


1996

Solving multi depot vehicle routing problem for Iowa recycled paper by Tabu Search heuristic

Supachai Pathumnakul
Iowa State University

Follow this and additional works at: <https://lib.dr.iastate.edu/rtd>

 Part of the [Industrial Engineering Commons](#), [Theory and Algorithms Commons](#), and the [Transportation Engineering Commons](#)

Recommended Citation

Pathumnakul, Supachai, "Solving multi depot vehicle routing problem for Iowa recycled paper by Tabu Search heuristic" (1996). *Retrospective Theses and Dissertations*. 16982.
<https://lib.dr.iastate.edu/rtd/16982>

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

**Solving multi depot vehicle routing problem for Iowa recycled paper by
Tabu Search heuristic**

by

Supachai Pathumnakul

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Major: Industrial Engineering

Major Professor: John C. Even, Jr.

Iowa State University

Ames, Iowa

1996

Copyright © Supachai Pathumnakul, 1996. All rights reserved.

Graduate College
Iowa State University

This is to certify that the Master's thesis of
Supachai Pathumnakul
has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

TABLE OF CONTENTS

ACKNOWLEDGMENTS	viii
1 OVERVIEW	1
Introduction	1
Statement of problem and objectives	3
2 LITERATURE REVIEW	5
Single depot vehicle routing problem (VRP)	5
Exact algorithms	7
Heuristic algorithms	8
Multi depot vehicle routing problem (MDVRP)	10
Exact algorithm	12
Heuristic algorithms	12
Tabu Search	15
3 TABU SEARCH HEURISTIC FOR MDVRP	19
The modified Clarke and Wright algorithm	20
Tabu Search concept	23
Short term memory and aggressive search	24
Tabu list type	24
Tabu list size	25
Aspiration criteria	27
Intensification and diversification search	27

Swap-tabu algorithm	27
Tabu Search heuristic for MDVRP	32
Algorithm	32
Neighborhood route	37
Insertion strategy	38
Total distance calculation	41
Conclusion of the heuristic	44
4 HEURISTIC APPLICATION	47
Iowa recycled paper problem	47
Characteristic of the problem	47
Assigning the pick up cities to the nearest depots	48
Construct the initial routes by the modified Clarke and Wright algorithm	48
Improve the initial solution by the swap-tabu algorithm	50
Improve the solution by the Tabu Search for MDVRP	54
Heuristic comparison	58
5 CONCLUSIONS	61
Conclusions	61
Recommendations for further studies	62
APPENDIX A COORDINATE OF CITIES AND DEPOTS OF IOWA RECYCLED PAPER PROBLEM	64
APPENDIX B EXAMPLE OF SOLVING IOWA RECYCLED PA- PER PROBLEM BY THIS HEURISTIC IN STEP BY STEP	67
APPENDIX C COORDINATE OF CITIES OF TEST PROBLEMS	77
APPENDIX D SELECTION OF THE PARAMETER VALUES	79
BIBLIOGRAPHY	83

LIST OF TABLES

Table 3.1	Summary of parameters used in this heuristic	46
Table 4.1	Assigning the pick up cities to the nearest depots	48
Table 4.2	Varying γ parameter in the modified Clarke and Wright algorithm	50
Table 4.3	Initial solution from the modified Clarke and Wright algorithm .	52
Table 4.4	The solution after using the swap-tabu algorithm	53
Table 4.5	Solutions obtained from varying parameters where initial α is 200	56
Table 4.6	The best solution of Iowa recycled paper problem after improving by Tabu search for MDVRP	57
Table 4.7	The best parameters for the problems from previous research . .	58
Table 4.8	Solution comparison between four different heuristics	59

LIST OF FIGURES

Figure 3.1	Vehicle routes including depot and one pick up point	21
Figure 3.2	Linking points i and j together	21
Figure 3.3	Superimposing artificial grid over the network	23
Figure 3.4	Procedure of Tabu Search using short term memory	25
Figure 3.5	Selecting the best admissible candidate	26
Figure 3.6	Swap-tabu procedure	28
Figure 3.7	Distance between the cities in the route	29
Figure 3.8	Initial route	29
Figure 3.9	Iteration 1	30
Figure 3.10	Iteration 2	31
Figure 3.11	Iteration 3	31
Figure 3.12	A problem contains three routes	38
Figure 3.13	Moving city A to route 1	39
Figure 3.14	The original solution before moving	40
Figure 3.15	Insert city 5 to arc 10-11 (the last-arc)	40
Figure 3.16	Insert city 5 to arc 11-12 (the next-arc)	41
Figure 3.17	Insert city 5 to arc 8-9 (the last-arc)	41
Figure 3.18	Insert city 5 to arc 9-3 (the next-arc)	42
Figure 3.19	The new route starting from depot 1	42
Figure 3.20	The new route starting from depot 2	43
Figure 3.21	The new route starting from depot 3	43

Figure 3.22	An original solution before insertion	44
Figure 3.23	The solution after moving city 6	45
Figure 4.1	Assign the pick up cities to the nearest depots	49
Figure 4.2	Initial solution from the modified Clarke and Wright algorithm .	51
Figure 4.3	The solution after using the swap-tabu algorithm	51
Figure 4.4	The best solution of Iowa recycled paper problem after improving by Tabu search for MDVRP	55
Figure 4.5	The best solution of the first problem solved by this heuristic . .	59
Figure 4.6	The best solution of the second problem solved by this heuristic	60
Figure B.1	Set all cities and depots on coordinate X-Y	67
Figure B.2	Assign the pick up cities to the nearest depots	68
Figure B.3	Initial solution from the modified Clarke and Wright algorithm .	69
Figure B.4	The best initial solution after using swap-tabu algorithm	70
Figure B.5	Iteration 1. move city 94 to the route that contains city 95	71
Figure B.6	Iteration 2. move city 10 to the route that contains city 30	72
Figure B.7	Iteration 17	72
Figure B.8	Iteration 18. construct the new route containing city 12 from depot 2	73
Figure B.9	Iteration 39	73
Figure B.10	Iteration 40. move city 16 to the route that contains city 11	74
Figure B.11	Iteration 50. move city 16 to the route that contains city 15	74
Figure B.12	Iteration 51	75
Figure B.13	Iteration 52. move city 88 to the route that contains city 34	75
Figure B.14	Iteration 53. move city 32 to the route that contains city 85	76
Figure B.15	Iteration 108. the best solution.	76

ACKNOWLEDGMENTS

This thesis would not have been possible without the help of several people, companies and organizations. First of all, I would like to express my sincere gratitude to my major professor, Dr. John C. Even, Jr., for his academic guidance, encouragement and friendship throughout my study.

Next, I would like to give special thank to my committee, Dr. Chu Chao - Hsien, who advised me to study and apply Tabu Search heuristic for this thesis. I also thank Dr. S. Keith Adams for his valuable input to this work.

I would like to thank Iowa Department of Natural Resources, Iowa Department of Transportation, Mason City Recycling Center, Cedar River Paper Company and Packaging Corporation of America Company for their invaluable data.

I would also like to thank all of my friends who have helped me to complete this work. Especially, I would like to thank Vara Varavithya and Nawapak Eua-anant for their suggestion in C programming language.

For my great friends, Denny Charlie, Chuang Wei-Chin, Masataka Tsuyama and Somgiat Dekrajangpetch, I would like to thank them for their helpful friendship.

Lastly, I would give all my thankfulness to my dear parents, Laddawan and Opas Pathumnakul. for their love and support during my entire graduate study.

1 OVERVIEW

Introduction

In the current decade, solid waste management is a very important factor in improving the environment. The lack of good management can lead to a lot of pollution and environment damage. There are many ways to manage solid waste such as source reduction, recycling, waste transformation and landfilling. From all of them, recycling is the second highest rank in hierarchy adopted by the U.S. Environmental Protection Agency (EPA) in the last few years. With the highest priority for source reduction, recycling helps reduce the amount of waste which requires disposal and also cuts the demand on the resource. Recycling involves three principle phases [43]:

- (1) The separation, collection and transportation of waste materials.
- (2) The preparation of these materials for reuse.
- (3) The reuse, reprocessing and remanufacture of these materials.

Collection and transportation phase is one of the major costs of the recycling method. Sometimes transportation and collection cost consumes more than 40% of the solid waste management budget [43]. Therefore, an efficient collection and transportation model can lead to significant savings in costs.

The nature of recycling transportation sends trucks from depots, manufacturers or brokers, to pick up collected materials from the sources such as landfills and backs to the same depot after the truck is filled to capacity. Based on this characteristic, the recycling transportation problem is similar to the vehicle routing problem, VRP, which

has been studied by many researchers.

The vehicle routing problem is a combinatorial optimization problem where the objective function is to minimize total distance under some side constraints. Solution is a set of pick up or delivery routes from depots to various supply or demand points that gives minimum total distance covered by the entire fleet. The basic constraints of the problem are vehicle capacity, depot capacity and time constraints. The VRP is different from other pick up and delivery problems in that the starting point and the ending point of each route is the same. In other word, the VRP is an M-traveling salesman problem including constraints such as vehicle capacity, depot capacity and time constraints. The VRP can be broadly divided into 2 different problems:

- (1) Single depot vehicle routing problem (VRP).
- (2) Multi depot vehicle routing problem (MDVRP).

Both of them are non-polynomial algorithm problems (NP Problem). The size of the problem depends on the number of pick up or delivery points. It is well known that NP problem is difficult to find the solution by exact algorithm especially in the large problems. Most of the research papers in the VRP area has been devoted to developing some efficient heuristic algorithms such as Saving heuristic, Tabu Search heuristic, Simulated Annealing, Sweep algorithm or Greedy heuristic for solving this class of problem.

Tabu Search heuristic (TS) is one of the most recent heuristic which has been applied to solve various non-polynomial combinatorial optimization such as scheduling problem, quadratic assignment problem and single depot vehicle routing problem. Glover and Laguna [21] provide a good definition for Tabu Search: “Tabu Search (TS) is a heuristic method designed to cross boundaries of feasibility or local optimality normally treated as barriers, and systematically to impose and release constraints to permit exploration of otherwise forbidden region”(p.70).

In this research, the concept of Tabu Search heuristic is developed and applied to

design the truck routes for some depots in Iowa to pick up recycled paper. The following section is the statement of this problem and objective of this research.

Statement of problem and objectives

The state of Iowa is one of the states in the U.S that uses recycling methods to manage solid waste especially in waste paper and paperboard. Paper and paperboard include the largest percentage, 35-45 percent, of the total municipal waste stream produced by Iowans. In 1995, estimated waste paper produced in the entire state of Iowa was about 280,000 tons of which 30 percent was recycled paper [29] [30]. There are three end-manufacturers and four brokers in Iowa which have extremely high depot capacity (see Appendix A) and can collect all recycled paper from 99 counties. Each county has various quantity of recycled paper (see Appendix A).

This study proposes to design a set of good routes for a fleet of trucks for 7 depots which are 3 end-manufacturers and 4 brokers to pick up recycled paper throughout the state of Iowa. This problem is a real-life problem that matches the multi depot vehicle routing problem including truck constraints, depot constraints and time constraints. The capacity of each truck is set to 20 tons which is a normal truck used in recycling area. For depot capacity, 4 of the depots are brokers where the capacity is unlimited since brokers can transfer their materials to manufacturers in state of Iowa or other neighbor states such as Minnesota or Illinois. The other depots are the end-manufacturers where the capacity is set to 1996 material plan capacity of each manufacturer (see Appendix A). Time constraint is ignored in this research since the correct load and unload time can not be collected. The business competition is also ignored in this research. The pick up points are located at every county-seat assuming that all cities in a county transfer their recycled paper to the county-seat.

The concept of Tabu Search heuristic is being applied to solve this problem since

Tabu Search is a flexible method for problems that have many side constraints. Many combinatorial optimization problems have been solved by Tabu Search. Good results are achieved in a short computation time. In this study, a good reasonable solution is also achieved by this new applied Tabu Search heuristic algorithm. This heuristic is also applied to the other two test multi depot vehicle routing problems with only vehicle capacity constraint from the previous research papers in which different heuristic algorithms were applied. The solutions show that this heuristic is one of the best heuristic for solving the multi depot vehicle routing problem.

Some selected literature reviews in the area of VRP, MDVRP and Tabu Search heuristic are presented in Chapter 2. The new applied Tabu Search heuristic algorithm for this problem is shown in Chapter 3. Chapter 4 includes the results of the heuristic applied to Iowa recycled paper problem and the other two test problems in OR literature. The conclusion is presented in Chapter 5.

2 LITERATURE REVIEW

As mentioned in the previous chapter, the vehicle routing problem can be divided into two different problems, single depot and multi depot problems. In the operation research literatures, most research papers involve solving the single depot problem. Many heuristic algorithms including Tabu Search have been developed and applied to solve such a problem. The MDVRP has been rarely studied and Tabu Search heuristic has not been applied. The following are the some literature reviews in VRP, MDVRP and Tabu Search.

Single depot vehicle routing problem (VRP)

Single depot vehicle routing problem can be formulated as follows by Golden et al. [22]:

Minimize

$$\sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^{NV} d_{ij} x_{ij}^k \quad (2.1)$$

subject to

$$\sum_{i=1}^n \sum_{k=1}^{NV} x_{ij}^k = 1 \quad (j = 2, \dots, n) \quad (2.2)$$

$$\sum_{j=1}^n \sum_{k=1}^{NV} x_{ij}^k = 1 \quad (i = 2, \dots, n) \quad (2.3)$$

$$\sum_{i=1}^n x_{ip}^k - \sum_{j=1}^n x_{pj}^k = 0 \quad (k = 1, \dots, nv) \quad (2.4)$$

$$(p = 1, \dots, n)$$

$$\sum_{i=1}^n Q_i \left(\sum_{j=1}^n x_{ij}^k \right) \leq p_k \quad (k = 1, \dots, NV) \quad (2.5)$$

$$\sum_{i=1}^n t_i^k \sum_{j=1}^n x_{ij}^k + \sum_{i=1}^n \sum_{j=1}^n t_{ij}^k x_{ij}^k \leq T_k \quad (k = 1, \dots, NV) \quad (2.6)$$

$$\sum_{j=2}^n x_{ij}^k \leq 1 \quad (k = 1, \dots, NV) \quad (2.7)$$

$$\sum_{i=2}^n x_{i1}^k \leq 1 \quad (k = 1, \dots, NV) \quad (2.8)$$

$$x_{ij}^k = 0 \text{ or } 1 \quad \text{for all } i, j, k. \quad (2.9)$$

$$x \in S \quad (2.10)$$

where

node 1 is the depot.

n = number of nodes.

NV = number of vehicles.

P_k = capacity of vehicle k .

T_k = maximum time allowed for a route of vehicle k .

Q_i = demand at node i ($Q_1 = 0$).

t_i^k = time required for vehicle k to deliver or collect at node i ($t_1^k = 0$).

t_{ij}^k = travel time for vehicle k from node i to node j ($t_{ii}^k = \infty$).

d_{ij} = shortest distance from node i to node j .

$$x_{ij}^k = \begin{cases} 1 & \text{if arc } (i,j) \text{ is traversed by vehicle } k. \\ 0 & \text{otherwise.} \end{cases}$$

x = matrix with components $x_{ij} \equiv \sum_{k=1}^{NV} x_{ij}^k$, specifying connections regardless of vehicle type.

Equation (2.1) states that total distance is to be minimized. Equations (2.2) and (2.3) assure that each node is visited by only one vehicle. Equation set (2.4) represents the route continuity. Equation set (2.5) is the vehicle capacity constraints. Equation set (2.6) is the time constraints. Equations (2.7) and (2.8) ensure that the number of available vehicles is not exceeded.

Most VRP solution strategies can be broadly divided into two different approaches: (1) Exact algorithm approach (2) Heuristic algorithm approach. The following is a brief summary of some exact algorithms and heuristic algorithms.

Exact algorithms

Exact algorithm is an algorithm based on problem formulations. Laporte and Nobert [33] divided it to three broad categories (1) Direct tree search method. (2) Dynamic programming and (3) Integer linear programming.

A branch and bound approach is one of the best method of solving routing problem. The most important parameter that determines efficiency of algorithm is the effectiveness of the bounds. Branch and bound method has been applied to solve VRP by many researchers. In [8], the authors bound the searching nodes by calculating the minimum spanning tree and compare the solution to the other two heuristic algorithms, saving heuristic and 3 optimal tour method. They find that branch and bound consumes a lot of computation time and the 3 optimal tour method gives the best result. In [32], Laporte, Mecure and Nobert develop branch and bound tree to be an exact algorithm for the asymmetrical capacitated vehicle routing problem. It is based on an integer linear programming formulation of the problem. Christofides and Mingozzi [9] relax state - space associated with a given dynamic programming recursion since the number of vertices causes an enormous state space graph. By this relaxation, the solution to the relaxed recursion provides a bound.

Set partitioning and column generation is an algorithm which has been used in a VRP. In [12], the routing problem with time window constraint is solved by column generation. In this paper, column generation is used to solve the LP relaxation of the set partitioning problem solved by simplex and branch and bound. Columns are constructed by a shortest path algorithm with time windows (SPTW). The exact algorithm with a heuristic method based on the set-partitioning (SP) formulation for VRP is presented

by Agarwal et al. [1]. The authors generate a small set of columns (reduce the problem) by using value of the objective function solved by LP relaxation as the lower bound and using a good heuristic solution which is extracted by solving a set-covering problem as upper bound. The optimal solution is achieved by solving set-partitioning over the set of columns generated.

Heuristic algorithms

Christofides [7] classifies VRP heuristic into three categories: (1) Constructive method. (2) Two phase method and (3) Incomplete-optimization. In the first category, a link between two points is sequentially added until all points have been assigned to some routes. The side constraints are checked for violation every time that a link is added. Cost savings or minimal total distance is used to motivate the choice of linking. In the two phase method, pick up or delivery points are first assigned to vehicles without route. Each route is constructed separately in the second phase by using a traveling salesman problem heuristic. In incomplete optimization methods, the methods apply some optimization algorithm and simply terminate before optimality.

The Clarke and Wright algorithm [11] is the saving heuristic. This heuristic is one of the earliest ones and the most well known heuristic in the VRP constructive method. The method starts with vehicle routes including depot and one other vertex. By merging any two routes, the vehicles are reduced to one and also solution cost or distance is reduced. The saving cost of combining two individual routes together is calculated as follow:

$$s_{ij} = c_{oi} + c_{oj} - c_{ij}$$

where

i, j = pick up or delivery points.

o = depot location.

c_{ij} = distance between point i and j .

The authors link i and j with highest s_{ij} with the condition that the new combined route

must satisfy all side constraints. Now i and j are considered as a single macro point. With this convention, the route combining procedure can be applied repeatedly.

There have been many modifications to the Clarke and Wright algorithm. Gaskell [14] and Yellow [46] independently present the scalar parameter concept to modify saving cost given by $s_{ij} - \theta_{ij}$ where θ is scalar parameter. The different solution is obtained by varying θ and the best of these is chosen. The best θ is different in different problems. It depends on the relative distance between pick up or delivery points and depot location. Golden et al. [22] modify the classical Clarke and Wright algorithm in two principle ways: (1) Reduce the problem by considering saving costs only in the close neighborhood. In this way, the computation time is faster than the original one. (2) Using a route shape parameter to define the saving cost. This concept is the same as scalar parameter in Gaskell and Yellow 's works.

The sweep algorithm is a heuristic in the two phase methods. It has been presented in the research of Wren and Holidays [45] and Gillett and Miller [18]. In Gillett and Miller's work, the problem is assumed Euclidean and the pick up or delivery points are located by their polar coordinates (θ_i, ρ_i) where θ_i is the angle and ρ_i is the ray length. The vertices are ranked in increasing order of their θ_i . The authors construct the feasible initial solution routes by means of assigning the unrouted vertex containing the smallest angle to a vehicle as long as its capacity is not exceeded. Each initial route is optimized separately by solving the corresponding TSP. The solution is improved by performing vertex exchanges between adjacent routes, if this exchange can reduce distance. The optimizing procedure is repeated after every exchange.

The other two-phase heuristic methods are developed by Chirstofides et al. [10] and Fisher and Jaikumar [13]. In [10], the least cost insertion criterion is used to cluster a number of clustering trials in the first phase. The different trials are produced by a user-controlled extra parameter. Each cluster is routed in the way of solving TSP.

In [13], Fisher and Jaikumar apply the concept of generalized assignment problem to

cluster the pick up or delivery points in the first phase. A fast algorithm for generalized assignment problem [14] is used to cluster the points. In this way, the solution in the clustering phase is optimal. Similar to Chirstofides et al. [7] in the second phase, each route is solved separately by a TSP heuristic.

Literature reviews of vehicle routing problem solved by Tabu Search heuristic are presented in Tabu Search section.

Multi depot vehicle routing problem (MDVRP)

Many practical problems can be modeled in the multi depot vehicle routing problem. Cassidy and Bennett [5] model the Transport routing and Multi-depot program (TRAMP). The program is used to organize the food transporting routes from schools which have cooking facilities to schools which can not cook by themselves in the southern half of the Inner London Education Authority's area. Ball et al. [2] implement some approximate solution strategies to determine an optimal fleet size with a common-carrier option for a chemical firm. The chemical firm has three depots and wants to move its products between origins and destinations over the U.S and some parts of Canada.

The integer programming formulation of MDVRP has minor changes from the single depot problem. The following formulation is a generic MDVRP formulation presented by Golden et al. [22].

Minimize

$$\sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^{NV} d_{ij} x_{ij}^k \quad (2.11)$$

subject to

$$\sum_{i=1}^n \sum_{k=1}^{NV} x_{ij}^k = 1 \quad (j = M + 1, \dots, n) \quad (2.12)$$

$$\sum_{j=1}^n \sum_{k=1}^{NV} x_{ij}^k = 1 \quad (i = M + 1, \dots, n) \quad (2.13)$$

$$\sum_{i=1}^n x_{ip}^k - \sum_{j=1}^n x_{pj}^k = 0 \quad (k = 1, \dots, nv) \quad (2.14)$$

$$(p = 1, \dots, n)$$

$$\sum_{i=1}^n Q_i \left(\sum_{j=1}^n x_{ij}^k \right) \leq p_k \quad (k = 1, \dots, NV) \quad (2.15)$$

$$\sum_{i=1}^n t_i^k \sum_{j=1}^n x_{ij}^k + \sum_{i=1}^n \sum_{j=1}^n t_{ij}^k x_{ij}^k \leq T_k \quad (k = 1, \dots, NV) \quad (2.16)$$

$$\sum_{i=1}^M \sum_{j=M+1}^n x_{ij}^k \leq 1 \quad (k = 1, \dots, NV) \quad (2.17)$$

$$\sum_{p=1}^M \sum_{i=M+1}^n x_{ip}^k \leq 1 \quad (k = 1, \dots, NV) \quad (2.18)$$

$$x_{ij}^k = 0 \text{ or } 1 \quad \text{for all } i, j, k. \quad (2.19)$$

$$x \in S \quad (2.20)$$

where

nodes 1,2,...,M are depots.

n = number of nodes.

NV = number of vehicles.

P_k = capacity of vehicle k.

T_k = maximum time allowed for a route of vehicle k.

Q_i = demand at node i ($Q_1 = 0$).

t_i^k = time required for vehicle k to deliver or collect at node i ($t_1^k = 0$).

t_{ij}^k = travel time for vehicle k from node i to node j ($t_{ii}^k = \infty$).

d_{ij} = shortest distance from node i to node j.

$$x_{ij}^k = \begin{cases} 1 & \text{if arc } (i,j) \text{ is traversed by vehicle } k. \\ 0 & \text{otherwise.} \end{cases}$$

x = matrix with components $x_{ij} \equiv \sum_{k=1}^{NV} x_{ij}^k$, specifying connections regardless of vehicle type.

$$S = \{(x_{ij}) : \sum_{i \in Q} \sum_{j \notin Q} x_{ij} \geq 1 \text{ for every proper subset } Q \text{ of } V \text{ containing nodes } 1, 2, \dots, M.\}$$

Equation (2.11) states that total distance is to be minimized. Equations (2.12) and (2.13) assure that each node is visited by only one vehicle. Equation set (2.14) represents the route continuity. Equation set (2.15) is the vehicle capacity constraints. Equation set (2.16) is the time constraints. Equations (2.17) and (2.18) ensure that the number of available vehicles is not exceeded.

Exact algorithm

The only exact algorithm for MDVRP found in OR literature is the work of Laporte et al. [34]. This paper investigates an integer programming formulation for MDVRP and presents an exact algorithm for these such problems. The formulation consists of degree constraints, subtour elimination constraints, chain barring constraints and integrality constraints. The branch and bound algorithm which initially relaxes the last three types of constraints is used to solve the problem.

Heuristic algorithms

The earliest heuristic for MDVRP is presented by Wren and Holiday [45]. Their heuristic consists of two phases: constructing a feasible initial solution and then refining the solution from the first phase. In the initial solution phase, all customer points are ordered and placed on a list. Customer points are assigned to depots and inserted onto a route by considering the least distance way by using the list generated. They generate four different initial solutions by selecting the different starting point in the list. The least cost solution is chosen to be the initial solution. In the second phase, they refine the initial solution by seven special procedures, inspect, single, pair, complain, delete, combine and disentangle.

Tillman and Cain [44] present an upperbound algorithm for solving MDVRP. Their heuristic begins with an initial solution such that each demand point is temporarily assigned to the nearest depot and a truck is also assigned to each demand point at the same time. A pair of points is chosen to link together by means of maximal saving cost at every iteration. This saving cost is slightly different from Clarke and Wright saving cost [11]. The points which are allowed to link are those which do not violate any constraints. When more than one point is assigned to a route that belongs to a specific depot, then all points in that route are permanent members of that depot.

Gillett and Johnson [17] present a multi-terminal vehicle dispatch algorithm. Their method includes three phases. The first phase is to assign demand points to depots by considering two criteria. The first is the minimum insertion cost, and the second considers the demand point's nearest neighbor. In this way, the demand points are clustered into many groups equaling to the number of depots. The sweep algorithm of Gillett and Miller [18] is used to construct the route for every clustering group in the second phase. In the third phase, demand points are allowed to move from routes in one depot to another depot, if that move reduces the total distance.

Golden, Magnanti, and Nguyen [22] use two different heuristics to solve MDVRP. The first one is based on the savings method of Tillman and Cain's savings procedure. They improve in the way which data are manipulated to allow fast computation and minimize storage requirements. This improvement is achieved in two ways. First they reduce the amount of data that must be manipulated at each step by superimposing a grid structure and only linking points in adjacent boxes of the grid. Second they exploit the symmetry of savings data to reduce the saving storage. Their second heuristic is a two phase heuristic. The pick up or delivery points are first allocated to depots, then routes are constructed by linking the points in the same depot. They allocate points to depots by assigning each point to the nearest depots. When pick up or delivery point i is equidistant from several depots, the ratio between the closest depot k_1 and the

second closest depot k_2 is determined. If the ratio $r(i) = d_i^{k_1} / d_i^{k_2}$ is greater than δ where $0 \leq \delta \leq 1$ (chosen parameter), the point i is a border point. Each border point is assigned to closest depot by means of savings procedure of Tillman and Cain. They then solve VRP at each depot by their own heuristic. The second heuristic is better than the first one when faces with a large problem.

Raft [39] develops a heuristic for MDVRP. The author separates the problems to five subproblems: the route assignment, the depot assignment, the vehicle assignment, the delivery period and the detailed route design. Each subproblem is solved separately and then connected to them by an iterative procedure. The important procedures are the route assignment, depot assignment and detailed route design. In route assignment, the author estimates the number of routes used in the problem and each route center. The demand points are assigned to the closest route center. The route centers are then reestimated and the demand points are reassigned. This procedure is repeated until the route center is not changed. In depot assignment, route centers and their associated demand points are assigned to the closest depot. The routes are then constructed by using Russell's heuristic [41] and improved by Lin's 3-optimal heuristic [36].

Laporte, Nobert and Taillieffer [35] examine a class of asymmetrical multi-depot vehicle routing problem and location routing problem. In the first step of their heuristic, the problem is transformed into an equivalent constraint assignment problem by using an appropriate graph representation. Optimal solutions are achieved by using the adapted Capaneto and Toth branch and bound algorithm for asymmetrical TSP [4].

Chao, Gloden and Wasil [6] present a new heuristic for MDVRP. Their heuristic is based on many heuristics which are previously used in solving MDVRP. The initial solution is set by grouping pick up or delivery points to the nearest depots, and then the modified Clarke and Wright algorithm [22] is used to construct the routes in each group. They improve the solution by five procedures: one point movement, clean up, 2-point improvement, re-initialization 1 and re-initialization 2, respectively.

Tabu Search

In the last few years, there are some research papers which involve solving the single depot vehicle routing problem by Tabu Search heuristic. They show that Tabu Search heuristic is one of the best methods that gives reasonably good solutions compared to the other heuristics.

Osman [37] develops Tabu Search heuristic with data structure to solve VRP. He finds that a special data structure for the Tabu Search algorithm can reduce computation time more than 50% compared to no data structure. In this heuristic, an initial solution is routed by a saving heuristic and then improved by Tabu Search. In Tabu Search step, a demand point is allowed to move to the other routes, if that move reduces the cost or distance. There are two selection strategies used to move: the best-admissible selection strategy, BA, and the first-best admissible strategy, FBA. The BA strategy selects the best admissible move from the current neighborhood which yields the greatest improvement or at least non - improvement in the objective function while the FBA strategy selects the first admissible move that yields an improvement in the objective value. After a move, that move is declared tabu for a next tabu size iteration except that that move satisfies the aspiration criteria. The aspiration criteria is set to a situation that a move yields a feasible cost that is lower than the lowest cost found before this iteration. The tabu size is calculated by a regression equation. The Tabu Search is stopped when the algorithm meets stopping criteria based on a maximum number of iterations after the best solution has been found. The maximum number is calculated by regression equation.

Semet and Taillard [42] apply Tabu Search to solving a real-life vehicle routing problem. The problem proposes to design a set of routes to distribute two different orders, bakery and other goods, to 45 grocery stores in the cantons of Vaud and Valais in Switzerland. The company has 21 trucks and 7 trailers. The trailer can be drawn by

a truck. Some stores called trailer stores can receive deliveries by the trucks drawing trailer but some called truck store can receive only the truck. The heuristic is started by a given solution, and then the solution is improved by Tabu Search. A move in this paper means removing an order from a route or inserting an order into a route. The costs of a move in every case are evaluated such as inserting an order for a trailer - store to a route between trailer store and a truck store. There are two tabu list types tested in this problem: one based on forbidding orders to be assigned to particular routes and the other based on forbidding stores to receive deliveries by particular route. In the first type, it means that an order which is selected to be inserted to the another route is forbidden to move back to the previous route for a next tabu size iteration. In the second type, it means that every order that originates from a given store is forbidden to be assigned back to a route from which any one of these orders was just removed. A move is allowed when that move gives the best improvement for the solution. The tabu list size is generated by randomly selected between 6 to 30. To speed up the Tabu Search, The authors reduce the neighborhood size to a quarter of the total moves.

Gendreau, Hertz and Laporte [16] present a Tabu Search heuristic for VRP. They construct an initial solution by means of their own heuristic for the TSP called GENIUS [15] and Tabu Search heuristic is used to improve solution. They avoid the infeasible solution by the means of a penalty function. When the solution is infeasible the penalty cost is added to the solution. The following equation is the penalty function used in this work:

$$F_1(s) = \sum_r \sum_{(v_i, v_j) \in R_r} C_{ij}$$

$$F_2(s) = F_1(s) + \alpha \sum_r [(\sum_{v_i \in R_r} q_i) - Q]^+ + \beta \sum_r [(\sum_{(v_i, v_j) \in C_{i,j}} + \sum_{v_i \in R_r} \delta_i) - L]^+$$

where

s = a set of m routes R_1, \dots, R_m .

$R_r = (v_0, v_{r_1}, v_{r_2}, \dots, v_0)$.

$v_0 = depot$.

$v_i \in R_r$ if v_i is a component of R_r .

$(v_i, v_j) \in R_r$ if v_i and v_j are two consecutive vertices of R_r .

$\alpha, \beta =$ positive parameters.

$[x]^+ = \max(0, x)$.

$C_{ij} =$ nonnegative distance between i and j .

$\delta_i =$ service time require by vertex i .

$q_i =$ nonnegative demand for vertex i .

$Q =$ vehicle capacity.

$L =$ limited time.

α and β are first set to 1. Then, after every 10 iterations, a parameter associated with a constraint that was always violated during the past 10 iterations is multiplied by 2 and a parameter associated with a constraint that was never violated during these 10 iterations is divided by 2 and otherwise the weights stay unchanged. For reducing the neighborhood of a move, a vertex can be only moved to the routes containing at least one of the nearest neighbors of that vertex. The tabu list size is randomly chosen in every iteration between 5 to 10. The concepts of intensification search and diversification search are used in this paper. In intensification search, the first half of the vertices that have the largest frequency move are allowed to search again. In diversification search, vertices that have been moved frequently but always give infeasible solution are penalized by adding to the objective function of the candidate solution a term proportional to the absolute frequency of movement of the vertex currently being moved. The value used for penalty value in diversification search is a constant equal to the product of three factors: (a) the absolute difference value between two successive values of the objective values of the objective function. (b) the square root of the neighborhood size and (c) a scaling factor equal to 0.01.

The other research papers related to Tabu Search are presented in the works of Punnen and Aneja [38], and Hooker and Natraj [26]. In [38], the authors apply Tabu Search heuristic to solve the resource-constrained assignment problem. Their adaptation

of Tabu Search uses, strategic oscillation, randomized short-term memory and multiple starts as a means of diversification search. In [26], Tabu Search heuristic is used to solve the vehicle routing and scheduling problem with time window constraint.

3 TABU SEARCH HEURISTIC FOR MDVRP

In this study, Tabu Search heuristic is applied to design a set of truck routes for the pick up of recycled paper from sources to depots. This problem is a MDVRP problem with capacity of truck and capacity of depot constraints. The heuristic developed in this study is a two phase heuristic. The first phase is to create a good initial solution and the second phase is an improvement of the initial solution. In the first phase, the initial solution is created as in Chao et al. [6]. The pick up points are assigned to the nearest depots and the initial routes are constructed by the modified Clarke and Wright algorithm [22]. The swap-tabu algorithm is used to reorder the pick up points in each route separately. The swap-tabu algorithm concept is not found in Chao et al. [6]. The solution which is obtained after swap-tabu algorithm is a good initial solution. In the second phase, the pick up points in each route are allowed to move to the other routes in the same depot or in different depots. The concept of Tabu Search heuristic is used for moving the pick up points. The entire algorithm can be presented as follow:

1. *First phase*-Initial solution construction.
 - (a) Assign the pick up points to the nearest depots.
 - (b) Construct a set of initial routes by means of the modified Clarke and Wright algorithm.
 - (c) Improve each route by swap-tabu algorithm.
2. *Second phase*-Solution improvement.

- Improve the initial solution by means of Tabu Search heuristic for MDVRP.

The following sections describe the modified Clarke and Wright algorithm, the Tabu Search concept, the swap-tabu algorithm and Tabu Search heuristic for MDVRP, respectively.

The modified Clarke and Wright algorithm

The modified Clarke and Wright algorithm is presented in the work of Golden, Magnanti and Nguyen [22]. Their heuristic is better than the classical Clarke and Wright algorithm in two ways: (1) Computation time is faster since they reduce the searching neighborhood. (2) The solution is more nearly optimal since the route shape parameter is used in the saving procedure.

At each step in the classical Clarke and Wright algorithm [11], the greatest positive savings S_{ij} is sought subject to the following conditions:

1. Points i and j are not already on the same route.
2. Neither i nor j are interior to an existing tour.
3. Vehicle availability is not exceeded.
4. Vehicle capacity is not exceeded.
5. Maximum route time is not exceeded.

To construct the route, points i and j are linked together by the way of savings procedure. The procedure is repeated until all points are routed. The algorithm starts with vehicle routes including depot and one pick up point as in Fig. 3.1

The costs of route 1 is $2d_{li}$ and route 2 is $2d_{lj}$. By merging these two routes as in Fig. 3.2, the distance is reduced $S_{ij} = 2d_{li} + 2d_{lj} - d_{lj} - d_{li} + d_{ij}$, so the saving equation in the basic Clarke and Wright algorithm is $S_{ij} = d_{li} + d_{lj} - d_{ij}$

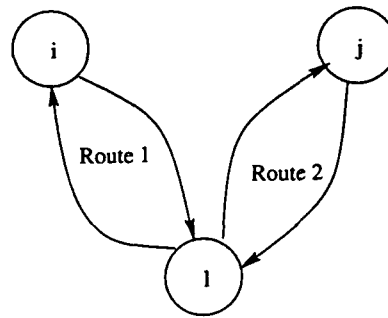


Figure 3.1 Vehicle routes including depot and one pick up point

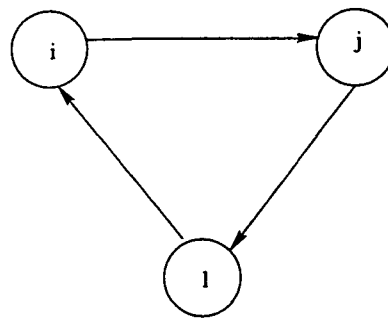


Figure 3.2 Linking points i and j together

where

S_{ij} : saving cost obtained by joining point i and j.

d_{1i} : distance from depot 1 to pick up point i.

d_{1j} : distance from depot 1 to pick up point j.

d_{ij} : distance from point i to j.

To select the greatest saving, all points are considered to link together and the saving cost in every pair linking is calculated. The linking that gives the greatest saving cost is selected to link. This procedure requires a lot of input data storage space and consumes a lot of computation time. For example: a problem which has 100 pick up points requires 100,000 storage spaces for storing the saving costs which are used for considering a linking of points. The modified Clarke and Wright algorithm is modified from the basic Clarke and Wright algorithm in two principle ways:

1. By using a route shape parameter γ to define a modified savings: $s_{ij} = d_{1i} + d_{1j} - \gamma d_{ij}$.

The best route structure is achieved by varying γ .

2. By considering saving costs only between points which close to each other.

In the first modification, the route shape parameter is presented by Yellow [46] and Gaskell [14]. When the parameter γ is increased from zero, greater stress is placed on the distance between points i and j more than their positions relative to depot. The best γ is different depending on the relative distance between pick up points and their depot locations.

In the second modification, rather than considering all pairwise linkings, only the linkings of greatest interest and convenience are considered. The greatest interest and convenience linkings are selected by the following procedure:

- Given the X and Y coordinates of the pick up points in the transportation network.
- Superimpose an artificial grid of width WIDTH and height HEIGHT over the network. The grid then can be divided into DIV^2 rectangular boxes of width WIDTH/DIV and height HEIGHT/DIV. Point i has box coordinate BX(i) and BY(i).
- The linking arcs between depot and pick up points or between pick up points and pick up points which are no further than one box are considered. In other words, if $|BX(i) - BX(j)| > 1$ or $|BY(i) - BY(j)| > 1$, the arc (i, j) is ignored. For example: in Fig. 3.3, DIV is 8 and the depot D can be considered to link with points 4, 10, 11, 5, 13, 14, 15, 16 and point 2 can be considered to link with 1, 15, 16, 8, 9, 3, 12, 11.

The value of DIV influences the accuracy of heuristic algorithm. The best value of DIV depends on the problem. If the number of pick up points is large, DIV should be large. If the number of pick up points is small, DIV should be small. The smaller the DIV, the larger is the number of arcs to be considered.

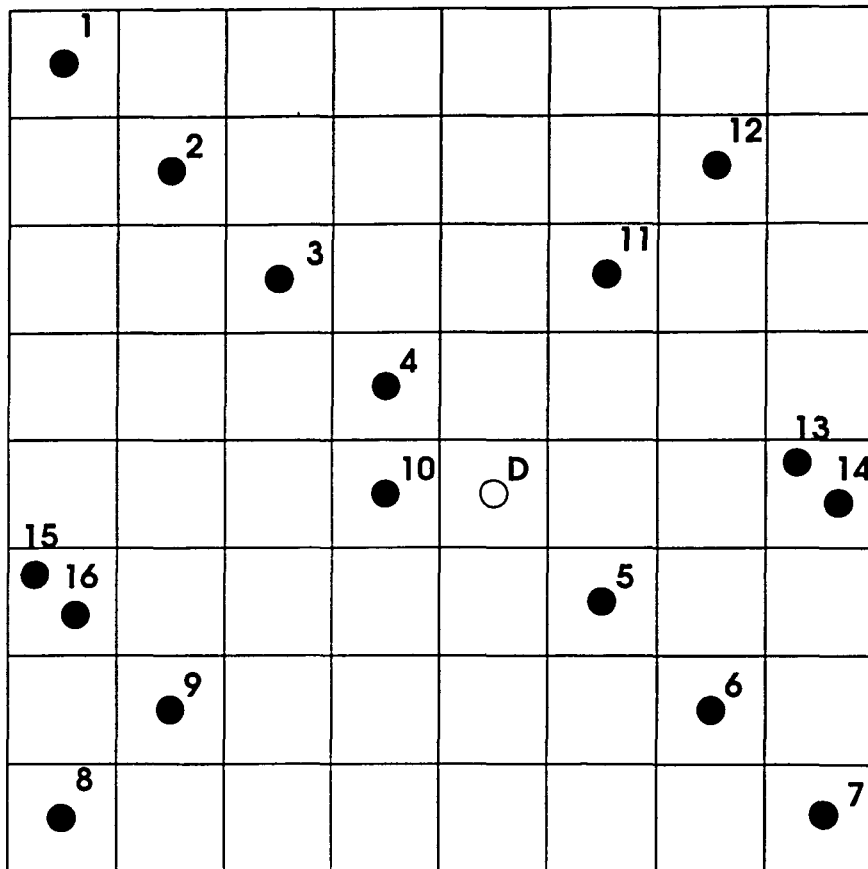


Figure 3.3 Superimposing artificial grid over the network

Tabu Search concept

In this section, the Tabu Search concept is condensed briefly from the works of Glover [19] [20], Glover and Laguna [21].

Tabu Search is a heuristic which is used to solve many kinds of optimization problems. It is an iterative technique which explores a set of the problem, denoted by X , by repeatedly making moves from one solution S to another solution \hat{S} located in the neighborhood $N(S)$ of S .

These moves are performed with the objective of reaching a good solution which is an optimal or nearly optimal solution. Tabu Search heuristic is composed of three principle concepts: (1) The use of short term memory and flexible attribute to explore a

move in the problem. (2) The control mechanism which is used to control the memory structure employment. (3) The use of long term memory to implement the strategies for diversifying and intensifying the search.

Short term memory and aggressive search

The short term memory is the core of Tabu Search. It is used to search for the best (highest evaluation) move which guarantees that the solution is not reverse or cyclic. The Tabu Search tries to avoid trapping in some local optima by using short term memory and suitable tabu list type to forbid some moves causing a reverse or cycle. The tabu status can be overridden, if a move satisfies aspiration criteria. In every iteration, the best move is selected to move. The best move is not only the move that can reduce the cost or distance but also the one that gives the least non-improvement at that iteration. For example in a VRP, at iteration 10 the total distance is 100 miles. No moves in iteration 11 give the total distance less than 100 but the move, moving city i to route j , gives the least non-improvement 110. This move is allowed in this iteration. The concept of using Tabu Search short term memory is presented in Fig. 3.4 and Fig. 3.5.

There are many stopping criteria in the Tabu Search, but the simplest ones are some logical combinations of the following:

- An optimal solution is found.
- The allowed maximum number of iterations for searching is met.
- The number of iterations performed since the lowest solution is met is greater than a specified maximum number of iterations.

Tabu list type

The tabu list type is a control mechanism used for preventing the reversal or cycling of some local optima. The appropriate type of tabu list depends on the problem. The

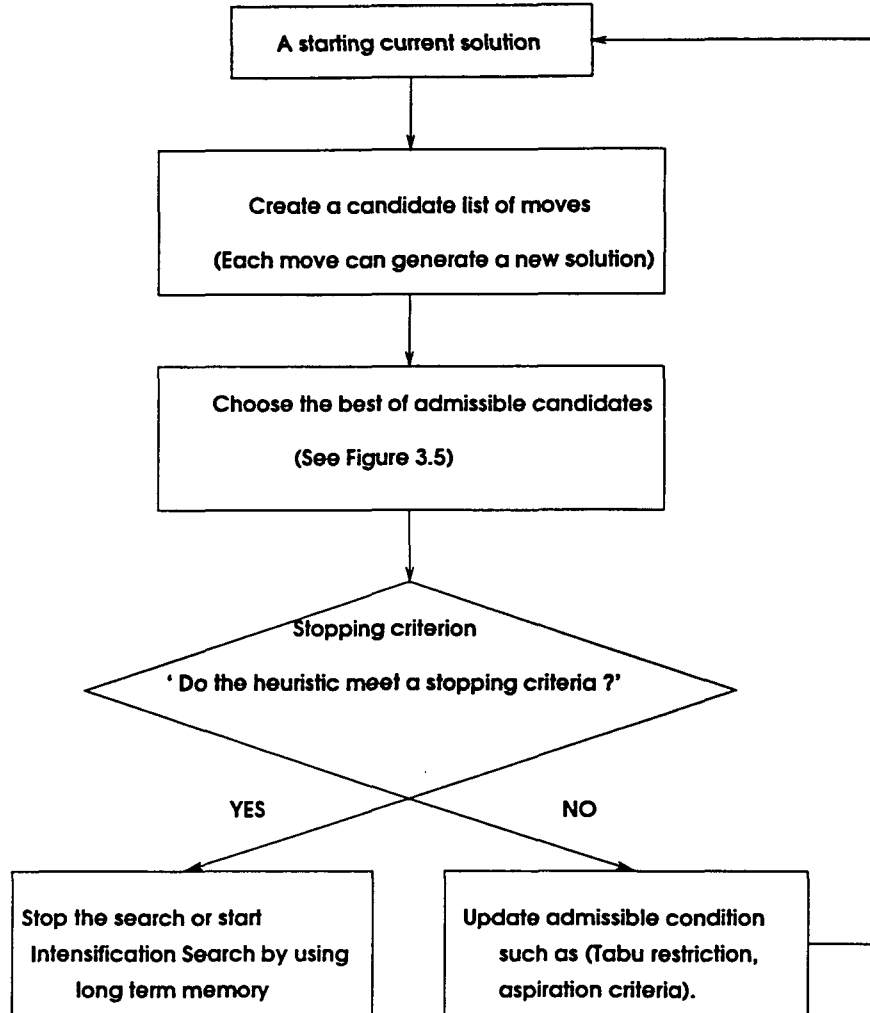


Figure 3.4 Procedure of Tabu Search using short term memory

tabu list type which is normally used in the large problem such as traveling salesman problem, vehicle routing problem in that point i which just was moved from position j can not be moved back to j again in some future iterations. In some small problem, there may be more than one tabu list type. It depends on the characteristic of the problem.

Tabu list size

The tabu list size is a parameter that designates the number of iterations that a move is forbidden. For example: in a VRP where the tabu list size is 5, a move declared

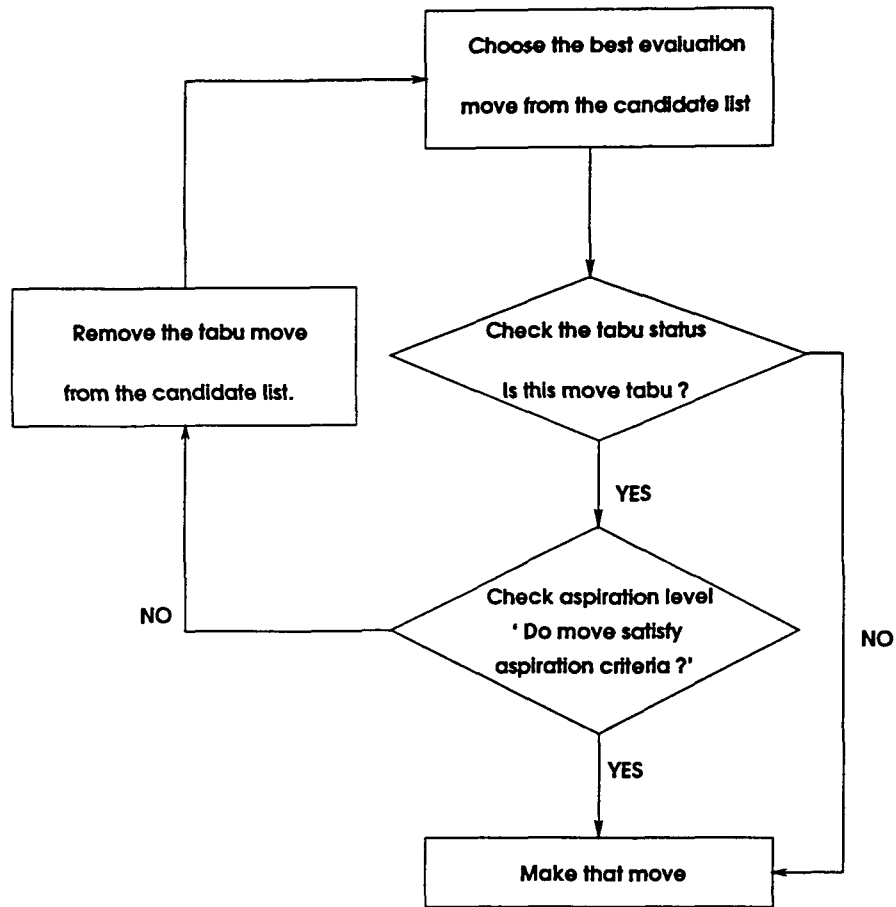


Figure 3.5 Selecting the best admissible candidate

tabu at iteration 1 is not allowed to move back to the last position until iteration 7. The best tabu list size is different in different problems. There are two types of tabu list size: (1) The static tabu list size which means the tabu list size is the same number in every iteration of the search. (2) The dynamic tabu list size which means the tabu list size is different in some iterations. The rule of the dynamic tabu list size is the random selection of the tabu list size between $0.9\sqrt{n}$ to $1.1\sqrt{n}$ where n is the number of dimensions in the problem such as the number of pick up points in VRP. This rule is not specific for every problem. It depends on the problem. Many researchers find that the dynamic tabu size is more effective than the static tabu size. The tabu list size is an the important factor in Tabu Search since when the tabu size is too large, the moves

will ignore the global minima or some good local minima. If the tabu size is too small, the moves will become cyclic in some local minima.

Aspiration criteria

An aspiration criteria is a control mechanism that helps to avoid returning to the previous solution. It is applied to a tabu attribute during the period that it remains tabu, overriding its tabu status. The normal aspiration criteria is that the tabu status is overridden when a tabu move gives a cost which is less than the lowest cost found so far. For example: in a VRP, a pick up point i is moved from position j in iteration 10 and is declared tabu for next 5 iterations. In iteration 14, the move of point i to position j gives the best solution (the least distance) obtained up to iteration 14. The move of point i to position j is allowed.

Intensification and diversification search

Intensification and Diversification are the strategies that exploit long term memory to improve the solution. Long term memory is used in Tabu Search in a kind of learning process.

The intensification strategies are the strategies that reinforce move combinations and solutions properties which usually occur in the good solution. The diversification strategies are the strategies that drive the search into the other regions. The penalty function is one of the methods that is used for diversification strategy in Tabu Search heuristic method.

Swap-tabu algorithm

It is well known that a good initial solution leads to speed up the reaching of the optimal or nearly optimal solution. The routes constructed by the modified Clarke

and Wright algorithm sometimes are not good routes. To improve this initial solution, the swap-tabu algorithm is applied to reorder the sequence of the cities in each route separately.

In this algorithm, the swap process is used to reorder the cities in each route and the concept of Tabu Search is applied to help the swap algorithm avoiding some local minima especially in the routes that contain many cities (see Fig. 3.6). A pair of cities is swapped, if the swap pair gives the best solution in that iteration. The aspiration criteria is the situation that a tabu swap pair can be swapped, if it gives the lowest distance found so far. The concepts of intensification and diversification searches are not applied in this algorithm.

For example: in an initial route, the cities 4, 59, 90, 19, 96, 97, 41, 40 are visited by a truck respectively (see Fig. 3.8). The distances between the cities are presented in Fig. 3.7. Assume that tabu size 2 is the appropriate number for this problem.

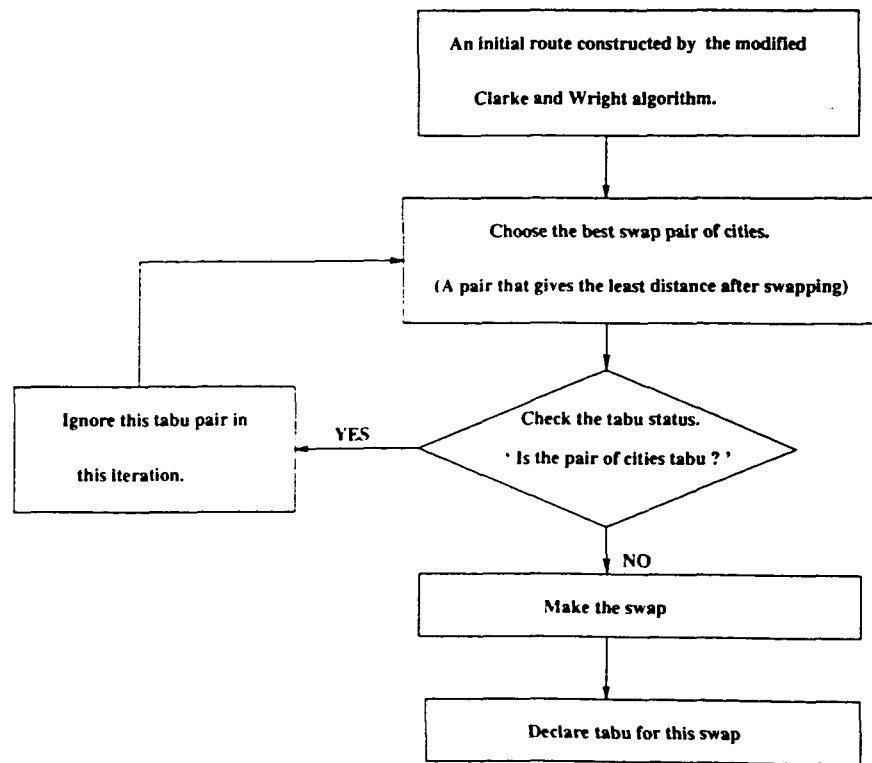


Figure 3.6 Swap-tabu procedure

CITY	4	59	90	19	96	97	41	40
4	—	52	63	75	93	141	98	120
59	52	—	27	127	79	49	58	80
90	63	27	—	100	52	22	35	57
19	75	127	100	—	50	79	93	77
96	93	79	52	50	—	30	45	27
97	141	49	22	79	30	—	23	45
41	98	58	35	93	45	23	—	22
40	120	80	57	77	27	45	22	—

Figure 3.7 Distance between the cities in the route

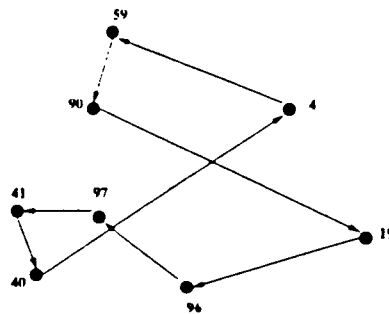


Figure 3.8 Initial route

- *Iteration 0 (initial solution).* The sequence of cities visited by a truck is 4, 59, 90, 19, 96, 97, 41, 40 where 4 is the depot. The total distance is 424 miles (see Fig. 3.8).
- *Iteration 1.* The best swap pair of cities is cities 19 and 41. This swap is the least increase distance in this iteration. The new order of cities is 4, 59, 90, 41, 96, 97, 19, 40 (see Fig. 3.9). The total distance is 465 miles. The swap between cities 19 and 41 is declared tabu for next 2 iterations.
- *Iteration 2.* The best swap pair of cities is cities 96 and 40. The new order of cities is 4, 59, 90, 41, 40, 97, 19, 96 (see Fig. 3.10). The total distance is 403 miles. The swap between cities 96 and 40 is declared tabu.
- *Iteration 3.* The best swap pair of cities is cities 96 and 19. The new order of cities is 4, 59, 90, 41, 40, 97, 96, 19 (see Fig. 3.11). The total distance is 336 miles. The swap between cities 96 and 19 is declared tabu. The tabu status of swapping between cities 19 and 41 is freed.

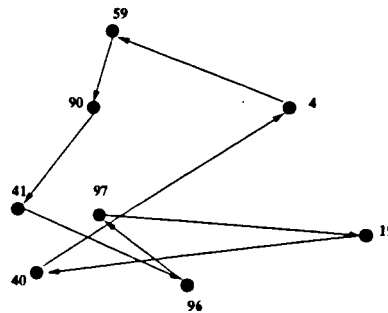


Figure 3.9 Iteration 1

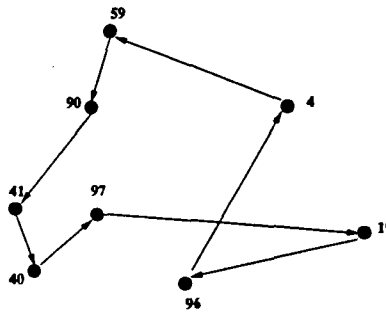


Figure 3.10 Iteration 2

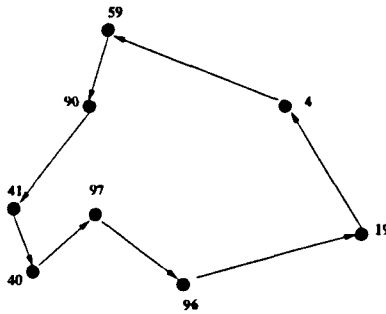


Figure 3.11 Iteration 3

Tabu Search heuristic for MDVRP

This algorithm is the core of this study. The Tabu Search concept is applied to improve the initial solution which is obtained from the modified Clarke and Wright algorithm and the swap-tabu algorithm. The concept of penalty function and diversification search in this study is developed from the work of Gendreau, Hezt and Laporte [16]. They use penalty function to avoid an infeasible solution and diversification in the search in solving single depot vehicle routing problem. This algorithm is composed of searching strategy, aspiration criteria, penalty function, tabu size, tabu list type and diversification search.

The move in this algorithm means that a pick up city is moved from current route to another route in the same depot or the different depots or to a new route which was not in the initial solution.

Algorithm

This section contains a description of the main algorithm in this heuristic. A solution is a set S of routes which can present as:

$$S = \left\{ \begin{array}{cccc} R_{11}, & R_{12}, & \dots, & R_{1n_1} \\ R_{11}, & R_{12}, & \dots, & R_{1n_1} \\ \vdots & \vdots & \dots & \vdots \\ R_{\bar{m}1}, & R_{\bar{m}2}, & \dots, & R_{\bar{m}n_m} \end{array} \right\}$$

where

R_{mn} : The route number n of depot number m .

$1 \leq m \leq \bar{m}$: Depot number.

n_m : The maximum number of routes in depot m .

The routes may be feasible or infeasible depending on the depot and vehicle capacity

constraints. For convenience, $V_i \in R_{mn}$ is used, if V_i is a member of R_{mn} and $(V_i, V_j) \in R_{mn}$, if V_i and V_j are two consecutive points of R_{mn} . The objective function is presented as follow:

$$F_1(S) = \sum_{m=1}^{\bar{m}} \sum_{n=1}^{n_m} \sum_{(V_i, V_j) \in R_{mn}} C_{ij} \quad (3.1)$$

$$F_2(S) = F_1(S) + \alpha \sum_{m=1}^{\bar{m}} \sum_{n=1}^{n_m} [(\sum_{v_i \in R_{mn}} q_i) - Q]^+ + \beta \sum_{m=1}^{\bar{m}} [(\sum_{n=1}^{n_m} \sum_{V_i \in R_{mn}} q_i) - D_m]^+ \quad (3.2)$$

where

C_{ij} : The distance between cities i and j .

α, β : Penalty parameters.

q_i : The quantity of materials in city i .

Q : The vehicle capacity.

D_m : Depot capacity of depot number m .

$[x]^+$: $\max(0, x)$

The equation (3.1) presents the total distance of a set of routes which is a feasible or infeasible solution. The equation (3.2) is the cost function of the search. It is one of the most important factors of the searching strategy. There are three terms in this equation. The first term is the total distance $F_1(S)$. The second and the third terms are the penalty function for the vehicle capacity and depot capacity constraints. If the solution is feasible, the second and the third terms equal zero and $F_1(S) = F_2(S)$. The total distance is the total cost. If the solution is infeasible, the total cost $F_2(S)$ will be greater than the total distance. The more violated the constraint, the greater is cost. This is a diversification of the search since if a move gives an infeasible solution, its cost may be higher than the other moves, so this move may not be selected to move.

The α is the parameter of vehicle penalty function and β is the parameter of depot penalty function. The weights of α and β depend on the characteristic of the problem.

If the vehicle capacity constraint is more serious than the depot capacity constraint, α parameter is set greater than β parameter.

The weights of α and β parameters are important factors in the searching strategy. If the weights of both are set too large, the solution is always local minima since in every iteration, only the move that gives a feasible solution is chosen to move. If the weights are too small, the best solution may be infeasible. The best way to obtain a good solution is to find appropriate α and β . The best α and β are the weights that produce a mix of feasible and infeasible solutions.

The following is the main procedure of the heuristic which is developed from Tabu Search concept.

- *Step 1 (Initialization)*

- The initial routes from the modified Clarke and Wright algorithm and swap-tabu algorithm.
- Set the iteration count $t = 1$.
- No move is tabu.
- Set initial α and β parameters.

- *Step 2 (City selection)*

- Consider the current solution S and randomly select a number of cities to be candidate moves.

- *Step 3 (Evaluation of all candidate move)*

- Repeat the following procedure for all candidate cities V
 - * Consider all potential moves of V from its current route R_c to another route R_x that contains at least a neighborhood of V in the same or

different depots or the new routes which contain no city. Repeat the following operations (Trial move operations) for all candidate moves.

1. Remove V from the current route R_c and insert V to a neighborhood route by an insertion algorithm which will be described in the next section.
2. Calculate the total cost of the trial solution S_t . If the move gives feasible solution, the total cost is $F_1(S_t)$ otherwise the total cost of this move is $F_2(S_t)$.
3. Move the city back to the previous route R_c .

- *Step 4 (Choose the best move)*

- The trial move that gives the least cost solution \acute{S} which may be an infeasible or feasible solution is selected to be an exact move.

- *Step 5 (Check tabu status)*

- If the exact move is tabu and does not satisfy the aspiration criteria, it is disregarded and eliminated from the candidate lists. Go to step 3.
- If the exact move is not tabu or satisfies the aspiration criteria. Go to step 6.

- *Step 6 (Execute the move)*

- Make the exact move.
- Update the new total distance and solution \acute{S}
- If $F_1(\acute{S}) \leq F_1(S^*)$ or $F_2(\acute{S}) \leq F_2(S^*)$, set S^* equal to \acute{S} and $F_1(S^*) = F_1(\acute{S})$ or $F_2(S^*) = F_2(\acute{S})$.

where

$F_1(\acute{S})$: The total cost of the new solution \acute{S} , if \acute{S} is the feasible solution.

$F_2(\acute{S})$: The total cost of the new solution \acute{S} , if \acute{S} is the infeasible solution.

$F_1(S^*)$: The least cost of the feasible solutions found so far.

$F_2(S^*)$: The least cost of the infeasible solutions found so far.

– Set \acute{S} to be the current solution S .

- *Step 7 (Declare tabu)*

– Declare the tabu status for the recent move.

- *Step 8 (Penalty adjustment)*

– If t (number of iterations) is a multiple of h , adjust α and β as follows:

- * If all previous h solutions were feasible with respect to vehicle capacity,

Set $\alpha = \alpha/2$.

- * If all previous h solutions were infeasible with respect to vehicle capacity,

Set $\alpha = 2\alpha$.

- * If all previous h solutions were feasible with respect to depot capacity,

Set $\beta = \beta/2$.

- * If all previous h solutions were infeasible with respect to depot capacity,

Set $\beta = 2\beta$.

- *Step 9 (Terminate check)*

– If $F_1(S^*)$ and $F_2(S^*)$ have not decreased for the last n_{max} iterations, stop otherwise go to step 2.

– If the maximum number of iterations is met, stop otherwise go to step 2.

The aspiration criteria in this heuristic is the same as the normal aspiration criteria in that the tabu status is overridden when a tabu move gives a cost which is less than the least cost found up to this iteration.

The tabu list type is that the city which is just moved is not allowed to move back to the previous route for a next number of iterations. The tabu list size is set to be the parameter. Both static tabu list size and dynamic tabu list size are used in this algorithm.

To control the mix between feasible solutions and infeasible solutions during the search, the concept of varying the weights of α and β parameters in the objective function is applied as in the work of Gendreau et al. [16]. The weight of α is doubled, if the h previous solutions were infeasible and it is halved, if the h previous solutions were feasible with respect to the vehicle capacity constraint. The weight of β parameter is considered in the same way as α parameter. In Gendreau et al. [16], the initial α and β are set to 1 but in this heuristic, initial weights of α and β are parameters. The best initial weights of α and β are achieved by trial and error. The initial weight of α depends on the ratio between the distance and the quantity in pick up cities. If α is small in a problem which has long distance between the cities and the pick up cities contain small quantity of materials, the solution is always infeasible. The move that saves a lot of distance but violates vehicle capacity is always selected to be the solution, for example; in a move that can reduce the distance 100 miles while exceeding the vehicle capacity 2 tons. If α is 1, the total cost is saved 98. This move is allowed since it can save a lot of cost. The moves which give infeasible solutions always occurred in every iteration, so the best solution will be an infeasible solution.

Neighborhood route

In many previous heuristic for MDVRP and VRP such as Chao et al. [6], a city is moved to every route to find the best move. This method consumes a lot of computation time. The heuristic in this study reduces the computation time by moving a city V to neighborhood routes. The neighborhood routes are the routes that contain at least a neighborhood city of city V . The move in this method reduces computation time and

guarantees the best move is found since moving to the neighborhood route is possible to obtain the lower cost solution.

For example: in Fig. 3.12, the move of city A to route 2 is allowed since route 2 contains cities B and C which are the neighbor cities of city A. The move of city A to route 1 is prohibited since it does not contain any neighborhoods of city A. Even though the moving of city A to route 1 is allowed, it is almost impossible to get a good solution from this move (see Fig. 3.13.)

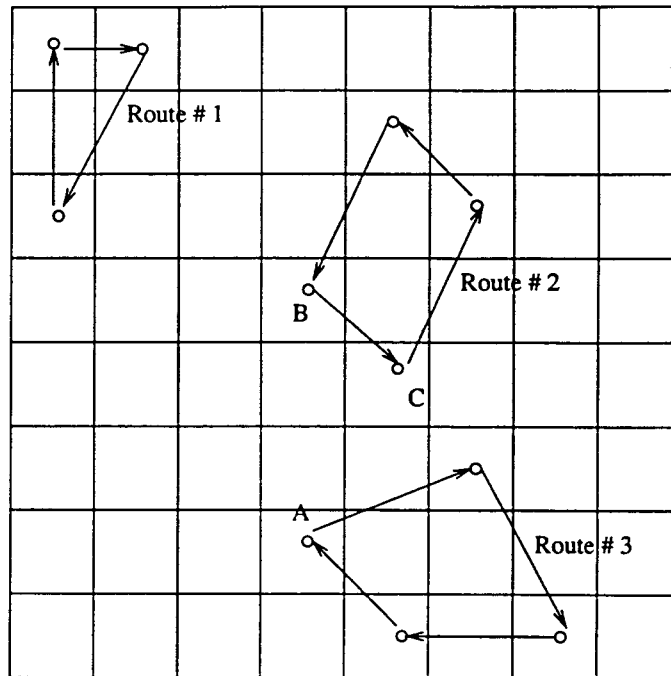


Figure 3.12 A problem contains three routes

Insertion strategy

After finding a set of neighbor routes, a candidate city V is moved there or the new routes. The city V is removed from the current route and inserted to the neighbor routes or the new route. The insertion to the neighbor routes is performed by inserting to the arcs that contain the neighbor city of the candidate city. There are two arcs that involve a neighbor city. An arc that begins from the neighbor city is called the next-arc. An arc

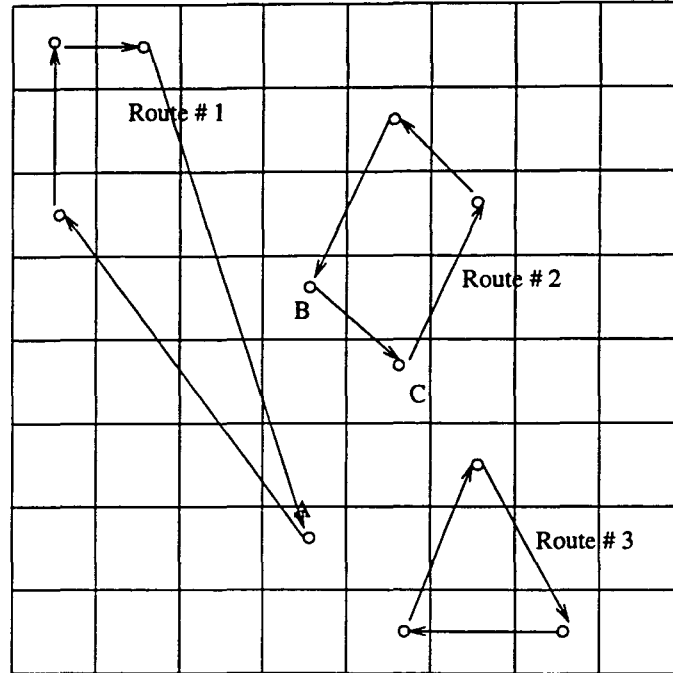


Figure 3.13 Moving city A to route 1

where the neighbor city is the destination is called the last-arc. For example: consider a problem as in Fig. 3.14. Cities 1, 2, 3 are depot. Route 1 visits cities 1, 4, 5, 6, 1. Route 2 visits cities 2, 10, 11, 12, 2 and route 3 visits cities 3, 7, 8, 9, 3. Assume that city 5 is a candidate city in this iteration. Cities 9 and 11 are the neighbor cities of city 5, so routes 2 and 3 are neighbor routes of city 5. For the route 2, the arc 10-11 is the last-arc and arc 11-12 is the next-arc. The insertion to route 2 can only happen between these two arcs (see Fig. 3.15 and Fig. 3.16)

For the route 3, arc 8-9 is the last-arc and arc 9-3 is the next-arc. The city 5 can only insert to these 2 arcs (see Fig. 3.17 and Fig. 3.18).

Assume that the insertion of city 5 to route arc 8-9 in route 3 gives the the lowest total cost compared to the other insertions. This solution is then brought to compare the cost with the new routes that contain only city 5. The new routes start from the depots and visit only city 5. There are three new routes that can occur for city 5, the route starting from depot 1 (see Fig. 3.19), the route starting from depot 2 (see

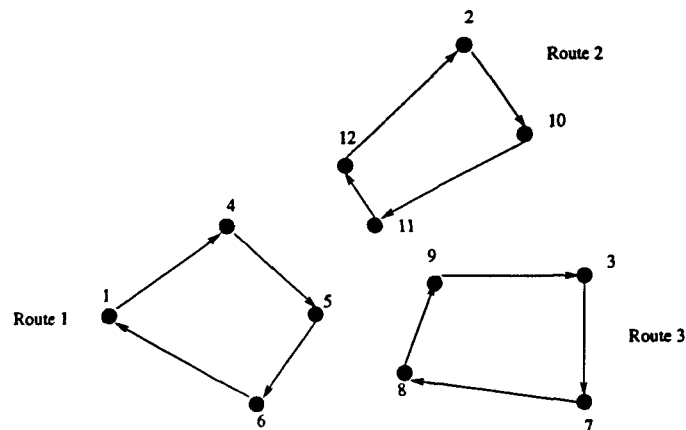


Figure 3.14 The original solution before moving

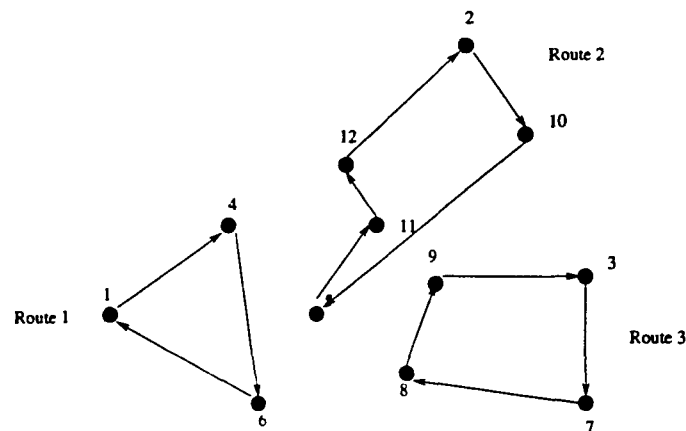


Figure 3.15 Insert city 5 to arc 10-11 (the last-arc)

Fig. 3.20) and the route starting from depot 3 (see Fig. 3.21). The solution that gives the lowest cost is selected to be a good candidate move of this iteration. In the above example, if inserting city 5 to route 2 arc 8-9 provides better solution than constructing the new routes; inserting city 5 to arc 8-9 is a good candidate move of this iteration. The total insertion procedure that is described above is repeated for every candidate city. In every candidate city, one good candidate move is achieved. The last selection is performed between the good candidate moves of every candidate city in that iteration. The best solution of the set of good candidate moves is chosen to be the exact move of that iteration.

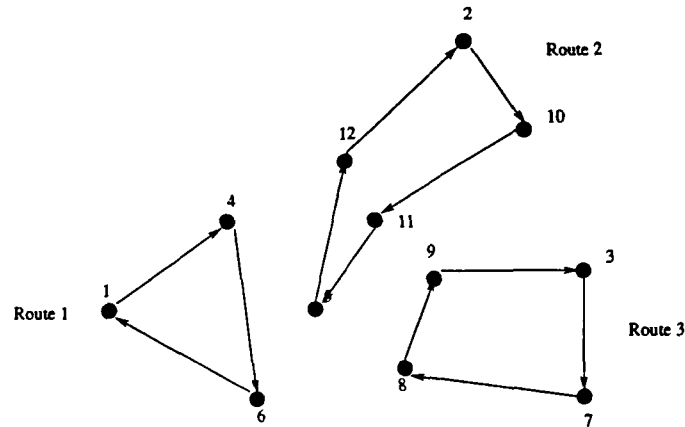


Figure 3.16 Insert city 5 to arc 11-12 (the next-arc)

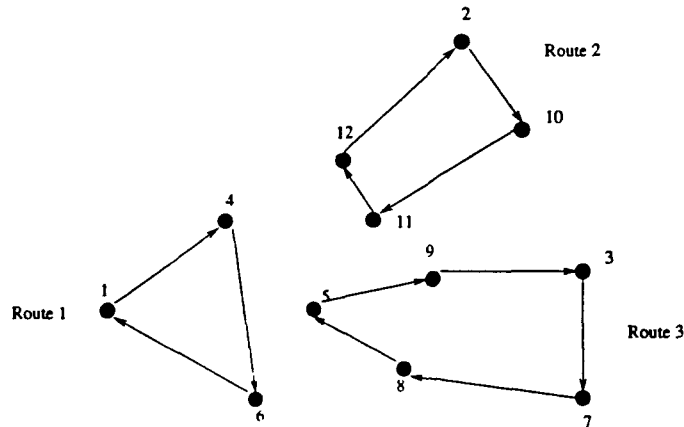


Figure 3.17 Insert city 5 to arc 8-9 (the last-arc)

Total distance calculation

One purpose of this heuristic is to reduce the computation time for reaching the optimal or nearly optimal solution. The computation time for calculating the total distance in every trial move is a factor causing a lot of computation time since it requires a lot of times to calculate the total distance in every iteration to select an exact move. In this heuristic, the behavior of moves are observed. There are two routes that are involved with a move; one is the route that the city is removed (removed-route) and another is the route that the city is inserted to (inserted-route). The other routes stay unchanged. For this behavior, the total distance can be calculated by updating the

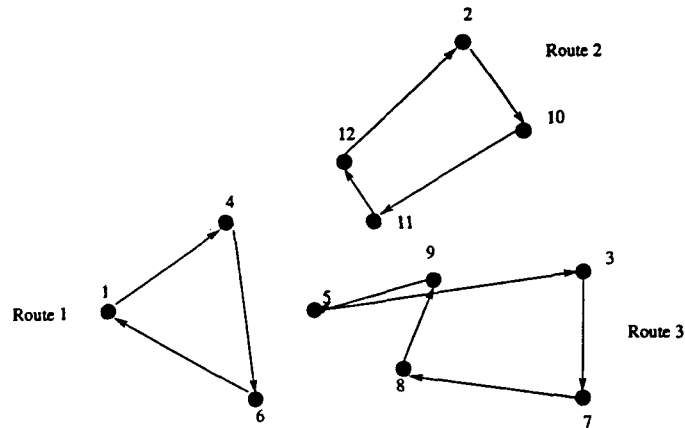


Figure 3.18 Insert city 5 to arc 9-3 (the next-arc)

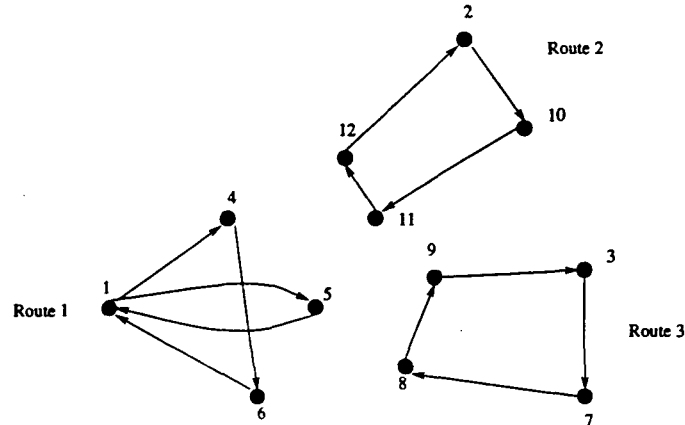


Figure 3.19 The new route starting from depot 1

involved routes. The distance of the removed-route is decreased while the the distance of the inserted-route is increased. The output from subtraction between the distance of these two routes is the distance changed in this move. For example: a problem as in Fig. 3.22, the total distance of the initial solution is 46 miles.

If the city 6 is moved from route 1 to insert to arc 8-9 of route 2, route 1 is removed-route and route 2 is the inserted-route. The distance in route 1 after removing city 6 is decreased 2 miles which is from $d_{56} + d_{61} - d_{51}$. The route 2 is increased 4 miles which is from $d_{86} + d_{69} - d_{89}$. The total distance, after moving city 6 to route 2, is increased 2 miles, so it is 48 miles (see Fig. 3.23).

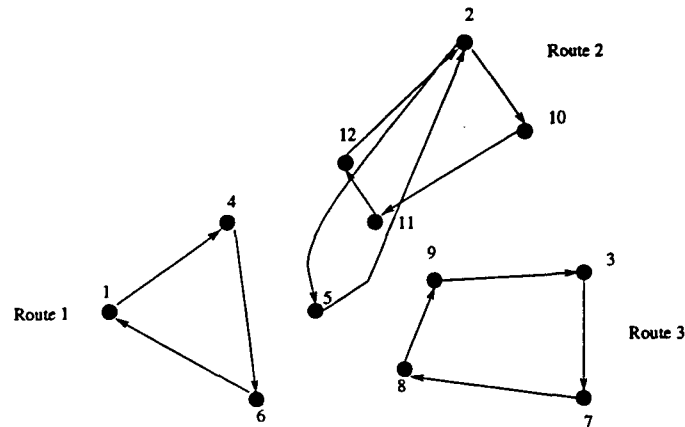


Figure 3.20 The new route starting from depot 2

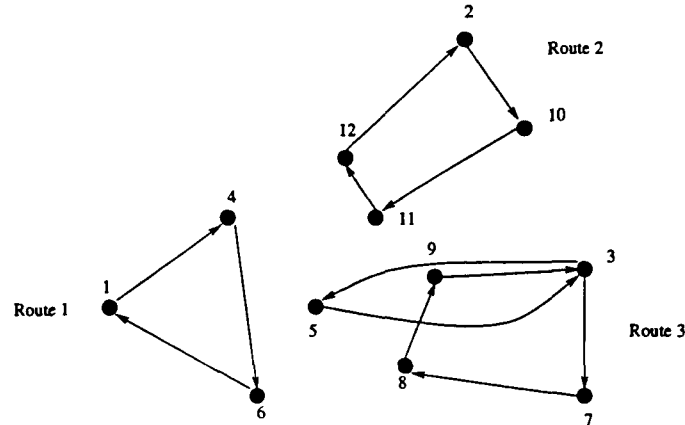


Figure 3.21 The new route starting from depot 3

From this observation, the distance of removed-route is decreased as in equation (3.3) and the distance of inserted-route is increased as in equation (3.4). The total distance is calculated by equation (3.5).

$$S_{de} = d_{ik} + d_{kj} - d_{ij} \quad (3.3)$$

$$S_{in} = d_{ak} + d_{kb} - d_{ab} \quad (3.4)$$

$$\dot{T} = T + S_{in} - S_{de} \quad (3.5)$$

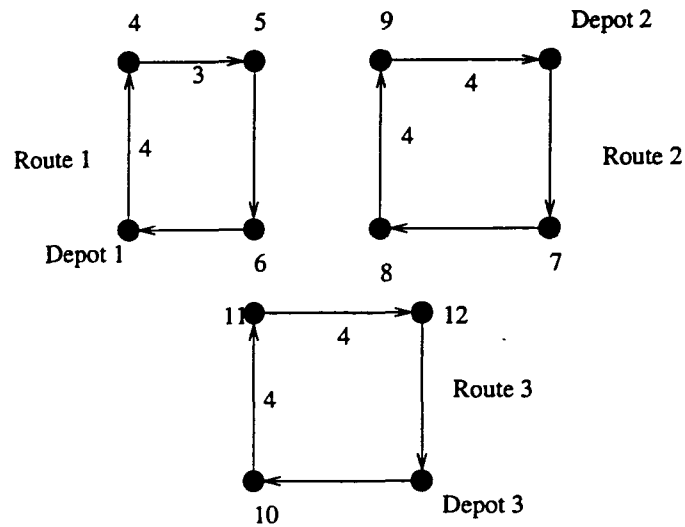


Figure 3.22 An original solution before insertion

where

S_{de} : Decreased distance in the removed-route.

S_{in} : Increased distance in the inserted route.

T' : The new total distance.

T : The previous total distance.

d_{ij} : Distance between city i and city j

k : The candidate city.

i : The city before the candidate city in the removed-route.

j : The city after the candidate city in the removed-route.

a : The city before the candidate city in the inserted-route.

b : The city after the candidate city in the inserted-route.

Conclusion of the heuristic

The heuristic in this study includes the assigning cities to the nearest depot algorithm, the modified Clarke and Wright algorithm, the swap-tabu algorithm and then the Tabu Search for MDVRP algorithm. These algorithms are utilized respectively for

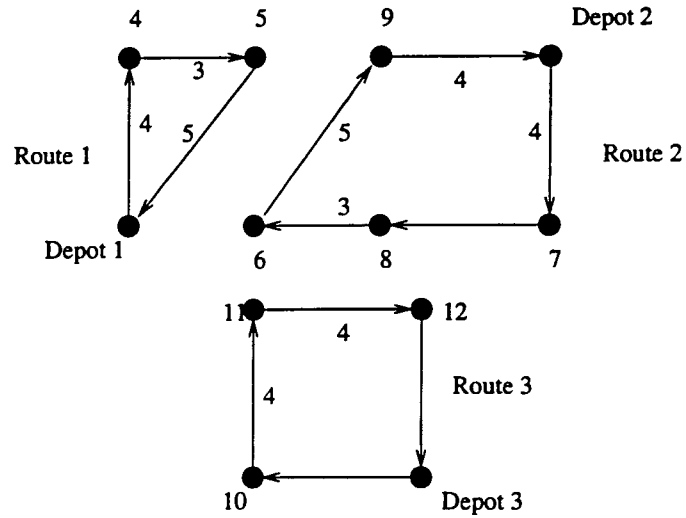


Figure 3.23 The solution after moving city 6

solving a specific MDVRP problem. From the entire heuristic, they contain a number of parameters which can be varied until the best solution is found. All parameters that concern this heuristic are summarized in Table 3.1. The heuristic application is presented in Chapter 4. The step by step of the procedure of this heuristic in solving Iowa recycled paper problem is shown in Appendix B.

Table 3.1 Summary of parameters used in this heuristic

Parameter	Description
WIDTH	The width of the input data. This parameter is used for superimposing an artificial grid in the modified Clarke and Wright algorithm.
HEIGHT	The height of the input data. This parameter is used for superimposing an artificial grid in the modified Clarke and Wright algorithm.
DIV	The number of rectangles in the artificial grid.
γ	The route shape parameter in the modified Clarke and Wright algorithm.
Tabu size	The tabu list size which is used in the swap-tabu algorithm and Tabu - Search for MDVRP algorithm. Tabu size in both algorithms are different.
α	Initial vehicle capacity constraint penalty function used in Tabu Search for MDVRP algorithm.
β	Initial depot capacity constraint penalty function used in Tabu Search for MDVRP algorithm.
h	The number of iterations that the Tabu Search for MDVRP will be adjusted the α and β .

4 HEURISTIC APPLICATION

In this chapter, the heuristic which is presented in the previous chapter is applied to solve the Iowa recycled paper problem. At the end of this chapter, this heuristic is tested by solving 2 MDVRP problems which are presented in previous research papers. The solutions obtained show that this heuristic is a good heuristic for solving MDVRP compared to the other heuristics.

Iowa recycled paper problem

Characteristic of the problem

Iowa recycled problem contains 7 depots and 92 pick up cities which locate at every county-seats. The pick up cities contain different quantity of recycled paper (see Appendix A). Each depot has a different capacity. The depots 1, 2 and 3 are manufacturers, so their capacity are set equal to 1996 plan capacity. The depot capacity of depot 4, 5, 6 and 7 is unlimited since they are brokers. The 20 ton capacity truck is the only type of truck that is used in this problem. The location of depots and pick up cities are presented in coordinate X-Y (see Appendix A).

The Manhattan Matrix is the method used to calculate the distance in this problem since it is close to the real distance between cities in the Midwest of the United States. The Manhattan Matrix distance can be calculated by the following formulation:

$$D_{AB} = |X_A - X_B| + |Y_A - Y_B| \quad (4.1)$$

where

D_{AB} : The distance between city A and city B.

X_A : The coordinate X of city A.

Y_A : The coordinate Y of city A.

Assigning the pick up cities to the nearest depots

The pick up cities are assigned to the nearest depots shown as in Table 4.1 and Fig. 4.1.

Table 4.1 Assigning the pick up cities to the nearest depots

Depots	Pick up cities
1	9 25 26 27 29 30 52 53 75 76 77 81 82
2	10 12 17 18 37 38 45 49 57 80
3	21 22 28 31 39 54 78 79 91 92 93 94 95 98 99
4	11 13 14 15 16 19 36 40 41 43 44 46 47 59 90 96 97
5	8 55 56 70 71 72 73 74 87
6	32 33 34 35 42 60 61 62 63 64 65 66 67 68 69 83 84 85 86 88 89
7	20 23 24 48 50 51 58

Construct the initial routes by the modified Clarke and Wright algorithm

After assigning all pick up cities to the nearest depots, the modified Clarke and Wright algorithm is applied to construct a set of routes in every group of pick up cities separately. Artificial grid is superimposed over the problem by the WIDTH and HEIGHT parameter values 300 and DIV parameter value 6. The WIDTH and HEIGHT

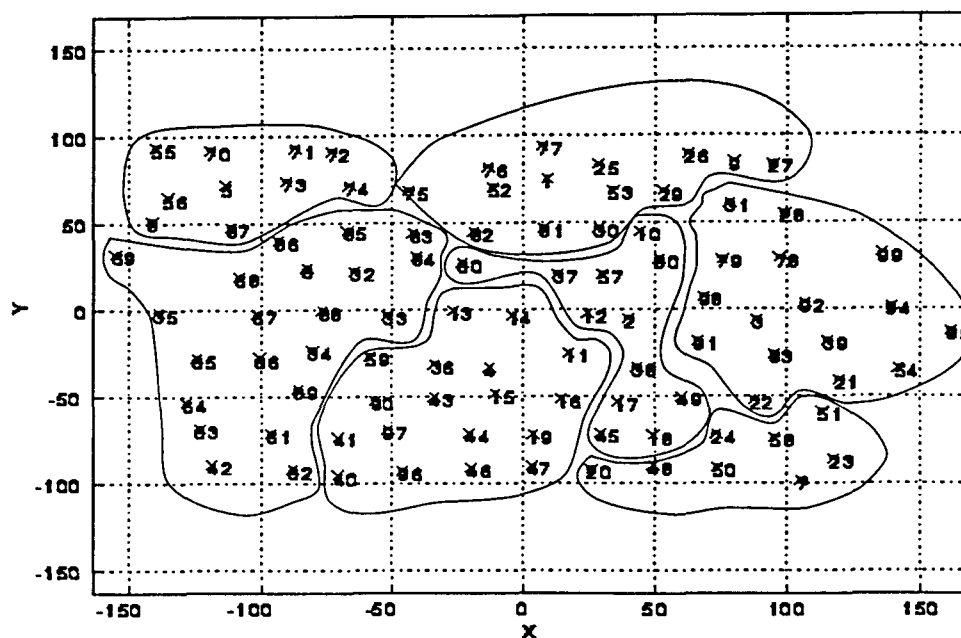


Figure 4.1 Assign the pick up cities to the nearest depots

represent the width and height of the artificial grid. The DIV value 6 is the best DIV in this problem since no cities are ignored in the construction procedure of the modified Clarke and Wright algorithm and also a good solution is obtained. In this problem, the DIV value 10 is the first tried number but some cities are missed in the construction procedure since there are too many rectangular grids (100 rectangular grids) in this problem. Some cities which are on a grid far from the other cities by more than one grid box they are not considered to be the neighborhood cities, so these cities are missed in the construction procedure. The DIV value 8 is found to be the greatest value that no cities are ignored in the construction procedure. The other values which are lower than 8 are also tried. The DIV value 6 is found to be the best DIV value in this problem. There is a tradeoff between the DIV value and the quality of routes; if the DIV value is small, the computation time is large since there are a great number of neighborhood cities to be considered in a constructing procedure but the routes always give a good result.

After superimposing the artificial grid, routes are constructed by the means of the

modified Clarke and Wright algorithm. Each route contains a set of cities and the total quantity of recycled paper is not more than 20 tons. In this step, the route shape parameter γ is varied until the best initial solution is found (see Table 4.2). The best γ for this problem is 0.98 and the total distance is 5,278.2 miles. The solution is shown in Table 4.3 and Fig. 4.2.

Improve the initial solution by the swap-tabu algorithm

The initial solution from the Table 4.3 is improved by the swap-tabu algorithm. In this step, each route contains the same cities as in the initial solution but the sequence of the cities in some routes is changed. The best solution which is presented in Table 4.4 and Fig. 4.3 is achieved when the tabu size is 2. The total distance of the best solution is 5009.7 miles.

Table 4.2 Varying γ parameter in the modified Clarke and Wright algorithm

γ	Total distance (miles)	No. of routes
1.008	5494.5498	32
1.005	5513.1929	32
1.000	5513.1929	32
0.98*	5278.2290*	30
0.95	5407.2036	30
0.90	5503.3472	30

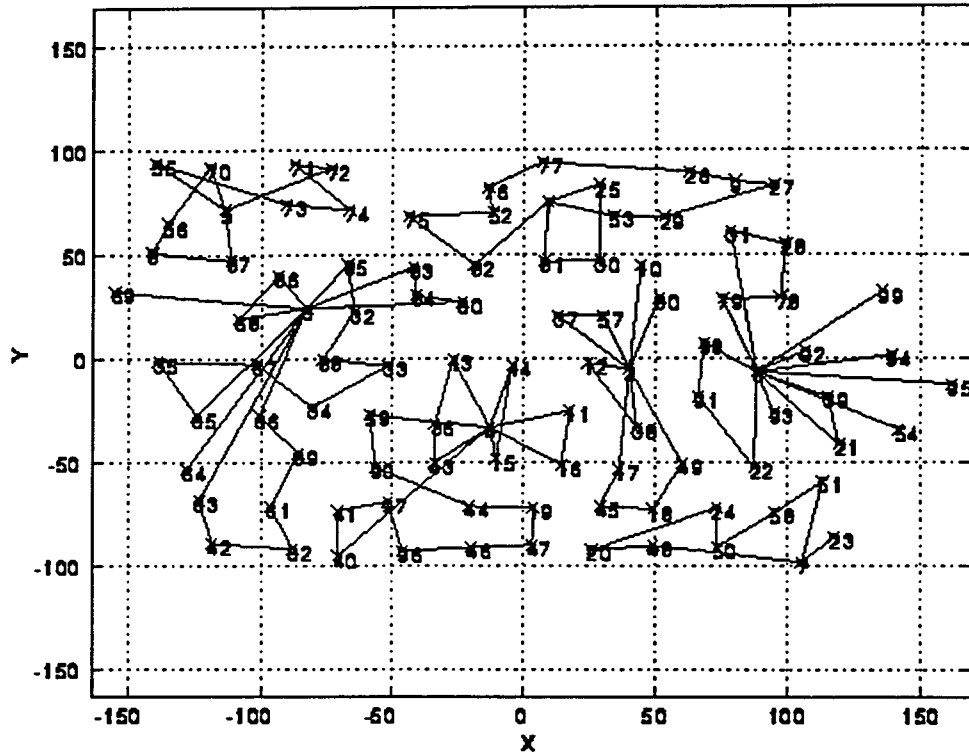


Figure 4.2 Initial solution from the modified Clarke and Wright algorithm

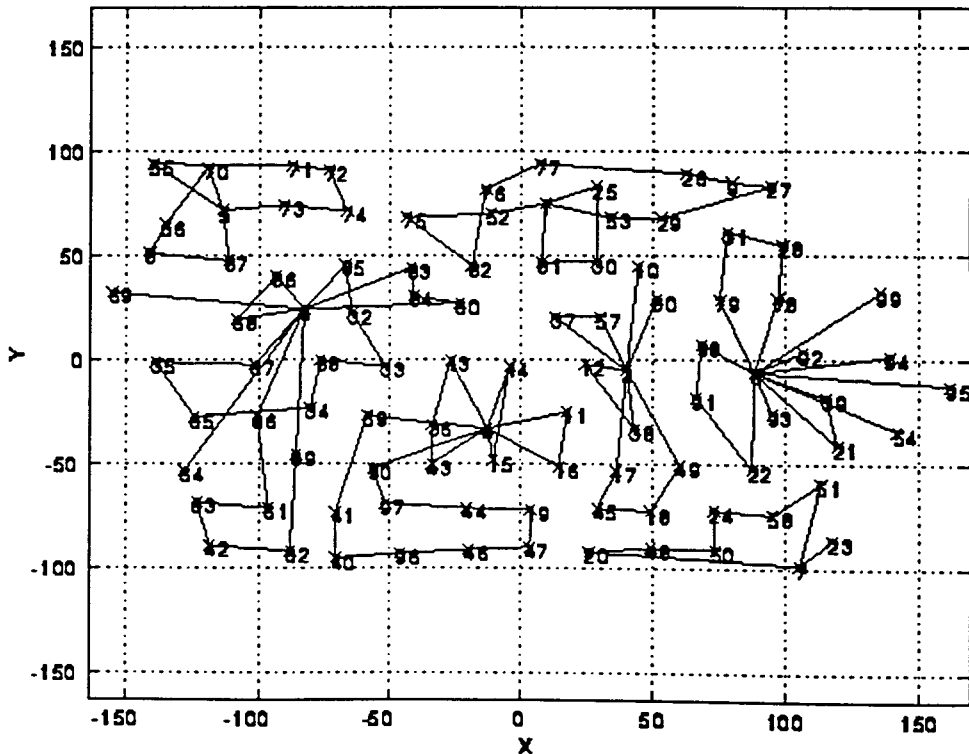


Figure 4.3 The solution after using the swap-tabu algorithm

Table 4.3 Initial solution from the modified Clarke and Wright algorithm

Depots	Routes	Sequence of the pick up cities	Capacity (tons)
1	1	1 82 75 52 76 77 26 9 27 29 53 1	19.44
	2	1 81 30 25 1	6.16
2	1	2 49 18 45 17 2	19.25
	2	2 37 57 2	5.23
	3	2 10 2	6.70
	4	2 80 2	19.99
	5	2 38 12 2	14.36
3	1	3 99 3	19.99
	2	3 95 3	14.04
	3	3 94 3	5.55
	4	3 54 3	19.99
	5	3 31 28 78 79 3	14.89
	6	3 21 39 3	16.31
	7	3 98 91 22 3	14.87
	8	3 93 3	19.99
	9	3 92 3	6.11
4	1	4 40 41 97 96 46 47 19 44 90 59 4	14.48
	2	4 43 36 13 4	16.98
	3	4 16 11 4	15.75
	4	4 15 14 4	19.86
5	1	5 72 71 74 73 55 5	18.59
	2	5 87 8 56 70 5	18.14
6	1	6 64 6	19.99
	2	6 63 42 62 61 89 66 6	17.34
	3	6 83 84 60 6	18.85
	4	6 69 6	19.99
	5	6 65 35 67 34 33 88 32 85 6	19.81
	6	6 86 68 6	4.59
7	1	7 48 20 24 50 58 51 7	15.06
	2	7 23 7	14.79

Table 4.4 The solution after using the swap-tabu algorithm

Depots	Routes	Sequence of the pick up cities	Capacity(tons)
1	1*	1 52 75 82 76 77 26 9 27 29 53 1	19.44
	2	1 81 30 25 1	6.16
2	1	2 49 18 45 17 2	19.25
	2	2 37 57 2	5.23
	3	2 10 2	6.70
	4	2 80 2	19.99
	5	2 38 12 2	14.36
3	1	3 99 3	19.99
	2	3 95 3	14.04
	3	3 94 3	5.55
	4	3 54 3	19.99
	5*	3 78 28 31 79 3	14.89
	6	3 21 39 3	16.31
	7	3 98 91 22 3	14.87
	8	3 93 3	19.99
	9	3 92 3	6.11
4	1*	4 59 41 40 96 46 47 19 44 97 90 4	14.48
	2	4 43 36 13 4	16.98
	3	4 16 11 4	15.75
	4	4 15 14 4	19.86
5	1*	5 55 71 72 74 73 5	18.59
	2	5 87 8 56 70 5	18.14
6	1	6 64 6	19.99
	2*	6 66 61 63 42 62 89 6	17.34
	3	6 83 84 60 6	18.85
	4	6 69 6	19.99
	5*	6 67 35 65 31 88 33 32 85 6	19.81
	6	6 86 68 6	4.59
7	1*	7 20 48 50 24 58 51 7	15.06
	2	7 23 7	14.79

Improve the solution by the Tabu Search for MDVRP

The solution in Fig. 4.3 is the best initial solution since every route is the optimal solution in itself but it is not the best solution of the problem since the sets of pick up cities in a route may not be the best sets. In this step, the pick up cities are allowed to move to the other routes or the new routes which contain no city. The method of inserting the city to the other routes as in chapter 3 is guaranteed by itself that the route after insertion is the optimal or nearly optimal route of the new set of cities. The parameters are run trial and error until the best solution is obtained. The static and dynamic tabu size are both tried between the range 5 - 30. A number of initial α value and h value are tried and the ones giving the best result are selected (see Table 4.5) .

From the Table 4.5, the dynamic tabu size provides better solution than the static tabu size and randomly selects from 20 to 24 as the best choice in this problem. If the tabu size is greater than 24 , the best solution is ignored from the searching procedure and if the tabu size is small, the solution is trapped in some local optimal solution. The h parameter that equals 6 is the best number of iterations to adjust the weights of α and β . Other h values are also tried; the h value between 4-10 is not much different in the results. The initial α and β are related to the tabu size and the h parameter. The β is not important in this problem since the solutions do not exceed the depot capacity. The suitable initial α for this problem is 200. The best solution is found in iteration 108 when the stopping criteria is the 300 maximum iterations or 100 iterations after the best solution was found. The best solution is shown in Table 4.6 and Fig. 4.4. Some different initial α values are found that they lead to the same results since the initial α is considered to adjust the weight in every h iterations. The least total distance in this problem obtained from this heuristic is 4,447.8 miles.

The tabu size parameter is found to be the major effect parameter compared to the other parameters since almost every time that the tabu size is changed a different solution is obtained.

The step by step of this algorithm is shown in Appendix B and comment on parameters is presented in Appendix D

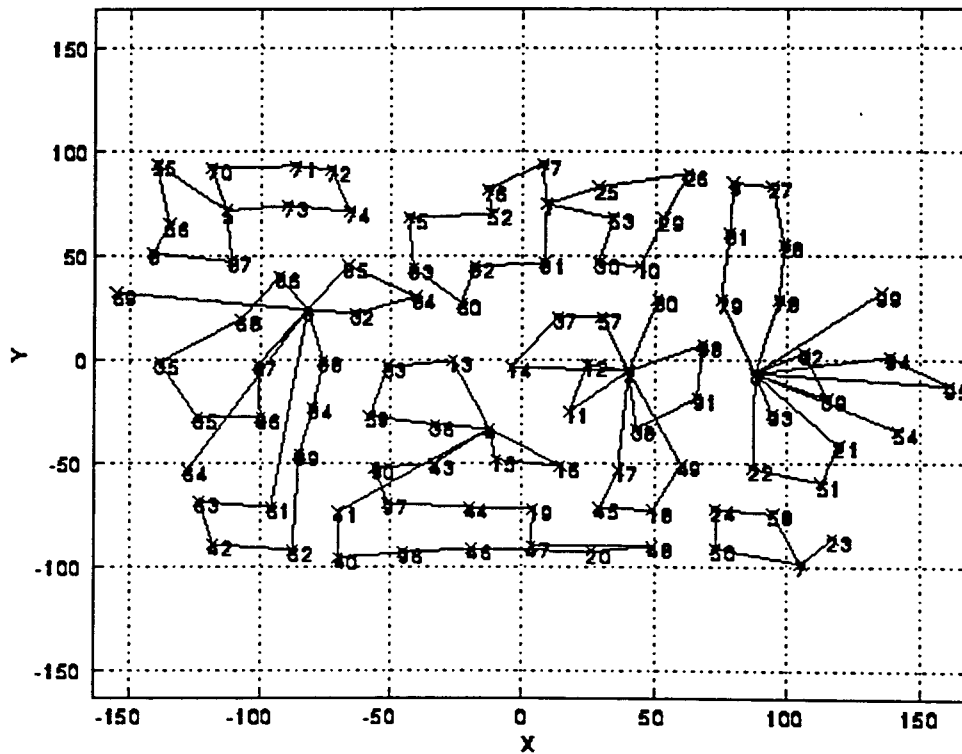


Figure 4.4 The best solution of Iowa recycled paper problem after improving by Tabu search for MDVRP

Table 4.5 Solutions obtained from varying parameters where initial α is 200

Type of tabu size	Tabu size	h	Total distance (miles)	No. of iteration that the best solution is found
Dynamic	20-24	8	4617.28	26
		7	4488.52	103
		6	4447.85*	108
		5	4602.65	54
		4	4580.52	53
Static	20	8	4627.28	26
		7	4488.52	103
		6	4473.36	108
		5	4602.65	54
		4	4580.52	53
Dynamic	5-9	6	4525.73	66
	10-14	6	4539.97	65
	15-19	6	4501.31	95
	25-29	6	4546.43	103
	20-29	6	4503.53	103
Static	3	6	4540.31	57
	5	6	4538.68	63
	8	6	4525.73	66
	10	6	4530.03	67
	14	6	4539.97	67
	15	6	4518.96	68
	25	6	4546.43	103
	30	6	4558.74	79
	40	6	4558.74	79

Table 4.6 The best solution of Iowa recycled paper problem after improving by Tabu search for MDVRP

Depots	Routes	Sequence of the pick up cities	Capacity (tons)
1	1	1 77 76 52 75 83 60 82 81 1	14.22
	2	1 53 30 10 29 26 25 1	18.51
2	1	2 49 18 45 17 2	19.25
	2	2 14 37 57 2	18.71
	3	2 80 2	19.99
	4	2 12 11 2	17.16
	5	2 98 91 38 2	16.76
3	1	3 99 3	19.99
	2	3 95 94 3	19.59
	3	3 54 3	19.99
	4	3 78 28 27 9 31 79 3	19.97
	5	3 22 51 21 3	16.85
	6	3 93 3	19.99
	7	3 92 39 3	10.99
4	1	4 41 40 96 46 20 48 47 19 44 97 90 43 4	19.77
	2	4 36 59 33 13 4	18.66
	3	4 15 16 4	14.42
5	1	5 70 71 72 74 73 5	17.50
	2	5 87 8 56 55 5	19.23
6	1	6 61 63 42 62 89 34 88 6	19.99
	2	6 85 84 32 6	18.04
	3	6 69 6	19.99
	4	6 86 68 35 65 66 67 6	15.41
	5	6 64 6	19.99
7	1	7 50 24 58 7	7.51
	2	7 23 7	14.79

Heuristic comparison

In this section, this heuristic is compared to the other heuristics previously used to solve MDVRP. There are no problems that exactly same as the Iowa recycled paper problem since all of the previous problems are concerned about vehicle capacity constraint or time constraint. They do not consider depot capacity constraint and the distance is calculated by Eucidean matrix. There are two problems in the work of Christofides and Eilon [8] that only involve the vehicle capacity constraint. The heuristic in this study is applied to solve these two problems by relaxing the depot capacity penalty function in the objective function and calculating the distance by Eucidean matrix. The first problem contains 50 cities and 4 depots (see Appendix C). The vehicle capacity is 80. The second problem includes the same cities and depots as in the first one but the vehicle capacity is 160. The parameters which give the best solutions of both problems are shown in Table 4.7. The solutions then are brought to compare with the solutions solved by the heuristics of Chao et al. [6], Gillett and Johnson [17] and Golden et al. [22] (see Table 4.8). The solutions of the first and the second problems solved by this heuristic are presented in Fig. 4.5 and Fig. 4.6 respectively.

Table 4.7 The best parameters for the problems from previous research

Problem	DIV	Type of tabu size	Tabu size	h	Solution (length units)
1	4	Static	10	6	591.00
2	4	Dynamic	9-13	6	476.04

Table 4.8 Solution comparison between four different heuristics

Problem	This heuristic (length units)	Chao (length units)	Gillett-Johnson (length units)	Golden (length units)
1	591.0	582.4	593.2	593.8
2	476.0	476.6	486.2	486.7

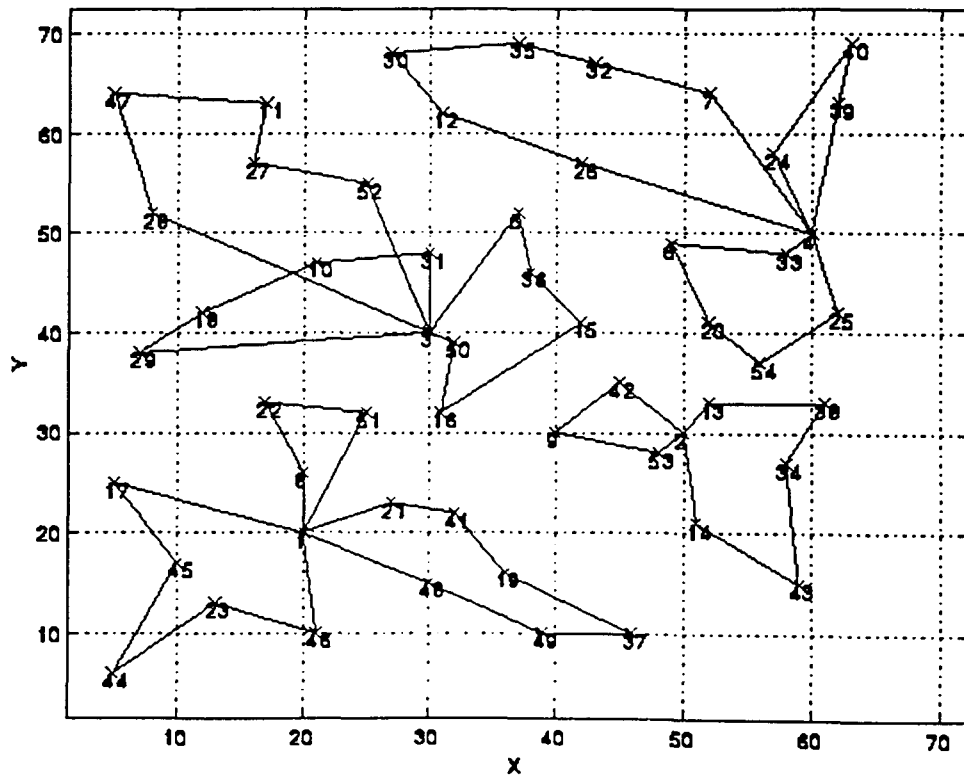


Figure 4.5 The best solution of the first problem solved by this heuristic

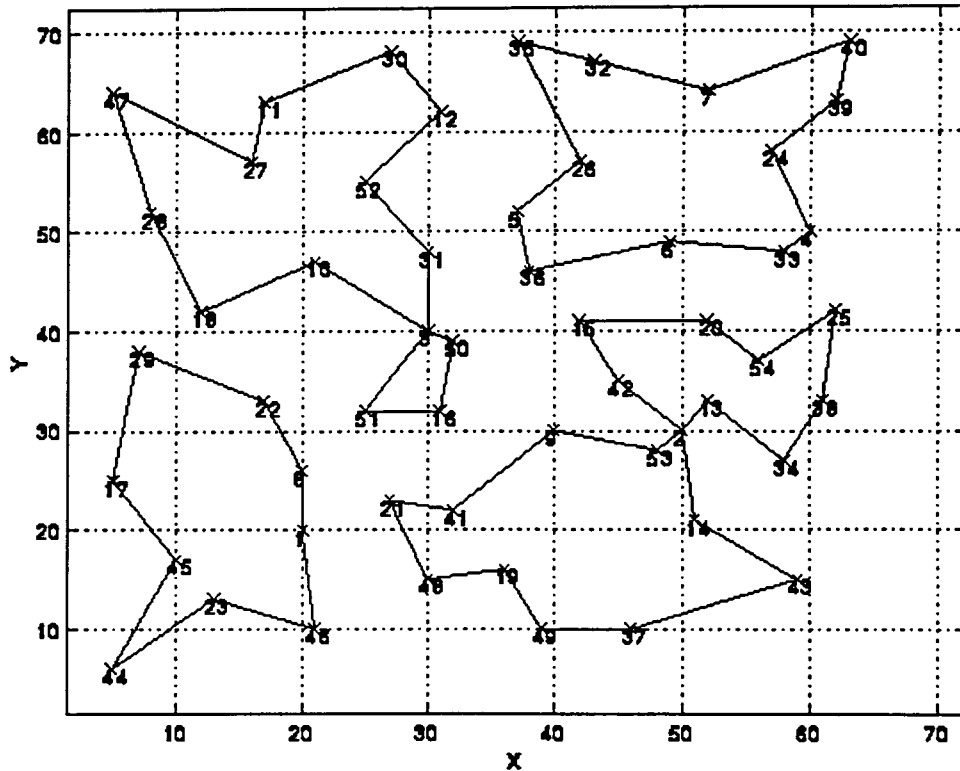


Figure 4.6 The best solution of the second problem solved by this heuristic

From Table 4.8, the solution in the first problem shows that this heuristic is worse than the heuristic of Chao et al. but better than Gillett and Johnson heuristic and Golden heuristic while this heuristic provides the best solution compared to the solutions from the other 3 heuristics in the second problem.

5 CONCLUSIONS

Conclusions

The main purpose of this study is to develop a heuristic to design a set of routes for transporting recycled paper in Iowa which is the multi depot vehicle routing problem. The business competition between depots and the time constraint are ignored but the solution here is one of the possible solutions. The heuristic developed in this study consists of three main parts: (1) The modified Clarke and Wright algorithm. (2) The swap-tabu algorithm and (3) The Tabu Search for MDVRP. The modified Clarke and Wright algorithm is used to construct an initial solution. The swap-tabu algorithm is applied for optimizing each initial route from the modified Clarke and Wright algorithm. The optimal or nearly optimal routes are necessary for the insertion strategy in this study for speeding up the search. The Tabu Search for MDVRP is the most important algorithm in this study. It is developed from the Tabu Search concept and the previous research concerning the single depot vehicle routing problem. The other additional strategies such as neighborhood route, insertion strategy and computation method that are developed in this study are the strategies that help to speed up the algorithm. The concepts of neighborhood route and computation method are shown that they can reduce the computation time and guarantee that good and correct solutions are found. The insertion strategy is the strategy that is used to insert a city to another route and by itself insures that the route including the new moving city is optimal or nearly optimal.

The results from chapter 4 show that the heuristic in this study is a good heuristic for

this problem and the other MDVRP problems even it is not the best heuristic compared to the heuristic of Chao et al. [6], this heuristic requires less procedure than the heuristic of Chao et al. since their heuristic includes 6 algorithms. The computer code for this heuristic is written in C programming language. The running CPU time depends on the number of pick up cities of the problem and the maximum number of iterations that needs to search. In Iowa recycled paper problem which contains 99 cities takes 3.5 minute in CPU time and runs for 300 iterations on a DEC 3000 work station at the Iowa State university computer center.

One of the main weakness in this heuristic is that there are many parameters used in this heuristic such as tabu size, initial α and β parameters in penalty function, h parameter, γ route shape parameter, WIDTH, HEIGHT and DIV parameters. The only way to find the good solution is to run trial and error all of parameters but by the observation in this study, the parameters γ , WIDTH, HEIGHT, DIV, h , initial α and β are minor effect to the results compared to the tabu size parameter, so this weakness is managed by easily selecting the minor effect parameters and focusing on varying the major effect parameter which is tabu size. On the other hand, this heuristic contains only one strict parameter. Therefore this heuristic even has many parameters but it is not difficult to control them. The suggestion of selecting parameters is presented in Appendix D.

Recommendations for further studies

An analysis of this heuristic yields a number of recommendations for further studies:

- Considering dynamic quantity in the pick up cities instead of static quantity as in this study since the dynamic quantity may be more realistic than the static quantity.

- Extending this heuristic for solving the MDVRP with time constraint which considers about 8 working hour per day for the truck driver.
- Extending this heuristic for solving the MDVRP with business competition constraint between the depots.
- Develop a method for finding a good initial parameter set instead of trial and error method which is used in this study.
- In this study, the only type of vehicle used is the 20 ton capacity trucks. If the other capacity trucks can be used to transport the recycled paper; the method of assigning the mixed trucks to the routes can be studied.
- The mix of materials such as paper, cardboard, plastic and glass in one pick up truck is possible for some depots which need many kinds of waste materials. The routes may be more complicated than the routes that transport only recycled paper.

APPENDIX A COORDINATE OF CITIES AND DEPOTS OF IOWA RECYCLED PAPER PROBLEM

The numbers shown here represent pick up cities and depots. The numbers from 1 to 7 represent depots and 8 to 99 are the pick up cities

No.	City	X-axis	Y-axis	Quantity ¹ (ton/day)	Depot capacity (ton/day)
1	MASON CITY	9.129	75.122	-	300
2	TOLEDO	40.2713	-5.134	-	300
3	CEDAR RAPIDS	88.200	-6.354	-	1300
4	DES MOINES	-12.498	-3.368	-	UNLIMITED
5	PRIMGHAR	-113.423	71.898	-	UNLIMITED
6	SAC CITY	-82.487	24.337	-	UNLIMITED
7	FORT MADISON	105.321	-98.422	-	UNLIMITED
8	LE MARS	-141.475	51.324	7.01	-
9	DECORAH	79.644	85.131	2.97	-
10	WAVERLY	44.494	45.171	6.70	-
11	NEWTON	17.801	-25.211	7.71	-
12	MARSHALLTOWN	24.663	-2.285	9.45	-
13	BOONE	-26.457	-0.373	4.61	-
14	NEVADA	-4.116	-3.368	13.48	-
15	INDIANOLA	-10.256	-48.240	6.38	-
16	KNOXVILLE	14.641	-51.336	8.04	-
17	OSKALOOSA	36.496	-53.401	8.17	-
18	OTTUMWA	49.222	-72.689	7.27	-
19	CHARITON	3.931	-72.1434	2.33	-
20	CENTERVILLE	26.210	-92.512	3.41	-
21	MUSCATINE	119.525	-41.387	11.43	-

No.	City	X-axis	Y-axis	Quantity (ton/day)	Depot capacity (ton/day)
22	WASHINGTON	87.337	-52.430	3.02	-
23	BURINGTON	117.473	-86.313	14.79	-
24	FAIRFIELD	73.1614	-72.1410	2.47	-
25	OSAGE	28.89	83.285	2.10	-
26	CRESCO	62.388	89.524	1.39	-
27	WAUKON	94.523	83.316	1.93	-
28	ELKADER	99.12	55.166	2.65	-
29	NEW HAMPTON	53.237	68.143	2.54	-
30	ALLISON	29.123	47.139	2.64	-
31	WEST UNION	78.278	61.324	3.65	-
32	ROCKWELL CITY	-64.115	22.463	1.97	-
33	JEFFERSON	-51.457	-3.4615	1.82	-
34	AUDUBON	-80.274	-23.300	1.18	-
35	ONAWA	-139.1706	-1.135	2.88	-
36	ADEL	-33.248	-31.233	10.16	-
37	ELDORA	13.342	20.51	3.14	-
38	MOTEZUMA	43.217	-33.165	4.91	-
39	TIPTON	115.137	-18.485	4.88	-
40	BEDFORD	-70.316	-95.437	1.17	-
41	CORNING	-70.486	-73.47	0.74	-
42	SIDNEY	-118.364	-89.318	1.99	-
43	WINTERSET	-33.53	-50.156	2.21	-
44	OSCEOLA	-20.129	-71.257	1.33	-
45	ALBIA	29.389	-71.259	2.11	-
46	LEON	-19.334	-91.211	1.34	-
47	CORYDON	3.435	-90.108	1.12	-
48	BLOOMFIELD	49.299	-90.170	1.74	-
49	SIGOURNEY	60.188	-50.610	1.70	-
50	KEOSAUQUA	73.4140	-91.1439	1.00	-
51	WAPELLO	113.214	-59.395	2.40	-
52	GARNER	-11.498	70.230	1.03	-
53	CHARLES CITY	34.320	68.382	3.14	-
54	DAVENPORT	141.755	-31.182	19.99	-
55	ROCK RAPIDS	-140.2145	94.317	2.69	-
56	ORANGE CITY	-135.344	65.319	7.10	-
57	GRUNDY CENTER	30.337	20.635	2.09	-
58	MT. PLEASANT	95.161	-74.460	4.04	-
59	GUTHRIE CENTER	-58.276	-27.167	2.07	-
60	WEBSTER CITY	-23.138	27.291	3.00	-
61	RED OAK	-96.174	-71.429	2.93	-

No.	City	X-axis	Y-axis	Quantity (ton/day)	Depot capacity (ton/day)
62	CLARINDA	-87.814	-91.915	4.10	-
63	GLENWOOD	-123.380	-68.517	3.39	-
64	COUNCIL BLUFFS	-128.156	-53.503	19.99	-
65	LOGAN	-124.324	-27.487	2.84	-
66	HARLAN	-100.469	-27.414	2.25	-
67	DENISON	-101.443	-2.493	2.85	-
68	IDA GROVE	-107.947	19.486	1.41	-
69	SIOUX CITY	-155.281	32.274	19.99	-
70	SIBLEY	-119.217	92.278	1.60	-
71	SPIRIT LAKE	-87.107	93.278	5.99	-
72	ESTHERVILLE	-73.322	91.451	3.12	-
73	SPENCER	-90.143	74.297	3.94	-
74	EMMETSBURG	-66.155	71.480	2.85	-
75	ALGONA	-43.292	68.412	2.37	-
76	FOREST CITY	-13.448	82.107	1.00	-
77	NORTHWOOD	7.335	94.255	0.67	-
78	MANCHESTER	97.196	29.298	2.44	-
79	INDEPENDENCE	75.299	28.490	6.15	-
80	WATERLOO	51.387	29.278	19.99	-
81	HAMPTON	8.267	46.723	1.42	-
82	CLARION	-18.322	45.280	2.40	-
83	DAKOTA CITY	-41.456	44.215	2.33	-
84	FORT DODGE	-40.499	30.380	13.52	-
85	POCAHONTAS	-66.717	45.487	2.55	-
86	STORM LAKE	-93.160	40.154	3.18	-
87	CHEROKEE	-111.284	47.511	2.43	-
88	CARROL	-76.311	-0.109	3.72	-
89	ATLANTIC	-85.488	-45.441	2.68	-
90	GREENFIELD	-56.153	-52.290	1.47	-
91	MARENGO	66.139	-18.160	4.58	-
92	ANAMOSA	106.473	3.484	6.11	-
93	IOWA CITY	94.866	-26.172	19.99	-
94	MAQUAKETA	138.679	1.473	5.55	-
95	CLINTON	161.321	-12.361	14.04	-
96	MT. AYR	-45.332	-93.109	0.88	-
97	CRESTON	-51.425	-69.20	2.03	-
98	VINTON	68.161	7.340	7.27	-
99	DUBUQUE	135.296	32.220	19.99	-

APPENDIX B EXAMPLE OF SOLVING IOWA RECYCLED PAPER BY THIS HEURISTIC IN STEP BY STEP

- *Initial solution*

- Set all cities and depots on coordinate X-Y (Fig. B.1).

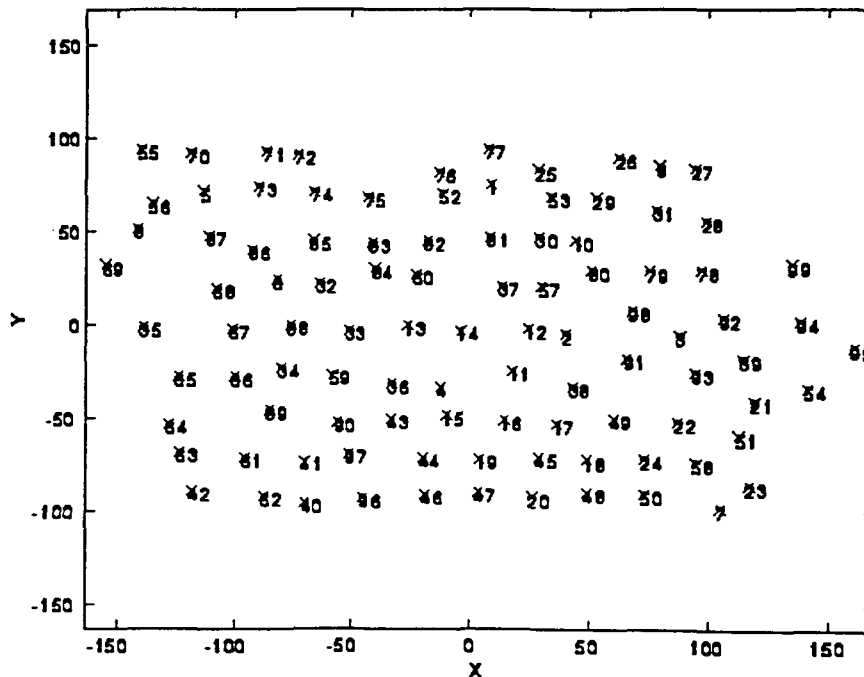


Figure B.1 Set all cities and depots on coordinate X-Y

- Assign the pick up cities to nearest depots (Fig. B.2).
- Superimpose artificial grid to all pick up cities and depots (Fig. B.2). In this problem, WIDHT parameter is 300. HEIGHT parameter is 300 and DIV

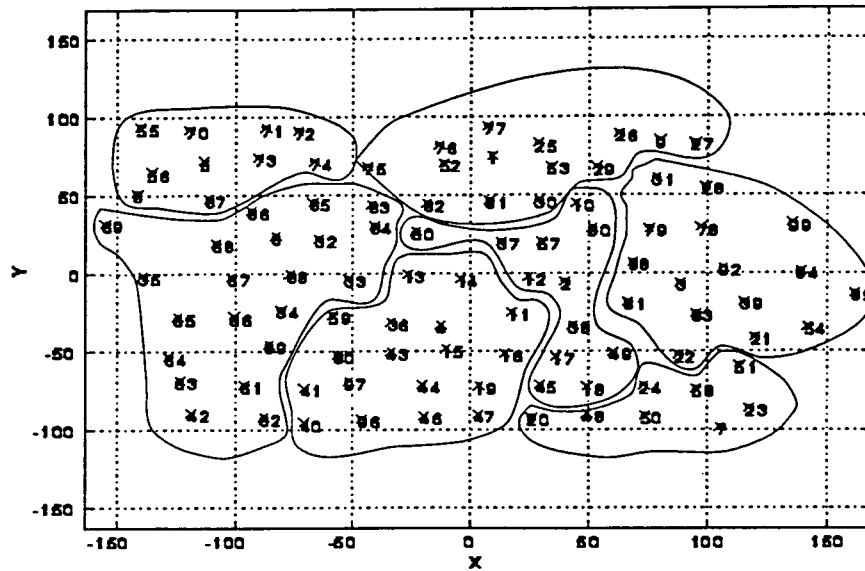


Figure B.2 Assign the pick up cities to the nearest depots

parameter is 6.

- Construct initial routes by the modified Clarke and Wright algorithm with γ 0.98 for the sets of cities which were grouped as Fig. B.2. The grid is used to define neighborhood cities for the modified Clarke and Wright algorithm. The cities which are no farther different than one box are considered to link together. Each route quantity is less than or equal to 20 tons (Fig B.3)
- Improve initial solution by the swap-tabu algorithm. Every route is improved by swap-tabu algorithm to optimize itself (Fig B.4). The solution here is the best initial solution. The total distance is 5009.72 miles.
- *Improve initial solution by Tabu Search for MDVRP*
 - Iteration 1, the move of city 94 to the route that contains city 95 and at the next-arc is the best move giving 4908.77 miles in total distance compared to the other moves. This move is the exact move in iteration 1 and city 94 is not allowed to move to the route that just left until iteration 22 (Fig. B.5).

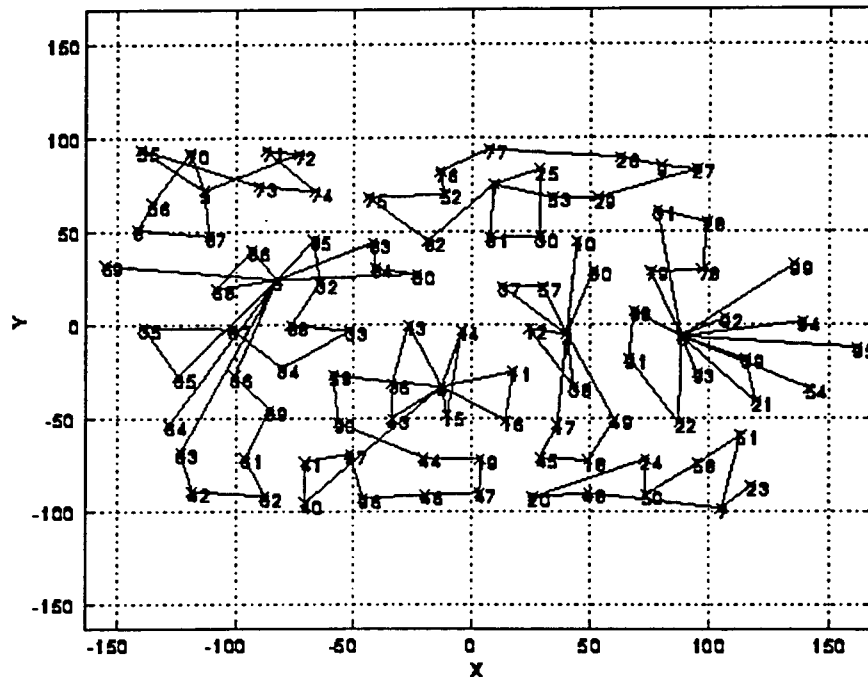


Figure B.3 Initial solution from the modified Clarke and Wright algorithm

- Iteration 2, the move of city 10 to the route that contains city 30 and at the last-arc is the best move giving 4833.56 miles in the total distance compared to the other moves in iteration 2. This move is the exact move of iteration 2 and city 10 is not allowed to move to the route that just left until iteration 23 (Fig B.6)
- Iteration 17, the routes are as Fig. B.7.
- Iteration 18, constructing the new route that contains only city 12 from depot 2 is the best move giving 4641.05 miles compared to the other moves (Fig. B.8).
- Iteration 39, the routes are as Fig. B.9.
- Iteration 40, the move of city 16 to the route that contains city 11 and at the last-arc is the best move giving 4636.63 miles in the total distance compared to the other moves in iteration 40. This move is the exact move in iteration

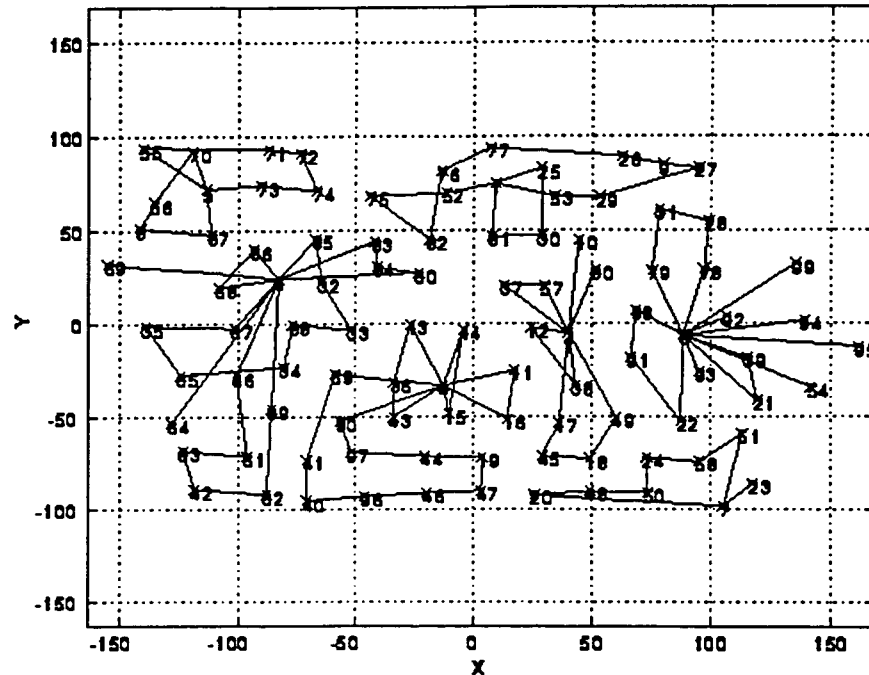


Figure B.4 The best initial solution after using swap-tabu algorithm

40 and city 16 is not allowed to move to the route that just left until iteration 62 (Fig B.10).

- Iteration 50, the move of city 16 to the route that contains city 15 and at the next-arc is the best move giving 4616.63 miles in the total distance compared to the other moves in iteration 50. The move of city 16 to this route is declared tabu at iteration 40 until iteration 62 but this move gives the best solution found up to this iteration, so the tabu status of moving city 16 to this route is overridden and allowed to move. The new tabu status is that the city 16 is not allowed to move to the route that just left until iteration 73 (Fig B.11).
- Iteration 51, the routes are as Fig. B.12.
- Iteration 52, the move of city 88 to the route that contains city 34 and at the next-arc is the best move giving 4602.99 miles in the total distance compared

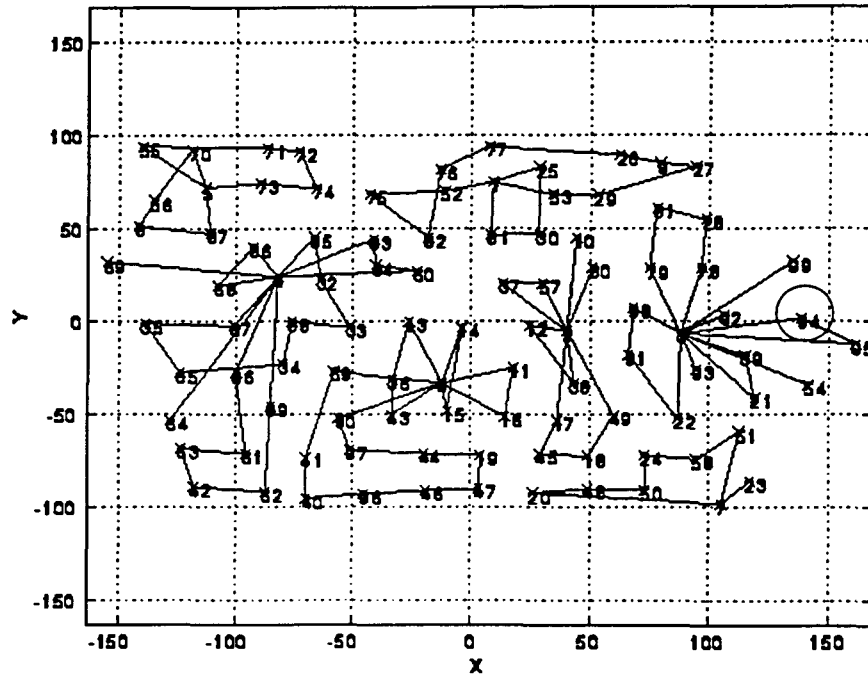


Figure B.5 Iteration 1, move city 94 to the route that contains city 95

to the other moves in iteration 52 . This move is the exact move in iteration 52 and city 88 is not allowed to move to the route that just left until iteration 75 (Fig B.13).

- Iteration 53, the move of city 88 to the route that contains city 67 and at the next-arc is the best move giving 4607.41 miles in the total distance compared to the other moves in iteration 53, but the move of city 88 to this route is declared tabu from iteration 52 to 75, and the total distance in this iteration is not satisfied aspiration criteria since the best total distance found up so far is 4602.99 miles at iteration 52, so this move is ignored. The move of city 32 to the route that contains city 85 is the second best solution providing 4608.19 miles in the total distance; it is selected to be the exact move in iteration 53 (Fig B.14).

- Iteration 108, the best solution (Fig. B.15).

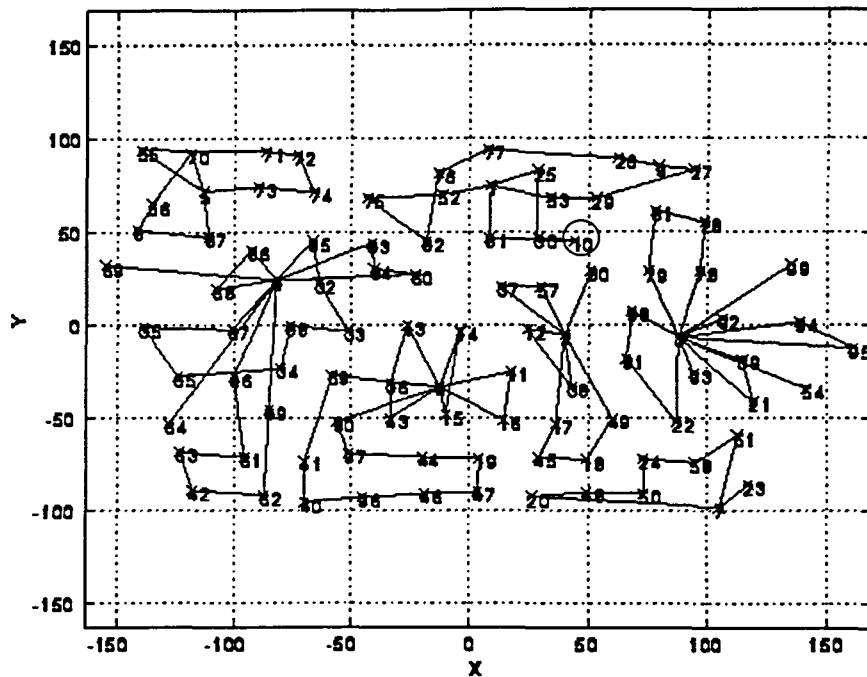


Figure B.6 Iteration 2, move city 10 to the route that contains city 30

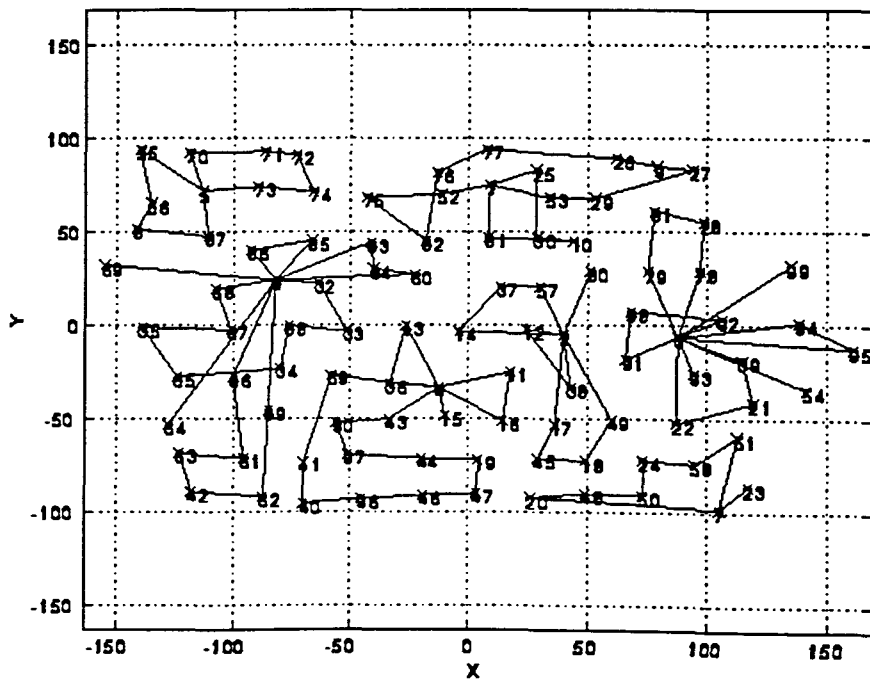


Figure B.7 Iteration 17

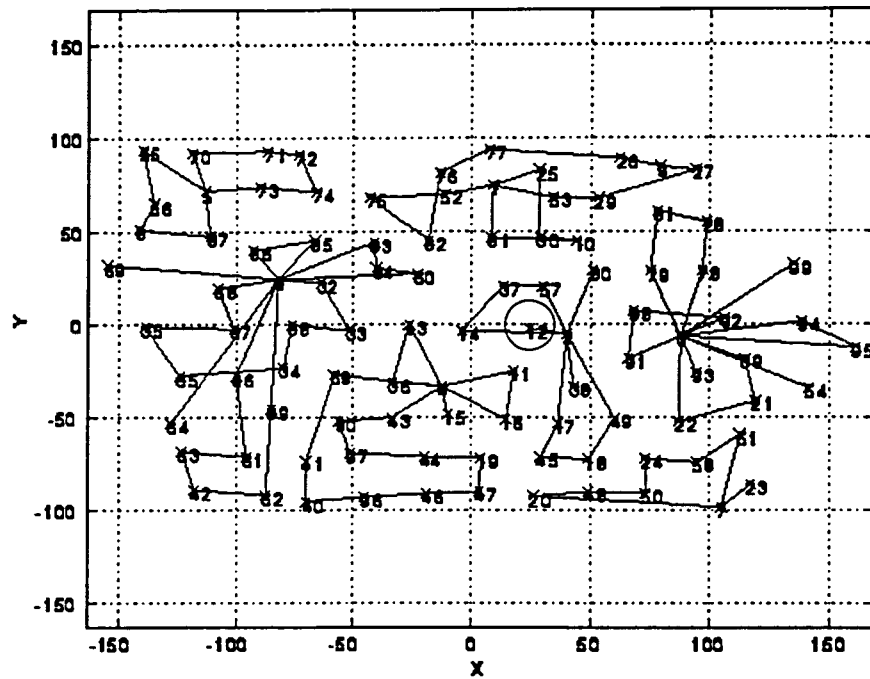


Figure B.8 Iteration 18, construct the new route containing city 12 from depot 2

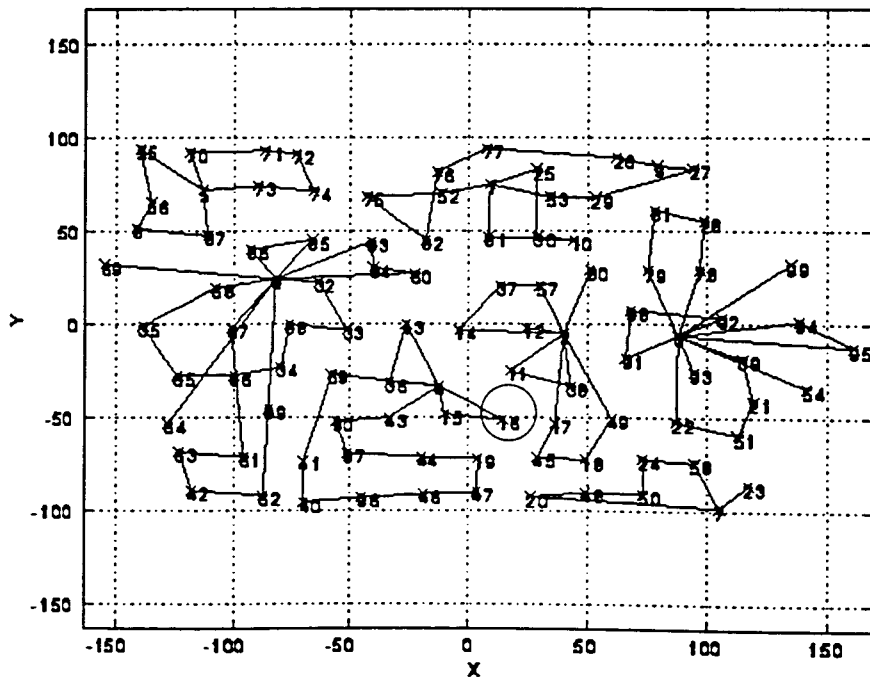


Figure B.9 Iteration 39

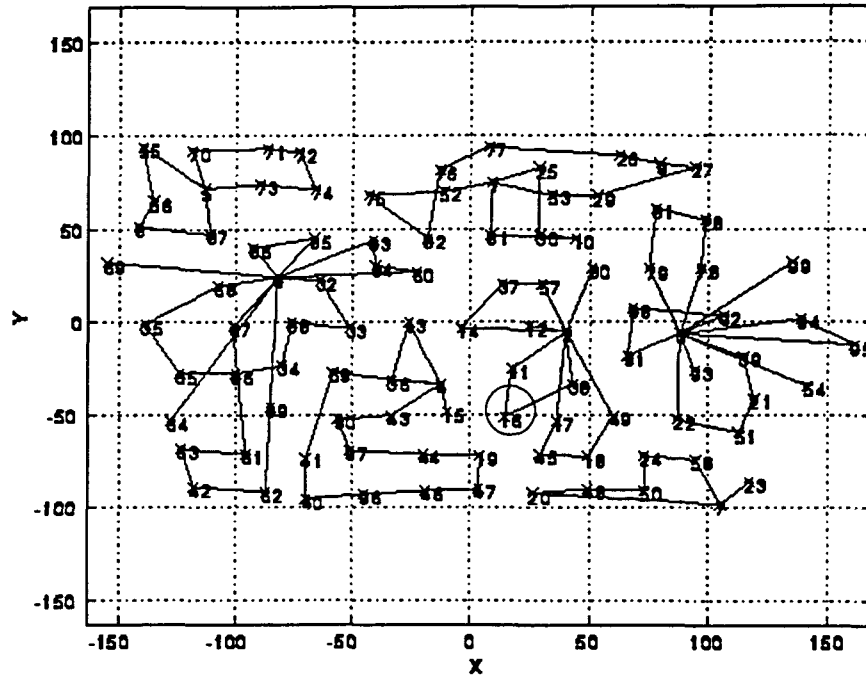


Figure B.10 Iteration 40, move city 16 to the route that contains city 11

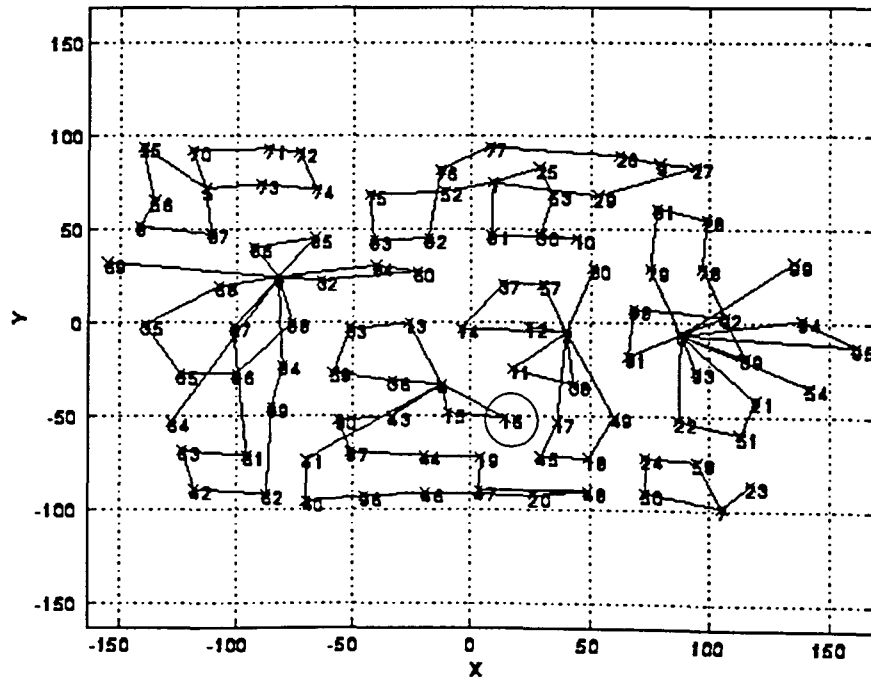


Figure B.11 Iteration 50, move city 16 to the route that contains city 15

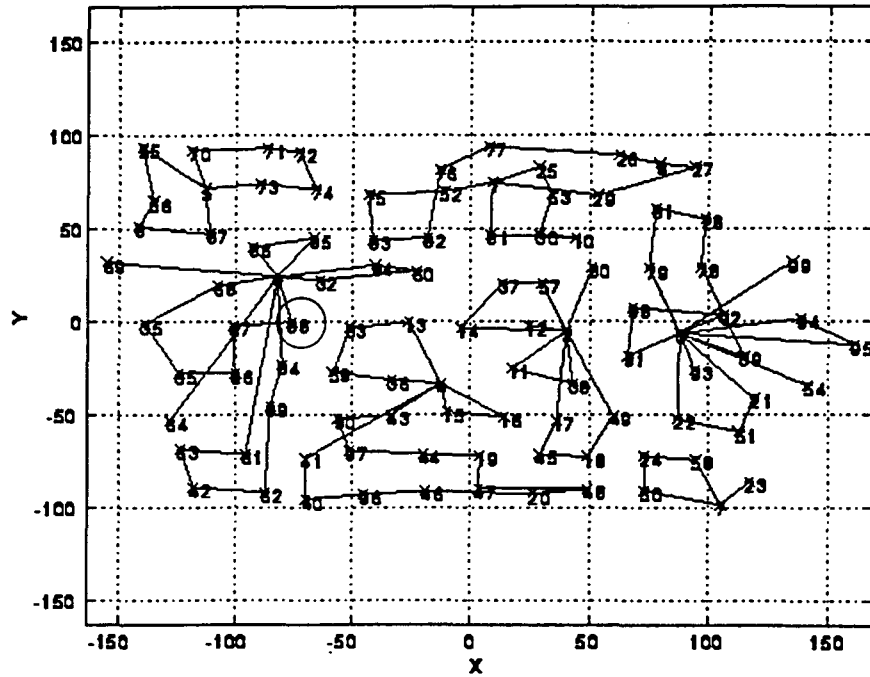


Figure B.12 Iteration 51

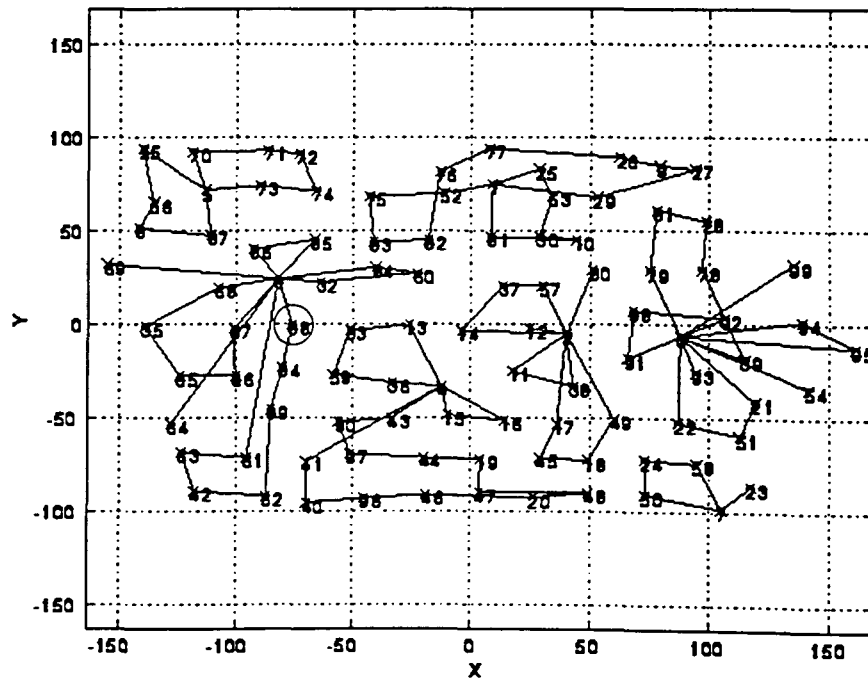


Figure B.13 Iteration 52, move city 88 to the route that contains city 34

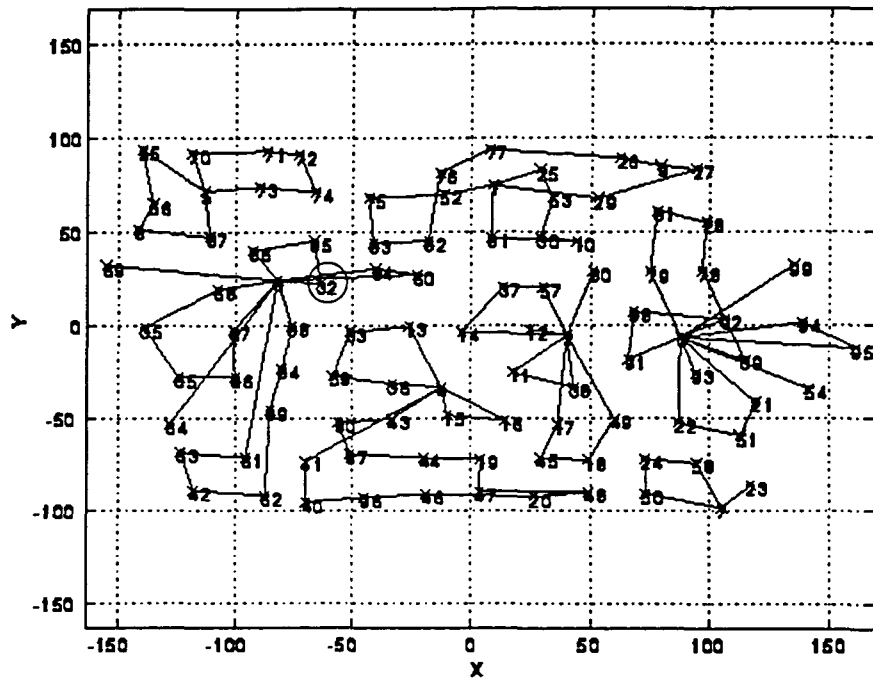


Figure B.14 Iteration 53, move city 32 to the route that contains city 85

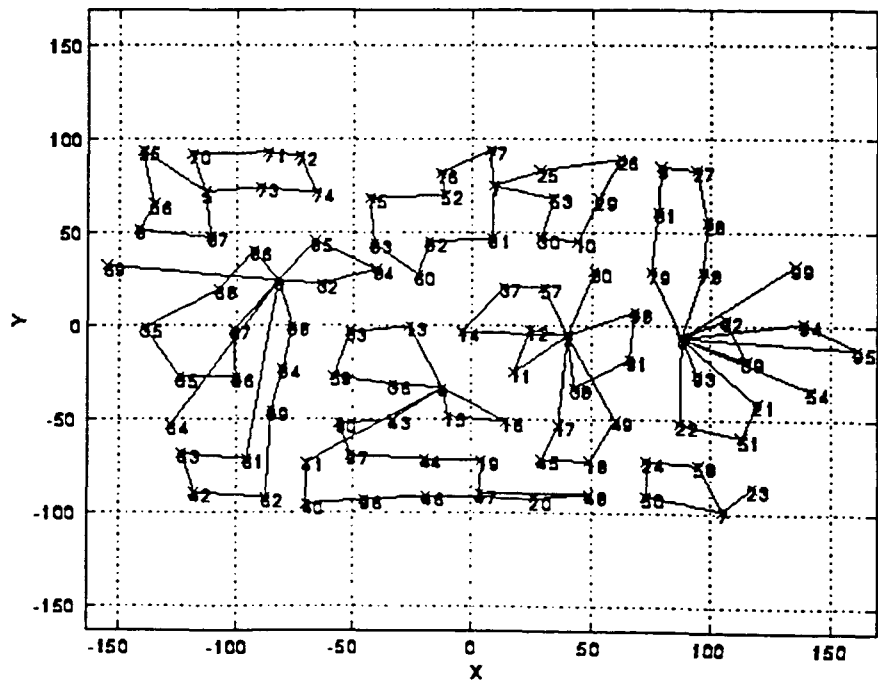


Figure B.15 Iteration 108, the best solution.

APPENDIX C COORDINATE OF CITIES OF TEST PROBLEMS

The numbers shown here represent pick up cities and depots. The numbers from 1 to 4 represent depots and 5 to 54 are the pick up cities

City	X-axis	Y-axis	Quantity (units)
1	20	20	0
2	50	30	0
3	30	40	0
4	60	50	0
5	37	52	7
6	49	49	30
7	52	64	16
8	20	26	9
9	40	30	21
10	21	47	15
11	17	63	19
12	31	62	23
13	52	33	11
14	51	21	5
15	42	41	19
16	31	32	29
17	5	25	23
18	12	42	21
19	36	16	10
20	52	41	15
21	27	23	3

City	X-axis	Y-axis	Quantity (units)
22	17	33	41
23	13	13	9
24	57	58	28
25	62	42	8
26	42	57	8
27	16	57	16
28	8	52	10
29	7	38	28
30	27	68	7
31	30	48	15
32	43	67	14
33	58	48	6
34	58	27	19
35	37	69	11
36	38	46	12
37	46	10	23
38	61	33	26
39	62	63	17
40	63	69	6
41	32	22	9
42	45	35	15
43	59	15	14
44	5	6	7
45	10	17	27
46	21	10	13
47	5	64	11
48	30	15	16
49	39	10	10
50	32	39	5
51	25	32	25
52	25	55	17
53	48	28	18
54	56	37	10

APPENDIX D SELECTION OF THE PARAMETER VALUES

The parameters which are commented here are based on the experience gained in this study of the Iowa recycled paper problem as well as two example problems from the paper of Christofide and Eilon [8].

WIDTH and HEIGHT

The WIDTH represents the width in the X axis while the HEIGHT represents the height of the Y axis of the artificial grid superimposed over the problem. The simplest way to define WIDTH is to set it that equals to the width between the smallest X coordinate and the largest X coordinate of the problem. HEIGHT is the height between the smallest Y coordinate and the largest Y coordinate of the problem.

DIV parameter in artificial grid

The DIV parameter is the number of rectangular elements in the artificial grid. The DIV parameter depends on the characteristic of the problem. The most important factor which can be considered to define the DIV for a problem which is the number of cities in the problem. If the number of pick up points is large, DIV should be large. The smaller the DIV, the larger is the number of arcs to be considered. As in this problem, the number of pick up cities is 99. The DIV must not greater than 8; if it is greater than 8 , some pick up cities are missed in the route construction procedure. On the other

hand, if there are too many rectangular grids, some cities which are on a grid far from the other cities by more than one grid box they are not considered to be neighborhood cities. The route construction procedure in the modified Clarke and Wright algorithm ignores cities that are on the grid box which are isolated from the other grids and those cities will not be on routes.

The numbers ranging from 1 to 8 can be selected to be the DIV. If the DIV is too small, the computation time for constructing routes will increase since the smaller DIV the greater number of neighborhood cities to be considered in a constructing procedure. The best range for this problem was found to be between 4-6. The DIV valued 6 means that there are 36 rectangular grids and the routes which are constructed under this artificial grid give the best distance. In the example problems which contain 50 pick up cities, the greatest DIV is 4. The best DIV values are in the range 2-4. By this observation, the method to find the best DIV parameter is to find the greatest DIV for that problem. The initial trial may be started by $DIV = \text{The number of cities}/10$, such as 10 in the Iowa problem and 5 in the example problems.

In Iowa recycled paper problem, DIV 10 is first tried but some cities are missed in the construction procedure. The DIV 9 is the next trial number and the solution still misses some cities. The DIV 8 is found that it is the greatest DIV that no cities are ignored in the construction procedure. The next step is to try the other DIV values that are smaller than the greatest number and selects the one that gives the best routes.

γ route shape parameter

The higher the γ the greater stress is placed on the distance between pick up city to pick up city more than their relative position distance to depot. The $\gamma = 1$ is the original Clarke and Wright algorithm value. The range of γ should be a value between 0.5 to 1.5; if it outs of this range, it will place too much emphasis on the distance between city to city or their relative distance to depots. In this study, the best range is found to be

the values between 0.9 to 1.1.

Tabu size

The tabu size parameter is the most important parameter in Tabu Search concept since almost every time that tabu size is changed a different solution is obtained. The only way for finding the best tabu size that appeared in the previous research is the trial and error method. The best way is to select the tabu size between $0.9\sqrt{n}$ to $1.1\sqrt{n}$ where n is the number of cities (such as 99) in this problem. In this study, the tabu size is tried between 5 - 30.

Initial α and β for the penalty function

Initial α and β values are the parameters that are used in the penalty function. The α is the weighting parameter for the truck capacity constraint and β is the weighting parameter for the depot capacity constraint. Good initial α and β choices help to speed up the algorithm to reach the optimal solution. Different initial α or β values can reach to the same solution since in this heuristic, the α and β are considered to adjust in every h iterations, so both parameters can simply be selected and tried with values such as 1, 10, 50, 100 or 200. The one that always gives good solution is selected to be the parameter value.

h parameter

The h parameter is the number of iterations that the algorithm will be considered to adjust the α and β . The h value of 6 is used in the examples, Iowa recycled paper and the other two test problems, of this study. Other values of h are also tried; the h between 4-10 is found to be not much different in the results.

Conclusion

Based on this study, all parameters can be divided into 2 class of parameters. The minor effect parameters which are WIDTH, HEIGHT, DIV, γ , initial α and β and h parameter. These parameters have small effect on the solution compared to the major effect parameter which is the tabu size. The tabu size has the major effect on the solution since almost every time that the tabu size is changed a different solution is obtained.

From these observations, the method to control parameters is simply to select the minor effect parameters as were suggested above and focus on varying tabu size.

BIBLIOGRAPHY

- ✓ [1] Y. Agarwal, K. Mathur and H. M. Salkin, "A Set-partitioning-based Algorithm for the Vehicle Routing Problem," *Networks*, Vol. 19, pp. 731-750, 1989.
- [2] M. O. Ball, B. L. Golden, A. A. Assad and L. D. Bodin, "Planning for Truck Fleet Size in the Presence of a Common-carrier Option," *Decision Sciences*, Vol. 14, pp. 103-120, 1983.
- [3] J. Brink, Personal Communication. Packaging Corporation of America, Tama, Iowa, 1996.
- [4] G. Carpaneto and P. Toth, "Some New Branching and Criteria for the Asymmetric Travelling Salesman Problem." *Management Science*, Vol. 26, pp. 736-743, 1980.
- [5] P. J. Cassidy and H. S. Bennett, "TRAMP — A Multi-depot Vehicle Scheduling System," *Operational Research Quarterly*, Vol. 23, No. 2, pp. 151-163, 1972.
- [6] I. M. Chao, B. L. Golden and E. Wasil, "A New Heuristic for the Multi-depot Vehicle Routing Problem that Improves upon Best-known Solutions," *American Journal of Mathematical and Management Sciences*, Vol. 13, Nos. 3&4, pp. 371-401, 1993.
- ✓ [7] N. Christofides, "Vehicle routing." in: E. L. Lawler, J. K. Lenstra, A.H.G. Rinnooy Kan and D. B. Shmoys, (eds.). *The Traveling Salesman Problem. A Guide Tour of Combinatorial Optimization*, Wiley, Chichester, pp. 431-448, 1985.

- [8] N. Christofides and S. Eilon, "An Algorithm for the Vehicle-dispatching Problem," *Operational Research Quarterly*, Vol. 20, No. 3, pp. 309-318, 1969.
- [9] N. Christofides and A. Mingozzi, "State-Space Relaxation Procedures for the Computation of Bounds to Routing Problems," *Networks*, Vol. 11, pp. 145-164, 1981.
- ✓ [10] N. Christofides, A. Mingozzi and P. Toth, "The Vehicle Routing Problem," in: N. Christofides, A. Mingozzi, P. Toth and C. Sandi(eds.), *Combinatorial Optimization*, Wiley, Chichester, England, pp.315-338, 1979.
- ✓ [11] G. Clarke and J. W. Wright, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," *Operations Research*, Vol. 12, No. 4, pp. 568-581, 1964.
- [12] J. Desrosiers, F. Soumis and M. Desrochers, "Routing with Time Windows by Column Generation," *Networks*, Vol. 14, pp. 545-565, 1984.
- [13] M. L. Fisher and R. Jaikumar, "A Generalized Assignment Heuristic for Vehicle Routing," *Networks*, Vol. 11. pp. 109-124, 1981.
- [14] T. Gaskell, "Bases for Vehicle Fleet Scheduling," *Operational Research Quarterly*, Vol.18, pp. 281-295, 1967.
- [15] M. Greau, A. Hertz and G. Laporte, "New Insertion and Postoptimization Procedures for the Traveling Salesman Problem," *Operations Research*, Vol. 40, pp. 1086-1094, 1992.
- [16] M. Gendreau, A. Hertz and G. Laporte, "A Tabu Search Heuristic for the Vehicle Routing Problem," *Management Science*, Vol. 40, No. 10, pp 1276-1290, 1994.
- [17] B. E. Gillett and J. G. Johnson, "Multi-Terminal Vehicle-Dispatch Algorithm," *Omega*, Vol. 4, No. 6, pp. 711-718, 1976.

- [18] B. E. Gillett and L. R. Miller, "A Heuristic Algorithm for the Vehicle-Dispatch Problem," *Operation Research*, Vol. 22, No. 2, pp. 340-349, 1974.
- [19] F. Glover, "Tabu Search: A Tutorial," *Interface*, Vol. 20, No. 4, pp 74-94, 1990.
- [20] F. Glover, "A User's Guide to Tabu Search," *Annals of Operations Research*, Vol. 41, pp. 3-28, 1993.
- [21] F. Glover and M. Laguna, "Tabu Search," in: C.R. Reeves(editor), *Modern Heuristic Techniques for Combinatorial Problems*, Oxford, New York, pp. 70-150, 1993.
- ✓ [22] B. L. Golden, T. L. Magnanti and H. Q. Nguyen, "Implementing Vehicle Routing Algorithms," *Networks*, Vol. 7, pp. 113-148, 1977.
- [23] B. L. Golden and E. A. Wasil, "Computerized Vehicle Routing in the Soft Drink Industry," *Operations Research*, Vol. 35, No. 1, pp. 6-17, 1987.
- [24] M. Hansen, Personal Communication, Iowa Department of Transportation, Ames, Iowa, 1996.
- [25] D. Hess, Personal Communication, Mason City Recycling Center, Mason City, Iowa, 1996.
- [26] J. N. Hooker and N. R. Natraj, "Solving a General Routing and Scheduling Problem by Chain Decomposition and Tabu Search," *Transportation Science*, Vol. 29, No. 1, pp. 30-44, 1995.
- [27] M. Imircian, Personal Communication, Cedar River Paper Company, Cedar Rapids, Iowa, 1996.
- [28] Iowa Department of Natural Resources, *Market Assessment for Plastic, Glass & Paper Recycling in Iowa*, Iowa Department of Natural Resources, Des Moines, Iowa, 1995.

- [29] Iowa Department of Natural Resources, *D. Paper*, Iowa Department of Natural Resources, Des Moines, Iowa, 1996.
- [30] Iowa Department of Natural Resources, *Planning Area Descriptions*, Iowa Department of Natural Resources, Des Moines, Iowa, 1996.
- [31] G. Laporte, "The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms," *European Journal of Operational Research*, Vol. 59, pp. 345-358, 1992.
- [32] G. Laporte, H. Mercure and Y. Nobert, "An Exact Algorithm for the Asymmetrical Capacitated Vehicle Routing Problem," *Networks*, Vol. 16, pp. 33-46, 1986.
- ✓ [33] G. Laporte and Y. Nobert, "Exact Algorithms for the Vehicle Routing Problem," in: S. Martello, G. Laporte, M. Minoux and C. Ribeiro (eds.), *Surveys in Combinatorial Optimization*, North-Holland, Amsterdam, pp. 147-184, 1987.
- [34] G. Laporte, Y. Nobert and D. Arpin, "Optimal Solutions to Capacitated Multidepot Vehicle Routing Problems," *Congressus Numerantium*, Vol. 44, pp. 283-292, 1984.
- [35] G. Laporte, Y. Nobert and S. Taillefer, "Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems," *Transportation Science*, Vol. 22, No. 3, pp. 161-172, 1988.
- [36] S. Lin, "Computer Solutions of the Traveling Salesman Problem," *Bell System Technical Journal*, Vol. 44, pp. 2245-2269, 1965.
- [37] I. H. Osman, "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem," *Annals of Operations Research*, Vol. 41, pp. 421-451, 1993.

- [38] A. P. Punnen and Y. P. Aneja, "A Tabu Search Algorithm for the Resource-Constrained Assignment Problem," *Journal of the Operation Research Society*, Vol. 46, pp. 214-220, 1995.
- [39] O. M. Raft, "A Modular Algorithm for an Extended Vehicle Scheduling Problem," *European Journal of Operational Research*, Vol. 11, pp. 67-76, 1982.
- [40] C. R. Rhyner, L. J. Schwartz, R. B. Wenger and M. G. Kohrell, *Waste Management and Resource Recovery*, Lewis, New York, 1995.
- [41] R. A. Russell, "An Effective Heuristic for the M - Tour Traveling Salesman Problem with Some Side Conditions," *Operations Research*, vol. 25, No. 3, pp. 517-524, 1977.
- [42] F. Semet and E. Taillard, "Solving Real-life Vehicle Routing Problems Efficiently Using Tabu Search.," *Annals of Operation Research*, Vol. 46, pp. 469-488, 1993.
- [43] G. Tchobanoglous, *Integrated Solid Waste Management: Engineering Principles and Management Issues*, McGraw-Hill, New York, 1993.
- [44] F. A. Tillman and T. M. Cain. "An Upperbound Algorithm for the Single and Multiple Terminal Delivery Problem," *Management Science*, Vol. 18, No. 11, pp. 665-683, 1972.
- [45] A. Wren and A. Holliday, "Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points," *Operational Research Quarterly*, Vol. 23, No. 23. pp. 333-344, 1972.
- [46] P. Yellow, "A Computational Modification to the Savings Method of Vehicle Scheduling," *Operation Research Quarterly*. Vol. 21, pp. 281-283, 1970.