UNIVERSITY OF THE PHILIPPINES MANILA
COLLEGE OF ARTS AND SCIENCES
DEPARTMENT OF PHYSICAL SCIENCES AND MATHEMATICS

The Vehicle Routing System of Makati

A Java-based Vehicle Route Finder Application,
Implementing a Two-Phase Heuristic

A Special Problem in partial fulfillment
Of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by

Jerome P. Mesa

April 2003

ACCEPTANCE SHEET

The Special Problem entitled "The Vehicle Routing System of Makati: A Java-based Vehicle Route Finder Application, Implementing a Two-Phase Heuristic" prepared and submitted by Jerome P. Mesa in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

Prof. Ma. Sheila A. Magboo
Adviser

EXAMINERS	APPROVED	DISAPPROVED
1. Prof. Gregorio Baes		
2. Prof. Ma. Sheila A. Magboo		
3. Dr. Vincent Peter C. Magboo		
4. Ms. Celina Umagtang		
5. Mr. Philip D. Zamora		

DATE

Accepted and approved as partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Vincent Peter C. Magboo, M.D., M.S.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences and Mathematics

Harry Engle, M.S.
Chair
Department of Physical Sciences and Mathematics

Marilou G. Nicolas, Ph.D.
Dean
College of Arts and Sciences

ABSTRACT

The Vehicle Routing Problem (VRP) is an important management problem in the field of physical distribution and logistics. Geographically distributed businesses in Makati that involve the delivery of goods to their customers within the city need a plan of route so as to minimize the costs of operations and to maximize their profit. The Vehicle Routing System of Makati is a java application that finds optimal routes from the depot node to the customers' nodes subjected to the vehicle capacity constraints and the customers' demands. The system implements the Two-Phase Heuristic for the Vehicle Routing Problem.

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND OF THE STUDY

Suppose a certain company in Makati would want to deliver quantities of goods to ten customers within the city and it has six delivery vehicles which can carry a fixed number of goods. The company may deploy all six vehicles without a plan of route as long as the ten customers are serviced. This, however, may lead to longer distances traveled by the delivery vehicles and this does not help maximize the company's profit as it makes the costs of operations high. The number of delivery vehicles used and the total distance traveled by these trucks can be minimized so that costs are reduced and profit is maximized. This is an instance of the Vehicle Routing Problem (VRP).

The Vehicle Routing Problem (VRP) is an important management problem in the field of physical distribution and logistics. It can be stated as follows. There are n geographically distributed customers and a fleet of m identical vehicles based at a given depot. The problem is to design m least-cost routes such that each customer is serviced exactly once and the total demand of all the customers in each route does not exceed the capacity of the vehicle that is assigned to execute the route [1]. The minimum cost of the routes is the minimum distance covered by the routes.

The Makati Route Finder System focuses on the route finding problem of the transportation industry. It finds the shortest route between two road junctions in Makati by implementing the Dijkstra's Shortest Path algorithm. This shortest path problem has many variants and one of these is the Vehicle Routing Problem [2].

B. STATEMENT OF THE PROBLEM

Businesses in Makati City concerning the delivery of materials or goods to their customers within the city need some management so that costs of operations will be reduced and profit will be maximized. Some businesses deploy many delivery vehicles to serve the customers without a plan of route. The vehicles can take any route from their starting point to the customers which may result in longer distances traveled. This leads to high costs of operations and low profit. At present, there is no automated system that handles logistics of delivery of goods in Makati.

C. OBJECTIVES

The Vehicle Routing System of Makati aims to do the following:

1. Enable the user to enter the number of vehicles used and the capacity of the vehicles
2. Enable the user to select nodes representing stations
3. Find vehicle routes given a starting road point, n road points, m number of vehicles, quantity of goods ordered at each n road points and the capacity of vehicles such that the total distance traveled is a minimum

D. SIGNIFICANCE OF THE STUDY

A vehicle routing software will provide a decision support tool for geographically distributed businesses in Makati. This will help keep the operating and investment costs as low as possible by providing routes with a minimum total distance traveled.

The software may also be used as a planner in the transport of goods used in: food industry, distribution systems, maintenance service, public service and transport, clothing textile industry, oil and chemical products, animal feed products, third-party transportation, pharmaceutical, electric household appliances and motor couriers [3].

E. SCOPE AND LIMITATIONS

The Vehicle Routing System of Makati uses two types of roads used in Makati Route Finder System to find the vehicle routes. Primary roads consist of streets or highways that allow rapid and efficient movement of large number of vehicles between sections of the city and across the urbanized area. Expressways and freeways are classified as Primary roads like EDSA, South Superhighway and Gil Puyat Avenue. Secondary roads also allow vehicle movement between areas and across the city but these roads provide direct access with the bordering property so movement of vehicles here is slower than in Primary roads [2].

Secondary roads interconnect residential, shopping, employment, and recreational areas. Although this road offers access to the abutting land uses, it does not penetrate identifiable neighborhoods. Examples of these are Kalayaan Avenue, JP Rizal, Jupiter Street, and streets that interconnect residential, commercial and recreational areas. These roads however don't include roads inside private residential areas and other villages. The Vehicle Routing System of Makati excludes roads from inside residential communities in accordance with the existing traffic rules in the city against using these roads for public use [2].

The system allows blocking of road nodes to illustrate instances like road constructions. Blocked nodes are considered impassable by the system.

The Vehicle Routing Problem solved by the system is limited to the Capacitated Vehicle Routing Problem that takes into consideration the vehicle capacity constraints and the quantity of goods associated with each node only. The system has one depot (starting node) only.

The objective in finding the vehicle routes is to minimize the cost of each route. The minimum cost pertains to the minimum distance covered by a route. This distance is determined by getting the sum of the distances between road junctions included in the route. The distance between two road junctions is determined by using the Makati Route Finder System's implementation of the Dijkstra's Shortest Path algorithm. Time is not considered by the system, meaning, the time the vehicles left the starting point and the time that they should return to the starting point are irrelevant. Because of this, the factor of traffic congestion is also not considered.

The system does not allow the user to add his road nodes as it may imply the construction of a new road. The researcher is responsible for the inclusion of all necessary road nodes in the system [2]. Also, the researcher is responsible for the definition of one-way and two-way roads.

II. REVIEW OF LITERATURE

Serna and Bonrostro in their study “MINMAX Vehicle Routing Problems: Application to School Transport in the Province of Burgos (Spain)” applied vehicle routing to school transport in the province of Burgos in Spain. They considered school transport a delicate problem from an economic, political and social viewpoint. They approached the problem from an economic viewpoint: minimize the cost of the routes which should not exceed a fixed time (60 min). A heuristic consisting of two parts was designed: a constructive algorithm to the solution of which a Local Search procedure was applied; and an algorithm based on a tabu search process which is applied to the solution obtained in the first part [4].

In another study by Xu and Kelly, “A Network-Flow Based Tabu Search Heuristic For the Vehicle Routing Problem”, they developed a new local search approach based on a network flow model that is used to simultaneously evaluate several customer ejection and insertion moves. This approach was used together with a direct customer swap procedure to solve the vehicle routing problem. Tabu search was incorporated into the procedure to overcome local optimality. More advanced issues were developed such as the intensification and diversification strategies to enhance the basic tabu search algorithm [5].

“A Savings Based Method for Real-Life Vehicle Routing Problems” by Poot and Kant described a Savings Based algorithm for the Extend Vehicle Routing Problem. This algorithm was compared with a Sequential Insertion algorithm on real-life data. Besides the traditional quality measures such as total distance traveled, total number of customers used, we also considered different non-standard quality measures to evaluate the "visually attractiveness" of the plans [6].

Rego in “Node Ejection Chains for the Vehicle Routing Problem: Sequential and Parallel Algortihms” presented a tabu search algorithm for the vehicle routing problem under capacity and distance restrictions. The neighborhood search is based on compound moves generated by a node-ejection chain process. During the course of the algorithm, two types of neighborhood structures were used and crossing infeasible solutions was allowed. Then, a parallel version of the algorithm, which exploits the moves' characteristics, is described. Parallel processing was used to explore the solution space more extensively and to accelerate the process [7].

Review of Systems

San Miguel Corporation created its own vehicle routing software with the aid of the company's Operations Research Department. Finished beer products are transported from three plants to 14 warehouses and are distributed to retailers using approximately 200 route trucks. Distribution from warehouses is done using conventional routes, distributors and pre-sell/ delivery routes. The project was initially done for sales districts in Metro Manila and eventually ran for all the districts throughout the country. For the pilot run, the number of routes required to serve the area was decreased by at least 43 percent [8].

Nemsys Network Models and System optimization created Routemate version 2 which is a vehicle routing and scheduling computer package designed to support the dispatcher in planning the routes and schedules of a mixed vehicle fleet. Routemate objective is the minimization of distribution costs under complex operational constraints. Advanced optimization techniques produce operative solution for a wide variety of distribution problems [9].

A typical application of vehicle routing is postal delivery. Routing of postal operations is a complex task. GeoRoute 5 of GIRO Inc. is a software created to solve the complex activity of mail delivery addressing many particularities and complexities of the said activity [10].

Advanced Delivery Systems, Inc. in U.S. uses ArcLogistics Route to run a contract delivery service for some 20 national and regional furniture and appliance retailers. The program is a stand-alone GIS (Geographic Information System) desktop software for solving vehicle routing and scheduling problems. It builds delivery routes based on vehicle characteristics, customer orders, and actual network drive times rather than straight distances [11].

Emet Software Products designed the ILOG Dispatcher and it is the transportation industry's most powerful optimization engine for creating vehicle routing and personnel scheduling and dispatching applications. Routing and dispatching applications reduce costs and improve customer service by optimizing vehicle routes - arming transportation firms with a competitive advantage. ILOG Dispatcher slashes application development time, accelerating deployment. The vehicle routing feature of the software targets businesses that deliver products to warehouses, sales points or directly to consumers, including manufacturing, third-party logistics, food and trade industries [12].

SpatialFX application provides services like Map Display and Interaction, Geocoding, Reverse Geocoding and Routing which are optimized for high speed computation and delivery to the client. The vehicle routing service lets users specify a start point and one or more destinations, and then see the route appear on a map, showing the "shortest distance" or "least time" path from the start point to the destination(s) along with textual driving directions [13].

The Makati Route Finder System provides an interactive map and a route finding functionality which implements the Dijkstra's Shortest Path Algorithm. The system lets the user choose the starting and end nodes and it generates the shortest route between these nodes. The system also provides the user with information about the ins and outs of the Makati City [2].

Mastrolilli cited all known techniques for solving the Vehicle Routing Problem [14]:

Exact Approach

- (up to 100 nodes) Branch and bound (Fisher 1994)

Heuristics

- Clark and Wright (1964)
- Hierarchical Approach (split + TSP)
- Fisher & Jaikumur (1981)
- Taillard (1993)
- Multi-route Improvement Heuristics
- Kinderwater and Savelsbergh (1997)

MetaHeuristics

- Tabu search, Rochat and Taillard (1995)
- Constraint Programming, Shown (1998)
- Tabu search Kelly and Xu (1999)
- Granular Tabu, Toth & Vigo (1998)
- Ant System, Gambardella & al. (1999)

III. THEORETICAL FRAMEWORK

Graph

A graph is a collection of vertices and edges. Vertices are simple objects that can have names and other properties; an edge is a connection between two vertices [15].

As shown in Figure 1, a path from a vertex x to y in a graph is a list of vertices in which successive vertices are connected by edges in the graph. A simple path is a path in which no vertex is repeated. Figure 2 illustrates a cycle as a path that is simple except that the first and the last vertices are the same [15].

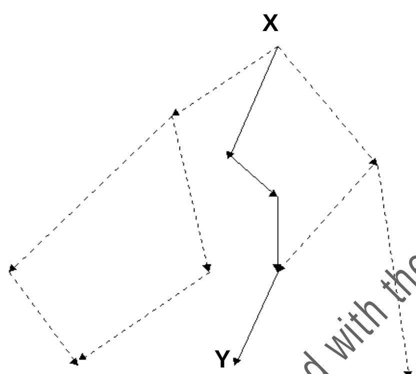

Figure 1. Illustration of a path

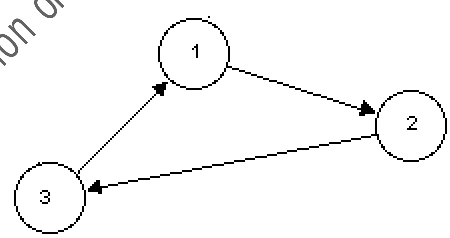

Figure 2. Illustration of a cycle

(1,2,3,1) is a cycle

Graphs may be classified as either directed or undirected. An undirected graph, as seen in Figure 3, is a finite set of vertices with a finite set of edges.. Both sets may be empty in which case it is called an empty graph. Each edge is associated with two vertices and the order of the vertices is unimportant [16]. In directed graphs, as shown in Figure 4, edges are “one-way”: an edge may go from x to y but not from y to x [15].

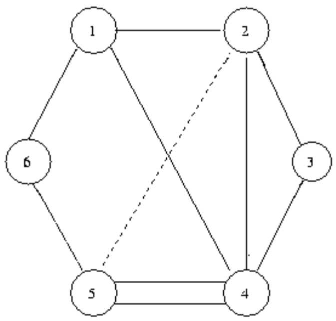

Figure 3. An Illustration of an undirected graph

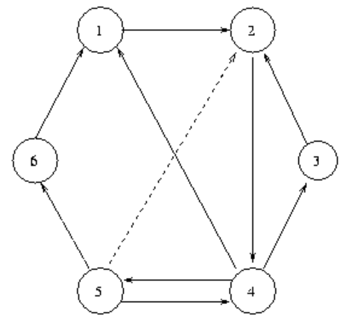

Figure 4. An Illustration of a directed graph

Weighted graphs (Figure 5) have integers (weights) assigned to each edge to represent distances or costs. Directed weighted graphs are sometimes called networks [15].

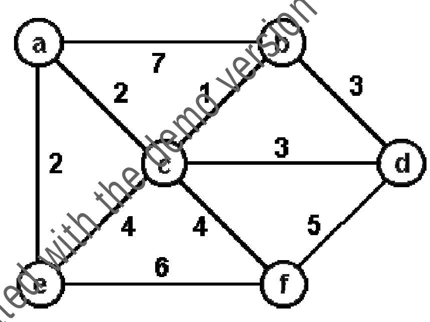

Figure 5. An Illustration of a weighted graph

Capacitated Vehicle Routing Problem

The capacitated vehicle routing problem (CVRP) is described as follows: n customers must be served from a unique depot. Each customer asks for a quantity qi of goods (i = 1,..., n) and a vehicle of capacity Q is available to deliver goods. Since the vehicle capacity is limited, the vehicle has to periodically return to the depot for reloading. In the CVRP, it is not possible to split customer delivery. Therefore, a CVRP solution is a collection of routes where each customer is visited only once and the total route demand is at most Q [14]. A route is a tour that begins at the depot, traverses a subset of the customers in a specified sequence and returns to the depot.

As illustrated in Figure 6, the CVRP may be stated from a graph theoretical point of view as follows: Let G = (C,L) be a complete graph with node (vertex) set C = (co, c1, c2,..., cn) and arc (edge) set L = (ci, cj): ci, cj ª C, i j. In this graph model, co is the depot and the other nodes are the customers to be served. Each node is associated with a fixed quantity qi of goods to be delivered (a quantity qo = 0 is associated to the depot co) to n customers by a fixed m number of vehicles. To each arc (ci, cj) is associated a value tij representing the distance between ci and cj. The goal is to find a set of routes of minimum total distance. Each route starts from and terminates at the depot co, each node ci (i = 1,..., n) must be visited exactly once, and the quantity of goods to be delivered on a route should never exceed the vehicle capacity Q [14].

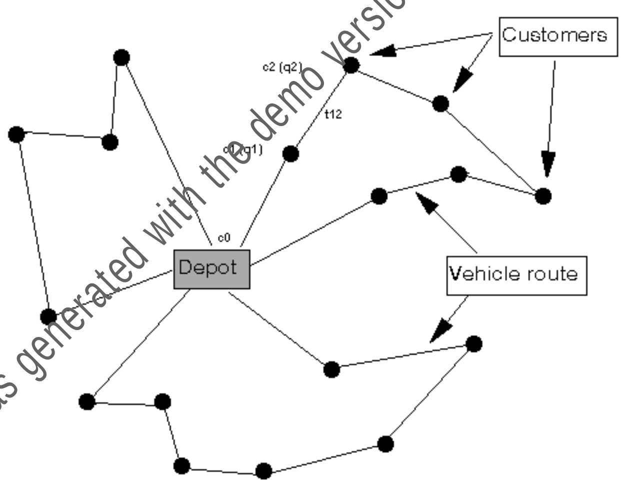

Figure 6. An Illustration of the Capacitated Vehicle Routing Problem

VRP Heuristics

Effective and efficient heuristics have been designed to solve the hard Vehicle Routing Problem (VRP). Early heuristics for the VRP concentrated on constructing routes based on cost saving criteria, as represented by Clarke and Wright (1964). These heuristics work fast, but their solutions can be easily improved by improvement procedures. More recent heuristics, employing the advanced metaheuristics (e.g., tabu search), such as Osman (1993), Taillard (1993), Gendreau, Hertz and Laporte (1994), Rochet and Taillard (1995), Stewart, Kelly and Laguna (1995), Rego and Roucairol (1996), and Xu and Kelly (1996), have shown dramatically improved performance by obtaining near optimal solutions for most benchmark problems [17].

One of the most competitive heuristics is the tabu search. Tabu search has been superior on VRPs since it has been studied and improved a lot since its introduction and a lot of VRP research has been done on the tabu search [17]. The heuristic comes in many forms but the central idea is the use of adaptive memory to keep track of information related to the exploration process [18].

In two-phase heuristics, the problem is decomposed into its two natural components: clustering of vertices into feasible routes and actual route construction, with possible feedback loops between the two stages. Two-phase heuristics can be divided into two classes: cluster-first, route-second methods and route-first, cluster-second methods. In the first case, vertices are first organized into feasible clusters, and a vehicle route is constructed for each of them. In the second case, a tour is first built on all vertices and is then segmented into feasible vehicle routes [20]. A set-partitioning-based heuristic for VRP makes use of the advantages of simple local search to produce initial solutions that are selected and assembled into superior solutions. Computational results for this algorithm with fourteen benchmark problems reveal its competitiveness in solving the VRP [17].

The Vehicle Routing System of Makati will use a two-phase heuristic to find optimal routes. The heuristic that will be used can be classified as cluster-first, route-second method. The Generation phase of the heuristic will generate the initial pool of routes from which the Integration phase will select routes that will be assembled into a solution (Figure 7). The

Generation phase will use an improved Clarke and Wright Savings Heuristic and the Integration phase will use a generic tabu search.

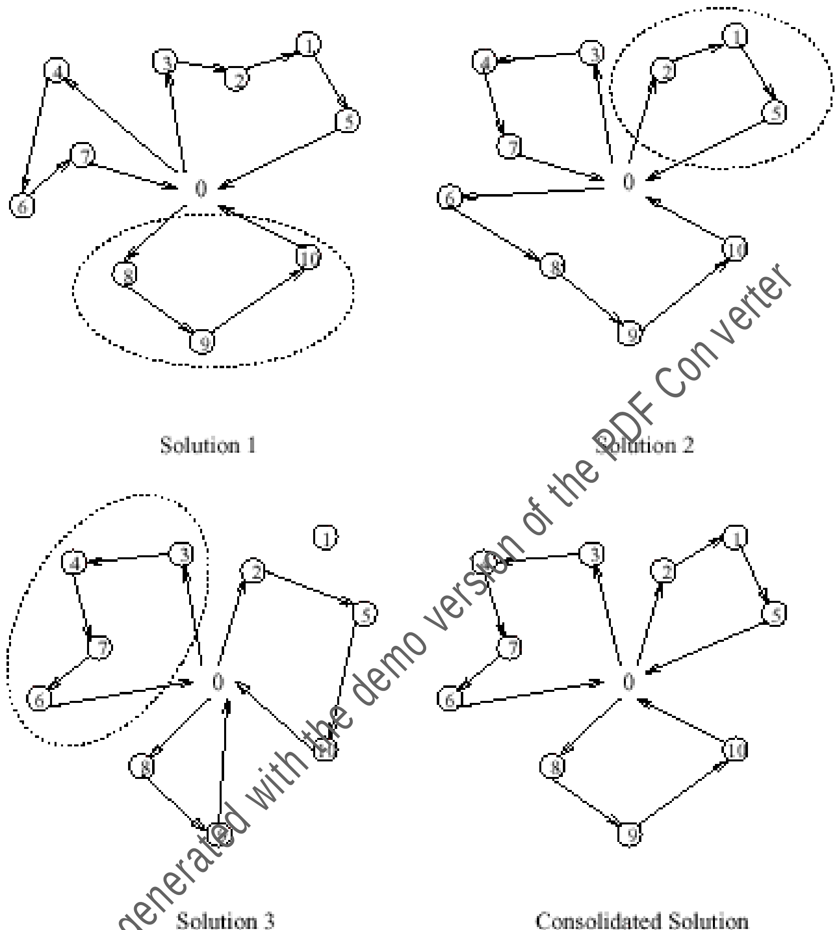

Figure 7. An Illustration of the Two-Phase Heuristic

Clarke and Wright Savings Heuristic

Clarke and Wright Savings Heuristic's basic idea is very simple. Consider a depot D and n demand points. Suppose that initially the solution to the VRP consists of using m vehicles and dispatching one vehicle to each one of the n demand points (Figure 8). The total route length of this solution is $2\sum_{i=1}^{n} d(D,i)$. If we now use a single vehicle to serve two points, say i or j, on a single trip (Figure 9), the total distance traveled is reduced by the amount s(i,j) = d(D,i) + d(D,j) – d(i,j). The quantity s(i,j) is known as the savings resulting from combining points i and j into a single tour. The larger s(i,j) is, the more desirable it becomes to combine i and j in a single tour. However, i and j cannot be combined if in doing so the resulting tour violates one or more of the constraints of the VRP.

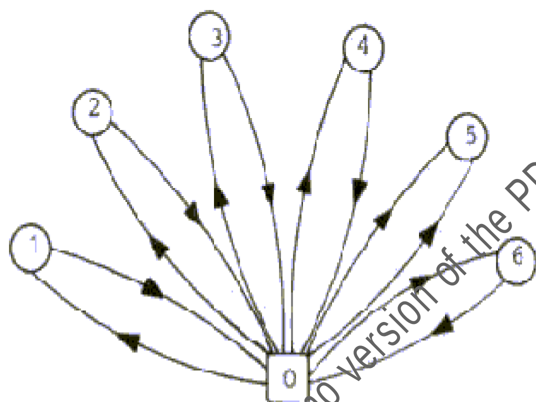

Figure 8. Initial Solution of the Clarke-Wright Savings Algorithm

Figure 9. Combining points i and j into a single tour

Paessens proposed the weighted savings measure as follows [17]:

$$S_{ij} = d_{oi} + d_{oj} – g*d_{ij} + f*| d_{oi} - d_{oj} |$$

Where 0 < g   3 and 0< f   1. g is used to place emphasis on the distance between the customers, rather than their positions relative to the depot. There are instances that customers distant from a chosen customer are often preferred to closer customers because of their greater savings value. The $f*| d_{oi} - d_{oj} |$ component encourages the matching of customers whose interdepot distances differ markedly. Suppose i is a customer distant from the depot, j is a customer close to the depot, and è$_i$ è$_j$ where è$_{i(\text{ or } j)}$ is the angle of customer i (or j) relative to the depot. This should be a good pairing, but s_{ij} will be small. The $f*| d_{oi} - d_{oj} |$ component corrects for this [17].

This modified function performs better than the classic Clarke and Wright function if the method is run multiple times using different f and g values. This produces differing solutions, thus increasing the chance of finding high quality solutions [17].

Initially, pairs of g and f values are obtained from the given range. For each pair of f and g values, their values are randomized by setting g=U(g-0.05,g+0.05) and f=U(f-0.05,f+0.05) before calling the savings method, where U(a,b) designates the uniform distribution between two real numbers a and b [17].

Tabu Search

Tabu search explores the solution space by moving at each iteration from a solution s to the best solution in a subset of its neighborhood. There may be times that the current solution may deteriorate from one iteration to the next. Thus, to avoid cycling, solutions possessing some attributes of recently explored solutions are temporarily declared tabu or forbidden [18].

The basic tabu search approach is to search for globally optimal solutions with the assistance of adaptive memory procedures. At each iteration, candidate neighborhood moves are evaluated that lead from the current solution to a new solution. Restrictions are imposed to classify certain moves tabu and thus forbid their selection. Conventionally, a non-tabu move with the highest evaluation is selected. The algorithm stops after a predefined number of iterations or a similar stopping rule is satisfied. Intensification and diversification strategies are highly important components of tabu search. More specifically, intensification strategies encourage move combinations by including/excluding good/bad solution attributes found historically, while diversification strategies drive the search into unexplored solution space [18].

If feasible solutions are found in the Generation phase, our TS heuristic starts with the best feasible solution found in that phase. If no feasible solution is found, we build a solution consisting of m routes in the pool as follows. Initially, mark all customers "uncovered." We select a route from the pool with the maximum number of uncovered customers, and mark the uncovered customers in the route "covered." We repeat the process until m routes are selected [17].

The neighborhood move is defined as the swap operation that replaces the route in the current solution by a route not in the current solution [17].

Consider a swap move of routes A and B (assuming that A is currently in the solution and B is not). If A and B are disjoint, i.e., there are no common customers between A and B, then the swap move is equivalent to removing all customers appearing in B and inserting all customers in A. This move is not desirable since it destroys a large amount of structure contained in the previous solution. Thus, the ideal swap move should be such that A and B do not differ too much in terms of customers they include. The similarity between two routes guarantees that B inherits most of A's structure. Based on this premise, we use the candidate list strategy to restrict the algorithm's attention to exchanging similar routes [17].

We define a similar route as follows. For route A, B is a similar route of A if the number of customers not in both routes does not exceed a predefined limit x. Then, let d = $\sum_{i=1}^{n} \frac{qi}{n}$ where n is the number of customers (or demand points), and D_A and D_B be the capacity of routes A and B respectively. Then in addition to the condition stated above, we add the condition $|D_A - D_B|$ (x+1)d. This new restriction can greatly narrow the search range as follows. We first find the first route in the pool with capacity no less than D_A –(x+1)d using binary search, and find the last route with capacity no more than D_A+ (x+1)d. Then we directly compare route B with each route in this range to see if the first similarity condition is satisfied [17].

Consider a swap move. When we exchange route A in the solution with route B not in the solution, we impose tabu restriction on A and do not allow it to move back for a certain number of iterations. We also forbid B to be removed from the solution for a period of time. After a swap move, there may be multi-covered and uncovered customers in the solution. Patching procedures where multi-covered customers are removed and uncovered customers are inserted in the solution solve this. Removing of multi-covered customers is done by finding the route that would produce the maximum distance savings if the customer is removed. Uncovered customers are inserted into routes in the solution while simultaneously satisfying the vehicle capacity. This is done by finding the route (or routes) where the uncovered customer can be inserted without violating the vehicle capacity constraint. The least-cost insertion position is then determined and the insertion executed. This procedure may not insert all uncovered customers into the routes which indicates that the solution is infeasible [17].

DEFINITION OF TERMS

Road points/nodes- these represent the road intersections on the map, where two road nodes or intersections can form a road segment or street. These are the points on the map where the user can choose his start and destination points [2].

Map views – the user may view several categories of map items at a time. He may filter out categories and even view the Land area layer excluding the map items. Zones may be viewed by filtering categories and displaying one category at a time.

i. Land Area Layer – is the 2D map itself which depicts the Makati land area along with its road network

ii. Output Routes Layer – where the output paths are drawn

Metaheuristics - refers to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality. The heuristics guided by such a meta-strategy may be high level procedures or may embody nothing more than a description of available moves for transforming one solution into another, together with an associated evaluation rule.

A heuristic is an algorithm for solving a problem which gives a good result but cannot guarantee an optimum. A metaheuristic is a heuristic which has wide applicability [19].

IV. DESIGN AND IMPLEMENTATION

Two-Phase Heuristic for the Vehicle Routing Problem

This heuristic has two phases, namely, the Generation phase which is used to produce the initial pool of routes, and the Integration phase which is used to find a high quality solution by solving the set-partitioning model [17].

Generation Phase

The generic procedure for the Generation phase is as follows:

Route Generation

A. (Initialization)

route_pool = ø

B. (Generate Routes)

Call a heuristic to produce $R_1,\dots,R_k$ where K is the resulting number of routes (vehicles) required to cover all customers. (Each route is feasible and satisfies all constraints). If K < m, where m is the number of vehicles available, the solution is feasible.

For this we will use Clarke and Wright savings algorithm with a modification of the weighted savings measure. The modification made by Gaskell, Yellow and Paessens is as follows:

(Eq. 1) $S(i,j) = d_{oi} + d_{oj} - g*d_{ij} + f*|d_{oi} - d_{oj}|$ where $0 < g$ 3 and $0 < f$ 1

Use random values of g and f for several iterations.

Step 1: Calculate the savings s (i,j) for every pair (i,j) of demand points.

Step 2: Rank the savings s(i,j) and list them in descending order of magnitude. This creates the "savings list." Process the savings list beginning with the topmost entry in the list (the largest s(i,j)).

Step 3: For the savings s(i,j) under consideration, include link (i,j) in a route if no route constraints will be violated through the inclusion of (i,j) in a route, and if either:

a. neither I nor j have already been assigned to a route, in which case a new route is initiated including both i and j.
b. Or, exactly one of the two points (i or j) has already been included in an existing route and that point is not interior to that route (a point is interior to a route if it is not adjacent to the depot D in the order of traversal of points), in which case the link (i,j) is added to that same route
c. Or, both i and j have already been included in two different existing routes and neither points is interior to its route, in which case the routes are merged.

Step 4: If the edges' list has not been exhausted, return to Step 3, processing the next edge; otherwise stop. The solution consists of the routes generated in Step 3. (Any points that have not been assigned to a route during Step 3 must each be served by a vehicle route that begins at the Depot D, visits the unassigned point and returns to D).

C. (Route Manipulation)

For i = 1 to K do

Check if R_i duplicates any route in route_pool.

If "no" then do

route_pool = route_pool U {R_i}

if |route_pool| = N, then terminate.

Goto B.

Integration Phase

Let

T = best feasible solution found in the Generation phase

r = route included in T

L = candidate list (similar list) of a route

r' = best route found in the candidate list. The best route is the route whose distance is less than any other distances of other routes.

k = counter variable (number of routes in solution T)

The essential characteristics of a Tabu Search procedure needed in the Integration phase is stated as follows:

A. Tabu search heuristic starts with the best feasible solution T found in the Generation phase. If no feasible solution is found, a solution consisting of m routes is built.
B. Choose a route r in the solution.
C. While k< number of routes in solution T, set k=k+1 and generate a candidate list L for a route r ª T
D. Find a best route r' ª L
E. Check the tabu list. If r' is in tabu list return to step C
F. Swap the routes r and r' if the distance of r' is less than the distance of r.
G. Update tabu list.
H. Let r be another route in solution T.
I. If k= number of routes in solution T then stop

J. Patch multi-covered and uncovered customers.

Example:

Solve the vehicle routing problem with 0 as the depot, (1,2,3,4,5,6) as the customers' nodes with (4,2,5,4,4,3) as the quantities demanded by the respective nodes, 3 vehicles, each with capacity 14.

Solution:

Let (g_1=0.6, f_1=0.4), (g_2=0.3, f_2=0.9), (g_3=1.2, f_3=0.1) and (g_4=2, f_4=1) whose values are taken randomly from Eq.1. We randomize each pair of values by setting g=U(g-0.05,g+0.05) and f=U(f-0.05, f+0.05) where U(a,b) designates a uniform distribution between the real numbers a and b. We obtain the following random values: