Waste management agency, KMA, 2010)

Because of different levels of wealth in the communities, Kumasi Metropolitan Assembly (KMA) has approved the following rates for garbage collection per month for door-to-door.

- First class: GH¢7.00 per month
- Second class: GH¢6.00 per month
- Third class: GH¢5.00 per month

1.1.2 WASTE MANAGEMENT IN KUMASI

Waste management in Kumasi is a complex issue that has been a major feature on the priority of successive municipal chief executives and waste management groups. Generally, existing facilities including sanitary facility are inadequate to serve the people, the ever escalating volumes of solid waste generated in the Kumasi municipality is overwhelming. Problems are encountered at all levels of waste management namely; poor road network, different housing characteristics making collection in some portion infeasible, increasing waste quantities due to urbanization, inadequate and obsolete waste collection equipment. The situation creates a suitable environment for the bleeding of disease vectors such as mosquitoes, flies, cockroaches and mice. In view of this, some of the inhabitants dispose of rubbish indiscriminately such as drainage channels; in fact the recent advent of polythene bags have even worsen the pride of waste management groups as they are seen everywhere in the city.

1.2 PROBLEM STATEMENT

Waste management has become very complex because of several factors such as finance, robust vehicles for routing, improper road network, haphazard way in which people build houses and so on, and so the stake holders expect to obtain maximum returns from her investment in waste management. Kwadaso estate our study area is not left out in these problems, looking at the volume of waste they generate in a day, two tonnes of waste daily. (W. M. G, KMA).

Some of the challenges with regards to waste collection some in the study area are listed below:

- They do 2 trips daily for 4 days
- The waste management group in the area do not have any laid down route to follow in the collection of waste
- They do not keep track of the mileage during the collection of waste as well as the inter-nodal distances
- They tend to pick any filled bin from a customer even if that customer has been served earlier in the week.
- The routing for picking waste in the area is arbitrary

1.3 OBJECTIVES OF THE STUDY

With the logistical constraints faced by both the government and the waste management groups, it becomes prudent to use the scares resources available to maximize output. The thesis is aimed at:

- To model selection of waste collection points as one centre clustering problem and determine the optimal clustering using vertex 1-centre clustering algorithm
- To model routing of cluster points as Capacitated Clustering Vehicle Routing Problem (CCVRP) and determine optimal cluster routing by Ant Colony Optimization (ACO) algorithm
- To determine optimal total routing for waste collection in Kwadaso Estate.

1.4 METHODOLOGY

Collection of waste from specific points in our streets is an arc routing problem. In this thesis, we shall use ant heuristics to model a formula which can enable us to collect waste on the streets using minimum path and shortest time. To effectively model the problem and solve to achieve our objectives, some organisations in waste management, such as Zoom lion Ghana limited, KMA waste management department and Ghana housing company limited were all contacted for one information or the other, in addition some information and references were obtained from both libraries and on the internet. Kwadaso site layout were used to determine the distances between the collection points from one customer to the with the help of a GIS software from Geomatic department, Kwame Nkrumah University of Science and Technology. Because the vehicle(s) is/are capacitated, our algorithm sort to first cluster the graph to obtain the capacity of the vehicle and then use the ant heuristics to find the shortest tour for collection. Software employed to solve the problem is matrix laboratory (MATLAB).

1.5 JUSTIFICATION

Uncollected waste has enormous consequence on health, economic and social life in general to the residents, the assemblies and the government at large so our thesis is aimed at reducing cost in the collection of waste. This can be achieved if we can have a model which could be used to cluster the area which will eliminate arbitrary routing. Have a model that can help in collecting waste in these clusters using minimum distance. It upon these reasons why our objectives is satisfied.

1.6 LIMITATIONS OF THE STUDY

Africa being a developing continent has its own problems with regard to accessing information from one source or the other for research purposes. Ghana, one of the developing countries in Africa is not an exception to this problem. In our quest to obtain information for our research work, we encountered some challenges some of which are categorised below.

- The right office to go for the required information
- Lack of street naming and numbering of houses
- Improper road network
- Lack of statistics from the waste management group concerning the area under study

1.7 ORGANISATION OF THE THESIS

The organisation of the thesis is as follows; in the first chapter we shall look at the background to solid waste collection and an introduction of the concept of heuristics are presented. Chapter two provides literature review of CARP and its related routing problems. Chapter three provides the mathematical formulation of solving (CARP) using ant heuristics, in chapter four, a real-life garbage collection problem, which exists in one portion of KMA in Ghana is solved using ant heuristics and the final chapter summarizes the main findings of the work and recommendations.

1.8 SUMMARY

In this chapter we looked at an introduction to Arc Routing Problem, the history of the research area, the definition of the problem and its variants. In the next chapter, we shall review pertinent literature in the field of Ant heuristics, CARP and its variants.

CHAPTER TWO

LITERATURE REVIEW

In the first part of this chapter, a review of ant heuristics and examples of heuristic techniques that have been used to solve the VRP for deliveries are presented. This is followed by some literature on VRP on solid waste collection, which includes previous work dealing with solid waste collection such as arc routing, as well as node routing.

2.1 ANT COLONY OPTIMIZATION

Ant Colony Optimization is one of the newest metaheuristic for the application to CO problems. The basic ideas of ACO were introduced in Marco Dorigo, (1992) and successively extended in Dorigo et al., (1999). In this section we present the description of ACO given in Dorigo and Di Caro, (1999). ACO was inspired by the foraging behavior of real ants. This behavior—as described by Deneubourg et al., (1990) enables ants to find shortest paths between food sources and their nest. Initially, ants explore the area surrounding their nest in a random manner. As soon as an ant finds a source of food, it evaluates quantity and quality of the food and carries some of this food to the nest. During the return trip, the ant deposits a pheromone trail on the ground. The quantity of pheromone deposited, which may depend on the quantity and quality of the food, will guide other ants to the food source. The indirect communication between the ants via the pheromone trails allows them to find the shortest path between their nest and food sources. This functionality of real ant colonies is exploited in artificial ant colonies in order to solve CO problems. In ACO algorithms the pheromone trails are simulated via a parametrized probabilistic model that is called the *pheromone model*. The pheromone model consists of a set of model parameters whose values are called the *pheromone values*. The basic ingredient of ACO algorithm is a constructive heuristic that is used for probabilistically constructing solutions using the pheromone values.

In general, the ACO approach attempts to solve a CO problem by iterating the following two steps:

- Solutions are constructed using a pheromone model, that is, a parametrized probability distribution over the solution space.
- The solutions that were constructed in earlier iterations are used to modify the pheromone values in a way that is deemed to bias the search toward high quality solutions.

2.2 Heuristics for delivery problems

Basically the VRP for delivery problems can be defined as delivering goods to a number of customers who have placed orders for a certain quantity of these goods from a central depot. Due to some constraints such as load, distance and time, a single vehicle may not be able to serve all the customers. The problem then is to determine the number of vehicles needed to serve the customers as well as the routes that will minimize the total distance travelled by the vehicles. Many heuristics have been introduced in the literature for searching for good solutions to the problem. For instance the savings algorithm of Clarke and Wright, (1964), the sweep algorithm of Gillett and Miller, (1974), the cluster-first, route-second heuristic of Fisher and Jaikumar, (1981), the path scanning heuristic of Golden, De Armon and Bakers, (1983), and the route-first, cluster-second heuristic of Beasley, (1983). A detailed survey of major developments in heuristics as well as exact algorithms for solving the VRP can be found in the recent paper by Laporte, (2009), but this is a still growing research area.

2.3 WASTE COLLECTION (VRP)

Dealing with a waste collection problem is different from the collection problem as discussed in the previous section. There is an additional constraint that needs to be considered in solving this problem. Instead of returning to the depot to unload the collected goods, in a waste collection problem vehicles need to be emptied at a disposal facility before continuing collecting waste from other customers. Thus, multiple trips to the disposal facility occur in this problem before the vehicles return to the depot empty, with zero waste. A complication in the problem arises when more than one disposal facilities are involved. Here one needs to determine the right time to empty the vehicles as well as to choose the best disposal facility they should go to so that the total distance can be minimized. For example it may not be optimal to allow the collection vehicle to become full before visiting a disposal facility. Increasing quantities of solid waste due to population growth, especially in urban areas, and the high cost of its collection are the main reasons why this problem has become an important research area in the field of vehicle routing. In the next two sections, previous work dealing with waste collection as arc routing problems and as node routing problems are reviewed.

2.4 ARC ROUTING PROBLEMS

Due to the large number of residential waste locations that have to be collected from this collection problem is often dealt with as an arc routing problem, whereas the collection of commercial waste is dealt with as a node routing problem. In this section some of the previous work dealing with arc routing problems for waste collection is reviewed. Chang, Lu and Wei, (1997) applied a revised multi-objective mixed-integer programming model (MIP) for analyzing the optimal path in a waste collection network within a geographic information system (GIS) environment. They demonstrated the integration of the MIP and the GIS for the management of solid waste in Kaohsiung, Taiwan. Computational results of three cases particularly the current scenario, proposed management scenario (without resource equity consideration) and modified management scenario (with resource equity requirement) are reported. Both the proposed and the modified management scenarios show solutions of similar quality. On average both scenarios show a reduction of around 36.46% in distance travelled and 6.03% in collection time compared to the current scenario.

Mourao and Almeida,(2000) solved a capacitated arc routing problem (CARP) with side constraints for a refuse collection VRP using two lower-bounding methods to incorporate the side constraints and a three-phase heuristic to generate a near optimal solution from the solution obtained with the first lower-bounding method. Then, the feasible solution from the heuristic represents an upper bound to the problem. The heuristic they developed is a route-first, cluster-second method.

Bautista and Pereira, (2004) presented an ant algorithm for designing collection routes for urban waste.

To ascertain the quality of the algorithm, they tested it on three instances from the capacitated arc routing problem literature i.e. Golden, DeArmon and Baker, (1983); Benavent et al., 1992; and Li and Eglese, (1996) and also on a set of real life instances from the municipality of Sant Boi del Llobregat, Barcelona. The characteristics of each dataset are presented. Computational results for Golden, DeArmon and Baker, (1983) and Benavent et al., (1992) are within less than 4% of the best known solution, and for Li and Eglese, (1996) dataset up to 5.08%. Mourao and Amado, (2005) presented a heuristic method for a mixed CARP, inspired by the refuse collection problem in Lisbon. The proposed heuristic can be used for directed and mixed cases. Mixed cases indicate that waste may be collected on both sides of the road at the same time (i.e. narrow street), whereas waste for the directed cases only can be collected on one side of the road. The authors reported computational results for the directed case on randomly generated data and for the mixed case on the extended CARP benchmark problems of Lacomme et al., (2002). Computational results for the directed problem, involving up to 400 nodes show the gap values (between their lower bound and upper bound values computed from their heuristic method) varying between 0.8% and 3%. For the mixed problem, comparison results with four other heuristics namely, extended Path-Scanning, extended Ulusoys, extended Augment-Merge and extended Merge are reported. They stated that they were able to get good feasible solutions with gap values (between the lower bound values obtained from Belenguer et al, (2003) and their upper bound values) between 0.28% and 5.47%.

Li, Borenstein and Mirchandani, (2008) solved a solid waste collection in Porto Alegre, Brazil which involves 150 neighbourhood, with a population of more than 1.3 million. They design a truck schedule operation plan with the purpose of minimizing the operating and fixed truck costs. In this problem the collected waste is discarded at recycling facilities, instead of disposal facilities. Furthermore, the heuristic approach used in this problem also attempts to balance the number of trips between eight recycling facilities to guarantee the jobs of poor people in the different areas of the city who work at the recycling facilities. Computational results indicate that they reduce the average number of vehicles used and the average distance travelled, resulting in a saving of around 25.24% and 27.21% respectively.

Mourao, Nunes and Prins, (2009) proposed two two-phase heuristics and one best insertion method for solving a sectoring arc routing problem (SARC) in a municipal waste collection problem. In SARC, the street network is partitioned into a number of sectors, and then a set of vehicle trips is built in each sector that aims to minimize the total duration of the trips. Moreover, workload balance, route compactness and contiguity are also taken into consideration in the proposed heuristics.

Ogwueleka, (2009) proposed a heuristic procedure which consists of a route first, cluster second method for solving a solid waste collection problem in Onitsha, Nigeria. Comparison results with the existing situation show that they use one less collection vehicle, a reduction of 16.31% in route length, a saving of around 25.24% in collection cost and a reduction of 23.51% in collection time.

In some cases, waste collection problems are solved as node and arc routing problems. For example Bautista, Fernandez and Pereira, (2008) transformed the arc routing into a node routing problem due to the road constraint such as forbidden turns for solving an urban waste collection problem in the municipality of Sant Boi de Llobregat, Barcelona with 73917 inhabitants using an ant colonies heuristic, which is based on nearest neighbour and nearest insertion methods. Computational results show that both methods produce less total distance compared with the current routes. In particular, routes from nearest neighbour and nearest insertion travel 35% and 37% less, respectively.

Furthermore, Santos, Coutinho-Rodriques and Current, (2008) presented a spatial decision support system (SDSS) to generate vehicle routes for multi-vehicle routing problems that serve demand located along arcs and nodes of the transportation network. This is mainly due to some streets which are too narrow for standard-sized vehicles to traverse, thus the demand along arcs as well as at network nodes are required for solving waste collection in Coimbra, Portugal.

2.5 NODE ROUTING PROBLEMS

If the location of every collection point is known when solving the waste collection problem then it is a node routing problem. Vehicles will travel from the depot to a customer and then to another customer, etc, to collect waste based on the sequence of visits on the vehicle route. This sequence includes trips to disposal facilities to empty the vehicle and the last visit would be the depot. In the next section, previous work dealing with node routing problems, particularly the skip problems and non-skip problems are reviewed. Note here that Sbihi and Eglese, (2007) have discussed the importance attached to waste management and collection in terms of the "green logistics" agenda.

2.6 ALGORITHMS FOR THE VRP

Since the VRP is an NP-hard problem, many approximation algorithms have been proposed in the literature. These algorithms can be classified into three groups: construction algorithms, improvement algorithms, and metaheuristics.

2.6.1. CONSTRUCTION ALGORITHMS

Construction algorithms are used to build an initial feasible solution for the problem. They build a feasible solution by inserting unrouted customers iteratively into current partial routes according to some specific criteria, such as minimum additional distance or maximum savings, until the route's scarce resources (e.g. capacity) are depleted Cordeau *et al.*, (1999). These types of algorithms are classified as either sequential or parallel algorithms. In a sequential algorithm routes are built one at a time whereas in a parallel algorithm many routes are constructed simultaneously.

2.5.2 SEQUENTIAL CONSTRUCTION ALGORITHMS

Sequential construction algorithms are mostly based on the *Sweep Heuristic* Gillet and Miller, (1974) and the *Savings Heuristic* Clarke and Wright, (1964). In the sweep heuristic, routes are constructed as an angle sweeps the location of nodes on a 2D space. In the savings heuristic, first routes are constructed in a predefined quantity and then new nodes are added to available nodes in order to obtain maximum savings.

Baker and Schaffer, (1986) proposed the first sequential construction algorithm. The algorithm is based on savings heuristic, and starts with all possible single customer routes in the form of depot – i – depot. Then two routes with the maximum saving are combined at each iteration. The saving between customers i and j is calculated as: $s_{ij} = d_{i0} + d_{0j} - G.d_{ij}$ where G is the route form factor and d_{ij} is the distance between nodes i and j.

Solomon, (1987) proposed *Time Oriented Nearest Neighborhood Heuristic*. Every route is initialized with the customer closest to the depot. At each iteration unassigned customer that is closest to the last customer is added to the end of the route. When there is no feasible customer, a new route is initialized.

Solomon, (1987) also proposed *Time-Oriented Sweep Heuristic*. First, customers are assigned to different clusters and then TSPTW problem is solved using the heuristics proposed by Savelsbergh, (1985).

2.6.3 PARALLEL CONSTRUCTION ALGORITHMS

Solomon, (1987) proposed a *Giant-Tour Heuristic*. In this heuristic, first of all, a giant route is generated as a travelling salesman tour without considering capacity and time windows. Then, it is divided into number of routes. Potvin and Rousseau, (1993) proposed parallelization of the *Insertion Heuristics*. Each route is initialized by selecting the farthest customer from the depot as a centre customer. Then, the best feasible insertion place for each not yet visited customer is computed. Customers with the largest difference between the best and the second best insertion place are inserted to the best feasible insertion place. Parallel algorithm in Foisy and Potvin, (1993) also constructs routes simultaneously using the Insertion Heuristics to generate the initial center customers.

Antes and Derigs, (1995) proposed another parallel algorithm based on the Solomon's heuristic. Offers comes to the customers from the routes, unrouted customers send a proposal to the route with the best offer, and each route accepts the best proposal.

2.6.4 IMPROVEMENT ALGORITHMS

Improvement algorithms try to find an improved solution starting from a considerably poorer solution. Almost all improvement algorithms for the VRP use an exchange neighbourhood to obtain a better solution. Exchange of neighbourhood can be intra or inter route Thangian and Petrovic, (1998). While *k*-opt procedure operates within a route, the relocate, exchange, and cross operators operate between routes.

Croes, (1958) introduced *k*-opt approach for single vehicle routes. In this heuristic, a set of links in the route are replaced by another set of *k* links. The Or-Opt exchange originally proposed for TSP by Or, (1976) removes a chain of at most three consecutive customers from the route and tries to insert this chain at all feasible locations in the routes.

In 1-1 exchange procedure connectors between nodes are replaced by connectors between nodes either in the same or in different route. 1-0 exchange move transfers a node from its current position to another position in either the same or a different route.

Christofides and Beasley (1984) proposed the *k*-node interchange for the first time to take time windows into account. In this heuristic, sets M_1 and M_2 are identified for each customer $i. M_1$ denotes the customer *i* and its successor $j. M_2$ denotes two customers that are closest to *i* and *j* on a different route than *i* and *j*. The elements of the sets M_1 and M_1 are removed and inserted in any other possible way. Osman and Christofides, (1994) introduced λ-interchange local search that is a generalization of the relocate procedure. λ, the parameter, denotes the maximum number of customer nodes that can be interchanged between routes.

Potvin and Rousseau, (1995) present two variants of 2-Opt and Or-Opt. For the 2-Opt, they proposed the consideration of every pair of links in different routes for removal. For the Or-

Opt, every sequence of three customers is considered and all insertion places are also considered for each sequence.

Schulze and Fahle, (1999) proposed shift-sequence algorithm. A customer is moved from one route to another checking all possible insertion positions. If an insertion is feasible after the removal of another customer, that customer is removed.

2.7 META HEURISTICS

In order to escape local optima and enlarge the search space, meta heuristic algorithms such as simulated annealing, tabu search, genetic algorithm, and ant colony algorithm have been used to solve the VRP Bräysy and Gendreau, (2001).

2.7.1 SIMULATED ANNEALING

Simulated Annealing (SA) is a stochastic relaxation technique. It is based on the annealing process of solids, where a solid is heated to a high temperature and gradually cooled in order to crystallize Bräysy and Gendreau, (2001). During the SA search process, the temperature is gradually lowered. At each step of the process, a new state of the system is reached. If the energy of the new state is lower than the current state, the new solution is accepted. But if the energy of the new state is higher, it is accepted with a certain probability. This probability is determined by the temperature. SA continues searching the set of all possible solutions until a stopping criterion is reached.

Thangiah *et al.*,(1994) used λ-interchange with λ=2 to define the neighbourhood and decrease the temperature after each iteration. In case the entire neighbourhood has been explored without finding and accepting moves the temperature is increased.

Chiang and Russell, (1996) proposed three different SA methods. First one uses modified version of the *k*-node interchange mechanism and second uses λ-interchange with λ=1. The third is based on the concept of tabu list of Tabu Search.

Tan *et al.*, (2001) proposed an SA heuristic. They defined a new cooling schedule. Thus, when the temperature is high, the probability of accepting the worse is high, when the temperature is decreased according to function given above; the probability of accepting worse is reduced.

Finally, Li and Lim, (2003) proposed an algorithm that finds an initial solution using Solomon's insertion heuristic and then starts local search from initial solution using proposed tabu-embedded simulated annealing approach.

2.7.2 TABU SEARCH

Tabu search (TS) presented by Glover, (1986) is a memory based local search heuristic. In TS, the solution space is searched by moving from a solution *s* to the best solution in its neighbourhood *N*(*s*) at each iteration. In order to avoid from a local optimum, the procedure does not terminate at the first local optimum and the solution may be deteriorated at the following iteration. The best solution in the neighbourhood is selected as the new solution even if it is poorer. Solutions having the same attributes with the previously searched solutions are put into tabu list and moving to these solutions is forbidden. This usually prevents making a move to solutions obtained in the last *t* iterations. TS can be terminated after a constant number of iterations without any improvement of the over all best solution or a constant number of iteration. Garcia *et al.*, (1994) applied TS to solve VRP for the first time. They generate an initial solution using Solomon's insertion heuristic and search the neighbourhood using 2-opt and Or-opt. Garcia et al., (1994) also parallelized the TS using partitioning strategy. One processor is used for controlling the TS while the other is used for searching the neighbourhood.

Thangiah et al., (1994) proposed TS with λ-interchange improvement method. They also combined TS with SA to accept or reject a solution. Potvin *et al.*, (1995) proposed an approach similar to Garcia et al., (1994) based on the local search method of Potvin and Rousseau, (1995). Badeau *et* al., (1997) generated a series of initial solutions. Then, they decomposed them into groups of routes and performed TS for each group using the exchange operator.

2.7.3 GENETIC ALGORITHM

The Genetic Algorithm (GA) is based on the Darwinian concept of evolution. Solutions to a problem are encoded as chromosomes and based on their fitness; good properties of solutions are propagated to a next generation Vacic and Sobh, (2002). The creation of the next generations involves four major phases:

1. Representation: The significant features of each individual in the population are encoded as a chromosome.

2. Selection: Two parent chromosomes are selected from the population.

3. Reproduction: Genetic information of selected parents is combined by crossover and two offspring of the next generation are generated.

4. Mutation: The gene sequence of small number of newly obtained is randomly swapped.

A new generation is created by repeating the selection, reproduction, and mutation phases until a specified set of new chromosomes have been created. Then the current population is set to the new population of chromosomes. Thangiah *et al.*, (1991) applied the GA to VRP for the first time. GA is proposed to find good clusters of customer. The routes within each cluster are then constructed with a cheapest insertion heuristic and λ–interchange are applied.

Thangiah *et al.*, (1995) generate initial population by clustering the customers randomly into groups and applying the cheapest insertion heuristic for each group. Then, 2-point crossover is used. GA of Potvin and Bengio, (1996) is performed on chromosomes of feasible solutions. Parents are randomly selected and two types of crossover are applied to these parents. The reduction of routes is obtained by two mutation operators. The routes are improved using Or-Opt at every *k* iterations.

2.7.4 ACO FOR CAPACITATED VEHICLE ROUTING PROBLEM

Bullneheimer *et al.*, (1998) applied the AS to the VRP with one central depot and identical vehicles for the first time. They set the number of ants (*m*) equal to the number of cities (*n*). Initially, each ant is placed at each customer. Then, ants construct vehicle routes by successively selecting cities, until all cities have been

visited. When there is no feasible city to visit, the depot is selected and a new route is started. City *j* is selected after city *i* according to following random-proportional rule:

$$p_{ij}^k = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}(\mu_{ij})^{\gamma}(\kappa_{ij})^{\lambda}}{\sum_{k \in \text{allowed}_k} (\tau_{ik})^{\alpha}(\eta_{ik})^{\beta}(\mu_{ik})^{\gamma}(\kappa_{ik})^{\lambda}}, & j \text{ allowed }_k \\ 0 & \text{otherwise} \end{cases}$$

$$\eta_{ij} = \frac{1}{d_{ij}}$$

μ_{ij}: Savings of visiting customer *j* after customer *i*

$$\mu_{ij} = d_{i0} + d_{j0} - d_{ij}$$

κ_{ji}: Capacity utilization through the visit of customer *j* after customer *i*

$$\kappa_{ji} = \frac{Q_i + q_j}{Q}$$

Q_i = Total capacity used including the capacity requirement of customer i

γ : Relative influence of the savings

λ : Relative influence of κ_{ji}

After routes are constructed using the proposed approach, 2-opt heuristic is applied to each route. Then, pheromone trail on arc (i, j) is updated according to:

$$\tau_{ij} = \rho.\tau_{ij} + \sum_{k=1}^{m} \Delta_{ij}^{k} + \sigma.\Delta_{ij}^{*}$$

If arc (i, j) is used by the k-th ant, the pheromone trail on that are increased by $\Delta_{ij}^{k} = \frac{1}{L_k}$. In addition, if arc ($i$, j) is on the so far best route, it is emphasized as if σ elitist ants used it. Each elitist ant increases the pheromone trail by $\Delta_{ij}^{*} = \frac{1}{L_*}$

STEP I: Initialize
STEP II: For I^{max} iterations do:

(a) For each ant $k = 1, \ldots, m$ generate a new solution

(b) Improve all vehicle route using the 2-opt heuristic

(c) Update the pheromone trial

Figure 2.1 A skeleton ACO algorithm applied to CVRP

Bullneheimer *et al.*, (1999) introduced an improved ACO algorithm for the VRP with one central depot and identical vehicles (Figure 2.1). Differences of this approach from

Bullneheimer *et al.*, (1998) are in random proportional rule and pheromone trail update. However, following parametrical savings function is used for the visibility:

$$\eta_{ij} = d_{i0} + d_{j0} - g.d_{ij} + f\left|d_{i0} - d_{j0}\right| = s_{ij} - (g-1)d_{ij} + f\left|d_{i0} - d_{j0}\right|$$

After an artificial ant has constructed a feasible solution, ants are ranked according to solution quality. Only the best ranked and elitist ants are used to update the pheromone trails. This update is done using equation $\tau_{ij} = \rho.\tau_{ij} + \Delta\tau_{ij} + \Delta\tau_{ij}^{*}$. They also used candidate lists for the selection of customers. Candidate lists are formed using nearest neighborhood.

Bell and McMullen, (2003) used ant colonies to solve the CVRP. Differences of this approach from Bullneheimer *et al.*, (1998) are in selection the next customer and pheromone trail update. Candidate lists are also formed using nearest neighbourhood. Selection of the next customer j is made using ACS approach. Thus, using equations $p_{ij}^{k} = \begin{cases} \arg \max_{j \text{ allowed}_i} \left\{\tau_{ij}.[\eta_{ij}]^{\beta}\right\}, & \text{if } q \le q_0 \\ P_{ij}^{k}, & \text{otherwise} \end{cases}$, where q is a random variable uniformly distributed over [0 1] and $q_0 \in [0, 1]$ is a parameter and $p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}].[\eta_{ij}]^{\beta}}{\sum_{k \text{ allowed}_j} [\tau_{ik}].[\eta_{ik}]^{\beta}}, & j \in \text{allowed}_k \\ 0, & \text{otherwise} \end{cases}$, each ant may either follow the most favourable path or randomly select a path to follow based on a probability distribution. Trail updating includes local updating of trails after each selection and global updating of the best solution route after all routes are constructed. These are respectively done with the following equations:

$$\tau_{ij} = (1-\rho).\tau_{ij} + \rho.\tau_0$$
$$\tau_{ij} = (1-\rho).\tau_{ij} + \rho.L^{-1}$$

Doerner *et al.*, (2001) proposed the savings based ant system approach (SbAS). The basic structure is identical to Bullneheimer *et al.*, (1999), but they use the savings algorithm to calculate visibility. The attractiveness is calculated by: $\xi_{ij} = (s_{ij})^{\beta}(\tau_{ij})^{\alpha}$

where s_{ij} is the savings of visiting customer *j* after customer *i*. Initially attractiveness values are sorted in non-increasing order and *k*-best combinations are considered at each decision step. If $allowed_k$ denotes the set of *k* feasible combinations (*i*, *j*) yielding the largest ξ_{ij}, the decision rule is given by:

$$p_{ij} = \begin{cases} \dfrac{(\xi_{ij})}{\sum\limits_{k \in \text{allowed}_k} (\xi_{ij})}, k \in \text{ allowed}_k \\ 0 \qquad\qquad\qquad , \text{otherwise} \end{cases}$$

After solutions are constructed, only the best ranked and elitist ants are used to update the pheromone trails.

2.8 Clustering analysis

The classification of objects into different groups sharing the same characteristics is termed as clustering. Clustering is a common technique for data mining, image analysis, biology and machine learning. Techniques which search for separating data in to convenient groups or clusters are termed as clustering analysis Everitt, (1974).

Most markets as well as customers are heterogeneous in their needs and preferences Clarke, (2009). In industrial markets, suppliers must carefully consider the nature and characteristics of their customers in order to satisfy them Hosseini, Maleki & Gholamian, (2010). Segmentation as a technique for forming customer groups for effective targeting is a widely researched area in marketing Simkin, (2008). Cluster analysis is a popular tool to segment markets. Simply stated, it is a technique for separation of customers into different groups such that each group of customers is collectively different from the customers in the other groups. Many methods of cluster analysis are available in the literature. But on a broad basis, clustering techniques can be divided into two groups classical (hard or deterministic) cluster analysis and probabilistic (fuzzy or soft) cluster analysis Budayan, (2008). A number of studies carried out in different fields compare the performance of these two different clustering approaches Budayan, Dikmen, & Birgonul, (2008). In a majority of these comparison studies, fuzzy clustering is discussed as the most popular form that has been adopted in diverse fields, presumably because it adds valuable diagnostics over hard clustering Ozer, (2001). The purpose of this paper is to introduce a relatively unexplored field of soft clustering technique for market segmentation. This technique is known as probabilistic-D clustering Israel and Iyigun, (2008).

2.6.5.1 HARD CLUSTER ANALYSIS

The term “hard cluster” analysis refers to all clustering techniques where the assignment of observations to cluster is deterministic. Stated differently, in hard clustering techniques each observation has 100% chance of belonging to one and only one cluster. There are two main groups of clustering methods, hierarchical and non-hierarchical clustering, each with many different sub-methods and algorithms. In agglomerative hierarchical methods, each observation is initially assigned to its own cluster and then merged with others based on a similarity measure. The algorithm continues until all data points form a single cluster solution.

In non-hierarchical methods such as k-means, an iterative partitioning algorithm is used that does not impose a hierarchical structure Budayan, (2008). We selected k-means, one of the most widely used clustering methods for segmentation, to compare with probabilistic-D clustering.

2.8.2 K-MEANS CLUSTER ANALYSIS

k-means cluster analysis is one of the most popular hard cluster analysis techniques Blattberg et al., (2008). In a classic application of this technique, the number of clusters k must be pre-specified. The algorithm then selects cluster centers and each of the observations in the data is assigned to a particular cluster based upon the shortest Euclidean distance of the data point from the cluster centers. It is an iterative procedure; once observations are assigned to cluster centers, new cluster centers are created by averaging the observations assigned to a cluster. Distances from these new cluster centers are calculated for all observations, and the assignment of observations to clusters continues until a convergence criterion is satisfied Budayan, (2008). This method has a number of advantages, such as its ability to handle large amounts of data points, and its ability to work with compact clusters (Budayan, 2008). However, it has its own set of limitations as well, such as the variables must be commensurable Blattberg et al., (2008), the number of clusters should be known beforehand, and it is sensitive to outliers and noise Budayan, (2008). In recent years, algorithms have been developed for an automatic (multi-stage) way of selecting the number of clusters, the k in k-means.

2.8.3 SOFT CLUSTER ANALYSIS

The term “soft cluster” analysis refers to all clustering techniques where assignment of observations to clusters is chance-based. In other words, in soft clustering techniques there is a chance that each observation could belong to any of the clusters. Thus, the probabilistic clustering technique assigns probabilities of cluster memberships to each observation; therefore, it is not deterministic. Soft clustering techniques overcome the limitation of forceful assignment of an observation to a single cluster and hence are more appealing in business situations where segments may not be clearly differentiable and may be overlapping in character Chuang, Chiu, Lin, and Chen, (1999). Fuzzy C means clustering is the most commonly known type of soft clustering. However, we discuss here a relatively new and a simpler method of soft clustering, as described below.

2.8.4 PROBABILISTIC-D CLUSTER ANALYSIS

As per Israel and Iyigun, (2008), in probabilistic-D (distance) clustering, “given clusters, their centers, and the distances of data points from these centers, the probability of cluster membership at any point is assumed inversely proportional to the distance from the center of the cluster in question.”

If, $P_k(x)$ = probability that the point x belongs to cluster Ck.

$d_k(x)$ = distance of point x from cluster C_k

Then: $P_k(x).\, d_k(x)$ = constant, depending on (x).

The clustering criterion being used here is Euclidean distances.

Mathematically as per Iyigun and Israel (2010): $P_k(x) = \dfrac{\prod_{j \neq k} d_j(x)}{\sum_{i=1}^{k} \prod_{j \neq i} d_j(x)}$

Probabilistic-D clustering has all the advantages of generic soft clustering techniques over hard clustering techniques such as k-means. Fuzzy C Means (FCM) cluster analysis is the most well known and widely researched technique in soft clustering Ozer, (2001). The main differences between FCM and probabilistic-D clustering is that while FCM determines the cluster centers as well as the distances between the cluster centers and observations simultaneously, in Probabilistic-D clustering the cluster centers are determined first. Then, based on those cluster centers, the distances (Euclidean) are calculated to assign probabilities of cluster membership. Our motivation to look for an approach other than FCM is as follows. First, FCM is known to be slow to converge, especially with large data sets Chuang et al., (1999). Second, in spite of our best efforts, we could not find a macro or algorithm to readily apply FCM using SAS®. Israel and Iyigun, (2008) argue that probabilistic-D clustering is a simpler process, is robust and gives a higher percentage of correct classifications. From a SAS® user point of view, application of probabilistic-D clustering should be easier because it can be built upon the familiar k-means output by extracting the distances from cluster centers and then using those distances to calculate the probabilities of cluster memberships.

CHAPTER THREE

METHODOLOGY

3.0 INTRODUCTION

In this chapter, we shall consider the methodology, mathematical formulations of vehicle routing problem in solid waste collection. The problem would be solved in three formulation stages

(i) we first find all pair shortest path by the use of Floyd Warshall algorithm;

(ii) secondary, we cluster the area based on the capacity of the collection vehicle, and;

(iii) thirdly, we use ant heuristics to find the shortest path

3.1 CAPACITATED CLUSTERING PROBLEM (CCP)

In real life, there is the need for moving goods/services from the service providers to various geographically dispersed points of service requesters. The final cost of delivery of service depends on transportation and routing. The requesters are grouped based on their needs/demands with optimal number of clusters and minimum cost of each service delivery. The provider has a lot of constraints, in delivering their service. These constraints include capacity of cluster, delivery cost, and number of clusters. The optimal consolidation of customer's orders into vehicle shipment is an important problem in logistics. This arises in a variety of applications like grouping order into load that fills the vehicle. The vehicle is then assigned to deliver the customer orders to each group from a single service provider. A provider can be a post office, solid waste group, etc. that initiates the service. An example of clustering with single service provider is shown in Figure 3.1 with three clusters of service requester and vehicles used for the service.

Clustering is a difficult combinatorial problem. Clustering algorithm can be hierarchical or partitioned. Hierarchical algorithms find successive clusters using previously established clusters, whereas partitioned algorithms determine all cluster at once.

Figure 3.1: An example of Capacitated Clustering

Another important property is whether the clustering uses symmetric or asymmetric distance. An important step in clustering is to select a distance measure that determines the similarity between items. This influences the shape of the cluster. The service requesters are clustered depending on their demands and distance. This forms the basis for determining optimal routing of transportation problem. The service providers are limited with the capacity of goods/service to be transported/ collected. This capacity limit is taken for the formation of clusters, often formulated as CCP. The CCP is a NP-Complete and Combinatorial Optimization Problem. The CCP is a special case of facility location problem and closely related to generalized assignment problem Geetha et el.,(2009). A simple facility location problem is the Fermat-Weber problem, in which a single facility is to be placed, with the only optimization criterion being the minimization of the sum of distances from a given set of point sites. More complex problems considered in this discipline include the placement of multiple facilities, constraints on the locations of facilities, and more complex optimization criteria. When deciding where to place a facility that serves geographically scattered client sites - whether the facility is a delivery centre, a distribution centre, a transportation hub, a fleet dispatch location, etc - a typical objective is to minimize the sum of the distances from the facility's location to the client sites. Definition: The CCP is defined as grouping '*n*' items into k clusters to minimize the route cost/distance with the specified capacity constraint.

3.2 PROBLEM FORMULATION

The CCP is considered to have *n* customers, whose demands are known and are distributed in the direct distance d_i. The *n* customers are grouped to form *k* clusters. Each cluster has $n_1, n_2, n_3, . . ., n_k$ number of customers with the condition that $\sum_{j=1}^{k} n_j = n$ where ***n*** is the total number of customers.

The problem is given with a set of

Customers : $r_1, r_2, r_3, . . ., r_n$

Distances : $d_1, d_2, d_3, . . ., d_n$

Demands : $q_1, q_2, q_3, . . ., q_n$

Capacity : Q

where $r_i \in R$ are the set of customers who are distributed in the direct distance (d_i), the demand (q_i) and capacity (Q) of cluster are positive integers.

Let X be a binary matrix, such that

$$x_{ij} = \begin{cases} 1, & \text{if customer } i \text{ is assigned to cluster } j \\ 0, & \text{otherwise} \end{cases}$$

The objective is to find X, which minimizes

$$\sum_{j=1}^{k}\sum_{i=1}^{n} d_{ij}x_{ij} \text{...}(3.2.1)$$

Subject to

$$\sum_{j=1}^{k} x_{ij} = 1,\ i = 1, 2, 3, \ldots, n \text{...}(3.2.2)$$

$$\sum_{i=1}^{n} d_i x_{ij} <= Q, \quad j = 1, 2, 3, \ldots, k \text{....................................}(3.2.3)$$

Where d_{ij} represents the closeness distance of customer i to the cluster j.

The objective function (3.2.1) strives to minimize the total distance of customers to the cluster. Constraint (3.2.2) ensures that each customer i is assigned to only one cluster j. The constraint (3.2.3) is to restrict that the total demand of the customer in the cluster should not exceed the cluster capacity Q.

3.2.1 ALGORITHM FOR THE PROPOSED WORK

In this study, the CCP is solved using one centre algorithm, which includes capacity as one of the constraints for clustering the loads along direct distances based on load with minimum distance from the cluster centre.

3.2.2 VERTEX 1- CENTRE CLUSTERING

The one centre algorithm assigns each point to the cluster whose centre (are also called as centroid) is nearest by the use of a priority measure to select the customers for a cluster. The customers are assigned to the nearest cluster based on maximum demand and minimum distance so the requester having larger demand are assigned to the cluster first and the requester with smaller demand can be easily packed in to other clusters. If customers are assigned based on distance alone, the number of clusters formed may not be optimal since customers with smaller demand may be assigned to the cluster before the customer with larger demand, which may lead to the formation of additional cluster.

3.2.3 VERTEX 1-CENTRE ALGORITHM

The major steps involved in the formation of the algorithm are described in the following section.

Calculate the number of clusters

It is calculated based on the demand (q_{i}) of the customer and capacity of cluster (Q) as

$$k = \sum_{i=1}^{n} \frac{d_i}{Q} \qquad \text{...(4)}$$

Select initial centroids

The initial k centroids are selected by arranging the customers based on their demand in their non-increasing order $q_1 > q_2 > q_3 > \ldots > q_n$. Then the first k customer with the highest loads becomes k centroids.

Assign the customer

The road (direct) distance between each requester to all the k centroids are found. Group all the customer r_i to the closest centroid j. To find the appropriate centroid j for r_i, we calculate a priority value as,

$$\text{Priority: } P_i = \frac{d_{ij}}{q_i} \quad \text{.............................(5)}$$

This priority determines the r_i which has the highest priority of having the centroid j. The selected r_i is assigned based on constraint (4). If constraint (4) is satisfied the selected r_i will be assigned to the next nearest centroid based on (5) and (4).

Convergence Criteria

The iterative procedure is repeated until there is no change in cluster formed.

3.3 ANT ALGORITHMS

Ant algorithms are one of the examples of swarm intelligence in which scientists study the behaviour patterns of bees, termites, ants, and other social insects in order to simulate processes. Ant algorithms were first proposed by Dorigo *et al.*, (1991) as an approach to solve combinatorial optimization problems like the travelling salesman problem (TSP) and quadratic assignment problem (QAP). Then, they have been applied to various other problems. In this section, we shall apply the ACO approach to solve capacitated clustering vehicle routing problems (CCVRP). The classical Vehicle Routing Problem (VRP) involves a set of delivery customers to be serviced by a fleet of vehicles housed at a central depot. The objective of the problem is to develop a set of vehicle routes originating and terminating at the depot such that all customers are serviced, the demands of the customers assigned to each route do not violate the capacity of the vehicle that services the route, and the total distance traveled by all vehicles is minimized. CCVRP is a variant of the VRP where the vehicles are not only required to deliver goods to customers but also to pick up goods from the customers.

3.3.1 REAL ANTS

The basic idea of ACO algorithms was inspired through the observation of swarm colonies and specifically ants Beckers et al., (1989). Insects like ants are social. That means that ants live in colonies and their behaviour is directed more to the survival of the colony as a whole, rather than to that of a single individual. Most species of ants are blind. However, while each ant is walking, it deposits on the ground a chemical substance called pheromone Dorigo and Caro, (1999). Ants can smell pheromone and when choosing their way, they tend to choose, in probability, paths with high pheromone density. The ants using the pheromone trail have the ability to find their way back to the food source. Then pheromone evaporates over time. It has been shown experimentally by Dorigo and Maniezzo, (1996) that the pheromone trail following behaviour can affect the detection of shortest paths. For example, a set of ants built a path to some food. An obstacle with two ends was then placed in their way, such that one end of the obstacle was more distant than the other. In the beginning, equal numbers of ants spread around the two ends of the obstacle. Since all ants have almost the same speed, the ants which chose the path of the nearer end of the obstacle returned before the ants that chose the path of the farther end (differential path effect). The amount of pheromone deposits by the ants on the shortest path increases more rapidly than the farther one and so, more ants prefer the shortest path. Finally, in time, the pheromone of the longest path evaporates and the path disappears. This cooperative work of the colony determines the insects' intelligent behaviour and has captured the attention of many scientists and the branch of artificial intelligence called *swarm intelligenc*e Leao et al., (2001), Huang et al., (1995)

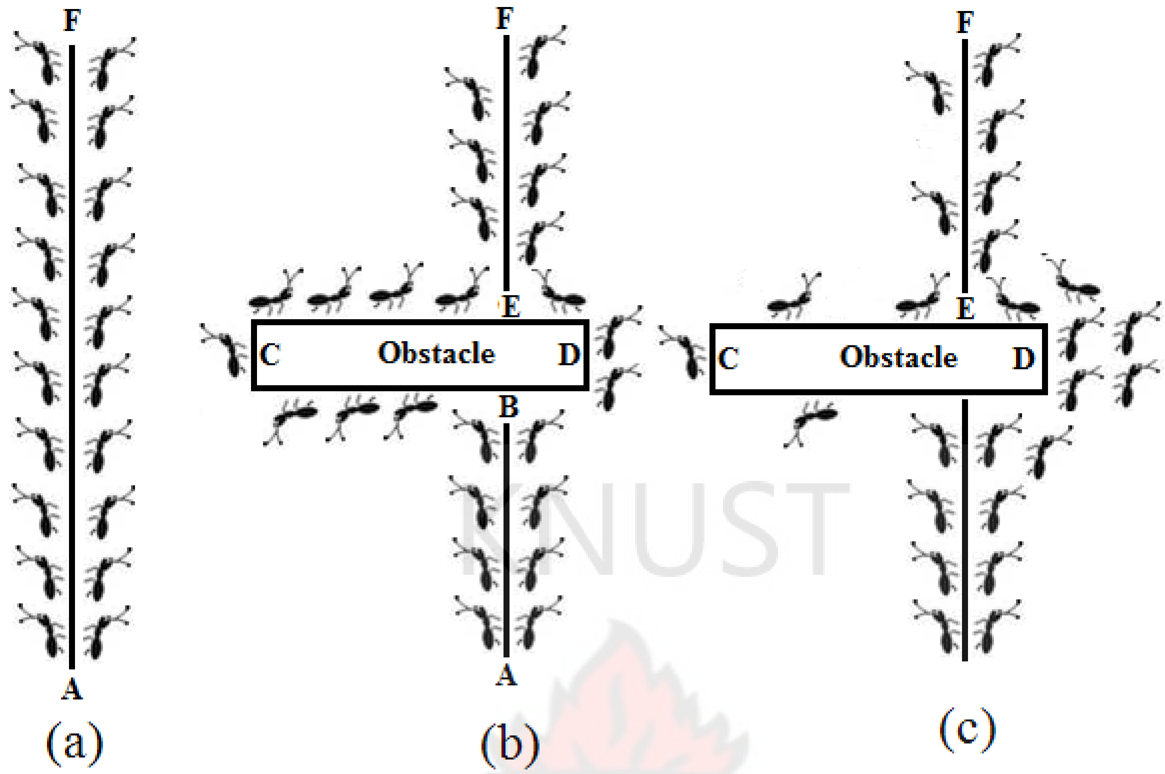

Figure 3.2 An example of the behaviour of real ants

Consider for example the experimental setting shown in Figure 3.2, there is a path along which ants are walking (for example from nest A to food source E). Suddenly an obstacle appears and the path is cut off. So at position B the ants walking from A to E (or at position D those walking in the opposite direction) have to decide whether to turn right or left. The choice is influenced by the intensity of the pheromone trails left by preceding ants. A higher level of pheromone on the right path gives an ant a stronger stimulus and thus a higher probability to turn right. The first ant reaching point B (or D) has the same probability to turn right or left (as there was no previous pheromone on the two alternative paths). Because path BDE is shorter than BCE the first ant following it will reach E before the first ant following path BCE. Shorter paths will receive pheromone reinforcement more quickly as they will be completed earlier than longer ones. The result is that an ant returning from F to E will find a stronger trail on path FEDB, as a consequence, the number of ants following path FED per unit time will be higher than the number of ants following FECB. This causes the quantity of pheromone on the shorter path to grow faster than on the longer one. Thus, the probability that any single ant chooses the path to follow is quickly biased towards the shorter one.

3.3.2 ARTIFICIAL ANTS

Now in artificial life, the Ant Colony Optimization (ACO) uses artificial ants, called agents, to find good solutions to difficult combinatorial optimization problems [Bonabeau, Press]. The behaviour of artificial ants is based on the traits of real ants, plus additional capabilities that make them more effective, such as a memory of past actions. Consider the example in Figure 3.2. The distances between D and H, between B and H, and between B and D are equal to 1. C is positioned in the middle of D and B. 30 new ants come to B from A and 30 to D from E at each time unit. Each ant walks at a speed of 1 per time unit and lays down a pheromone trail of intensity 1 at time *t*. Evaporation occurs in the middle of the successive time interval (*t*+1, *t*+2).

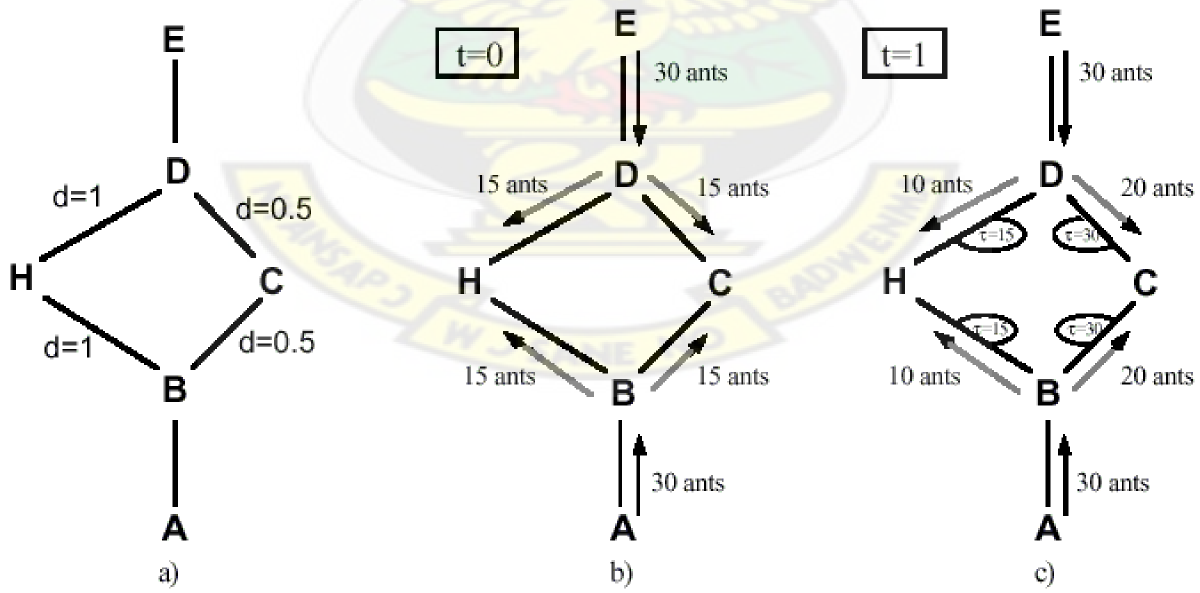

Figure 3.3 The behaviour of artificial ants on a path with time

At t =0 30 ants are in B and 30 in D. As there is no pheromone trail they randomly choose the way to go. Thus, approximately 15 ants from each node will go toward H and 15 toward C.

At t=1 30 new ants come to B from A. They sense a trail of intensity 15 on the path that leads to H, laid by the 15 ants that went through B-H-D. They also sense a trail of intensity 30 on the path to C, obtained as the sum of the trail laid by the 15 ants that went through B-C-D and by the 15 ants that went through D-C-B. The probability of choosing a path is therefore biased. The expected number of ants going toward C will be the double of those going toward H: 20 versus 10, respectively. The same is true for the new 30 ants in D which came from E. This process continues until all of the ants eventually choose the shortest path E. I Gokce (2004).

In brief, if an ant has to make a decision about which path to follow it will most probably follow the path chosen heavily by preceding ants, and the more the number of ants following a trail, the more attractive that trail becomes to be followed.

In the ant meta-heuristic, a colony of artificial ants cooperates in finding good solutions to discrete optimization problems. Artificial ants have two characteristics. On the one hand they imitate the following behaviour of real ants:

- *Colony of cooperating individuals:* Like real ant colonies, ant algorithms are composed of entities cooperating to find a good solution. Although each artificial ant can find a feasible solution, high quality solutions are the result of the cooperation. Ants cooperate by means of the information they concurrently read/write on the problem states they visit.

- *Pheromone trail:* While real ants lie pheromone on the path they visit, artificial ants change some numeric information of the problem states. This information takes into account the ant's current performance and can be obtained by any ant accessing the state. In ant algorithms pheromone trails are the only communication channels among the ants. It affects the way that the problem environment is perceived by the ants as a function of the past history. Also an evaporation mechanism, similar to real pheromone evaporation, modifies the pheromone. Pheromone evaporation allows the ant colony to slowly forget its past history so that it can direct its search towards new directions without being over-constrained by past decisions.

• *Shortest path searching and local moves:* The aim of both artificial and real ants is to find a shortest path joining an origin to destination sites. Like real ants artificial ants move step-by-step through adjacent states of the problem.

• *Stochastic state transition policy:* Artificial ants construct solutions applying a probabilistic decision to move through adjacent states. As for real ants, the artificial ants only use local information in terms of space and time. The information is a function of both the specifications and pheromone trails induced by past ants.

On the other hand, they are enriched with the following capabilities:

(i) artificial ants can determine how desirable states are.

(ii) artificial ants have a memory that keeps the ants' past actions.

(iii) artificial ants deposit an amount of pheromone, which is a function of the quality of the solution found.

(iv) the way that artificial ants lies pheromone is dependent on the problem.

(v) ant algorithms can also be enriched with extra capabilities such as local optimization, backtracking, and so on, that cannot be found in real ants.

3.4 ANT SYSTEM (AS)

In AS, K artificial ants probabilistically construct tours in parallel exploiting a given pheromone model. Initially, all ants are placed on randomly chosen cities. At each iteration, each ant moves from one city to another, keeping track of the partial solution it has constructed so far. The algorithm has two fundamental components:

(i) the amount of pheromone on arc (i, j), τ_{ij}

(ii) desirability of arc (i, j), η_{ij}

where arc (i, j) denotes the connection between cities i and j.

At the start of the algorithm an initial amount of pheromone τ_0 is deposited on each arc: $\tau_{ij} = \tau_0 = \frac{K}{L_O}$, where L_0 is the length of an initial feasible tour and K is the number of ants. In AS, the initial tour is constructed using the nearest-neighbor algorithm; however, another TSP heuristic may be utilized as well. The desirability value (also referred to as visibility or heuristic information) between a pair of cities is the inverse of their distance $\eta_{ij} = \frac{1}{d_{ij}}$ where d_{ij} is the distance between cities i and j. So, if the distance on the arc (i, j) is long, visiting city j after city i (or vice-versa) will be less desirable.

Each ant constructs its own tour utilizing a transition probability: an ant k positioned at a city i selects the next city j to visit with a probability given by

$$p_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}]^\alpha.[\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{ik}]^\alpha.[\eta_{ik}]^\beta}, & j \in N_i^k \\ 0 & ,\text{otherwise} \end{cases}$$

where, N_i^k denotes the set of not yet visited cities; α and β are positive parameters to control the relative weight of pheromone information τ_{ij} and heuristic information η_{ij}. After each ant has completed its tour, the pheromone levels are updated. The pheromone update consists of the pheromone evaporation and pheromone reinforcement. The pheromone evaporation refers to uniformly decreasing the pheromone values on all arcs. The aim is to prevent the rapid convergence of the algorithm to a local optimal solution by reducing the probability of repeatedly selecting certain cities. The pheromone reinforcement process, on the other hand, allows each ant to deposit a certain amount of pheromone on the arcs belonging to its tour. The aim is to increase the probability of selecting the arcs frequently used by the ants that construct short tours. The pheromone update rule is the following:

$$\tau_{ij} \leftarrow (1-\rho).\tau_{ij} + \sum_{k=1}^{K} \Delta\tau_{ij}^{k} \quad \forall (i,\ j)\ .$$

In this formulation $\rho\,(0<\rho\leq 1)$ is the pheromone evaporation parameter and $\Delta\tau_{ij}^{k}$ is the amount of pheromone deposited on arc (i, j) by ant k and is computed as follows:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{L_k}, \text{if k}^{\text{th}} \text{ ant uses path } (i,\ j) \text{ in its tour} \\ 0 \quad , \text{otherwise} \end{cases}$$

where L_k is the tour length constructed by the k-th ant.

3.5 IMPROVEMENT OF ANT SYSTEM

The success of ant heuristic lie sorely on the door steps of the pheromone trial. A substantial research on ACO has focused on how to improve AS all in the aim of improving the tour length. Some of these AS improvement algorithms are

(i) Elitist Ant System (EAS);

(ii) Rank Based Ant System (AS_{rank});

(iii) Ant Colony System (ASC) and

(iv) Max-Min Ant System (MMAS)

3.5.1 ELITIST ANT SYSTEM (EAS)

In the EAS an elitist strategy is implemented by further increasing the pheromone levels on the arcs belonging to the best tour achieved since the initiation of the algorithm. That best-so-far tour is referred to as the “global-best” tour. The pheromone update rule is as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{K} \Delta\tau_{ij}^{k} + w\Delta\tau_{ij}^{gb} \qquad \forall(i,\ j)$$

Here, w denotes the weight associated with the global-best tour and τ_{ij}^{gb} is the amount of pheromone deposited on arc $(i,\ j)$ by the global-best ant and calculated by the following formula:

$$\tau_{ij}^{gb} = \begin{cases} \dfrac{1}{L^{gb}}, & \text{if the global best ant uses arc } (i,\ j) \text{ in its tour} \\ 0, & \text{otherwise} \end{cases}$$

here L^{gb} is the length of global-best tour.

3.5.2 RANK BASED ANT SYSTEM (Rank AS)

In the *AS*rank a rank-based elitist strategy is adopted in an attempt to prevent the algorithm from being trapped in a local minimum. In this strategy, w best ranked ants are used to update the pheromone levels and the amount of pheromone deposited by each ant decreases with its rank. Furthermore, at each iteration, the global-best ant is allowed to deposit the largest amount of pheromone. The pheromone update rule is given by:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{w-1} (w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{gb} \qquad \forall(i,\ j)$$

3.5.3 ANT COLONY SYSTEM (ACS)

The ACS attempts to improve AS by increasing the importance of exploitation versus exploration of the search space. This is achieved by employing a strong elitist strategy to update pheromone levels and a pseudo-random proportional rule in selecting the next node to visit. The strong elitist strategy is applied by using the global-best ant only to increase the pheromone levels on the arcs that belong to the global-best tour:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{gb} \quad \forall(i,\ j)$$

The mechanism of the pseudo-random proportional rule is as follows: an ant *k* located at customer *i* may either visit its most favorable customer or randomly select a customer. The selection rule is the following:

$$j^k = \begin{cases} \underset{j \in N_i^k}{\operatorname{argmax}} \tau_{ij}^{\alpha}\eta_{ij}^{\beta}\tau_{ij} & z \le z_0 \\ J^k, & \text{otherwise} \end{cases}$$

where *z* is a random variable drawn from a uniform distribution $U[0,1]$ and z_0 $(0 \le z_0 \le 1)$ is a parameter to control exploitation versus exploration. j^k is selected according to the probability distribution p_{ij}^k. ACS also uses local pheromone updating while building solutions: as soon as an ant moves from city *i* to city *j* the pheromone level on arc (*i*, *j*) is reduced in an attempt to promote the exploration of other arcs by other ants. The local pheromone update is performed as follows:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0$$

where ξ is a positive parameter less than 1.

Similar to ACS, uses either the global-best ant or the iteration-best ant alone to reinforce the pheromone.

3.6 VEHICLE ROUTING PROBLEM WITH SOLID WASTE COLLECTION

In this section, we shall first consider the problem of bin collection; look at the problem formulation and the mathematical model for the problem.

3.6.1 PROBLEM DESCRIPTION

Capacitated clustering vehicle routing problem (CCVRP) deals with a single depot collection system servicing a set of customers by means of a homogeneous fleet of vehicles, i.e. all vehicles have the same capacity. The customers require only one type of service: solid waste collection, the vehicles leaves the depot empty, collect the waste of each customer to the dump site and return to the depot empty. The objective is to find the set of vehicle routes servicing all the customers with the minimum total distance.

In CCVRP, each customer must be serviced exactly once. The graph have been clustered according the capacity of the vehicle, by the time the vehicle collects the load from the last customer in the cluster, the vehicle will be full and each customer in the cluster might have been served.

3.6.2 PROBLEM FORMULATION

Mathematically, CCVRP is described by a set of homogenous vehicles V, a set of customers R, and a complete directed graph $G(N, A)$. The graph consists of $(n+1)$ vertices where the customers are denoted by 1, 2, ..., n and the depot is represented by the vertex 0 with $(n + 1)$ as the dumpsite. $A = \{(i, j) : i, j \in N, i \neq j\}$ denote the set of arcs that represents connections between the depot and the customers and among the customers. No arc terminates at vertex O and no arc originates from vertex $(n + 1)$, distance (d_{ij}) is associated with each arc (i, j). Each vehicle has capacity Q and each customer (node) i is characterized by its direct distance and the pick up demand qi. Finally, Q, d_{ij}, and q_i are assumed to be non-negative integers. The CCVRP determines a set of paths (routes) such that:

(i) each vehicle travels exactly one route;

(ii) each customer is visited only once by one of the vehicles completely satisfying its demand and supply;

(iii) the load carried by a vehicle between any pair of adjacent customers on the route must not exceed its capacity; and

(iv) total distance given by the sum of the arcs belonging to these routes is minimal.

3.6.3 MATHEMATICAL FORMULATION FOR CCVRP

$$\text{Minimize } Z = \sum_{i \in N} \sum_{j \in N} \sum_{v \in V} d_{ij} x_{ijv} \qquad (3.6.1)$$

Subject to

$$\sum_{j \in N} \sum_{v \in V} x_{ijv} = 1 \qquad \forall i \in R \qquad (3.6.2)$$

$$\sum_{i \in N} \sum_{v \in V} x_{ijv} = 1 \qquad \forall j \in R \qquad (3.6.3)$$

$$\sum_{j \in N} x_{0jv} = 1 \qquad \forall v \in V \qquad (3.6.4)$$

$$\sum_{i \in N} x_{ikv} = \sum_{j \in N} x_{kjv} \qquad \forall k \in R, v \in V \qquad (3.6.5)$$

$$\sum_{i \in N} x_{i\,n+1\,v} = 1 \qquad \forall v \in V \qquad (3.6.6)$$

$$\sum_{i \in R} q_i \left[\sum_{j \in N} x_{ijv} \right] \le Q \qquad \forall v \in V \qquad (3.6.7)$$

$$x_{ijv} \in \{0,1\} \qquad \forall i,\ j \in N, \forall v \in V \qquad (3.6.8)$$

Equation (3.6.1) ensures that the objective function aims at minimizing the total travel distance, Equations (3.6.2) and (3.6.3) guarantees that for each cluster, a customer is visited exactly once, equations (3.6.4), (3.6.5) and (3.6.6) ensure that each vehicle leaves the depot O, after arriving at the customer the vehicle leaves that customer again, and finally arrives at the dumpsite (*n* + 1). Equation (3.6.7) state that no vehicle is loaded more than its capacity and equation (3.6.8) are the binary constraints.

3.7 ACO ALGORITHM FOR OUR PROPOSED WORK

The construction graph G = (N, A), where the set A fully connects the components N, is identical to the problem graph, that is the set of states of the problem corresponds to the set of all possible partial tours.

An initial solution is first obtained using the nearest-neighbor heuristic: start at the depot and then select the not yet visited closest feasible customer as the next customer to be visited.

Each artificial ant has a memory called tabu list. The tabu list forces the ant to make legal tours. It saves the cities already visited and forbids the ant to move already visited cities until a tour is completed.

After all cities are visited, the tabu list of each ant will be full. The shortest path found is computed and saved. Then, tabu lists are emptied. This process is iterated for a user-defined number of cycles.

Suppose there are n nodes and b_i is the number of ants at city i. Consider the following notation:

$K = \sum_{i=1}^{n} b_i$: Total number of ants

N : Set of customers to be visited

$tabu_k$: Tabu list of the k-th ant

$tabu_k(s)$: s-th customer visited by the k-th ant in the tour

τ_{ij} (t) : Intensity of trail on edge between customer i and customer j at time t

η_{ij} : Visibility of edge between customer i and customer j

η_{ij} is usually assumed as the inverse of the distance between customer *i* and customer *j* (d_{ij})

Thus, $\eta_{ij} = \frac{1}{d_{ij}}$

After *K* artificial ants are randomly placed on customers, the first element of each ant's tabu list is set to be equal to its starting customers. Then, they move to unvisited customers. The probability of moving from customer *i* to customer *j* for the *k*-th ant is defined as: (p_{ij}^k)

$$p_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}]^\alpha.[\eta_{ij}]^\beta}{\sum_{k \in \text{allowed}_k} [\tau_{ik}]^\alpha.[\eta_{ik}]^\beta}, & j \in \text{ allowed} \\ 0, & \text{otherwise} \end{cases}$$

where $allowed_k = \{N - tabu_k\}$, α and β are parameters that control the relative importance of pheromone trail versus visibility.

3.7.1. HEURISTIC INFORMATION

Generally the ant approaches developed for solving TSP and VRP the visibility value between a pair of customers is the inverse of their distance, thus $\eta_{ij} = \frac{1}{d_{ij}}$.

3.7.2. INITIAL PHEROMONE TRIALS

In most of the ant colony based algorithms to VRP, initial pheromone trails τ_0 is set equal to the inverse of the best known route distances found for the particular problem. However, it was found that $\tau_0 = \frac{1}{n}$.

When the initial route is constructed, it is started at the depot and the customer with the highest φ_{0j} value is selected as the first customer to be visited. Then, the tour is constructed by selecting the not yet visited feasible customer with the highest φ_{ij} at each time.

3.7.3 ROUTE CONSTRUCTION PROCESS

It is assumed that the number of ants is equal to the number of customers, initially each ant is positioned at each customer. Then, each ant constructs its own tour by successively selecting a not yet visited feasible customer. The choice of the next customer to visit is based on proportional fitness (Roulette Wheel) in conjunction with the information of both the pheromone trails and the visibility of that choice given in equation $\varphi_{ij} = \tau_{ij}[\eta_{ij}]^{\beta}$, τ_{ij} denotes the amount of pheromone on arc (i, j) and β is power weighting parameter that weights the consistency of arc (i , j).

3.7.4 PHEROMONE UPDATE

Our pheromone update consists of an improved ant system strategy. In this strategy our pheromone update rule is as follows: $\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{r=1}^{k} \Delta\tau_{ij}^{r} + \frac{Q}{L^k}$

where Q is a constant based on the number of nodes in the cluster , L^k is the length of tour of ant K and $\rho, 0 \le \rho \le 1$, is the evaporation factor, which determines the strength of an update.

In order to get more insight of the algorithm, we shall consider a five (5) node TSP problem. The objective is to find a minimum tour required to visit all the five (5) customers on the nodes. A connectivity matrix of the graph is given in Table 1. The values given in the table denotes the distance ‘’d’’ between customer nodes and it is assumed to be a symmetric TSP problem, in which $d_{ij} = d_{ji}$

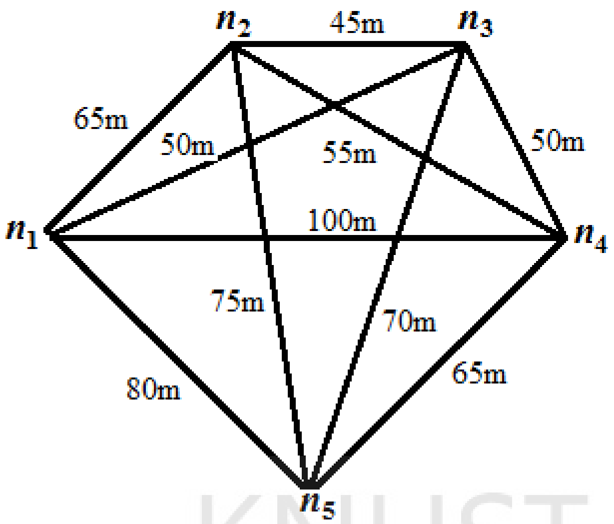

Figure 3.4: An example of a TSP problem

Table 3.1**:** Connectivity matrix of a TSP example shown in Figure 3.4

	n_1	n_2	n_3	n_4	n_5
n_1	0	65	50	100	80
n_2	65	0	45	55	75
n_3	50	45	0	50	70
n_4	100	55	50	0	65
n_5	80	75	70	65	0

Each edge in the graph is given an initial pheromone value $\tau_0 = \frac{1}{n}$, where $n = 5$. Let the heuristic value $\eta_{ij} = \dfrac{1}{d_{ij}}$

The probability of selecting an edge is given by $p_{ij}^k = \dfrac{[\tau_{ij}]^\alpha * [\eta_{ij}]^\beta}{\sum_{l \in N} [\tau_{il}]^\alpha * [\eta_{il}]^\beta}$, where N = 4 (the set of neighbouring customers (nodes)), α and β are parameters that control the relative weight of pheromone trial and heuristic value. In this example, the values of α and β are taken as 1 and 2 respectively.

Table 3.2: Heuristic value (η_{ij}) for each edge is as shown in Figure 3.4

	n_1	n_2	n_3	n_4	n_5
n_1	0	0.0150	0.0200	0.0100	0.0125
n_2	0.0150	0	0.0222	0.0182	0.0133
n_3	0.0200	0.0222	0	0.0200	0.0143
n_4	0.0100	0.0182	0.0200	0	0.0154
n_5	0.0125	0.0133	0.0143	0.0154	0

Since there are 5 nodes, we take the size of the colony as 5, each ant will start its tour from different node. For example, the first ant starts from customer n_1, the second ant will start from node n_2, and so on.

Table 3.3: Initial pheromone value (τ_0) for each edge is as shown in Figure 3.4

	n_1	n_2	n_3	n_4	n_5
n_1	0	0.2	0.2	0.2	0.2
n_2	0.2	0	0.2	0.2	0.2
n_3	0.2	0.2	0	0.2	0.2
n_4	0.2	0.2	0.2	0	0.2
n_5	0.2	0.2	0.2	0.2	0

ITERATION 1

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}}{\sum_{l \in N} [\tau_{il}(t)]^{\alpha} * [\eta_{il}]^{\beta}}$$

$$p_{1,2}^{1}(1) = \frac{[\tau_{1,2}(1)]^{1} \times [\eta_{1,2}]^{2}}{\sum_{l \in 4} [\tau_{1l}(1)]^{1} \times [\eta_{1l}]^{2}}$$

$$p_{1,2}^{1}(1) = \frac{[0.20]^{1} \times [0.015]^{2}}{[0.20 \times (0.015)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.01)^{2}] + [0.20 \times (0.0125)^{2}]}$$

$$= \frac{4.5 \times 10^{-5}}{1.7625 \times 10^{-4}} = 0.2553$$

$$p_{1,3}^{1}(1) = \frac{[0.20]^{1} \times [0.02]^{2}}{[0.20 \times (0.015)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.01)^{2}] + [0.20 \times (0.0125)^{2}]}$$

$$= \frac{8 \times 10^{-5}}{1.7625 \times 10^{-4}} = 0.4539$$

$$p_{1,4}^{1}(1) = \frac{[0.20]^{1} \times [0.01]^{2}}{[0.20 \times (0.015)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.01)^{2}] + [0.20 \times (0.0125)^{2}]}$$

$$= \frac{2 \times 10^{-5}}{1.7625 \times 10^{-4}} = 0.1135$$

$$p_{1,5}^{1}(1) = \frac{[0.20]^{1} \times [0.0125]^{2}}{[0.20 \times (0.015)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.01)^{2}] + [0.20 \times (0.0125)^{2}]}$$

$$= \frac{3.125 \times 10^{-5}}{1.7625 \times 10^{-4}} = 0.1773$$

The first ant starts the tour from node 1, there are four neighbouring cities to be considered by the ant. The probability of choosing any edge leading to another node is as calculated above and the results shown the table below.

n_2	n_3	n_4	n_5
0.2553	0.4539	0.1135	0.1773

Using proportional selection (Roulette wheel), the ant chooses the next node say n_3, the ant will update its memory and put node 1 and 3 in its Tabu List. When the ant arrives at node 3, there are three nodes left to visit, the probability of choosing these nodes is calculated below.

$$p_{3,2}^{1}(1) = \frac{[0.20]^{1} \times [0.0222]^{2}}{[0.20 \times (0.0222)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.0143)^{2}]}$$

$$= \frac{9.8568 \times 10^{-5}}{2.19466 \times 10^{-4}} = 0.4491$$

$$p_{3,4}^{1}(1) = \frac{[0.20]^{1} \times [0.02]^{2}}{[0.20 \times (0.0222)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.0143)^{2}]}$$

$$= \frac{8 \times 10^{-5}}{2.19466 \times 10^{-4}} = 0.3645$$

$$p_{3,5}^{1}(1) = \frac{[0.20]^{1} \times [0.0143]^{2}}{[0.20 \times (0.0222)^{2}] + [0.20 \times (0.02)^{2}] + [0.20 \times (0.0143)^{2}]}$$

$$= \frac{4.0898 \times 10^{-5}}{2.19466 \times 10^{-4}} = 0.1864$$

n_2	n_4	n_5
0.4491	0.3645	0.1864

Using proportional selection (Roulette wheel), the ant chooses the next node say n_4, the ant will update its memory and put node 1, 3and 4 in its Tabu List. When the ant arrives at node 4, there are two nodes left to visit, the probability of choosing these nodes is calculated below.

$$p_{2,4}^{1}(1)=\frac{[0.20]^{1}\times[0.0182]^{2}}{[0.20\times(0.0182)^{2}]+[0.20\times(0.0154)^{2}]}$$

$$=\frac{6.6248\times10^{-5}}{1.1368\times10^{-4}}=0.5828$$

$$p_{2,5}^{1}(1)=\frac{[0.20]^{1}\times[0.0154]^{2}}{[0.20\times(0.0182)^{2}]+[0.20\times(0.0154)^{2}]}$$

$$=\frac{4.7432\times10^{-5}}{1.1368\times10^{-4}}=0.4172$$

n_2	n_5
0.5828	0.4172

Using proportional selection (Roulette wheel), the ant chooses the next node say n_5, the ant will update its memory and put node 1, 3, 4 and 5 in its Tabu List. When the ant arrives at node 5, there is only one node left to visit, the next process will certainly take node 2.

The path that was built by ant 1 is: $n_1 \rightarrow n_3 \rightarrow n_4 \rightarrow n_5 \rightarrow n_2$

The length L of this path is $L = n_1n_3 + n_3n_4 + n_4n_5 + n_5n_2$

$$= 50 + 50 + 65 + 75$$
$$= 240\text{m}$$

The remaining ants will make their tour according to the same procedure. The following table summarizes the solutions built by all ants. The last column in Table 3 is the gain obtained by each ant. Since the longest distance between nodes is 100m, the solution built by the ant must not exceed 4×100 = 400. Thus, the gain of each ant can be formulated as $\frac{400}{L}$, with L as the length of the path of solution.

Table 3.4: Solutions built by the ants in the first iteration

Ant	Path	Length of the path (L)	$\Delta\tau = \frac{400}{L}$
Ant 1	$n_1 \to n_3 \to n_4 \to n_5 \to n_2$	240	1.6667
Ant 2	$n_2 \to n_1 \to n_5 \to n_3 \to n_4$	265	1.5094
Ant 3	$n_3 \to n_2 \to n_4 \to n_5 \to n_1$	245	1.6327
Ant 4	$n_4 \to n_5 \to n_3 \to n_2 \to n_1$	245	1.6327
Ant 5	$n_5 \to n_3 \to n_2 \to n_1 \to n_4$	280	1.4286

When all ants finish their tour, they will back track and update the pheromone along their path by putting additional pheromone $(\Delta\tau)$, which is proportional to the gain obtained by the ant. $\Delta\tau_{ij} = \sum_{k=1}^{k} \Delta\tau_{ij}^{k}$, where $\Delta\tau_{ij}^{k}$ is the added pheromone to the arcs in the tour ant k has visited.

The new pheromone value is given by $\tau^{\text{new}} = (1-\rho)\tau^{\text{old}} + \Delta\tau$, where ρ is the evaporation constant taken as ρ = 0.5 in this example.

Edge $n_1 n_2$ was used by ant 2, 4 and ant 5, therefore

$$\tau^{\text{new}} = (1-0.5)0.2 + 1.5094 + 1.6327 + 1.4286$$
$$= 4.6707$$

Edge $n_1 n_3$ was used by ant 1 only, therefore $\tau^{\text{new}} = (1-.5)0.2 + 1.6667 = 1.7667$

Edge $n_1 n_4$ was used by ant 5, therefore $\tau^{new} = (1-0.5)\times 0.2 + 1.4286 = 1.5286$

Edge $n_1 n_5$ was used by ant 2 and ant 3, therefore $\tau^{new} = (1-.5)0.2 + 1.6327 + 1.6327 = 3.3654$

This is done for all the edges and the new pheromone matrix at the end of iteration 1 is as shown below.

Table 3.5: Pheromone values for each edge after iteration 1

	n_1	n_2	n_3	n_4	n_5
n_1	0	4.6707	1.7667	1.5286	3.3654
n_2	4.6707	0	4.794	1.7328	1.7668
n_3	1.7667	4.794	0	3.2761	4.6708
n_4	1.5286	1.7328	3.2761	0	5.0321
n_5	3.3654	1.7668	4.6708	5.0321	0

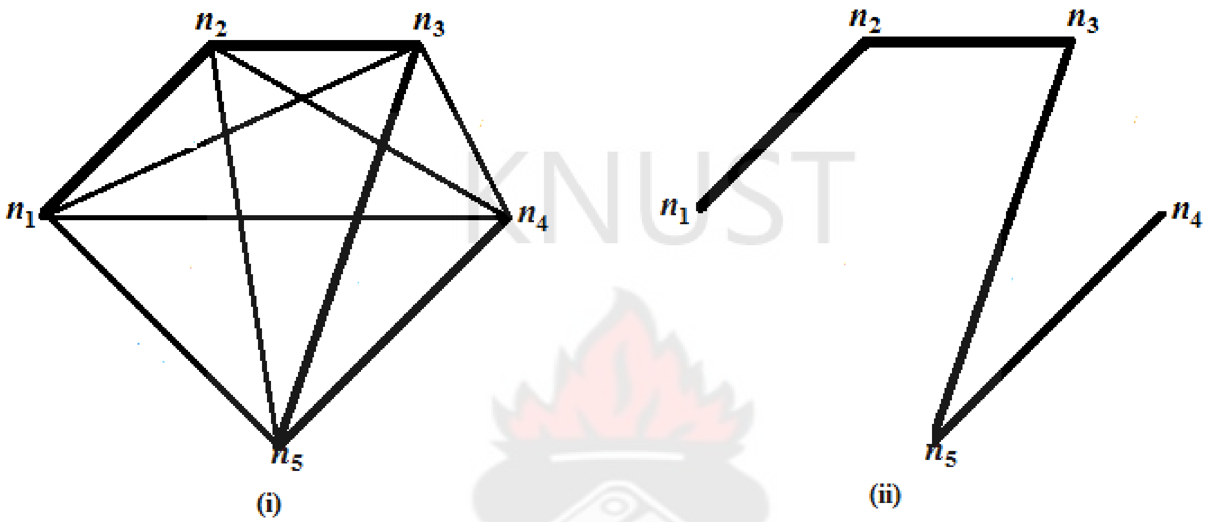

Figure 3.5: (i) Visualization of pheromone values and (ii) Best solution built in the first iteration.

Figure 4 (i) shows the visualization of pheromone values on the edges, where the darker lines indicate higher pheromone on the edge. The best solution found by the heuristic in the first iteration is shown in Figure 4 (ii)

ITERATION 2

The same procedure is repeated as done in the first iteration. However, the initial pheromone values on all the edges have changed, thus the probability of selecting a certain edge will also change. The higher the pheromone on the edge, the more attractive it is for an ant to choose. After going through the whole procedure again, the table below summarizes the solutions built by the ants.

Table 3.6: Solutions built by the ants in the second iteration

Ant	Path	Length of the path (L)	$\Delta\tau = \frac{400}{L}$
Ant 1	$n_1 \rightarrow n_3 \rightarrow n_2 \rightarrow n_4 \rightarrow n_5$	215	1.8605
Ant 2	$n_2 \rightarrow n_4 \rightarrow n_5 \rightarrow n_3 \rightarrow n_1$	240	1.6667
Ant 3	$n_3 \rightarrow n_2 \rightarrow n_1 \rightarrow n_5 \rightarrow n_4$	255	1.5686
Ant 4	$n_4 \rightarrow n_5 \rightarrow n_1 \rightarrow n_2 \rightarrow n_3$	255	1.5686
Ant 5	$n_5 \rightarrow n_3 \rightarrow n_2 \rightarrow n_4 \rightarrow n_1$	270	1.4815

The pheromone update and pheromone evaporation procedures are then performed and the results is as shown in Table 6 below

Table 3.7: Pheromone values for each edge after iteration 2

	n_1	n_2	n_3	n_4	n_5
n_1	0	5.4726	4.4106	0.7643	4.8199
n_2	5.4726	0	8.8762	5.8751	0.8834
n_3	4.4106	8.8762	0	1.6381	5.4836
n_4	0.7643	5.8751	1.6381	0	9.1805
n_5	4.8199	0.8834	5.4836	9.1805	0

Figure 3.6 (i) below shows the visualization of pheromone values on each edge, the thick lines represents edges with high pheromone values, which corresponds to the path followed by the initial iteration.

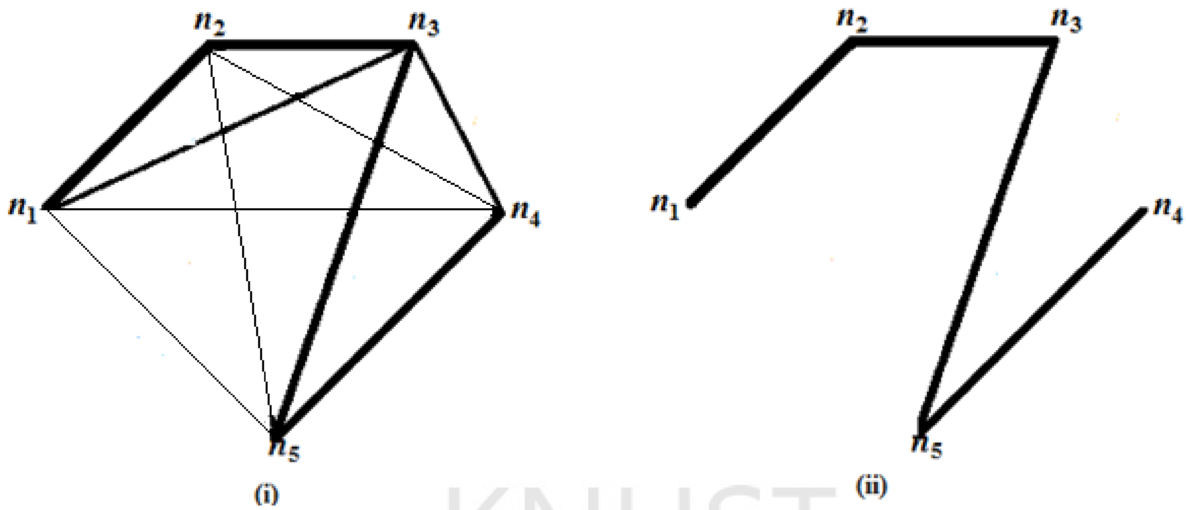

Figure 3.6: (i) Visualization of pheromone values and (ii) Best solution built in the second iteration.

The thickness of these lines corresponds to high pheromone values, which means that more ants are using those edges (see Table 5). The faint lines represent few ant usage on those edges and with time virtually all the pheromone values on those edges will turn to zero (0) and no ant will use those edges, since it will become less attractive to the ants. The algorithm will continue until a best tour length is found which will represent the shortest path to be used for the collection in that cluster. After five iterations we obtained a minimum route as $n_1 \rightarrow n_3 \rightarrow n_2 \rightarrow n_4 \rightarrow n_5$ which gives a total distance of 215m

3.8 SUMMARY

In this chapter, we looked at the mathematical formulation of a clustering problem and its variants, we also looked at the problem formulation and mathematical model for solving a capacitated clustering vehicle routing problem using Ant heuristics and worked an illustrative example using ant heuristics to find the shortest path was put forward.

CHAPTER FOUR

DATA COLLECTION AND ANALYSIS

4.0 INTRODUCTION

In this chapter, we shall look at how the data for the work was obtained, how it was used for the intended analysis based on the method(s) discussed in chapter three.

4.1 DATA COLLECTION

The study is intended to find the minimum route in order to collect the waste at Kwadaso estate in Kumasi. By its name, Kwadaso estate is a residential area built by state housing corporation, now state housing company. The site layout for the said area was obtained from state housing company and with the help of a software called GIS, we found all the direct distances between the adjacent node(s). Figure 4.1shows the load on each node and Figure 4.2 shows the direct distances between adjacent node(s) of the area under study.

Table 4.1 Node number and its corresponding load

Service points (nodes)	1	2	3	4	5	6	7	8	9
Load (q_i)	3	5	6	6	2	1	3	3	6

10	11	12	13	14	15	16	17	18	19
1	3	3	4	2	2	4	6	6	2

. . .

. . .

148	148	150	151	152	153	154	155	156	157
5	2	2	3	3	4	3	4	4	2

Table 4.2 Road distances between customer points from the site graph above

	1	2	3	4	5	6	7	.	.	.	151	152	153	154	155	156	157
1	0	56	inf	80	inf	inf	inf	.	.	.	inf	inf	inf	inf	inf	inf	inf
2	56	0	44	74	inf	inf	inf	.	.	.	inf	inf	inf	inf	inf	inf	inf
3	inf	44	0	inf	inf	inf	inf	.	.	.	inf	inf	inf	inf	inf	inf	inf
4	80	74	inf	0	53	inf	inf	.	.	.	inf	inf	inf	inf	inf	inf	inf
5	inf	inf	inf	53	0	44	inf	.	.	.	inf	inf	inf	inf	inf	inf	inf
6	inf	inf	inf	inf	44	0	35	.	.	.	inf	inf	inf	inf	inf	inf	inf
7	inf	inf	inf	Inf	inf	35	0	.	.	.	inf	inf	inf	inf	inf	inf	inf
.	.	..	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
151	inf	inf	inf	inf	inf	inf	inf	.	.	.	0	44	inf	inf	inf	inf	inf
152	inf	inf	inf	inf	inf	inf	inf	.	.	.	44	0	56	inf	inf	inf	inf
153	inf	inf	inf	inf	inf	inf	inf	.	.	.	inf	56	0	50	inf	inf	inf
154	inf	inf	inf	inf	inf	inf	inf	.	.	.	inf	inf	50	0	35	inf	inf
155	inf	inf	inf	inf	inf	inf	inf	.	.	.	inf	inf	inf	35	0	30	inf
156	inf	inf	inf	inf	inf	inf	inf	.	.	.	inf	inf	inf	inf	30	0	25
157	inf	inf	inf	inf	inf	inf	inf		.	.	inf	inf	inf	inf	inf	25	0

Table 4.3 All pair shortest path from table 4.2 by Floyd Warshall's Algorithm

	1	2	3	4	5	6	7	.	.	.	151	152	153	154	155	156	157
1	0	56	100	80	133	177	212	.	.	.	599.5	643.5	699.5	749.5	784.5	814.5	839.5
2	56	0	44	74	127	171	206	.	.	.	593.5	637.5	693.5	743.5	778.5	808.5	833.5
3	100	44	0	118	171	215	250	.	.	.	637.5	681.5	737.5	787.5	822.5	852.5	877.5
4	80	74	127	0	53	97	132	.	.	.	519.5	563.5	619.5	669.5	704.5	734.5	759.5
5	133	127	171	53	0	44	79	.	.	.	466.5	510.5	566.5	616.5	651.5	681.5	706.5
6	177	171	215	97	44	0	35	.	.	.	422.5	466.5	522.5	572.5	607.5	637.5	662.5
7	212	206	250	132	79	35	0	.	.	.	387.5	431.5	487.5	537.5	572.5	602.5	627.5
.	.	..	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
151	599.5	599.5	599.5	599.5	599.5	599.5	599.5	.	.	.	0	44	100	150	185	215	240
152	643.5	643.5	643.5	643.5	643.5	643.5	643.5	.	.	.	44	0	56	106	141	171	196
153	699.5	699.5	699.5	699.5	699.5	699.5	699.5	.	.	.	100	56	0	50	85	115	140
154	749.5	749.5	749.5	749.5	749.5	749.5	749.5	.	.	.	150	106	50	0	35	65	90
155	784.5	784.5	784.5	784.5	784.5	784.5	784.5	.	.	.	185	141	85	35	0	30	55
156	814.5	814.5	814.5	814.5	814.5	814.5	814.5	.	.	.	215	171	115	65	30	0	25
157	839.5	839.5	839.5	839.5	839.5	839.5	839.5		.	.	240	196	140	90	55	25	0

4.2 DATA ANALYSIS

After road distances between service points (nodes) have been obtained, we also counted all the collection points and the total bins, which gave 157 nodes with total bins of 588. A code was then written to give all shortest pair distances based on Floyd Warshall's algorithm which gave us 157× 157 matrix. We then applied our vertex 1- centre clustering algorithm to first generate stable centres after 1000 iterations, and then used these centres to generate our capacitated clusters based on load on each node and the distance from the centre. The total load to be collected in each cluster is based on the capacity of the vehicle (103 x 240 litre capacity bins) and all road distances were measured (in metres) from the graph is of scale 1:2.

4.3 RESULTS FROM VERTEX 1-CENTRE ALGORITHM

Table 4.4 Stable cluster Centres and their respective clusters

Cluster Centres = 128 6 48 31 146 106

cluster1 = 113 114 115 116 118 119 120 121 122 123 124 125 126 127 128
129 130 131 132 133 134 135 139 140

cluster2 = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
25 29 30 37 38 39 40 57 58 59 60 61

cluster3 = 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 63
64 65 66 67 68 69 70 71 72 73 74 75 76

cluster4 = 19 20 21 22 23 24 26 27 28 31 32 33 34 35 36 78 79
80 81 82 83 62 77

cluster5 = 84 85 86 87 88 89 105 108 109 136 137 138 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157

cluster6 = 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 106 107 110 111 112 117 141

4.4 MODIFIED ACO (ASOPTION) RESULTS

Our modified ASoption was then used on each of the clusters to find the minimum tour of each ant and then select the best ant tour. The result for each ant in a cluster is as shown in the Tables below.

Table 4.5 Cluster, the ants tour length and the best ant tour for cluster 1

cluster1 = 113 114 115 116 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 139 140

```
Distance_Covered_By_Ant =
                      431.4000
                      382.9333
                      495.1417
                      535.5250
                      412.0250
                      436.0167
                      435.3500
                      395.1417
                      447.2333
                      336.3292
                      377.7042
                      478.5167
                      537.7167
                      472.2667
                      562.3583
                      402.5375
                      453.9375
                      548.1583
                      476.1875
                      421.3667
                      579.4333
                      371.7500
                      527.0167
                      463.4375
```

Ant_Best_Tour = 116 → 118 → 119 → 122 → 128 → 129 → 134 → 135 → 139 → 140 → 132 → 133 → 131 → 130 → 127 → 126 → 125 → 124 → 123 → 114 → 113 → 115 → 120 → 121

Table 4.6 Cluster, the ants tour length and the best ant tour for cluster 2

```
cluster2 = 1     2     3     4     5     6     7     8     9    10    11
          12    13    14    15    16    17    18    25    29    30    37
          38    39    40    57    58    59    60    61

Distance_Covered_By_Ant =

   1.0e+003 *

    1.0307
    0.8281
    0.8883
    0.9243
    0.9750
    1.0953
    1.0756
    0.9422
    1.0377
    0.8052
    0.8718
    0.8461
    0.8256
    0.9297
    0.9202
    0.9065
    0.8771
    0.8972
    0.8125
    1.0137
    0.9162
    0.9850
    0.9674
    0.9856
    0.8614
    0.9761
    0.8225
    0.9249
    0.8237
    0.8605
```

Ant_Best_Tour = 3 → 2 → 1 → 4 → 5 → 6 → 7 → 8 → 10 → 11 → 9 → 30 → 37 → 39 → 40 → 38 → 29 → 25 → 14 → 13 → 12 → 16 → 17 → 15 → 18 → 61 → 60 → 59 → 58 → 57

Table 4.7 Cluster, the ants tour length and the best ant tour for cluster 3

```
cluster3 = 41    42    43    44    45    46    47    48    49    50    51

           52    53    54    55    56    63    64    65    66    67    68

           69    70    71    72   73    74    75    76

Distance_Covered_By_Ant =

                     1.0e+003 *

                       1.4121
                       1.2712
                       1.3023
                       1.1712
                       1.3141
                       0.9527
                       1.0574
                       1.0644
                       1.3382
                       1.3160
                       1.2337
                       1.2653
                       1.2879
                       1.2987
                       1.2576
                       1.3518
                       1.2214
                       1.2168
                       1.1528
                       1.4724
                       1.1402
                       1.3601
                       1.3600
                       1.3688
                       1.1711
                       1.3623
                       1.4798
                       1.0427
                       1.0501
                       1.0223
```

Ant_Best_Tour = 56 → 55 → 54 → 53 → 52 → 51 → 50 → 49 → 48 → 63 → 64 → 70 → 69 → 68 → 67 → 71 → 66 → 65 → 72 → 76 → 75 → 74 → 73 → 44 → 45 → 46 → 47 → 43 → 42 → 41

Table 4.8 Cluster, the ants tour length and the best ant tour for cluster 4

```
cluster4 = 19    20    21    22    23    24    26    27    28    31    32
           33    34    35    36    78    79    80    81    82    83    62
           77

Distance_Covered_By_Ant =

                    1.0e+003 *

                      0.8653
                      1.1950
                      1.2371
                      1.2064
                      1.1986
                      0.9538
                      1.0545
                      1.0885
                      0.8616
                      1.1004
                      1.1288
                      1.2015
                      0.9560
                      1.0083
                      0.8731
                      1.0752
                      0.9972
                      0.8040
                      0.8602
                      0.9514
                      0.9838
                      1.1804
                      0.8921
```

Ant_Best_Tour = 77 → 78 → 79 → 80 → 81 → 82 → 83 → 34 → 35 → 36 → 26 → 27 → 28 → 22 → 23 → 24 → 21 → 20 → 19 → 33 → 32 → 31 → 62

Table 4.9 Cluster, the ants tour length and the best ant tour for cluster 5

```
cluster5 = 84    85    86    87    88    89   105   108   109   136   137
           138   142   143   144   145   146   147   148   149   150   151
           152   153   154   155   156   157
Distance_Covered_By_Ant =
                            1.0e+003 *

                               0.9496
                               0.9463
                               0.7677
                               0.9925
                               0.8265
                               0.7377
                               0.9167
                               0.9324
                               1.0552
                               0.8263
                               0.8710
                               1.1871
                               0.9332
                               0.8610
                               0.9767
                               0.9156
                               0.8856
                               0.8465
                               0.9870
                               0.8355
                               0.9034
                               1.0359
                               0.6628
                               0.9624
                               0.9970
                               1.0041
                               0.7515
                               1.0312
```

Ant_Best_Tour =142 → 143 → 144 → 145 → 146 → 147 → 105 → 109 → 108 → 138 → 137 → 136 → 150 → 151 → 152 → 148 → 149 → 84 → 86 → 87 → 88 → 89 → 85 → 153 → 154 → 155 → 156 → 157

Table 4.10 Cluster, the ants tour length and the best ant tour for cluster 6

```
cluster6 = 90    91    92    93    94    95    96    97    98    99   100
           101   102   103   104   106   107   110   111   112   117   141
```

```
Distance_Covered_By_Ant =

          591.3500
          568.5250
          589.7750
          644.6250
          619.6833
          579.3833
          594.0250
          720.6833
          567.4917
          769.5833
          600.6917
          552.7750
          663.1833
          683.9000
          547.3333
          680.3500
          718.2500
          625.3833
          582.9833
          577.2333
          569.6500
          562.6917
```

Ant_Best_Tour = 90 → 91 → 92 → 93 → 94 → 95 → 96 → 110 → 111 → 112 → 97 → 98 → 99 → 100 → 101 → 102 → 103 → 104 → 107 → 106 → 117 → 141

4.5 DISCUSSIONS

The node started by the best tour ant and the last node visited by the best tour ant can be interchanged based on the distance from the depot and the dumpsite, we therefore divide the nodes (junctions) of the entire sector into two, namely, level 1 and level 2 junctions, where level 1 junction is the starting node and level 2 is the last node before leaving the sector Amponsah and Salhi (2003).

In cluster 1, total distance used best the best ant = (336.33×2)m, and the best starting node is 116 and leave the cluster from node 127.

In cluster 2, total distance used by the best ant = (805.2×2) m, and the best starting node is 57 and leave the cluster from node 1.

In cluster 3, total distance used by the best ant = (952.7×2)m, and the best starting node is 64 and leave the cluster from node 41.

In cluster 4, total distance used best the best ant = (804×2)m, and the best starting node is 77 and leave the cluster from node 80.

In cluster 5, total distance used by the best ant = (737.7×2) m, and the best starting node is 85 and leave the cluster from node 142.

In cluster 6, total distance used by the best ant = (547.33×2)m, and the best starting node is 111 and leave the cluster from node 141.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.0 INTRODUCTION

The study was aimed at finding the minimum tour that can be used to collect solid waste at Kwadaso estate. After formulating the problem as vehicle routing problem, ant heuristic was implemented on each cluster to find the minimum tour. The chapter gives conclusion and recommendations for further research work.

5.1 CONCLUSION

Our modified ACO algorithm (ASoption code) was tested on some benchmark problems such as Ulysses 22 and Berlin 52, which gave comparable results.

We have been able to model selection of waste collection points as one centre clustering problem and determine optimal cluster using vertex 1-centre clustering algorithm. We have model the routing of cluster points as CCVRP and determine optimal cluster routing by ACO algorithm. We have also been able to determine optimum routing for waste collection in Kwadaso Estate with a total routing distance in the area as (672.66 + 1610.4 + 1905.4 + 1608 + 1475.4 + 1094.66)m = 6599.2m = 6.6km

Our results have given a six truck collection in the area instead of 8 trucks as done currently.

Our results have also given us an average of 1.1km routing distance per cluster as compared to their estimated tour length of 1.85km for a full load.

5.2 RECOMMENDATION

We recommend the model to stake holders, since the model has been able to solve the problem of:

- Arbitrary routing
- An undefined inter nodal distances between collection points and total tour distance
- Covering long distances for collection

We also recommend further research into collection of solid waste using capacitated clustering based on Euclidean distances.

References

1. Amponsah S. K. and Salhi S. (2003) The Investigation of Capacitated Arc Routing Problem: The collection of garbage in developing countries, PhD thesis (page 76), University of Birmingham, United Kingdom.
2. Badeau P., Gendreaou M., Guertin F., Potvin J. and Taillard E. D. (1997). A parallel tabu search heuristic for the vehicle routing problem with time windows," *Transportation Research*,(55), pp.109-122.
3. Bautista J. and Pereira J. (2004). Ant algorithms for urban waste collection routing. *Ant Colony Optimization and Swarm Intelligence, Proceedings, 3172*, 302-309.
4. Bautista J., Fernandez E. and Pereira J. (2008). Solving an urban waste collection problem using ants heuristics. *Computers & Operations Research, 35*(9), 3020-3033. doi:10.1016/j.cor.2007.01.029
5. Beasley J. E. (1983). Route first-Cluster second methods for vehicle routing. *Omega,* 11(4), 403-408.
6. Belenguer J. M., E. Benavent, Lacomme P. and Prins C. (2003). Heuristics and lower bounds for the mixed capacitated arc routing problem. Working Paper, Univ. Tecnologie de Troyes, France.
7. Benavent E., Campos V., Corberan A. and Mota E. (1992). The capacitated arc routing problem: lower bounds. *Networks*, 22: 669–690. doi: 10.1002/net.3230220706
8. Blattberg R., Kim B. and Neslin S. (2008). *Database Marketing: Analyzing and Managing Customers*. New York: Springer Science
9. Bonabeau E., Dorigo M., and Theraulaz G."Intelligence: From Natural to Artificial Systems", Oxford University Press.
10. Braysy O. and Gendreau M. (2001). "Metaheuristics for the Vehicle Routing Problem with Time Windows," Sintef Technical Report STF42 A01025, Department of Mathematics and Statistics, University of Vaasa, Finland.
11. Budayan C. (2008). *Strategic group analysis: Strategic perspective, differentiation and performance in construction*. Doctoral dissertation, Middle East Technical University
12. Budayan C., Dikmen I. and Birgonul T. (2008). Comparing the performance of traditional cluster analysis, self organizing maps and fuzzy C - means method for strategic grouping. *Expert Systems with Applications*, 36, pp 11772– 11781

13. Bullnheimer B., Hartl R. F. and Strauss C. (1998). "Applying the ant system to the vehicle routing problem," *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Boston.
14. Bullnheimer B., Hartl R. F. and Strauss C. (1999) "An improved ant system algorithm for the vehicle routing problem," *Annals of Operations Research,* vol.89, pp.319-328.
15. Chang N. B., Lu H. Y and Wei Y. L. (1997). GIS technology for vehicle routing and scheduling in solid waste collection systems. *Journal of Environmental Engineering- Asce, 123*(9), 901-910.
16. Chiang W. and Russell R. (1997). "A reactive tabu search metaheuristic for the vehicle routing problem with time windows", *INFORMS Journal on Computing*, (9), pp.417-430.
17. Chiang W., Russell R. (1996). "Simulated annealing metaheuristics for the vehicle routing problem with time windows," *Annals of Operations Research*, (63), pp.3-27.
18. Chuang K., Chiu M., Lin C. and Chen J. (1999). Model free functional MRI analysis using Kohonen clustering neural network and Fuzzy C Means. *IEEE Transactions on Medical Imaging*, 18(12), pp 1117 – 1128
19. Clarke A. (2009). Bridging industrial segmentation theory and practice. *Journal of Business-to-Business Marketing*, 16(4), pp 343 – 373
20. Clarke G. and Wright J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research, 12*(4), 568-581.
21. Clarke G. and Wright W. (1964). "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, (12), pp.568-581.
22. Clarke G. and Wright W.. (1964). "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, (12), pp.568-581.
23. Croes G. A. (1958). "A method for solving the traveling salesman problems," *Operations Research*, (6), pp.791-812, 1958.
24. Deneubourg J. L, Aron S., Goss S., and Pasteels J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behaviour*, 3: 159–168.
25. Doerner K. F., Hartl R. F. and Reimann M. (2000). "Ants solve time constrained pick-up and delivery problems with full truckloads," Technical Report, Lehrstuhl für Produktion und Logistik Institut für BWL, Wien.

26. Dorigo M, Maniezzo Vittorio, Colorni Alberto (1996). The Ant System: Optimization by a colony of cooperating agents [J]. IEEE Transactions on Systems, Man, and Cybernetics- -Part B,1996, 26(1)
27. Dorigo M. (1992). *Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale*. PhD thesis, Politecnico di Milano, Milan, Italy, 1992.
28. Dorigo M. and Di Caro G. (1999). The Ant Colony Optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill, London, UK.
29. Dorigo M., Di Caro G., and Gambardella L. M. (1999). Ant algorithms for discrete optimization *Artificial Life*, 5(2):137–172.
30. Everitt B. (1974) Cluster Analysis, Heinemann, London.
31. Fisher M. L. and Jaikumar R. (1981). A generalized assignment heuristic for vehicle routing. *Networks, 11*(2), 109-124.
32. Garcia B. D., Potvin J. and Rousseau J. (1994). "A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints," *Computers and Operations Research*, vol.21, no.9, pp.1025-1033, 1994.
33. Geetha S., Ponthalir G. and Vanathi P. T. (2009). PSG college of technology, Tamil Nadu, India.
34. Gillett B. E. and Miller R. L. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research, 22*(2), 340.
35. Glover F. and Laguna M. (1993). "Tabu search," *Modern Heuristic Techniques for Combinatorial Problems*, Blackwell, Oxford, pp.76-150, 1993.
36. Gokce E. I. (2004). Masters thesis, School of Engineering and Natural Sciences, sabanci University
37. Golden B. L, Dearmon J. S and Baker E. K. (1983). Computational experiments with algorithms for a class of routing-problems. *Computers & Operations Research, 10*(1), 47-59.
38. Hosseini S., Maleki A. and Gholamian M. (2010). Cluster analysis using data mining approach to develop CRM methodology to access the customer loyalty. *Expert Systems with Applications*, 37, pp 5259 – 5264
39. Israel A. and Iyigun C. (2008). Probabilistic D Clustering. *Journal of Classification*, 25, pp 5 – 26
40. Iyigun C. and Israel A. (2010). Semi-supervised probabilistic distance clustering and the uncertainty of classification in

41. Lacomme P., Prins C. and Ramdane-Cherif W. (2002). Fast algorithm for general arc routing problems. Presented in: IFORS 2002 Conference, Edinburgh, UK.
42. Laporte G. (2009). Fifty years of vehicle routing. *Transportation Science, 43*(4), 408-416. doi:10.1287/trsc.1090.0301
43. Li H. and Lim A. (2003). "Local Search with annealing-like restarts to solve the VRPTW," *European Journal of Operational Research*, Corrected Proof, Article in Press.
44. Li J., Borenstein D. and Mirchandania P. B. (2008). Truck scheduling for solid waste collection in the city of Porto Alegre, Brazil. *Omega-International Journal of Management Science, 36*(6), 1133-1149. doi:10.1016/j.omega.2006.04.007
45. Li L. Y. O and Eglese R. W. (1996). An interactive algorithm for vehicle routing for winter-gritting. *Journal of the Operational Research Society*, 47, 217–228.
46. Mourão M. C and Almeida M. T. (2000). Lower-bounding and heuristic methods for a refuse collection vehicle routing problem. *European Journal of Operational Research,121*(2), 420-434. doi:10.1016/S0377-2217(99)00045-4
47. Mourao M. C, Nunes A. C and Prins C. (2009). Heuristic methods for the sectoring arc routing problem. *European Journal of Operational Research, 196*(3), 856-868. doi:10.1016/j.ejor.2008.04.025
48. Mourão M. C. and Amado L. (2005). Heuristic method for a mixed capacitated arc routing problem: A refuse collection application. *European Journal of Operational Research, 160*(1), 139-153. doi:10.1016/j.ejor.2004.01.023
49. Ogwueleka T. Ch. (2009). Municipal solid waste characteristics and management in Nigeria. *Iranian Journal of Environmental Health Science & Engineering*, 6(3), 173-180.
50. Ozer M. (2001). User segmentation of online music services using fuzzy clustering. *Omega Int J of Management Science*, 29, pp 193 – 206
51. Potvin J. and Bengio S.. (1996). "The vehicle routing problem with time windows-part II: genetic search," *INFORMS Journal on Computing*, (8), pp.165-172.
52. Potvin J., Kervahut T., Garcia B. L. and Rousseau J. M. (1995). "The vehicle routing problem with time windows; part I: tabu search," *INFORMS Journal on Computing*, (8), pp.158-164.
53. Randall M. and Montgomery J. (2002). "Candidate set strategies for ant colony optimization," Technical Report, School of Information Technology, Bond University, QLD 4229, Australia.

54. Reimann M., Doerner K. and Hartl R. F. (2002). "Insertion based ants for vehicle routing problems with backhauls and time windows,"*Ant Algorithms*, Springer LNCS 2463, Berlin, pp.135–147.
55. Reimann M., K. Doerner K. and Hartl R. F. (2003). "Analyzing a Uni.ed Ant System for the VRP and Some of Its Variants," *EvoWorkshops 2003*, LNCS 2611, pp.300-310.
56. Santos L., Coutinho-Rodrigues J. and Current J. R. (2008). Implementing a multivehicle multi-route spatial decision support system for efficient trash collection in portugal. *Transportation Research Part A-Policy and Practice*, *42*(6), 922-934. doi:10.1016/j.tra.2007.08.009
57. Savelsbergh M. W. P. (1985). "Local search for routing problems with time windows," Annals of *Operations Research*, (4), pp.285-305, 1985.
58. Sbihi A.and Eglese R. W. (2007). Combinatorial optimization and green logistics. *4ora Quarterly Journal of Operations Research*, *5*(2), 99-116. doi:10.1007/s10288-007-0047-3
59. Schulze J. and Fahle T. (1999). "A parallel algorithm for the vehicle routing problem with time window constraints," *Combinatorial Optimization: Recent Advances in Theory and Praxis, Special volume of Annals of Operations Research*, 86, pp.585-607.
60. Simkin L. (2008). Achieving market segmentation from B2B sectorisation. *Journal of Business & Industrial Marketing*, 23(7), pp 464 – 474
61. Stützle T. and Hoos H. H. (1999). "MAX-MIN ant system and local search for combinatorial optimization problems," *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, pp. 313-329, Boston.
62. Tan K. C., Lee L. H., Zhu Q. L. and Ou K. (2001). "Artificial intelligence heuristics in solving vehicle routing problems with time windows," *Engineering Applications of Artificial Intelligence*, (14), pp.825-837.
63. Thangiah S., Osman I. H. and Sun T. (1994). "Hybrid genetic algorithm, simulated annealing and tabu search method for vehicle routing problems with time windows," Technical Report, UCK/OR94/4, Institute of Mathematics and Statistics, University of Kent, Canterbury, UK.

APPENDIX A; MATLAB CODE FOR ALL PAIR SHORTEST PATH

```
function z = floyd(w)
%% INITIALIZATION OF SOME USEFUL VARIABLES

    n = size(w);
    d = zeros(n);
    pred = zeros(n);
    for i = 1:n
        for j = 1:n
            d(i,j) = w(i,j);
            pred(i,j) = inf;
        end
    end


    for k = 1:n
        for i = 1:n
            for j = 1:n
                if (d(i,k) + d(k,j) < d(i,j))
                    d(i,j) = d(i,k) + d(k,j);
                    pred(i,j) = k;
                end
            end
        end
    end
    z = d;
```

APPENDIX B; MATLAB CODE FOR CAPACITATED CLUSTERING

```
function c = clusterNew11(q,d)
n = length(d);
capacity = 103;
k = ceil(sum(q)/capacity);
dd = q;    % load
dist = d;   % distance matrix from FLOYD

%sort and pick the first k centroid
[d_val,d_indx] = sort(dd);
d_indx1(1,:) = d_indx(end-k+1:end); % actual centroid

%create a new distance table from dist (P[i])
for i = 1:n
    dist(i,:) = dist(i,:)/dd(i);
end

min_check(1) = sum(dd(d_indx1(1,:)));  %sum of centroid from each iteration

for it = 2:1001 %iterative panel

    d_column = dist(:,d_indx1(it-1,:));
```

```
cluster1 = [];
clus_sum1 = 0;

cluster2 = [];
clus_sum2 = 0;

cluster3 = [];
clus_sum3 = 0;

cluster4 = [];
clus_sum4 = 0;

cluster5 = [];
clus_sum5 = 0;

cluster6 = [];
clus_sum6 = 0;

ix=1;
    while ~ isinf(d_column) | ix <= 1000
        if d_column == inf
            break
        else
            for i = 1:n
                [val,indx] = min(d_column(i,:));

                if clus_sum1 <= capacity & indx == 1
                    if d_column(i,:) == inf
                       d_column(i,:) = inf;
                    else
                        cluster1 = [cluster1,i];
                        temp = d_column(i,:);
                        d_column(i,:) = inf;
                        clus_sum1 = clus_sum1 + dd(i);
                        if clus_sum1 >= capacity
                            clus_sum1 = clus_sum1 - dd(i);
                            d_column(i,:) = temp;
                            cluster1(end) = [];

                            d_column(:,indx) = inf;
                        end
                    end
                elseif clus_sum2 <= capacity & indx == 2
                    if d_column(i,:) == inf
                        d_column(i,:) = inf;
                    else
                        cluster2 = [cluster2,i];
                        temp = d_column(i,:);
                        d_column(i,:) = inf;
                        clus_sum2 = clus_sum2 + dd(i);
                        if clus_sum2 >= capacity
                            clus_sum2 = clus_sum2 - dd(i);
                            d_column(i,:) = temp;
                            cluster2(end) = [];

                            d_column(:,indx) = inf;
                        end
                    end
                elseif clus_sum3 <= capacity & indx == 3
```

```
        if d_column(i,:) == inf
            d_column(i,:) = inf;
        else
            cluster3 = [cluster3,i];
            temp = d_column(i,:);
            d_column(i,:) = inf;
            clus_sum3 = clus_sum3 + dd(i);
            if clus_sum3 >= capacity
                clus_sum3 = clus_sum3 - dd(i);
                d_column(i,:) = temp;
                cluster3(end) = [];

                d_column(:,indx) = inf;
            end
        end
    elseif clus_sum4 <= capacity & indx == 4
        if d_column(i,:) == inf
           d_column(i,:) = inf;
        else
            cluster4 = [cluster4,i];
            temp = d_column(i,:);
            d_column(i,:) = inf;
            clus_sum4 = clus_sum4 + dd(i);
            if clus_sum4 >= capacity
                clus_sum4 = clus_sum4 - dd(i);
                d_column(i,:) = temp;
                cluster4(end) = [];

                d_column(:,indx) = inf;
            end
        end
    elseif clus_sum5 <= capacity & indx == 5
        if d_column(i,:) == inf
            d_column(i,:) = inf;
        else
            cluster5 = [cluster5,i];
            temp = d_column(i,:);
            d_column(i,:) = inf;
            clus_sum5 = clus_sum5 + dd(i);
            if clus_sum5 >= capacity
                clus_sum5 = clus_sum5 - dd(i);
                d_column(i,:) = temp;
                cluster5(end) = [];

                d_column(:,indx) = inf;
            end
        end
    elseif clus_sum6 <= capacity & indx == 6
        if d_column(i,:) == inf
            d_column(i,:) = inf;
        else
            cluster6 = [cluster6,i];
            temp = d_column(i,:);
            d_column(i,:) = inf;
            clus_sum6 = clus_sum6 + dd(i);
            if clus_sum6 >= capacity
                clus_sum1 = clus_sum1 - dd(i);
                d_column(i,:) = temp;
                cluster6(end) = [];
```

```
                        d_column(:,indx) = inf;
                    end
                end
            end
        end
        ix = ix+1;
    end
end

% Determination of new centroid from the cluster point
sub_cluster = d(cluster1,cluster1);
d_sub = max(sub_cluster,[],2);
[minVal d_ind]= min(d_sub);
d_indx2(1) = cluster1(d_ind);

sub_cluster = d(cluster2,cluster2);
d_sub = max(sub_cluster,[],2);
[minVal d_ind] = min(d_sub);
d_indx2(2) = cluster2(d_ind);

sub_cluster = d(cluster3,cluster3);
d_sub = max(sub_cluster,[],2);
[minVal d_ind] = min(d_sub);
d_indx2(3) = cluster3(d_ind);

sub_cluster = d(cluster4,cluster4);
d_sub = max(sub_cluster,[],2);
[minVal d_ind] = min(d_sub);
d_indx2(4) = cluster4(d_ind);

sub_cluster = d(cluster5,cluster5);
d_sub = max(sub_cluster,[],2);
[minVal d_ind] = min(d_sub);
d_indx2(5) = cluster5(d_ind);

sub_cluster = d(cluster6,cluster6);
d_sub = max(sub_cluster,[],2);
[minVal d_ind] = min(d_sub);
d_indx2(6) = cluster6(d_ind);

d_indx1(it,:) = d_indx2;

if
sum([clus_sum1,clus_sum2,clus_sum3,clus_sum4,clus_sum5,clus_sum6]) ==
sum(q)
    min_check(it) = sum(dd(d_indx1(it,:)));
else
    min_check(it) = inf;
end

if it == 1000
    [min_clust_val,min_clust_indx] = min(min_check);
    stable_cluster = d_indx1(min_clust_indx,:)
    d_indx1(it,:) = d_indx1(min_clust_indx-1,:);
elseif it == 1001
    cluster1 = cluster1
    pause
```

```
            cluster1i = dist(cluster1,cluster1);
            ACOReview(cluster1i)

disp('=====================================================================
===============================================')

            pause
            cluster2 = cluster2
            cluster2i = dist(cluster2,cluster2);
            ACOReview(cluster2i)

disp('=====================================================================
===============================================')

            pause
            cluster3 = cluster3
            cluster3i = dist(cluster3,cluster3);
            ACOReview(cluster3i)

disp('=====================================================================
===============================================')

            pause
            cluster4 = cluster4
            cluster4i = dist(cluster4,cluster4);
            ACOReview(cluster4i)

disp('=====================================================================
===============================================')

            pause
            cluster5 = cluster5
            cluster5i = dist(cluster5,cluster5);
            ACOReview(cluster5i)

disp('=====================================================================
===============================================')

            pause
            cluster6 = cluster6
            cluster6i = dist(cluster6,cluster6);
            ACOReview(cluster6i)

disp('=====================================================================
===============================================')
            break
        end
end
 c = clus_sum1+clus_sum2+clus_sum3+clus_sum4+clus_sum5+clus_sum6;
 total_sum =
length(cluster1)+length(cluster2)+length(cluster3)+length(cluster4)+length(
cluster5)+length(cluster6);
end
```

APPEDIX C; MATLAB CODE FOR ANT HEURISTIC

```
function ACOReview(clusterX,p_cluster)

%% START declare of own Variable for testing
Dimension = length(p_cluster);

% ====================================
%=====================================
global clus_No
clus_No = getLength(p_cluster,clusterX);
% ====================================
%=====================================

NodeWeight = [];
Name =  'cluster';
% END declare of own Variable for testing


disp([num2str(Dimension),' nodes from this ',Name,' has been read in']);
disp(['AS start at ',datestr(now)]);
%%%%%%%%%%%%% the key parameters of Ant System %%%%%%%%%



MaxITime=1e3;
AntNum=Dimension; %depends on # of nodes
alpha=1;
beta=5;
rho=0.65;
%%%%%%%%%%%%% the key parameters of Ant System %%%%%%%%%
fprintf('Showing Iterative Best Solution:\n');

finalOutput = ...
AS(NodeWeight,AntNum,MaxITime,alpha,beta,rho);

disp(['AS stop at ',datestr(now)]);

function
[GBTour,GBLength,Option,IBRecord]=AS(WeightMatrix,AntNum,MaxITime,alpha,bet
a,rho)
%% (Ant System) date:070427
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Reference£º
% Dorigo M, Maniezzo Vittorio, Colorni Alberto.(1996)
Modified by D. Otoo (2012)
global ASOption Problem AntSystem clus_No
ASOption = InitParameter(AntNum,alpha,beta,rho,MaxITime);
Problem = InitProblem(WeightMatrix);
AntSystem = InitAntSystem();

% =====================================
clust_index = clus_No.m_clust;
%======================================

ITime = 0;
IBRecord = [];
```

```
while 1
    InitStartPoint();
    for step = 2:ASOption.n
        for ant = 1:ASOption.m
            P = CaculateShiftProb(step,ant);
            nextnode = Roulette(P,1);
            RefreshTabu(step,ant,nextnode);

        end
    end

    ITime = ITime + 1;
    CaculateToursLength();

    GlobleRefreshPheromone();
    ANB = CaculateANB();
    [GBTour,GBLength,IBRecord(:,ITime)] = GetResults(ITime,ANB);

    if Terminate(ITime,ANB)
       Ant_Tour = AntSystem.tours
       Distance_Covered_By_Ant = AntSystem.lengths
       Ant_Best_Tour = AntSystem.bestTour;
       Ant_Best_Tour = clust_index(Ant_Best_Tour)
        break;
    end
end
Option = ASOption;
%% -----------------------------------------------------------------
function ASOption = InitParameter(AntNum,alpha,beta,rho,MaxITime)
global clus_No

ASOption.n = clus_No.n;
ASOption.m = AntNum;
ASOption.alpha = alpha;
ASOption.beta = beta;
ASOption.rho = rho;
ASOption.MaxITime = MaxITime;
ASOption.OptITime = 1;
ASOption.Q = 10;
ASOption.C = 100;
ASOption.lambda = 0.15;
ASOption.ANBmin = 2;
ASOption.GBLength = inf;
ASOption.GBTour = zeros(clus_No.n,1);
ASOption.DispInterval = 10;
rand('state',sum(100*clock));
%% -----------------------------------------------------------------
function Problem = InitProblem(WeightMatrix)
global ASOption
n = ASOption.n;
MatrixTau = (ones(n,n)-eye(n,n))*ASOption.C;
Distances = WeightMatrix;
SymmetryFlag = false;
if isempty(WeightMatrix)
    Distances = CalculateDistance;
    SymmetryFlag = true;
end
Problem = struct('dis',Distances,'tau',MatrixTau,'symmetry',SymmetryFlag);
%% -----------------------------------------------------------------
```

```
function AntSystem = InitAntSystem()
global ASOption clus_No
AntTours = zeros(ASOption.m,ASOption.n);
ToursLength = zeros(ASOption.m,1);
% % ==================================
% % ==================================
AntBestTour = zeros(1,clus_No.n);
AntSystem =
struct('tours',AntTours,'lengths',ToursLength,'bestTour',AntBestTour);
%% --------------------------------------------------------------
function InitStartPoint()
global AntSystem ASOption clus_No
AntSystem.tours = zeros(ASOption.m,ASOption.n);
rand('state',sum(100*clock));
AntSystem.tours(:,1) = randperm(clus_No.n)';
%randint(ASOption.m,1,[1,ASOption.n]);
AntSystem.lengths = zeros(ASOption.m,1);
%% --------------------------------------------------------------
function Probs = CaculateShiftProb(step_i, ant_k)
global AntSystem ASOption Problem
CurrentNode = AntSystem.tours(ant_k, step_i-1);
VisitedNodes = AntSystem.tours(ant_k, 1:step_i-1);
tau_i = Problem.tau(CurrentNode,:);
tau_i(1,VisitedNodes) = 0;
dis_i = Problem.dis(CurrentNode,:);
dis_i(1,CurrentNode) = 1;
Probs = (tau_i.^ASOption.alpha).*((1./dis_i).^ASOption.beta);
if sum(Probs) ~= 0
    Probs = Probs/sum(Probs);
else
    NoVisitedNodes = setdiff(1:ASOption.n,VisitedNodes);
    Probs(1,NoVisitedNodes) = 1/length(NoVisitedNodes);
end
%% --------------------------------------------------------------
function Select = Roulette(P,num)
m = length(P);
flag = (1-sum(P)<=1e-5);
Select = zeros(1,num);
rand('state',sum(100*clock));
r = rand(1,num);
for i=1:num
    sumP = 0;
    j = ceil(m*rand);
    while (sumP<r(i)) && flag
        sumP = sumP + P(mod(j-1,m)+1);
        j = j+1;
    end
    Select(i) = mod(j-2,m)+1;
end
%% --------------------------------------------------------------
function RefreshTabu(step_i,ant_k,nextnode)
global AntSystem
AntSystem.tours(ant_k,step_i) = nextnode;
%% --------------------------------------------------------------
function CaculateToursLength()
global ASOption AntSystem
x = CalculateDistance;
p = AntSystem.tours;

Lengths = zeros(ASOption.m,1);
```

```
for j=1:ASOption.n
    pRow = p(j,:);
    sumRow = 0;
    for i=1:ASOption.n-1
        sumRow = sumRow + x(pRow(i),pRow(i+1));
    end
    Lengths(j) = sumRow;
end

[BestVal,BestIdx] = min(Lengths);
BestTour = p(BestIdx,:);

AntSystem.bestTour = BestTour;
AntSystem.lengths = Lengths;
%% -----------------------------------------------------------------
function [GBTour,GBLength,Record] = GetResults(ITime,ANB)
global AntSystem ASOption
[IBLength,AntIndex] = min(AntSystem.lengths);
IBTour = AntSystem.tours(AntIndex,:);
if IBLength<=ASOption.GBLength
    ASOption.GBLength = IBLength;
    ASOption.GBTour = IBTour;
    ASOption.OptITime = ITime;
end
GBTour = ASOption.GBTour';
GBLength = ASOption.GBLength;
Record = [IBLength,ANB,IBTour]';
%% -----------------------------------------------------------------
function GlobleRefreshPheromone()
global AntSystem ASOption Problem
AT = AntSystem.tours;
TL = AntSystem.lengths;
sumdtau=zeros(ASOption.n,ASOption.n);
for k=1:ASOption.m
    for i=1:ASOption.n
        sumdtau(AT(k,i),AT(k,i))=sumdtau(AT(k,i),AT(k,i))+ASOption.Q/TL(k);
        if Problem.symmetry
            sumdtau(AT(k,i),AT(k,i))=sumdtau(AT(k,i),AT(k,i));
        end
    end
end
Problem.tau=Problem.tau*(1-ASOption.rho)+sumdtau;
%% -----------------------------------------------------------------
function flag = Terminate(ITime,ANB)
global ASOption
flag = false;
if ANB<=ASOption.ANBmin || ITime>=ASOption.MaxITime
    flag = true;
end
%% -----------------------------------------------------------------
function ANB = CaculateANB()
global ASOption Problem
mintau = min(Problem.tau+ASOption.C*eye(ASOption.n,ASOption.n));
sigma = max(Problem.tau) - mintau;
dis = Problem.tau - repmat(sigma*ASOption.lambda+mintau,ASOption.n,1);
NB = sum(dis>=0,1);
ANB = sum(NB)/ASOption.n;
%% -----------------------------------------------------------------

function Distances = CalculateDistance
global clus_No
```

```
Distances = clus_No.Vals;

%% ----------------------------------------------------------------
function clus_No = getLength(n,clusterI)
clus_No.n = length(n);
clus_No.Vals = n;
clus_No.m_clust = clusterI;
```