KWAME NKRUMAH UNIVERSITY UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI

INSTITUTE OF DISTANCE LEARNING

OPTIMAL ROUTE TOUR OF THE REGIONAL CAPITALS IN GHANA

CASE STUDY: WEST AFRICAN EXAMINATIONS COUNCIL'S QUESTION PAPERS DEPOT INSPECTION PLAN

BY

AMOS YAW SARFO (BSC CHEMICAL ENGINEERING)

A THESIS SUBMITTED TO THE INSTITUTE OF LEARNING, KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE MASTER OF SCIENCE DEGREE IN INDUSTRIAL MATHEMATICS

JUNE, 2014

DECLARATION

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

SARFO YAW AMOS (PG6324511) ……. ……..

(Student's name & index number) Signature Date

Certified by

MR. CHARLES SEBIL ……. ……..

(Supervisor's name) Signature Date

Certified by

PROF. S.K. AMPONSAH …….. ……..

(Head of Department) Signature Date

DEDICATION

This work is dedicated to my dad Mr. John Sarfo, my guardians Mr. and Mrs Emmanuel Mensah, for their prayers and financial assistance towards the completion of my course.

ACKNOWLEDGEMENT

I am very grateful to the Almighty Jehovah who strengthens and inspires me in all my endeavors. My sincere appreciation also goes to my dad John Sarfo for his prayers, financial support and encouragement to my success. Dad God richly bless you for the love you showed to me.

Special thanks to Mr. Charles Sebil my project supervisor, for having time to read, correct and his useful suggestion which helped me in the writing of the project. I have no other way of showing my heartfelt appreciation than saying a very big THANK YOU and God bless you.

I also like to express my gratitude to all the lecturers in the department of mathematics, KNUST, Kumasi for their advice and support.

ABSTRACT

The travelling salesman problem is considered to be a classic example of what is known as a tour problem. Essentially, any type of tour problem involves making a series of stops along a designated route and making a return journey without ever making a second visit to any previous stop. Generally, a tour problem is present when there is concern on making the most of available resources such as time and mode of travel to accomplish the most in results.

The objective of the thesis is to use Simulated Annealing to determine the optimal route for visiting all the ten regions in Ghana, save time and minimize expenditure.

This study formulated a real-life problem of WAEC as a TSP, modelled as network problem and a matlab program was prepared using simulated annealing algorithm in solving the problem. It was observed that the route that gave minimum achievable inspection plan was 3 – 9 – 6 – 10 – 8 – 5 – 2 – 4 – 7 – 1 at the minimum distance of 2229 km. Thus;

Koforidua → Ho → Tamale → Bolgatanga → Wa → Sunyani → Kumasi → Takoradi → Cape Coast → Accra

Hence an officer assigned to inspect the various question paper depots can start from Accra and end in Accra following the order above.

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

1.0 INTRODUCTION

Ghana is a sub-Saharan African country with a population of 24,391,823million according to the 2010 population census and annual population growth of 2.4.

Ghana covers a total land area of about 92456sqm or 239460sqkm. It has ten (10) administrative regions namely; Greater Accra, Western, Eastern, Central, Volta, Ashanti, Brong-Ahafo, Northern, Upper East and Upper West. It also has 170 districts.

Ghana contains about 32,250km of road of which about 12,000km are main roads. Approximately 6,000km are paved; the remainder is gravel, crushed stones or graded earth. The country's rail network is 953km in length; all track is 1.067m (narrow) gauge and all but 32km are single track. Poor rural infrastructure has been blamed for problems in agriculture, partly because transportation costs accounts for about 70% of the difference between farm price and retail prices. Only about one-third of the feeder road network can carry vehicular traffic.

The Government has no plans of extending the railway system beyond its limited coverage of the southwestern regions of the country. Figures indicate a downward trend in passenger traffic from a high of 389 per km in 1988 to 277 per km in 1990. The government has instead focused on improving the road system. Since 1985 all trunk roads and about 2,900km of feeder roads as well as a number of bridges and drainage systems have been undergoing repairs.

The travelling salesman or salesperson problem (TSP) is one of the most well-known optimization problems in the literature. It has attracted the attention of many researchers over the last half a century because of its simple problem description but simultaneously its associated difficulty in obtaining an optimal solution efficiently. The travelling-salesman problem involves a salesman who must make a tour of a number of cities using the shortest path available. For each number of cities n, the number of paths which must be explored is n!, causing this problem to grow exponentially rather than as a polynomial.

The travelling salesperson problem (TSP) is a classic model for various production and scheduling problems. Many production and scheduling problems ultimately can be reduced to the simple concept that there is a salesperson who must travel from city to city (visiting each city exactly once) and wishes to minimize the total distance travelled during his tour of all n cities. Obtaining a solution to the problem of a salesperson visiting n cities while minimizing the total distance travelled is one of the most studied combinatorial optimization problems. While there are variations of the TSP, the Euclidean TSP is NP-hard (Schmitt and Amini, 1998; Falkenauer, 1998). The interest in this particular type of problem is due to how common the problem is and how difficult the problem is to solve when n becomes sufficiently large.

In this chapter of the thesis, an overview of the travelling salesman problem would be given; a brief description of the problem statement of the thesis is also presented together with the objectives, the methodology, the justification and the organization of the thesis.

1.1 BACKGROUND OF THE STUDY

The Travelling Salesman Problem (TSP) is a problem whose solution has eluded many mathematicians for years. Currently there is no solution to the TSP that has satisfied mathematicians. The TSP has a very rich history. Historically, mathematics related to the TSP was developed in the 1800's by Sir William Rowan Hamilton and Thomas Penyngton Kirkman, Irish and British mathematicians, respectively. Specifically, Hamilton was the creator of the Icosian Game in 1857. It was a pegboard with twenty holes that required each vertex to be visited only once, no edge to be visited more than once, and the ending point being the same as the starting point. This kind of path was eventually referred to as a Hamiltonian circuit. However, the general form of the TSP was first studied by Karl Menger in Vienna and Harvard in the late 1920's or early 1930's.

TSPs were first studied in the 1930s by mathematician and economist Karl Menger in Vienna and Harvard. It was later investigated by Hassler Whitney and Merrill Flood at Princeton. Applegate et al., (1994) solved TSP containing 7,397 cities. Later in 1998, they solved it using 13,509 cities in United States. In 2001, the authors found the optimal tour of 15,112 cities in Germany. Later in 2004, TSP of visiting all 24,978 cities in Sweden was solved; a tour of length of approximately 72,500 kilometers was found and it was proven that no shorter tour exists. This is currently the largest solved TSP.

The travelling salesman problem (TSP) is a typical example of a very hard combinatorial optimization problem. The problem is to find the shortest tour that passes through each vertex in a given graph exactly once. The TSP has received considerable attention over the last two decades and various approaches are proposed to solve the problem. As early as in 1954, optimal solution to travelling salesman problem with 49 numbers of cities

has been obtained. In 1970's Held and Karp used minimum spanning tree to solve the TSP with 64 cities. In 1971, Bellmore and Malone solved TSP using sub tour elimination .In 1980's, Crowder and Padberg solved the problem with 318 cities using cutting-plane method. In 1991 Grötschel and Holland proposed a solution for large scale TSP. Applegate et al., (1998, 2001 and 2004) proposed solution for TSP using cuts that solved 13509,15112 ,24978 cities respectively. The solutions worked well up to 5000 cities and can be used up to 33,810 cities.

1.2 STATEMENT OF THE PROBLEM

Nowadays, the route management is very important to make sure the user can arrive at the destination the fastest. In the transportation industry, the route that will be generated should consider the cost and time constraints which are dependent on the distance travelled using the route. Although from human logical thinking, the route can be generated easily but the calculation of checking the route whether it is optimal route or not is difficult and will take long time to be implemented.

WAEC Ghana, during its various Examination seasons, sends officers to inspect the various question paper depots and examination centres where security materials are kept to ascertain whether the regulations regarding the safety of the materials are complied with in the various regions in Ghana.

An officer moves from the various regional capitals and is expected to visit as many examination depots and centres as possible on each route within each journey.

Inasmuch as to manage time and minimize cost, one of the best and sure ways is by taking the optimal route of an officer in all his/her inspection/ visitation.

The specific form of problem that this thesis seeks to solve is to mathematically model the West African Examinations Council Depot inspection problem as travelling salesman problem (TSP) and solve the problem.

1.3 OBJECTIVES

The objective of this research is to determine the optimal route for visitation and to model the West African Examinations Council depot inspection problem in all the regional capitals of Ghana.

1.4 JUSTIFICATION

WAEC Ghana, during its various Examination seasons, sends officers to inspect the various question paper depots and examination centres where security materials are kept to ascertain whether the regulations regarding the safety of the materials are complied with in the various regions in Ghana.

At the time of this work, records show that there is no laid down procedure for determining which routes to be used by inspection officers. The routes are chosen arbitrarily and sometimes the driver's discretion is the determining factor. The maximum number of centres they normally visit were three on a route.

This thesis will seek to address the problem of WAEC officers travelling more distances and will minimize the total distance of the inspection so as to manage time and minimize expenditure.

1.5 METHODOLOGY

The West African Examinations Council's question papers depot inspection will be modeled as a Travelling Salesman Problem (TSP). Simulated Annealing (SA) is the method that will be used to solve the TSP model. This is because the SA is a generic probabilistic metaheuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space when solving TSP.

1.6 ORGANIZATION OF THE THESIS

Chapter one deals with introduction, chapter two talks about the literature review, chapter three methodology, chapter four is the collection of data, analysis and discussion. Conclusion and recommendations are discussed in chapter five.

1.7 SUMMARY

In chapter one, we presented brief history of Ghana and introduction to TSP, background to the study, statement of the problem, objectives of the thesis and methodology.

In the next chapter, we shall review some literature in the field of TSP and SA.

CHAPTER 2

LITERATURE REVIEW

2.0 INTRODUCTION

In this chapter we shall review some works done by other people in the field of TSP and SA where we shall look at the various methods used to solve the TSP and their findings.

In 2005, Cook et al computed an optimal tour through a 33,810-city instance given by a microchip layout problem, currently the largest solved TSPLIB instance. For many other instances with millions of cities, solutions can be found that are guaranteed to be within 1% of an optimal tour.

Dantzig et al (1954), also used linear programming (LP) relaxation to solve the integer formulation by suitably chosing linear inequality to the list of constraints continuously. Eastman (1958), Held and Karp (1970), Smith et al, Carpaneto and Toth, Balas and Christofides proposed branch and bound (B & B) algorithm based on assignment problem relaxation of the original TSP formulation. Some Branch and Cut (B & C) based exact algorithms were developed by Crowder and Padperg, Padberg and Hong, Grotschel and Holland.

Ratliff and Rosenthall (1983) studied a problem of order-picking associated with material handling in a warehouse. Assume that at a warehouse an order arrives for a certain subsets of the items stored in the warehouse. Some vehicle has to collect all items of this order to ship them to the customer. The relation to the TSP is immediately

seen. The storage locations of the items correspond to the nodes of the graph. The distance between two nodes is given by the time needed to move the vehicle from one location to the other. The problem of finding a shortest route for the vehicle with minimum pickup time can now be solved as a TSP.

Davoian and Gorlatch (2005) presented a new modification of the Genetic Algorithm (GA) for solving the classical Travelling Salesman Problem (TSP), with the objective of achieving its efficient implementation on multiprocessor machines. The authors described the new features of our GA as compared to existing algorithms, and developed a new parallelization scheme, applicable to arbitrary GAs. In addition to parallel processes and iterative data exchanges between the involved populations, our parallel implementation also contains a generation of new possible solutions (strangers), which eliminates typical drawbacks of GA and extends the search area. The proposed algorithm allows for acceleration of the solution process and generates solutions of better quality as compared with previously developed GA versions.

Applegate et al (1994) solved a traveling salesman problem which models the production of printed circuit boards having 7,397 holes (cities), and in 1998, the same authors solved a problem over the 13,509 largest cities in the U.S. For problems with large number of nodes as cities the TSP becomes more difficult to solve.

Kenneth and Ruth (2007) studied a new multi-period variation of the M-travelling salesman problem. The problem arises in efficient scheduling of optimal interviews among tour brokers and vendors at conventions of the tourism and travel industry. In

classical travelling salesman problem vocabulary, a salesman is a tour broker at the convention and a city is a vendor's booth. In this problem, more than one salesman may be required to visit a city, but at most one salesman per time period can visit each city. The heuristic solution method presented is polynomial and is guaranteed to produce a non-conflicting set of salesmen's tours. The results of an implementation of the method for a recent convention are also reported.

Bellmore and Hong considered a Single Depot Multiple Travelling Salesman Problem (SDMTSP) where each salesman is available for service at a specific cost and the edge costs need not satisfy triangle inequality. Since the objective is to reduce the total cost travelled by the salesmen, there could be situations when the optimal solution will not necessitate using all the salesmen. Bellmore and Hong provide a way of transforming this single depot MTSP to a standard TSP for the asymmetric case.

Hong and Padberg present a more elegant transformation for the same problem in Rao discusses the symmetric version of the SDMTSP in Jonker and Volgenant give an improved transformation for a variant of the symmetric, SDMTSP where each salesman has to visit at least one target. Currently, there is no transformation available for the MDMTSP when each salesman must return to his initial depot for more than 2 depots.

Bernd and Peter (1996) presented an approach which incorporates problem specific knowledge into a genetic algorithm which is used to compute near-optimum solutions to travelling salesman problems (TSP). The approach is based on using a tour construction heuristic for generating the initial population, a tour improvement heuristic for finding

local optimal in a given TSP search space, and new genetic operators for effectively searching the space of local optima in order to find the global optimum. The quality and efficiency of solutions obtained for a set of TSP instances containing between 318 and 1400 cities are presented.

The problem of scheduling buses is investigated by Angel et al as a variation of the mTSP with some side constraints. The objective of the scheduling is to obtain a bus loading pattern such that the number of routes is minimized, the total distance travelled by all buses is kept at minimum, no bus is overloaded and the time required to traverse any route does not exceed a maximum allowed policy.

Svestka and Huckfeldt report an application for deposit carrying between different branch banks. Here, deposits need to be picked up at branch banks and returned to the central office by a crew of messengers. The problem is to determine the routes of messengers with a total minimum cost.

Zakir (2010) presented a new crossover operator, Sequential Constructive crossover (SCX), for a genetic algorithm that generates high quality solutions to the travelling salesman Problem (TSP). The sequential constructive crossover operator constructs an offspring from a pair of parents using better edges on the basis of their values that may be present in the parents' structure maintaining the sequence of nodes in the parent chromosomes. The efficiency of the SCX is compared as against some existing crossover operators; namely, edge recombination crossover (ERX) and generalized N-

point crossover (GNX) for some benchmark TSPLIB instances. Experimental results show that the new crossover operator is better than the ERX and GNX.

Lenstra and Rinnooy Kan describe two similar applications, where the first application consists of finding the routes of a technical crew, which has to visit telephone boxes in North Holland. The second application involves designing the routes of vehicles to visit 200 mailboxes in Utrecht, such that the number of vehicles used is minimized.
Another application of the mTSP in crew scheduling is reported by Zhang et al., who investigate the problem of scheduling multiple teams of photographers to a large number of elementary and secondary schools.

Croes (1958) proposed a variant of 3-opt together with an enumeration scheme for computing an optimal tour. He solved the Dantzig-Fulkerson-Johnson 49-city example in 70 hours by hand. He also solved several of the Robacker examples in an average time of 25 minutes per example.

Bock (1958) describes a 3-opt algorithm together with an enumeration scheme for computing an optimal tour. The author tested his algorithm on some 10-city instance using an IBM 650 computer.

Gilbert and Hofstra describe an application of a multiperiod variation of the mTSP, where the problem arises in scheduling interviews between tour brokers and vendors of the tourism industry. Each broker corresponds to a salesman who must visit a specified set of vendor booths, which are represented by a set of *T* cities.

The algorithm of Held and Karp (1971) was the basis of some major publications in 1974. In one case, Hansen and Krarup (1974) tested their version of Held-Karp (1971) on the 57-city instance of Karg and Thompson 1964 and a set of instances having random edge lengths.

Smith and Thompson, 1977 presented some improvements to the Held-Karp algorithm tested their methods on examples which included the 57-city instance of Karg and Thompson 1964 and a set of ten 60-city random Eucliean instances. The decade ended with a survey on algorithms for the TSP and the asymmetric TSP in Buckard, (1979).

The first computer implementation of the Dantzig-Fulkerson-Johnson cutting-plane method for solving the traveling salesman problem, written by Martin, used subtour inequalities as well as cutting planes of Gomory's type. The practice of looking for and using cuts that match prescribed templates in conjunction with Gomory cuts was continued in computer codes of Miliotis, Land, and Fleischmann.

Grotschel, Padberg, and Hong advocated a different policy, where the template paradigm is the only source of cuts; furthermore, they argued for drawing the templates exclusively from the set of linear inequalities that induce facets of the TSP polytope. These policies were adopted in the work of Crowder and Padberg, in the work of Grotschel and Holland, and in the work of Padberg and Rinaldi; their computer codes produced the most impressive computational TSP successes of the nineteen eighties.

Eventually, the template paradigm became the standard frame of reference for cutting planes in the TSP.

In the next chapter, we will be considering some of the methods used to solve the TSP and extensively use the simulated annealing method to solve the campaign visitation of a presidential aspirant which will be modeled as a TSP.

CHAPTER 3

METHODOLOGY

3.0 INTRODUCTION

This chapter focuses on the overview, formulation of the TSP model, some terminologies associated with TSP and other methods used to solve the TSP.

The chapter also discusses the application of the TSP and other linkages with other problems.

3.1 OVERVIEW

Many managerial problems, like routing problems, facility location problems, scheduling problems, network design problems, can either be modeled as combinatorial optimization problems, or solve combinatorial optimization problems as sub-problems. A very commonly researched combinatorial optimization problem in this and other contexts is the Traveling Salesman Problem (TSP). In a TSP, we are given a weighted graph with n nodes, and are required to find a tour in the graph visiting each node exactly once such that the sum of the costs of the edges or arcs in the tour is the minimum possible. The number n is commonly referred to as the size of the TSP. TSPs serve as a representation of many managerial problems, especially in logistics. Many more problems, though not obviously related to the TSP can be modeled as TSPs. A large number of other problems are not equivalent to solving TSPs, but solve TSPs as sub-problems.

The traveling salesman problem (TSP) is one which has commanded much attention of mathematicians and computer scientists specifically because it is so easy to describe and

so difficult to solve. The problem can simply be stated as: if a traveling salesman wishes to visit exactly once each of a list of *m* cities (where the cost of traveling from city *i* to city j is c_{ij}) and then return to the home city, what is the least costly route the traveling salesman can take? A complete historical development of this and related problems can be found in Hoffman and Wolfe (1985).

The importance of the TSP is that it is representative of a larger class of problems known as combinatorial optimization problems. The TSP problem belongs in the class of combinatorial optimization problems known as NP-complete. Specifically, if one can find an efficient algorithm (i.e., an algorithm that will be guaranteed to find the optimal solution in a polynomial number of steps) for the traveling salesman problem, then efficient algorithms could be found for all other problems in the NP-complete class. To date, however, no one has found a polynomial time algorithm for the TSP. Does that mean that it is impossible to solve *any* large instances of such problems? Many practical optimization problems of truly large scale are solved to optimality routinely. In 1994, Applegate, et. al. solved a traveling salesman problem which models the production of printed circuit boards having 7,397 holes (cities), and, in 1998, the same authors solved a problem over the 13,509 largest cities in the U.S. So, although the question of what it is that makes a problem "difficult" may remain open, the computational record of specific instances of TSP problems coming from practical applications is optimistic.

How are such problems tackled today? Obviously, one cannot consider a brute force approach. In one example of a 16 city traveling salesman problem -- the problem of Homer's Ulysses attempting to visit the cities described in *The Odyssey* exactly once -- there are 653,837,184,000 distinct routes, (Grötschel and Padberg, 1993). Enumerating

all such roundtrips to find a shortest one took 92 hours on a powerful workstation. Rather than enumerating all possibilities, successful algorithms for solving the TSP problem have been capable of eliminating most of the roundtrips without ever explicitly considering them.

3.2 TERMINOLOGIES

Graph theory: is the study of graphs, which are mathematical structures used to model pairwise relations between objects. A *graph* in this context is made up of *vertices* or *nodes* and lines called *edges* that connect them. A graph may be *undirected*, meaning that there is no distinction between the two vertices associated with each edge, or its edges may be *directed* from one vertex to another.

Hamiltonian path: is a path that uses each vertex of a graph exactly once.

Hamiltonian circuit: is a path that uses all the vertexes of a graph and ends with a starting vertex.

The first step to solving instances of large TSPs must be to find a good mathematical formulation of the problem. In the case of the traveling salesman problem, the mathematical structure is a graph where each city is denoted by a point (or node) and lines are drawn connecting every two nodes (called arcs or edges). Associated with every line is a distance (or cost). When the salesman can get from every city to every other city directly, then the graph is said to be *complete*. A round-trip of the cities corresponds to some subset of the lines, and is called a tour or a Hamiltonian cycle in graph theory. The length of a tour is the sum of the lengths of the lines in the round-trip.

3.3 FORMULATION OF THE TSP MODEL

Depending upon whether or not the direction in which an edge of the graph is traversed matters, one distinguishes the *asymmetric* from the *symmetric* traveling salesman problem. To formulate the *asymmetric* TSP on *m* cities, one introduces zero-one variables

$$x_{ij} = \begin{cases} 1 \text{ if the edge } i \to j \text{ is in the tour} \\ 0 \text{ otherwise} \end{cases}$$

and given the fact that every node of the graph must have exactly one edge pointing towards it and one pointing away from it, one obtains the classic assignment problem. These constraints alone are not enough since this formulation would allow "subtours", that is, it would allow disjoint loops to occur. For this reason, a proper formulation of the asymmetric traveling salesman problem must remove these subtours from consideration by the addition of "subtour elimination" constraints. The problem then becomes

$$min \sum_{j=1}^{m} \sum_{i=1}^{m} CijXij$$

$$s.t \sum_{j=1}^{m} Xij = 1 \qquad for\ i = 1, \dots, m$$

$$\sum_{i=1}^{m} Xij = 1 \qquad for\ j = 1, \dots, m$$

$$\sum_{i=k}^{m} \sum_{j=k}^{m} Xij \leq \lfloor K \rfloor - 1 \qquad for\ all\ K \subset \{1, \dots, m\}$$

$$Xij = 0 \; or \; 1 \qquad for \; all \; i, j$$

where *K* is any nonempty proper subset of the cities *1,...,m*. The cost c_{ij} is allowed to be different from the cost c_{ji}. Note that there are m(m-1) zero-one variables in this formulation.

To formulate the *symmetric* traveling salesman problem, one notes that the direction traversed is immaterial, so that $c_{ij} = c_{ji}$. Since direction does not now matter, one can consider the graph where there is only one arc (undirected) between every two nodes. Thus, we let $x_j \; \varepsilon \; \{0,1\}$ be the decision variable where *j* runs through all edges *E* of the undirected graph and c_j is the cost of traveling that edge. To find a tour in this graph, one must select a subset of edges such that every node is contained in exactly two of the edges selected. Thus, the problem can be formulated as a 2-matching problem in the graph G^v having m(m-1)/2 zero-one variables, i.e. half of the number of the previous formulation. As in the asymmetric case, subtours must be eliminated through subtour elimination constraints. The problem can therefore be formulated as:

$$min 1/2 \sum_{j=1}^{m} \sum_{k=J(j)}^{m} CkXk$$

$$s.t \sum_{k=J(j)}^{m} Xk = 2 \qquad for \; all \; j = 1, \dots, m$$

$$\sum_{j=E(k)}^{m} Xj \leq \lfloor K \rfloor - 1 \qquad for\ all\ K \subset \{1, \ldots, m\}$$

$$Xj = 0\ or\ 1 \qquad for\ all\ j \epsilon E$$

where $J(j)$ is the set of all undirected edges connected to node j and $E(K)$ is the subset of all undirected edges connecting the cities in any proper, nonempty subset K of all cities. Of course, the symmetric problem is a special case of the asymmetric one, but practical experience has shown that algorithms for the asymmetric problem perform, in general, badly on symmetric problems. Thus, the latter need a special formulation and solution treatment.

The figure below shows an example graph used for the travelling salesman problem

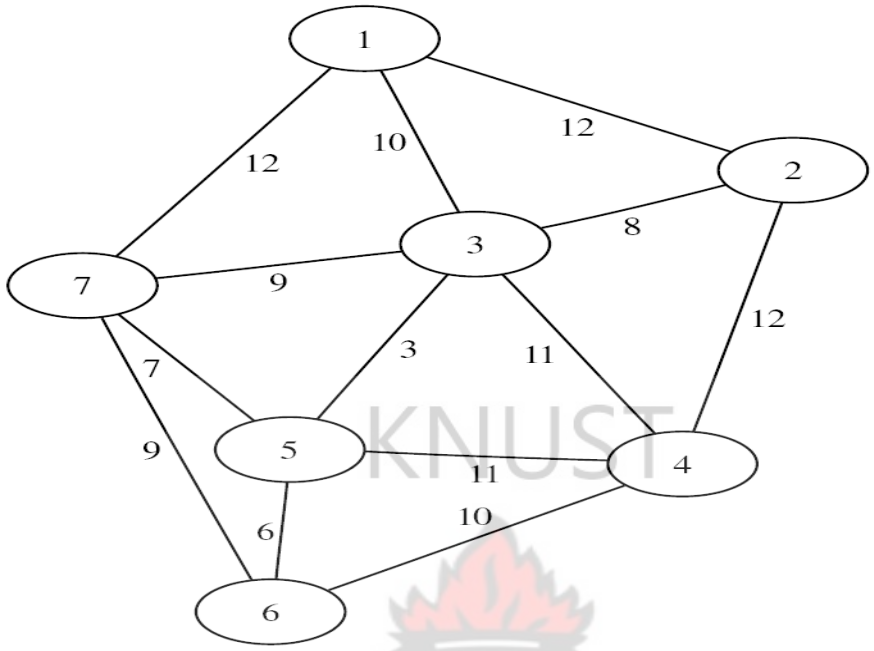

Figure 3.1 Travelling Salesman Problem

We will use the travelling salesmen problem (TSP) on the graph as an example problem for the metaheurisics discussed. Travelling Salesman Problem (TSP):

A salesman spends his time visiting n cities (or nodes) cyclically. Starting from the home city, the salesman wishes to determine which route to follow to visit each city exactly once before returning to the home city so as to minimize the total distance of the tour.

The difficulty of the travelling salesman problem increases rapidly as the number of cities increases. For a problem with n cities and a link between every pair of cities, the number of feasible routes to be considered is (n-1)!/2. Due to enormous difficulty in solving the TSP, heuristic methods guided by the meta-heuristics, address such

problems. Heuristic methods involve sequence of feasible trial solutions, where each solution is obtained by making certain adjustment in current trial solution.

3.4 METHODS FOR SOLVING THE TSP

3.4.1 SUBTOUR REVERSAL

Adjusting a sequence of cities visited in the current solution by selecting a subsequence of the cities and simply reversing the order.

Eg. Initial trial solution is the following sequence of cities visted: 1-2-3-4-5-6-7-1 with total distance = 69.

While reversing the sequence 3-4, we obtain new trial solution: 1-2-4-3-5-6-7-1 with total distance = 65.

Neighbors: We say 2 tours/solutions/cycles are neighbors if we can transform one to the other by a subtour reversal.

Degree of Neighbor: The degree of a neighbor A to B equals the minimum number of subtour reversals required to get from A to B.

Local Minimum: A local minimum is when no neighbors are better i.e no neighbour's subtour gives a better solution. Problems like TSP have many local minima's. If we look into the gradient search approach to solve TSP, the steps are:

- Pick a cycle
- Take best neighbor
- Repeat until local minima is obtained.

Because of local mininma, this may not yield good solution.

3.4.2 TABU SEARCH

The key process is finding the local optima and then continue the search by allowing non improving moves (or illegal moves, called as tabu moves) to the best solutions in the neighborhood of the local optima. One of the features of tabu search is to avoid bad solutions which have already been explored i.e use of memory to guide the search by using tabu list to record recent searches. Essentially, Tabu search makes some moves illegal by maintaining a list of 'tabu' moves.

For example, if A is a neighbor of B in the TSP then B is a neighbor of A. But if you have already chosen B over A, there might not be any reason to search A again.

Some Useful Definitions

- Intensify: To intensify the search is to search the local area (portion of feasible region) more thoroughly.
- Diversify: To diversify the search is to force the search away from the current solution (to unexplored areas of feasible region).
- Length of Tabu List: The length of the list signifies the balance between intensify/diversify.

The Tabu Search Algorithm

- Initialize
- Iteration:

- Compare all possible moves
- Take best (even if it is worse than the current solution)
- Update List
- Stop after a fixed time or CPU usage, or there are no feasible moves.

The optimal solution is the best solution so far.

Travelling Salesman Example

Initial trial solution: 1-2-3-4-5-6-7-1 Distance = 69

Tabu list: Null

Iteration 1: Reverse 3-4

Deleted links: 2-3 and 4-5

Added links: 2-4 and 3-5

Tabu List: (2-4), (3-5)

New trials Solution: 1-2-4-3-5-6-7-1 Distance = 65

Iteration 2: Reverse: 3-5-6

Delete links: 4-3 and 6-7

Add links: 4-6 and 3-7

Tabu List: 2-4, 3-5, 4-6, 3-7

New Solution: 1-2-4-6-5-3-7-1 Distance = 64

Keep running for more iteration until algorithm terminates at a point where we obtain the best trial solution as the final solution. While running Tabu search it is necessary to keep track of the Tabu list and the best solution so far. So, delete few links from the tabu list after some number of iterations and while doing this, delete the oldest links of tabu list.

Advantage

This method is easy to integrate with other methods.

Disadvantage

Tabu search approach is to climb the hill in the steepest direction and stop at top and then climb downwards to search for another hill to climb. The drawback is that a lot of iterations are spent climbing each hill rather than searching for tallest hill.

3.4.3 GENETIC ALGORITHMS

Genetic Algorithms are a class of optimization algorithms based on survival of the fittest" and combining solutions (parents and children). Here feasible solutions to particular problems correspond to members of species. Fitness of each member is measured by the objective function. The basic idea is that each possible solution is a member of entire population of trial solutions, and any given population is keeping track of multiple solutions.

Parents in a genetic algorithm are selected at random from the available population, and the new trial solutions (children) are created from the parents. When these children are added to the population they occasionally have mutations which add more variety to the population.

Important property:

When going through a genetic algorithm, a good solution is more likely to survive and hence more likely to reproduce.

A framework for a Genetic Algorithm

1. Choose the initial population of individuals
2. Evaluate the fitness of each individual in that population
3. Repeat on this generation until termination: (time limit, sufficient fitness achieved, etc.)

(a) Select the best-fit individuals for reproduction

(b) Breed new individuals through crossover and mutation operations to give birth to offspring

(c) Evaluate the individual fitness of new individuals

(d) Replace least-fit population with new individuals

- Initialize the population of solution
- Randomly Select Parents to combine
- Generate Child based on parents
    - keep the child (add to population) if good or adds improvements or
    - reject if infeasible or bad{ mutate randomly
    - Possibly throw out all solutions if necessary.

Many decisions affect the effectiveness of genetic algorithm for any particular problem:

- Population Size

The population size indicates how much of the search space the GA will search in each iteration. Smaller size could mean the algorithm takes smaller time to find the optimal solution. Similarly when the size is large the algorithm take a longer time in sampling the large number of chromosomes in order to obtain the best chromosome.

Selection Rule

The general selection process involves reproduction, crossover and mutation operations. The selection process is used to generate a new population from the current one. The objective is to select individuals from the high fitness range . It is used for selecting individuals for crossover and mutation.

Combination rule

Two chromosomes are chosen at random. The one with the higher fitness is then selected.

The process is repeated until the required numbers of chromosomes are obtained.

Mutation Rule

Mutation operation is performed on individual chromosome whereby the alleles are changed probabilistically.

Stopping Rule

The algorithm terminates when a set of conditions are satisfied. At that point the best solution is taken as the global solution or the algorithm may terminate if one or more of the following are satisfied;

i. A specified number of total iteration is completed.

ii. A specified number of iteration is completed within which the solution of best fitness has not changed.

iii. A standard deviation of the generation of the population approaches a given value.

iv. The average fitness of the generation of the population does not differ significantly from the solution of best fitness.

A genetic algorithm is often good for solving hard optimization problems which can easily be represented in binary.

An example solution for the TSP using a genetic algorithm:

Some characteristics of the TSP when represented to be used with a genetic algorithm need a rule for joining parents (or 2 subtours). Parents are tours and the current city is the home city of a path.

Algorithm

- Identify all links from current in either parent that are not already in the tour.

- Randomly select one of the available.
- Check for mutation.
- The next city is one these
- Use this link to complete the tour.

Example

1-2-4-6-5-3-7-1

1-7-6-4-5-3-2-1

Generate a child:

1-2-4-5-6-7-3-1

Parents are selected using a fitness function, if we were given the choice amongst the following:

1. $f(x_1) = 69$
2. $f(x_2) = 65$
3. $f(x_3) = 79$
4. $f(x_4) = 86$

We would choose numbers 1 and 2 as they are the lowest distance. Alternatively for some problems we would choose to select those which have the highest fitness.

Have mutations to randomly allow other links as these help genetic algorithms to explore a new/better feasible solution. Essence is that parents generate children (new trial solutions) who share some features of both parents. Since the fittest members of the population are more likely to become parents than others, a genetic algorithm tends to improving populations of trial solutions as it proceeds.

3.4.4 HARMONY SEARCH

In computer science and operations research, Harmony Search (HS) is a phenomenon-mimicking algorithm (also known as metaheuristic algorithm, soft computing algorithm or evolutionary algorithm) inspired by the improvisation process of musicians proposed by Zong Woo Geem in 2001. In the HS algorithm, each musician (= decision variable) plays (= generates) a note (= a value) for finding a best harmony (= global optimum) all together. Proponents claim the following merits

- HS does not require differential gradients, thus it can consider discontinuous functions as well as continuous functions.
- HS can handle discrete variables as well as continuous variables.
- HS does not require initial value setting for the variables.
- HS is free from divergence.
- HS may escape local optima.
- HS may overcome the drawback of GA's building block theory which works well only if the relationship among variables in a chromosome is carefully considered. If neighbor variables in a chromosome have weaker relationship than remote variables, building block theory may not work well because of crossover operation. However, HS explicitly considers the relationship using ensemble operation.
- HS has a novel stochastic derivative applied to discrete variables, which uses musician's experiences as a searching direction.
- Certain HS variants do not require algorithm parameters such as HMCR and PAR, thus novice users can easily use the algorithm.

Basic Harmony Search Algorithm

Harmony search tries to find a vector $\mathbf{x}$which optimizes (minimizes or maximizes) a certain objective function.

The algorithm has the following steps:

Step 1: Generate random vectors ($\mathbf{x}^1, \ldots, \mathbf{x}^{hms}$) as many as hms(harmony memory size), then store them in harmony memory (HM).

$$\mathbf{HM} = \begin{bmatrix} x_1^1 & \cdots & x_n^1 & | & f(\mathbf{x}^1) \\ \vdots & \ddots & \vdots & | & \vdots \\ x_1^{hms} & \cdots & x_n^{hms} & | & f(\mathbf{x}^{hms}) \end{bmatrix}.$$

Step 2: Generate a new vector $\mathbf{x}'$. For each component x_i',

- with probability $hmcr$(harmony memory considering rate; 0 ≤ $hmcr$≤ 1), pick the stored value from HM: $x_i' \leftarrow x_i^{int(u(0,1)*hms)+1}$
- with probability $1 - hmcr$, pick a random value within the allowed range.

Step 3: Perform additional work if the value in Step 2 came from HM.

- with probability par(pitch adjusting rate; 0 ≤ par≤ 1), change x_i' by a small amount: $x_i' \leftarrow x_i' + \delta$or $x_i' \leftarrow x_i' - \delta$for discrete variable; or $x_i' \leftarrow x_i' + fw \cdot u(-1, 1)$for continuous variable.
- with probability $1 - par$, do nothing.

Step 4: If $\mathbf{x}'$ is better than the worst vector $\mathbf{x}^{Worst}$ in HM, replace $\mathbf{x}^{Worst}$ with $\mathbf{x}'$.

Step 5: Repeat from Step 2 to Step 4 until termination criterion (e.g. maximum iterations) is satisfied.

The parameters of the algorithm are

- hms= the size of the harmony memory. It generally varies from 1 to 100. (typical value = 30)
- $hmcr$= the rate of choosing a value from the harmony memory. It generally varies from 0.7 to 0.99. (typical value = 0.9)
- par= the rate of choosing a neighboring value. It generally varies from 0.1 to 0.5. (typical value = 0.3)
- δ= the amount between two neighboring values in discrete candidate set.
- fw(fret width, formerly bandwidth) = the amount of maximum change in pitch adjustment. This can be (0.01 × allowed range) to (0.001 × allowed range).

It is possible to vary the parameter values as the search progresses, which gives an effect similar to simulated annealing.

Parameter-setting-free researches have been also performed. In the researches, algorithm users do not need tedious parameter setting process.

3.4.5 SIMULATED ANNEALING (SA)

The process of simulated annealing is inspired by the physical process of annealing from chemistry. Annealing involves slowly heating the metal and then slowly cooling the substance by varying the temperatures until it reaches the low energy stable state, resulting in a global reduction in energy, but locally it may result in an increase in energy. These changes in energy follow a Boltzmann distribution.

The key idea in simulated annealing algorithm is to select an appropriate temperature schedule which needs to specify the initial, relatively large value of T and then decrease the value of T. Starting with relatively large values of T makes probability of acceptance relatively large, which enables the search to proceed in almost all random directions. Gradually decreasing the value of T as the search proceeds gradually decreases the probability of acceptance, which emphasizes on climbing upwards.

In 1983 Kirk Patrick showed how Simulated Annealing of Metropolis could be adapted to solve problems in Combinatorial Optimization.

The following analogy was made

1. a) Annealing looks for system state at a given temperature.

   b) Optimization looks for feasible solution of the combinatorial problems

2. a) Cooling of the metal is to move from one system state to another

   b) Search procedure (algorithm scheme) tries one solution after another in order to find the optimal solution.

3. a) Energy function is used to determine the system state and energy

   b) Objective (cost) function is used to determine a solution and the objective function value.

4. a) Energy results in evaluation of energy function and the lowest energy state corresponds to stable state.

   b) Cost results in evaluation of objective function and the lowest objective function value corresponds to the optimal solution

5. a) Temperature controls the system state and the energy

   b) A control parameter is used to control the solution generation and the objective function value

Simulated annealing (SA) is a generic probabilistic metaheuristic for the global optimization problem of applied mathematics, namely locating a good approximation to the global minimum of a given function in a large search space. It is often used when the search space is discrete (e.g., all tours that visit a given set of cities). For certain problems, simulated annealing may be more effective than exhaustive enumeration — provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution.

The basic iteration

At each step, the SA heuristic considers some neighbouring state s' of the current state s, and probabilistically decides between moving the system to state s' or staying in state s. These probabilities ultimately lead the system to move to states of lower energy. Typically this step is repeated until the system reaches a state that is good enough for the application, or until a given computation budget has been exhausted.

The neighbours of a state

The neighbours of a state are new states of the problem that are produced after altering a given state in some well-defined way. For example, in the traveling salesman problem

each state is typically defined as a permutation of the cities to be visited. The neighbours of a state are the set of permutations that are produced, for example, by reversing the order of any two successive cities. The well-defined way in which the states are altered in order to find neighbouring states is called a "move" and different moves give different sets of neighbouring states. These moves usually result in minimal alterations of the last state, as the previous example depicts, in order to help the algorithm keep the better parts of the solution and change only the worse parts. In the traveling salesman problem, the parts of the solution are the city connections.

Searching for neighbours of a state is fundamental to optimization because the final solution will come after a tour of successive neighbours. Simple heuristics move by finding best neighbour after best neighbour and stop when they have reached a solution which has no neighbours that are better solutions. The problem with this approach is that the neighbours of a state are not guaranteed to contain any of the existing better solutions which mean that failure to find a better solution among them does not guarantee that no better solution exists. This is why the best solution found by such algorithms is called a local optimum in contrast with the actual best solution which is called a global optimum. Metaheuristics use the neighbours of a solution as a way to explore the solutions space and although they prefer better neighbours they also accept worse neighbours in order to avoid getting stuck in local optima. As a result, if the algorithm is run for an infinite amount of time, the global optimum will be found.

Acceptance probabilities

The probability of making the transition from the current state s to a candidate new state s' is specified by an *acceptance probability function* $P(e, e', T)$, that depends on the energies $e = E(s)$ and $e' = E(s')$ of the two states, and on a global time-varying parameter T called the *temperature*. States with a smaller energy are better than those with a greater energy. The probability function P must be positive even when e' is greater than e. This feature prevents the method from becoming stuck at a local minimum that is worse than the global one.

When T tends to zero, the probability $P(e, e', T)$ must tend to zero if $e' > e$ and to a positive value otherwise. For sufficiently small values of T, the system will then increasingly favor moves that go "downhill" (i.e., to lower energy values), and avoid those that go "uphill." With $T = 0$ the procedure reduces to the greedy algorithm, which makes only the downhill transitions.

In the original description of SA, the probability $P(e, e', T)$ was equal to 1 when $e' < e$ — i.e., the procedure always moved downhill when it found a way to do so, irrespective of the temperature. Many descriptions and implementations of SA still take this condition as part of the method's definition. However, this condition is not essential for the method to work.

The P function is usually chosen so that the probability of accepting a move decreases when the difference $e' - e$ increases—that is, small uphill moves are more likely than

large ones. However, this requirement is not strictly necessary, provided that the above requirements are met.

Given these properties, the temperature T plays a crucial role in controlling the evolution of the state s of the system vis-a-vis its sensitivity to the variations of system energies. To be precise, for a large T, the evolution of s is sensitive to coarser energy variations, while it is sensitive to finer energy variations when T is small

General schema for Simulated Annealing Algorithm.

a. Generate a starting solution *S* and set the initial solution *S* * = *S*.
b. Determine a starting temperature *T*.
c. While not yet at *equilibrium* for this temperature, do the following:
d. Choose a *random neighbour S** of the current solution.
e. Set Δ = *Length*(*S**) = *Length*(*S*).
f. If ≤ 0 (downhill move):

   Set *S* = *S**.

   If *Length*(*S*) < *Length*(*S* *), set *S* * = *S*.
g. If length(S) < length(S*) (uphill move):

   Choose a random number *r* uniformly from [0, 1].

   If $r < e^{-\Delta/T}$, set *S* = *S**.
h. End "While not yet at equilibrium" loop.
i. Lower the temperature *T*.
j. End "While not yet frozen" loop.
k. Return *S* *.

Travelling Salesman Example

Considering Figure 3.1

Taking the initial solution to be in the tour in the order: 1-2-3-4-5-6-7-1 using the parameters;

$T_o = 20 \quad T_{k+1} = \alpha T_k \quad \alpha = 0.5$

Stop when $T < 0.1$

First Iteration

Assuming x^o =1-2-3-4-5-6-7-1

$d(x^o) = d(1,2)+d(2,3)+d(3,4)+d(4,5)+d(5,6)+d(6,7)+d(7,1) = 69$

Using the sub-tour reversal as local search to generate the new solution x^1 =1-3-2-4-5- 6-7-1

$d(x^1) = d(1,3)+d(3,2)+d(2,4)+d(4,5)+d(5,6)+d(6,7)+d(7,1)=68$

$\delta = d(x^1) - d(x^o) = 68-69 = -1$

Since $\delta < 0$, set $x^o \leftarrow x^1$

Updating the temperature $T_1 = \alpha T_o = 0.5(20) =10$

Second Iteration

$d(x^o) = 68$

By the sub-tour reversal as local search to generate the new solution 1-2-3-5-4-6-7-1

x^1 =1-2-3-5-4-6-7-1

$d(x^1)=d(1,2)+d(2,3)+d(3,5)+d(5,4)+d(4,6)+d(6,7)+d(7,1)=65$

$\delta= d(x^1) - d(x^o) =65-68 = -3$

Since $\delta < 0$, set $x^o \leftarrow x^1$

Updating the temperature, $T_2 = 0.5(10) = 5$

Third Iteration

$d(x^o)=65$

Using the sub-tour reversal as local search to generate the new solution 1-2-3-4-6-5-7-1

x^1 =1-2-3-4-6-5-7-1

$d(x^1)=d(1,2)+d(2,3)+d(3,4)+d(4,6)+d(6,5)+d(5,7)+d(7,1)=66$

$\delta= d(x) - d(x) = 66-65 =1$

Since $\delta>0$, then apply Boltzmann's condition $m = e^{-\delta/T2} = 0.81$

A random number would be generated from a computer say θ

If $m>\theta$ then set $x^o \leftarrow x^1$ otherwise $x^1 \leftarrow x^o$

Updating the temperature, $T_3 = 0.5(5) = 2.5$

This process will continue until the final temperature and the optimal solution are obtained.

3.5 APPLICATION OF TSP AND LINKAGES WITH OTHER PROBLEMS

Drilling of printed circuit boards

A direct application of the TSP is in the drilling problem of printed circuit boards (PCBs) (Grötschel et al., 1991). To connect a conductor on one layer with a conductor on another layer, or to position the pins of integrated circuits, holes have to be drilled through the board. The holes may be of different sizes. To drill two holes of different diameters consecutively, the head of the machine has to move to a tool box and change the drilling equipment. This is quite time consuming. Thus it is clear that one has to choose some diameter, drill all holes of the same diameter, change the drill, drill the holes of the next diameter, etc. Thus, this drilling problem can be viewed as a series of TSPs, one for each hole diameter, where the 'cities' are the initial position and the set of all holes that can be drilled with one and the same drill. The 'distance' between two cities is given by the time it takes to move the drilling head from one position to the other. The aim is to minimize the travel time for the machine head.

Overhauling gas turbine engines

An application found by Gerard (1994) is overhauling gas turbine engines in aircraft. Nozzle-guide vane assemblies, consisting of nozzle guide vanes fixed to the circumference, are located at each turbine stage to ensure uniform gas flow. The placement of the vanes in order to minimize fuel consumption can be modeled as a symmetric TSP.

X-Ray crystallography

Analysis of the structure of crystals (Dreissig & Uebach, 1990) is an important application of the TSP. Here an X-ray diffractometer is used to obtain information about the structure of crystalline material. To this end a detector measures the intensity of Xray reflections of the crystal from various positions. Whereas the measurement itself can be accomplished quite fast, there is a considerable overhead in positioning time since up to hundreds of thousands positions have to be realized for some experiments. In the two examples that we refer to, the positioning involves moving four motors. The time needed to move from one position to the other can be computed very accurately. The result of the experiment does not depend on the sequence in which the measurements at the various positions are taken. However, the total time needed for the experiment depends on the sequence. Therefore, the problem consists of finding a sequence that minimizes the total positioning time. This leads to a traveling salesman problem.

Computer wiring

(Lenstra & Rinnooy Kan, 1974) reported a special case of connecting components on a computer board. Modules are located on a computer board and a given subset of pins has to be connected. In contrast to the usual case where a Steiner tree connection is desired, here the requirement is that no more than two wires are attached to each pin. Hence we have the problem of finding a shortest Hamiltonian path with unspecified starting and terminating points. A similar situation occurs for the so-called testbus wiring. To test the manufactured board one has to realize a connection which enters the board at some specified point, runs through all the modules, and terminates at some specified point. For

each module we also have a specified entering and leaving point for this test wiring. This problem also amounts

to solving a Hamiltonian path problem with the difference that the distances are not symmetric and that starting and terminating point are specified.

The order-picking problem in warehouses

This problem is associated with material handling in a warehouse (Ratliff & Rosenthal, 1983). Assume that at a warehouse an order arrives for a certain subset of the items stored in the warehouse. Some vehicle has to collect all items of this order to ship them to the customer. The relation to the TSP is immediately seen. The storage locations of the items correspond to the nodes of the graph. The distance between two nodes is given by the time needed to move the vehicle from one location to the other. The problem of finding a shortest route for the vehicle with minimum pickup time can now be solved as a TSP..

Vehicle routing

Suppose that in a city *n* mail boxes have to be emptied every day within a certain period of time, say 1 hour. The problem is to find the minimum number of trucks to do this and the shortest time to do the collections using this number of trucks. As another example, suppose that n customers require certain amounts of some commodities and a supplier has to satisfy all demands with a fleet of trucks. The problem is to find an assignment of customers to the trucks and a delivery schedule for each truck so that the capacity of each truck is not exceeded and the total travel distance is minimized. Several variations of these two problems, where time and capacity constraints are combined, are common

in many realworld applications. This problem is solvable as a TSP if there are no time and capacityconstraints and if the number of trucks is fixed (say m). In this case we obtain an m -salesmen problem. Nevertheless, one may apply methods for the TSP to find good feasible solutions for this problem (see Lenstra & Rinnooy Kan, 1974).

Mask plotting in PCB production

For the production of each layer of a printed circuit board, as well as for layers of integrated semiconductor devices, a photographic mask has to be produced. In our case for printed circuit boards this is done by a mechanical plotting device. The plotter moves a lens over a photosensitive coated glass plate. The shutter may be opened or closed to expose specific parts of the plate. There are different apertures available to be able to generate different structures on the board. Two types of structures have to be considered. A line is exposed on the plate by moving the closed shutter to one endpoint of the line, then opening the shutter and moving it to the other endpoint of the line. Then the shutter is closed. A point type structure is generated by moving (with the appropriate aperture) to the position of that point then opening the shutter just to make a short flash, and then closing it again. Exact modeling of the plotter control problem leads to a problem more complicated than the TSP and also more complicated than the rural postman problem. A real-world application in the actual production environment is reported in (Grötschel et al., 1991).

CHAPTER 4

COLLECTION OF DATA, ANALYSIS OF DATA AND RESULTS

4.1 INTRODUCTION

This chapter deals with the collection and gathering of data, analysis and evaluation of the data and interpretation of the results.

4.2 DISTANCE MATRIX FOR THE 10 REGIONAL CAPITALS IN GHANA IN KILOMETERS (KM).

Table 4.1 is the distance matrix table, taken from Transport Department of WAEC and it shows the various links of connecting question paper depots and examination centres for officer assigned to all the ten (10) regions of Ghana in kilometers (km).

For cities which have no direct link the minimum distance along the edges is considered. The cells indicated zero shows that there is no distance.

C_{ij} = the distance from city i to city j

$C_{ii} = C_{jj} = 0$ = There is no distance

C_{ij}	Accra	K'si	K'dua	T'di	S'nyani	T'male	C.Coast	Wa	Ho	Bolga
ACCRA	0	270	85	218	400	658	144	740	165	810
KUMASI	270	0	194	242	130	388	221	470	162	558
KOFORIDUA	85	194	0	303	324	582	229	664	162	752
TAKORADI	218	242	303	0	372	683	74	765	362	853
SUNYANI	400	130	324	372	0	300	351	378	486	470
TAMALE	658	388	582	683	300	0	609	314	476	170
CAPE COAST	144	211	229	74	351	609	0	691	309	779
WA	740	470	664	765	378	314	691	0	790	368
HO	165	356	162	362	486	476	309	790	0	914
BOLGATANGA	810	558	752	853	470	170	779	360	914	0

Table 4.1 Distance matrix for the 10 regional capitals in Ghana in kilometers (Km)

For the purpose of this work, numbers have been allocated to the ten major city depots in Ghana. This is illustrated in the table below.

REGIONAL DEPOT	NUMBER ALLOCATED
Accra	1
Kumasi	2
Koforidua	3
Takoradi	4
Sunyani	5
Tamale	6
Cape Coast	7
Wa	8
Ho	9
Bolgatanga	10

Table 4.2: numbers allocated to various regional capitals.

4.2 FORMULATION OF THE TSP MODEL FOR THE WAEC DEPOT INSPECTION PROBLEM.

The problem can be defined as follows: Let *G = (V,E)* be a complete undirected graph with vertices *V*, *|V|=n*, where n is the number of cities, and edges *E* with edge length *dij* for *(i,j)*.

We focus on the symmetric TSP case in which $C_{ij} = C_{ji}$, for all *(i,j)*.

We formulate this minimization problem as an integer programming, as shown in Equations (1) to (5).

$$P1: \min \sum_{i \in v} \sum_{j \in v} Cij\ Xij \qquad (1)$$

$$\sum_{\substack{j \in v \\ j \neq i}} Xij = 1 \qquad j \in v \qquad (2)$$

$$\sum_{\substack{i \in v \\ i \neq j}} Xij = 1 \qquad j \in v \qquad (3)$$

$$\sum_{i \in s} \sum_{j \in s} Xij \leq \lfloor s \rfloor - 1 \ \forall s \subset v, s \neq \phi\ Xij = 0 or 1 \quad i,j \in v \qquad (4)$$

$$Xij = 0\ or\ 1 \qquad i,j \in v \qquad (5)$$

The problem is an assignment problem with additional restrictions that guarantee the exclusion of subtours in the optimal solution. Recall that a subtour in *V* is a cycle that does not include all vertices (or cities). Equation (1) is the objective function, which minimizes the total distance to be travelled.

Constraints (2) and (3) define a regular assignment problem, where (2) ensures that each city is entered from only one other city, while (3) ensures that each city is only departed

to on other city. Constraint (4) eliminates subtours. Constraint (5) is a binary constraint, where $X_{ij} = 1$ if edge *(i,j)* in the solution and $X_{ij} = 0$, otherwise.

4.3 ANALYSIS

To satisfy constraints (2) and (3) we choose the random

Initial tour $(x^0) = 3 – 9 – 2 – 5 – 6 – 10 – 8 – 7 – 4 – 1$

From the objective function (1) the initial distance = $d(x^0) = d(3,9) + d(9,2) + d(2,5) + d(5,6) + d(6,10) + d(10,8) + d(8,7) + d(7,4) + d(4,1) = 2360$ Km

The initial temperature is taken to be $(T_o) = 3540.00$ and $\alpha = 0.99$

Temperature is updated by using the formula $T_{k+1} = \alpha T_k$ where k is the number of iterations. Stop when $T \leq 51.97$.

Simulated annealing algorithm was used to find the final solution. An HP dual core computer with processor 2.66GHz was used in ascertaining the final solution after 841 iterations in 182.72 seconds. The execution time varied with the number of iterations.

4.4 RESULTS

After performing 841 iterations, the optimal tour = 3 – 9 – 6 – 10 – 8 – 5 – 2 – 4 – 7 – 1

Thus, d(3,9) + d(9,6) + d(6,10) + d(10,8) + d(8,5) + d(5,2) + d(2,4) + d(4,7) + d(7,1) = 2229 Km.

The optimal tour was found to be the same after it was run ten times.

The optimal tour is therefore as follows:

Koforidua → Ho → Tamale → Bolgatanga → Wa → Sunyani → Kumasi → Takoradi → Cape Coast → Accra

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION

The travelling salesman problem is a traditional problem that has to do with making the most efficient use of resources while at the same time spending the least amount of energy in that utilization. The designation for this type of problem hails back to the days of the travelling salesman, who often wished to arrange travel distances in a manner that allowed for visiting the most towns without having to double back and cross into any given town more than once.

In a wider sense, the travelling salesman problem is considered to be a classic example of what is known as a tour problem. Essentially, any type of tour problem involves making a series of stops along a designated route and making a return journey without ever making a second visit to any previous stop. Generally, a tour problem is present when there is concern on making the most of available resources such as time and mode of travel to accomplish the most in results. Finding a solution to a tour problem is sometimes referred to as discovering the least-cost path, implying that the strategic planning of the route will ensure maximum benefit with minimum expenditure incurred.

TSP is a very attractive problem for the research community because it arises as a natural sub-problem in many applications concerning everyday life. Indeed, each application, in which an optimal ordering of a number of items has to be chosen in a way that the total cost of a solution is determined by adding up the costs arising from two successive items, can be modelled as a TSP instance. Thus, studying TSP can never be considered as an abstract research with no real importance.

The study shows clearly that, any WAEC officer who has been assigned to inspect the question papers depots within the ten (10) regions of Ghana must consider the tour order below.

The order is as follows:

Koforidua → Ho → Tamale → Bolgatanga → Wa → Sunyani → Kumasi → Takoradi → Cape Coast → Accra

5.2 RECOMMENDATION

The use of mathematical models has proved to be efficient in the computation of optimum results and gives a systematic and transparent solution as compared with an arbitrary method.

Management will benefit from the proposed approach for officers who would be assigned to inspect various examination centres in order to visit more centres on a route at a minimized travel distance. We therefore recommend that our TSP model should be adopted by WAEC for its depot inspection planning.

Students can use this work as reference for further research covering all the regions in Ghana.

REFERENCES

1. Applegate D, Bixby R.E., Chvatal V., and Cook W. (1994) "Finding cuts in the TSP" a preliminary report distributed at The Mathematical Programming Symposium, Ann Arbor, Michigan.
2. Applegate, D., Bixby, R., Chvatal, V., Cook, W., and Helsghaun, K. (2004). ―The Sweden Cities TSP Solution‖. http://www.tsp.gatech.edu//sweeden/cities/cities.htm.
3. Applegate D, Bixby R., Chvatal V and Cook W. (1998). On the Solution of the Traveling Salesman Problems. Documenta Mathematica – Extra Volume ICM, chapter 3, pp. 645-656
4. Applegate, D., Bixby, R., Chvatal, V., and Cook, W. (1994): Finding Cuts in the TSP (A preliminary report), Tech. rep., Mathematics, AT&T Bell Laboratories, Murray Hill, NJ.
5. Amponsah, S .K and F.K Darkwah (2007) .Lecture notes on operation Research ,IDL KNUST 62-67
6. Angel R D, Caudle W L, Noonan R and Whinston A (1972). Computer assisted school bus scheduling. Management Science, Vol. 18, pp.279–88.
7. Balas, E. and Simonetti, N. (2001). ―Linear Time Dynamic Programming Algorithms for New Classes of Restricted TSPs: A Computational Study.‖ *INFORMS Journal on Computing*. 13(1): 56-75.
8. Bellmore, M. and Nemhauser, G.L. (1968). "The Traveling Salesman Problem: A Survey". *Operations Research*. 16:538-558.
9. Bernd F. and Peter M., (1996).New Genetic Local Search Operators for the Travelling Salesman Problem.
10. Bock, F. (1958). "An Algorithm for Solving Traveling-Salesman' and Related Network Optimization Problems". Research Report, Operations Research Society of America Fourteenth National Meeting: St. Louis, MO. Problems". Research Report, Operations Research Society of America Fourteenth National Meeting: St. Louis, MO.
11. Burkard, R.E. (1979). "Traveling Salesman and Assignment Problems: A Survey". In: *Discrete Optimization 1*. P.L. Hammer, E.L. Johnson, and B.H.

Korte, eds. *Annals of Discrete Mathematics Vol. 4*, North-Holland: Amsterdam. 193-215.

12. Carpaneto, G., Toth, P., (1980). "Some new branching and bounding criteria for the asymmetric traveling salesman problem", Management Science 21, pp.736–743.
13. Carpaneto, G., Dell'Amico, M. and Toth, P., (1995), ―Exact Solution of Large-scale Asymmetric Traveling Salesman Problems‖, ACM Transactions on Mathematical Software, 21, pp.394–409.
14. Croes, G.A. 1958. "A Method for Solving Travelling-Salesman Problems". Operations Research. 6:791-812.
15. Crowder, H. and Padberg, M.W. (1980). "Solving Large-Scale Symmetric Travelling Salesman Problems to Optimality". Management Science. 26:495-509.
16. Dantzig, G.B., Fulkerson, D.R., and Johnson, S.M. (1954). "Solution of a Large-Scale Traveling-Salesman Problem". Operations Research. 2: 393-410.
17. Dorigo, M. and Gambardella, L.M. (1996). Ant Colonies for the Travelling Salesman Problem. University Libre de Bruxelles, Belgium.
18. Dreissig W. and Uebaeh W., (1990). Personal communication. Linear programming with pattern constraints. PhD thesis, Department of Economics, Harvard University, Cambridge, MA
19. Davoian K. and Gorlatch S., (2005). Modified Genetic Algorithm for the Travelling Salesman Problem and Its Parallelization. Proceeding (453) Artificial Intelligence and Applications
20. Eastman, W.L. (1958). "Linear Programming with Pattern Constraints". Ph.D. Dissertation. Harvard University: Cambridge, MA.
21. Gomory, R.E. (1996). "The Traveling Salesman Problem". In: Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems. IBM: White Plains, NY. 93-121.
22. Geem, Z. W., Kim, J. H., Loganathan, G. V., (2001). A New Heuristic Optimization Algorithm: Harmony Search
23. Grötschel .M and Padberg M., (1993). "Ulysses 2000: In Search of Optimal

Solutions to Hard Combinatorial Problems," Technical Report, New York University Stern School of Business.

24. Grötschel, M.,and Holland, (1991) “Solution of large-scale travelling salesman problems”, Mathematical Programming 51(2), 141–202.
25. Gerard R., (1994). The Traveling Salesman: Computational Solutions for TSP Applications. Springer-Verlag.
26. Gilbert C. K. and Hofstra R. B., (2007). A New Multiperiod Multiple Travelling Salesman Problem with Heuristic and Application to a Scheduling Problem. Journal of the institute for operations research and management science.
27. Golden B.L., Wasil E.A., Kelly J.P, and. Chao I-M, (1998). Metaheuristics in Vehicle Routing. In Fleet Management and Logictics, T.G. Crainic and G. Laporte (eds), Kluwer, Boston, 33-56.
28. Hoffman A. J. and Wolfe P. (1985), "History" in *The Traveling Salesman Problem*, Lawler, Lenstra, Rinooy Kan and Shmoys, eds., Wiley, 1-16 Holland, J, (1975).Adaptation *in Natural and Artificial Systems*, Michigan University Press
29. Helbig, H.K. and Krarup, J. (1974). "Improvements of the Held-Karp Algorithm for the Symmetric Traveling-Salesman Problem". *Mathematical Programming*. 7:87-96.
30. Held, M. and Karp, R.M. (1970). "The Traveling-Salesman Problem and Minimum Spanning Trees". Operations Research. 18:1138-1162.
31. Held, M. and Karp, R.M. (1971). "The Traveling-Salesman Problem and Minimum Spanning Trees: Part II". Mathematical Programming. 1:6-25.
32. Hong, S. (1972). "A Linear Programming Approach for the Traveling Salesman Problem". Ph.D. Thesis. The Johns Hopkins University: Baltimore, MD.
33. Johnson, D.S., Gutin, G. McGeoch, L.A., Yeo, A., Zhang, W., and Zverovitch, A. (2002). “Experimental Analysis of Heuristics for the Asymmetric Traveling Salesman Problem”. In: Gutin G and Punnen H (eds). The Traveling Salesman Problem and it Variations. Kluwer Academic Publishers.
34. Karg, R.L. and. Thompson, G.L. (1964). "A Heuristic Approach to Solving Travelling Salesman Problems". Management Science. 10:225-248.

35. Kirkpatrick, **S.,** Gelatt, C.D. Jr., and Vecchi, M.P. (1983). Optimizations by simulated annealing. Science, 220, 671-681.
36. Lawler, E.J., Lenstra, J.K., Rinnoy Kan, A.H.G., and Shmoys, D.B. (1985). ―The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization‖. John Wiley & Sons: New York, NY.
37. Morton, G. and Land, A.H. (1955). "A Contribution to the Travelling-Salesman' Problem". *Journal of the Royal Statistical Society, Series B*. 17:185-194.
38. Padberg, M.W. and Rinaldi, G. (1987). "Optimization of a 532-city Symmetric Traveling Salesman Problem by Branch and Cut". *Operations Research Letters*. 6:17.
39. Ratliff H. D. and Rosenthal A. S., (1983). Order-Picking in a Rectangular Warehouse: A Solvable Case for the Travelling Salesman Problem. Operations Research, Vol. 31, pp. 507-521.
40. Svestka J. A. and Huckfeldt V. E., (1973). Computational experience with an m-salesman travelling salesman algorithm. Management Science, Vol. 19, No. 7, pp. 790–9.
41. Smith, T.H.C. and Thompson, G.L. (1977). "A LIFO Implicit Enumeration Search Algorithm for the Symmetric Traveling Salesman Problem using Held and Karp's 1-Tree Relaxation". In: *Studies in Integer Programming*. P.L. Hammer, E.L. Johnson, B.H. Korte, and G.L. Nemhauser (eds.). *Annals* of Discrete Mathematics 1: North-Holland, Amsterdam. 479-493.
42. Volgenant, T. and Jonker, R. (1982). A Branch and Bound Algorithm for the Symmetric Traveling Salesman Problem Based on the 1-Tree Relaxation. *European Journal of Operational Research.* 9:83-89.
43. Notes on Tabu Search Retrieved from http//itc.ktu.it/itc32/Misev32.pdf
44. Wikipedia, the free encyclopedia - Moore's law (2009). Retrieved from http://en.wikipedia.org/wiki/Moore's_law
45. Zakir H. A., (2010). Genetic Algorithm for the Travelling Salesman Problem using Sequential Constructive Crossover Operator. International Journal of Biometrics and Bioinformatics Volume (3): Issue (6)
46. Zhang, W. (2004). Phase Transitions and Backbones of the Asymmetric

Traveling Salesman Problem. Journal of Artificial Intelligence Research. 21:471-497.

APPENDIX

MATLAB PROGRAM

```
>> %function simanneal()

% **********Read distance (cost) matrix from Table 4.1 ******

>> d = xlsread('distance.xlsx');

>> d_orig = d;

start_time = cputime;

summ=0;

dim1 = size(d,1);

dim12 = size(d);

for i=1:dim1

d(i,i)=10e+06;

end

for i=1:dim1-1

for j=i+1:dim1

d(j,i)=d(i,j);

end

end

%d

% *****************Initialise all parameters**********************

d1=d;

tour = zeros(dim12);

cost = 0;

min_dist=[];

short_path=[];
```

```
%*****************************************************************
%************Initialize Simulated Annealing paratemers************
%T0 Initial temperature is set equal to the initial solution value
>> Lmax = 400; %Maximum transitions at each temperature
ATmax = 200; %Maximum accepted transitions at each temperature
alfa = 0.99; %Temperature decrementing factor
Rf = 0.0001; %Final acceptance ratio
Iter_max = 1000000; %Maximum iterations 13
start_time = cputime;
diary output.txt
% ********Generate Initial solution - find shortest path from each node****
% if node pair 1-2 is selected, make distance from 2 to each of earlier
%visited nodes very high to avoid a subtour
k = 1;
for i=1:dim1-1
min_dist(i) = min(d1(k,:));
short_path(i) = find((d1(k,:)==min_dist(i)),1);
cost = cost+min_dist(i);
k = short_path(i);
% prohibit all paths from current visited node to all earlier visited nodes
d1(k,1)=10e+06;
for visited_node = 1:length(short_path);
d1(k,short_path(visited_node))=10e+06;
end
end
```

```
tour(1,short_path(1))=1;
for i=2:dim1-1
tour(short_path(i-1),short_path(i))=1;
end
%Last visited node is k;
%shortest path from last visited node is always 1, where the tour
%originally started from
last_indx = length(short_path)+1;
>> short_path(last_indx)=1;
tour(k,short_path(last_indx))=1;
cost = cost+d(k,1);
% A tour is represented as a sequence of nodes startig from second node (as
% node 1 is always fixed to be 1
crnt_tour = short_path;
best_tour = short_path;
best_obj =cost;
crnt_tour_cost = cost;
obj_prev = crnt_tour_cost;
fprintf('\nInitial solution\n');
crnt_tour
fprintf('\nInitial tour cost = %d\t', crnt_tour_cost);
nbr = crnt_tour;
T0 = 1.5*crnt_tour_cost;
T=T0;
iter = 0;
```

```
iter_snc_last_chng = 0;

accpt_ratio =1;

%*********Perform the iteration until one of the criteria is met***********

%1. Max number of iterations reached**************************************

%2. Acceptance Ratio is less than the threshold

%3. No improvement in last fixed number of iterations

while (iter < Iter_max && accpt_ratio > Rf)

iter = iter+1;

trans_tried = 0;

trans_accpt = 0;

while(trans_tried < Lmax && trans_accpt < ATmax)

trans_tried = trans_tried + 1;
```


Initial solution

crnt_tour =

3 9 2 5 6 10 8 7 4 1

Initial tour cost = 2360

```
city1 = round(random('uniform', 1, dim1-1));

city2 = round(random('uniform', 1, dim1-1));

while (city2 == city1)

city2 = round(random('uniform', 1, dim1-1));

end
```

```
if (city2>city1)
i=city1;
j=city2;
else
i=city2;
j=city1;
end
nbr(i)=crnt_tour(j);
nbr(j)=crnt_tour(i);
if i==1
if j-i==1
nbr_cost=crnt_tour_cost-d(1,crnt_tour(i))+d(1,crnt_tour(j))-
d(crnt_tour(j),crnt_tour(j+1))+d(crnt_tour(i),crnt_tour(j+1));
else
nbr_cost=crnt_tour_cost-d(1,crnt_tour(i))+d(1,crnt_tour(j))-
d(crnt_tour(j),crnt_tour(j+1))+d(crnt_tour(i),crnt_tour(j+1))-
d(crnt_tour(i),crnt_tour(i+1))+d(crnt_tour(j),crnt_tour(i+1))-d(crnt_tour(j-
1),crnt_tour(j))+d(crnt_tour(j-1),crnt_tour(i));
end
else
if j-i==1
nbr_cost=crnt_tour_cost-d(crnt_tour(i-1),crnt_tour(i))+d(crnt_tour(i-1),crnt_tour(j))-
d(crnt_tour(j),crnt_tour(j+1))+d(crnt_tour(i),crnt_tour(j+1));
else
nbr_cost=crnt_tour_cost-d(crnt_tour(i-1),crnt_tour(i))+d(crnt_tour(i-1),crnt_tour(j))-
d(crnt_tour(j),crnt_tour(j+1))+d(crnt_tour(i),crnt_tour(j+1))-
d(crnt_tour(i),crnt_tour(i+1))+d(crnt_tour(j),crnt_tour(i+1))-d(crnt_tour(j-
1),crnt_tour(j))+d(crnt_tour(j-1),crnt_tour(i));
```

```
end

end

delta = nbr_cost - crnt_tour_cost;

prob1 = exp(-delta/T);

prob2 = random('uniform',0,1);

if(delta < 0 || prob2 < prob1)

summ = summ+delta;

crnt_tour = nbr;

crnt_tour_cost = nbr_cost;

trans_accpt = trans_accpt + 1;

if crnt_tour_cost < best_obj

best_obj = crnt_tour_cost;

best_tour = crnt_tour;

end

else

nbr = crnt_tour;

nbr_cost = crnt_tour_cost;

end

end

accpt_ratio = trans_accpt/trans_tried;

fprintf('\niter# = %d\t, T = %2.2f\t, obj = %d\t, accpt ratio=%2.2f', iter,T,crnt_tour_cost,accpt_ratio);

if crnt_tour_cost == obj_prev

iter_snc_last_chng = iter_snc_last_chng + 1;

else

iter_snc_last_chng = 0;
```

```
end
if iter_snc_last_chng == 10
fprintf('\n No change since last 10 iterations');
break;
end
obj_prev = crnt_tour_cost;
T = alfa*T;
iter = iter + 1;
end
fprintf('\nbest obj = %d', best_obj);
fprintf('\n best tour\n');
best_tour
end_time = cputime;
exec_time = end_time - start_time;
fprintf('\ntime taken = %f\t\n', exec_time);
diary off
```