Vehicle Routing with Time Windows using Genetic Algorithms

Sam R. Thangiah

Artificial Intelligence and Robotics Laboratory
Computer Science Department
Slippery Rock University
Slippery Rock
PA 16057
U.S.A.

Abstract - In vehicle routing problems with time windows (VRPTW), a set of vehicles with limits on capacity and travel time are available to service a set of customers with demands and earliest and latest time for servicing. The objective is to minimize the cost of servicing the set of customers without being tardy or exceeding the capacity or travel time of the vehicles. As finding a feasible solution to the problem is NP-complete, search methods based upon heuristics are most promising for problems of practical size. In this paper we describe GIDEON, a genetic algorithm heuristic for solving the VRPTW. GIDEON consists of a global customer clustering method and a local post-optimization method. The global customer clustering method uses an adaptive search strategy based upon population genetics, to assign vehicles to customers. The best solution obtained from the clustering method is improved by a local post-optimization method. The synergy a between global adaptive clustering method and a local route optimization method produce better results than those obtained by competing heuristic search methods. On a standard set of 56 VRPTW problems obtained from the literature the GIDEON system obtained 41 new best known solutions.

Key words: Genetic Algorithms, Vehicle routing, Time Windows, Multi-Objective.

? . 1 Introduction

The problem we address is the Vehicle Routing Problem with Time Windows (VRPTW). The VRPTW involves routing a fleet of vehicles, with limited capacities and travel times, from a central depot to a set of geographically dispersed customers with known demands within specified time windows. The time windows are two-sided, meaning that a customer must be serviced at or after its earliest time and before its latest time. If a vehicle reaches a customer before the earliest time it results in idle or waiting time. A vehicle that reaches a customer after the latest time is tardy. A service time is also associated with servicing each customer. The route cost of a vehicle is the total of the traveling time (proportional to the distance), waiting time and service time taken to visit a set of customers.

The VRPTW arises in a wide array of practical decision making problems. Instances of the VRPTW occur in retail distribution, school bus routing, mail and newspaper delivery, municipal waste collection, fuel oil delivery, dial-a-ride service and airline and railway fleet routing and scheduling. Efficient routing and scheduling of vehicles can save government and industry millions of dollars a year. The current survey of vehicle routing methodologies are available in [2] and [12]. Solomon and Desrosiers[26] provide an excellent survey on vehicle routing with time windows.

In this paper we describe GIDEON, a genetic algorithm heuristic for solving the VRPTW. GIDEON is a cluster-first route-second method that assigns customers to vehicles by a process we call Genetic Sectoring and improves on the routes using a local post-optimization procedure. The Genetic Sectoring method uses a genetic algorithm to adaptively search for sector rays that partition the customers into sectors or clusters served by each vehicle. It ensures that each vehicle route begins and ends at the depot and that every customer is serviced by one vehicle. The solutions obtained by the Genetic Sectoring method are not always feasible and are improved using local post-optimization methods by moving customers between clusters.

The paper is arranged in the following form. Section 2 gives a mathematical formulation of the VRPTW. Section 3 gives a description of the GIDEON system. Section 4 describes the results of computational testing on a standard set of VRPTW problems obtained from the literature. Section 5 is the computational analysis of the solutions obtained from the GIDEON system and with respect to competing heuristics. Section 6 contains the summary and concluding remarks.

? . 2 Mathematical Formulation for the VRPTW

The notation and expressions used in the model are useful in explaining the genetic search. We present a mixed -integer formulation of the vehicle routing problem with time window constraints. Our formulation is based upon the model defined by Solomon[28]. The following notations will

help in the description of the GIDEON system. In the mixed integer formulation the indices i,j=1,...,N and k=1,...,K.

Parameters:

K = number of vehicles.
N = number of customers (0 denotes the central depot).
T = maximum travel time permitted for a vehicle.
Ci = customer i.
C_0 = the central depot.
V_k = vehicle route k.
O_k = total overload for vehicle route k.
T_k = total tardiness for vehicle route k.
D_k = total distance for a vehicle route k.
R_k = total route time for a vehicle route k.
Q_k = total over-route time for a vehicle route k.
t_{ij} = travel time between customer i and j (proportional to the Euclidean distance).
v_k = maximum capacity of vehicle k.
t_i = arrival time at customer i.
f_i = service time at customer i.
w_i = waiting time before servicing customer i.
e_i = earliest time allowed for delivery to customer i.
l_i = latest time allowed for delivery to customer i.
q_{ik} = total demand of vehicle k until customer i.
r_{ik} = travel time of vehicle k until customer i (including service time and waiting time).
p_i = polar coordinate angle of customer i.
s_i = pseudo polar coordinate angle of customer i.
F = fixed angle for Genetic Sectoring, $\text{Max}[p_i,...,p_n]/2K$, where n = 1,...,N.
B = length of the bit string in a chromosome representing an offset, B=3.
P = population size of the Genetic Algorithm, P=50.
G = number of generations the Genetic Algorithm is simulated, G=1000.
E_k = offset of the k^{th} sector, i.e, decimal value of the k^{th} bit string of size B.
I = a constant value used to increase the range of E_i.
S_k = seed angle for sector k.
S_0 = initial seed angle for Genetic Sectoring, S_0=0.
α = weight factor for the distance.
β = weight factor for the route time.
η = penalty weight factor for an overloaded vehicle.
γ = penalty weight factor for exceeding maximum route time in a vehicle route.
κ = penalty weight factor for the total tardy time in a vehicle route.

Variables:

$$y_{ik} = \begin{cases} 1, \text{ if i is serviced by vehicle k} \\ 0, \text{ otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1, \text{ if the vehicle k travels directly from i to j} \\ 0, \text{ otherwise} \end{cases}$$

The mixed integer formulation for the vehicle routing problem is stated as follows:

$$\textbf{(VRPTW)} \quad \text{Min} \quad \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K} c_{ijk}\, x_{ijk} \tag{2.1}$$

Subject to:

$$\sum_{i=0}^{N} q_{ik} y_{ik} \le v_k, \quad k = 1,\ldots,\text{K}. \tag{2.2}$$

$$\sum_{i=0}^{N}\sum_{j=0}^{N} y_{ik}\,(t_{ij} + f_i + w_i) \le R_k \quad k = 1,\ldots\text{K}. \tag{2.3}$$

$$y_{ik} = 0 \text{ or } 1; \quad i = 0,\ldots,1; \; k = 1,\ldots,\text{K} \tag{2.4}$$

$$x_{ijk} = 0 \text{ or } 1; \quad i,j = 1,\ldots,\text{N}; \; k = 1,\ldots,\text{K}. \tag{2.5}$$

$$\sum_{k=1}^{K} y_{ik} = \begin{cases} K, i = 0 \\ 1, i = 1, \ldots, \text{N} \end{cases} \tag{2.6}$$

$$\sum_{j=0}^{N} x_{ijk} = y_{jk}, \; j = 0,\ldots,\text{N}; \quad k = 1,\ldots,\text{N}. \tag{2.7}$$

$$\sum_{j=0}^{N} x_{ijk} = y_{ik}, \; i = 1,\ldots,\text{N}; \; k = 1,\ldots,\text{N}. \tag{2.8}$$

$$t_j \ge t_i + s_i + t_{ij} - (1 - x_{ijk}) \cdot T, \quad i,j = 1,\ldots\text{N}, \quad k = 1,\ldots,\text{K}. \tag{2.9}$$

$$f_i \le t_i < l_i, \; i = 1,\ldots,\text{N}. \tag{2.10}$$

$$t_i \ge 0, \; i = 1,\ldots,\text{N}. \tag{2.11}$$

The objective is to minimize the total vehicle travel time (2.1) subject to vehicle capacity, travel time and arrival time feasibility constraints. A feasible solution for the VRPTW services all the customers without the vehicle exceeding the maximum capacity of the vehicle (2.2) or the travel time of the vehicle (2.3). In addition, each customer can be served by one and only one vehicle (2.6). Travel time for a vehicle is sum total of the distance travelled by the vehicle including the waiting and service time. Waiting time is the amount of time that a vehicle has to wait if it arrives at a customer location before the earliest arrival time for that customer. Constraints (2.6) and (2.7) ensure that the same vehicle visits each customer. The time feasibility constraints for the problem are defined in (2.9), (2.10) and (2.11). The constraint (2.9) ensures that the arrival time between two customers are compatible. The constraint (2.10) enforces the arrival time of a vehicle at a

customer site to be within the customers earliest and latest arrival times and (2.11) ensures that the arrival time of the vehicle at a customer location is always positive.

The vehicle routing problem (VRP), without time windows, is NP-complete [3][18]. Solomon[28] and Savelsbergh[24] indicate that the time constrained problem is fundamentally more difficult than a simple VRP even for a fixed fleet of vehicles. Savelsbergh[24] has shown that finding a feasible solution for a VRPTW using a fixed fleet size is NP-complete. Due to the intrinsic difficulty of the problem, search methods based upon heuristics are most promising for solving practical size problems [1][9][17][23][23][24][25][27][28][28]. Heuristic methods often produce optimum or near optimum solutions for large problems in a reasonable amount of computer time. Therefore the development of heuristic algorithms that can obtain near optimal feasible solutions for large VRPTW are of primary interest.

The GIDEON system that we propose to solve the VRPTW is a cluster-first route-second heuristic algorithm that solves an approximation of the mathematical model described in (2.1). The algorithm has two phases consisting of a global search strategy to obtain clusters of customers and a local post-optimization procedure that improves the solution. The clustering of customers is done using a Genetic Algorithm(GA) and the local post-optimization method moves and exchanges customers between routes in order to improve the solution. The two processes are run iteratively a finite number of times to improve the solution quality.

? . 3 The GIDEON System

The global search strategy for clustering customers in GIDEON is done using a Genetic Algorithm(GA). GA's are a class of heuristic search algorithms based upon population genetics[6][7][16]. As they are inherently adaptive, genetic algorithms can converge to near optimal solutions in many applications. They have been used to solve a number of combinatorial problems[4][5][15][19]. The GA is an iterative procedure that maintains a pool of candidates

simulated over a number of generations. The population members are referred to as chromosomes. The chromosomes are fixed length strings with a finite number of binary values. Each chromosome has a fitness value assigned to it based upon a fitness function. The fitness value determines the relative ability of the chromosome to survive over the generations. Chromosomes with high fit values have a higher probability of surviving into the next generation compared to chromosomes with low fit values. At each generation, chromosomes are subjected to selection, crossover and mutation. Selection allows chromosomes with high fit values to survive into the next generation. Crossover splices chromosomes at random points and exchanges it with other spliced chromosomes. Mutation changes the bit value of a chromosome to its complementary value. Selection and crossover search a problem space exploiting information present in the chromosomes by selecting and recombining primarily those offspring that have high fitness values. These two processes eventually produce a population of chromosomes with high performance characteristics. The mutate operator is a secondary operator that prevents premature loss of information by randomly mutating bits in a chromosome. For a detailed description of this process refer to [11].

The local post-optimization procedure in GIDEON improves a solution by using a λ-interchange local optimization method. The λ-interchange local optimization method is based on the interchange of customers between sets of routes. The λ-interchange generation mechanism can be described as follows. Given a solution to the problem represented by a set of routes $S = \{R_1,...,R_p,...,R_q,...,R_k\}$, where each route is the set of customers serviced on this route, a λ-interchange between a pair of routes R_p and R_q is a replacement of a subset of customers $S_1 \subseteq R_p$ of size $|S_1| \leq \lambda$ by another subset $S_2 \subseteq R_q$ of size $|S_2| \leq \lambda$ to get two new routes $R'_p = (R_p - S_1) \cup S_2$, $R'_q = (R_q - S_2) \cup S_1$ and a new neighboring solution $S' = \{R_1,...,R'_p,...,R'_q,...,R_k\}$. The neighborhood $N_\lambda(S)$ of a given solution S is the set of all neighbors S′ generated in this way for a given value of λ.

The order in which the neighbors are searched is specified as follows for a given solution $S = \{R_1,...,R_p,...,R_q,...,R_k\}$:

$$(R_1, R_2),(R_2, R_3),..., (R_1, R_k), (R_2, R_3),...,(R_2, R_k),...,(R_{k-1}, R_k)$$

Hence a total number of $\frac{K \times (K-1)}{2}$ different pairs of routes (R_p, R_q) are examined to define a cycle of search. For the iterative local search heuristics, the same permutation is used throughout the search. For a given pair of routes (R_p, R_q), one must also define the search order for the customers to be exchanged. Here we consider the cases of λ=1 and λ=2, that result in one or two customers being shifted from one route to another or exchanges between two routes. The λ-interchange method results in customers either being shifted from one route to another, or customers being exchanged between routes. The search in the neighborhood of the current solution applies the operators in the following order on each pair of routes: (0,1), (1,0), (1,1), (0,2), (2,0), (2,1), (1,2) and (2,2). The operator (0,1) on routes (R_p, R_q) indicates a shift of one customer from route *q* to route *p*. The operators (1,0), (2,0) and (0,2) indicate a shift of one or two customers from one route to another. The operator (1,1) on routes (R_p, R_q) indicates an exchange of one customer between route *p* and *q*. The operators (1,2), (2,1) and (2,2) are defined similarly and indicate an exchange of customers between two vehicle routes. Finally, for a given operator and a given pair of routes, the customers are considered sequentially and systematically along the routes in order.

The λ-interchange local optimization method shifts and exchanges customers between routes until no more improvements are found. In both shift and exchange procedures, improved solutions are accepted if the insertion results in the reduction of the total cost for routing the vehicles. The shift and exchange heuristics have theoretical properties [20] and have been implemented successfully in many combinatorial problems [21][22][29][30]. The local post-optimization procedure for the GIDEON system uses $\lambda = 2$ for improving solutions.

The search space used by GIDEON is a relaxation of the feasible region of the mathematical model proposed in (2.1). The mathematical model (2.1) is approximated by the GIDEON system by a relaxation of the capacity, route time and time window constraints in a Lagrangian Relaxation fashion. The cost function used by the GIDEON system drives the search for a good feasible solution by penalizing violation of capacity, route or time window constraints. The objective function used by the GIDEON system is stated as:

$$(\overline{\mathbf{VRPTW}}) \qquad \text{Min} \sum_{ijk} c_{ijk} x_{ijk} \qquad (2.12)$$

where,

$$c_{ijk} = \alpha \times t_{ij} + \beta \cdot \times (t_i + f_i + t_{ij}) + \eta \times \max\{0, (q_{ik} - v_k)\} + \kappa \cdot \times \max\{0, (r_{ik} - l_i)\} + \gamma \cdot \times \max\{0, (t_i + f_i + t_{ij} - T)\}$$

The cost function includes components weighted by coefficients α for distance, β for route time and penalty weighting factors η for vehicle overload, γ for travel time in excess of the allocated route time for the vehicle and κ for tardiness. The GIDEON system explores for feasible solutions to the VRPTW with weights that drive the model towards feasibility in the VRPTW problem. The weights for GIDEON were derived empirically and set at $\alpha = 0.5$, $\beta = 0.05$, $\eta = 50$, $\kappa = 25$ and $\gamma = 50$. The weights are biased towards finding a feasible solution in comparison to reducing the total distance and route time. The main priority of the cost function (2.12) is to obtain a feasible solution. Therefore the coefficients of the cost function (2.12) gives higher priority to reducing tardy and overloading vehicles, followed by vehicles that exceed the maximum allotted route time for a vehicle. If there are no violation of the capacity, time feasibility and route time constraints, then the coefficients of the cost function (2.12) is to reduce the total distance followed by the total route time. The weights for the coefficients of the cost function were chosen to first obtain a feasible solution and then minimize the total distance and route time. The cost function (2.12) was experimented with other weight values, values that gave higher weights to the cost coefficients α and β and lower weights to η, α and γ but these resulted in infeasible or solutions of poor quality. The GIDEON system uses a the cluster-first route-second method to solve a VRPTW. That is, given a set of customers and a central depot, the system clusters the customers using the GA, and

the customers within each sector are routed using the cheapest insertion method [13]. The clustering of customers using a GA is referred to as Genetic Sectoring. Genetic Sectoring has been successfully used to solve vehicle routing and scheduling problems with varying constraints [28][29]. The GIDEON system allows exploration and exploitation of the search space to find good feasible solutions with the exploration being done by the GA and the exploitation by the local post-optimization procedure.

The GENESIS [14] genetic algorithm software was used in the implementation of the GIDEON system. The chromosomes in GENESIS are represented as bit strings. The sectors(clusters) for the VRPTW are obtained from a chromosome by subdividing it into K divisions of size B bits. Each subdivision is used to compute the size of a sector. The fitness value for the chromosome is the cost function (2.12) for of serving all the customers computed with respect to the sector divisions derived from it.

In an N customer problem with the origin at the depot, the GIDEON system replaces the customer angles $p_1,...,p_N$ with pseudo polar coordinate angles $s_1,...,s_N$. The pseudo polar coordinate angles are obtained by normalizing the angles between the customers so that the angular difference between any two adjacent customers is equal. This allows sector boundaries to fall freely between any pair of customers that have adjacent angles, whether the separation is small or large. The customers are divided into K sectors, where K is the number of vehicles, by planting a set of "seed" angles, $S_0,...,S_K$, in the search space and drawing a ray from the origin to each seed angle. The initial number of vehicles, K, required to service the customers is obtained using Solomon's insertion heuristic[28]. The initial seed angle S_0 is assumed to be 0^o. The first sector will lie between seed angles S_0 and S_1, the second sector will lie between seed angles S_1 and S_2, and so on.

The Genetic Sectoring process assigns a customer, C_i, to a sector or vehicle route, V_k, based on the following equation:

$$C_i \text{ is assigned to } V_k \quad \text{if } S_k < s_i <= S_{k+1}, \text{ where } k= 0,...,K-1$$

Customer C_i is assigned to vehicle V_k if the pseudo polar coordinate angle s_i is greater than seed angle S_k but is less than or equal to seed angle S_{k+1}. Each seed angle is computed using a fixed angle and an offset from the fixed angle. The fixed angle, *F*, is the minimum angular value for a sector and assures that each sector gets represented in the Genetic Sectoring process. The fixed angle is computed by taking the maximum polar coordinate angle within the set of customers and dividing it by 2*K*. The offset is the extra region from the fixed angle that allows the sector to encompass a larger or a smaller sector area. The GA is used to search for the set of offsets that will result in the minimization of the total cost of routing the vehicles. If a fixed angle and its offset exceeds 360^o, then that seed angle is set to 360^o thereby allowing the Genetic Sectoring process to consider vehicles less than K to service all its customers. Therefore *K*, the initial number of vehicles with which the GIDEON system is invoked, serves as the upper bound on the number of vehicles that can be used for servicing all the customers.

The bit size representation of an offset in a chromosome *B* was set at 3 bits. Bit size representations larger than 3 were experimented with and resulted in poor quality solutions. The decimal conversion of 3 bits results in a range of integer values between 0 and 7. The decimal values retrieved from the subset of a chromosome are multiplied by a constant *I* that increases the range of the offset. The value derived from the decimal conversion of the bit values times the constant value *I* are proportionately mapped to the offsets with the value 0 as a 0^o offset and the bit value 10.5 and 10.5^o as the maximum offset. Figure ?.1 describes the chromosome mapping used to obtain the offsets.

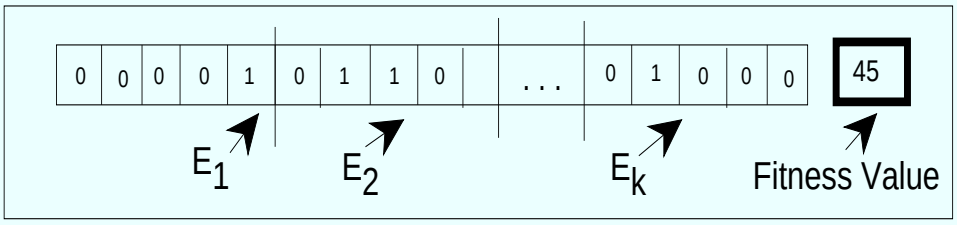

Figure ?.1. Representation of the offsets using a chromosome.

The seed angles are derived from the chromosome using the following equation:

$$S_i = S_{i-1} + F + (E_i \cdot I) \qquad (2.13)$$

The fitness value of a chromosome is the total cost of routing K vehicles for servicing N customers using the sectors formed from the set of seed angles derived from the chromosome. The seed angles are derived using the fixed angle and the offsets from the chromosomes. The customers within the sectors, obtained from the chromosomes, are routed using the cheapest insertion method. The cheapest insertion method takes each unrouted customer in the sector and each edge $\{i, j\}$ in the current tour and computes the cost of inserting the unrouted customer between i and j. The unrouted customer that has the least insertion cost at edge $\{i, j\}$ is selected to be inserted between i and j. The cost of inserting customer C_i into route V_k using the cheapest insertion method is calculated using the insertion cost function:

$$\text{insertion cost of } C_i = \alpha D_k + \beta R_k + \eta O_k + \kappa T_k + \gamma Q_k \qquad (2.14)$$

The insertion cost function (2.14) will accept infeasible solutions if the reduction in total distance is high enough to allow either a vehicle to be overloaded or be tardy. Overloading and tardiness in a vehicle route are penalized in the insertion cost function (2.14). The weights for insertion cost (2.14) were set at $\alpha = 0.5$, $\beta = 0.05$, $\eta = 50$, $\kappa = 50$, and $\gamma = 25$.

In the GIDEON system each chromosome represents a set of offsets for a VRPTW. Therefore, a population of *P* chromosomes usually has *P* different solutions for a VRPTW. That is, there may be some chromosomes in the population that are not unique. At each generation, *P* chromosomes are evaluated for fitness. The chromosomes that have the least cost will have a high probability of surviving into the next generation through the selection process. As the crossover operator exchanges a randomly selected portion of the bit string between the chromosomes, partial information about sector divisions for the VRPTW is exchanged between the chromosomes. New information is generated within the chromosomes by the mutation operator.

The GIDEON system uses selection, crossover and mutation to adaptively explore the search space for the set of sectors that will minimize the total cost of the routes over the simulated generations for the VRPTW. The GIDEON system would utilize more computer time than traditional heuristic algorithms because every time the Genetic Sectoring process is invoked it has to evaluate $P \cdot G$ vehicle routes, where *P* is the population size and *G* is number of generations to be simulated. The worst case running time bounds for the Genetic Sectoring process is $O(\frac{N}{K}^2)$ as there is on the average $\frac{N}{K}$ customers for each route and $\frac{N}{K}$ edges have to be checked for each of the *v* routes.

The parameter values for the number of generations, population size, crossover and mutation rates for the Genetic Sectoring process were set at 1000, 50, 0.8 and 0.001. During the simulation of the generations, the GIDEON system keeps track of the set of sectors obtained from the genetic search that has the lowest total route cost. The genetic search terminates either when it reaches the number of generations to be simulated or if all the chromosomes have the same fitness value. The best set of sectors obtained after the termination of the genetic search does not always result in a feasible solution. The solution obtained from the GA is improved using the local post-optimization procedure that shifts and exchanges customers between the vehicle routes. The post-optimization procedure is similar to the 2-opt procedure, as it deletes two arcs and inserts it

into a location of a different route that has the lowest cost. The worst case running time bounds for the local post-optimization process is $O((\frac{N}{K})^2)$ as there is on the average $\frac{N}{K}$ customers for each route and on the average $\frac{N}{K}$ edges have to be checked.

The local post-optimization process is carried out until no more improvements can be made to the solution obtained from the GA. At the termination of the local post-optimization procedure, the customers are ranked in order of the sectors, and within the sectors in the sequence in which they are visited by the vehicles. The customer angles, $p_1,..., p_N$, are replaced with pseudo polar coordinate angles $s_1,..., s_N$ in order of the customer rank. The assignment of pseudo polar coordinate angles, using route and customer sequence, clusters together customers with geographical and temporal characteristics that can be serviced by a single vehicle.

The customers with the new pseudo polar coordinate angles are once again used to form new sectors using the GA. The best set of sectors obtained from the GA using the new customer polar coordinate angles are improved using the local post-optimization procedure. This iteration between Genetic Sectoring process and local post-optimization procedure is carried out a predetermined number of times and was set at 5. The flow of the GIDEON system is as follows:

Step 1: Set the number of cluster-route iterations: *itermax* = 3.
Set the current iteration number: *iter*=0.
Set the bit string size for the offset: *Bsize*=5.
Step 2: Sort the customers in order of their polar coordinate angles, and
assign pseudo polar coordinate angles to the customers.
Set the lowest global route cost to infinity: *g*=∞.
Set the lowest local route cost to infinity: *l*=∞.
Step 3: Increment the number of iterations: *iter*=*iter*+1.
If *iter* > *itermax*, go to Step 7.
Step 4: If GA has terminated, go to Step 5.
For each chromosome in the population:
For each bit string of size *BSize*,
calculate the seed angle,
sector the customers, and
route the customers within the sectors using the cheapest insertion method.
If the cost of the current set of sectors is lower than *l*,
set l to the current route cost, and

save the set of sectors in *lr*.
If the cost of the current set of sectors is lower than *g*,
set *g* to the current route cost, and
save the set of sectors in *gr*.
Do Selection, Crossover and Mutation on the chromosomes.
Go to Step 4.

Step 5: Do local post-optimization using the route *lr*.
If no improvements can be made to route *lr*,
go to Step 6.
If the current improved route has lower cost than *l*,
set *l* to the current cost, and
save the set of sectors in *lr*.
If the current improved route has lower cost than *g*,
set *g* to the current cost, and
save the set of sectors in *gr*.
Go to step 5.

Step 6: Rank the customers of route lr in order of the sectors, and within the sectors in order of the sequence in which they are visited.
Sort the customers by the rank.
Assign pseudo polar coordinates to the customers in order of the sorted rank.
Go to Step 3.

Step 7: Stop the Genetic Sectoring Heuristic with a local post-optimization solution.

The Genetic Sectoring and local post-optimization procedures are symbiotic as the Genetic Sectoring is a meta-search strategy that forms the sectors and the local post-optimization procedure gives adjacency information about the customers back to the Genetic Sectoring process. These two procedures derive information from each other in order to obtain a good feasible solution.

? . 4 Computational Results

GIDEON was run on a set of 56 VRPTW problems in six data sets denoted R1, C1, RC1, R2, C2, and RC2, developed by Solomon [28]. Solomon generated vehicle routing problems with two time windows using the standard set of vehicle routing test problems from Christofides et al. [3].The vehicle routing problems with two time windows were generated by assigning earliest and latest time windows to each of the customers in addition to the service time required by each of the customers. In terms of time window density (the percentage of customers with time windows), the

problems have 25%, 50%, 75%, and 100% time windows. Each of the problems in these data sets has 100 customers. The fleet size to service them varied between 2 and 21 vehicles.

For the R1 data set, without time window constraints, a fleet of 10 vehicles, each with a capacity of 200 units, was required to attain a feasible solution. Each of the customers in the R1 data set required 10 units of service time and a maximum route time of 230 units.In the C1 data set, each customer required 90 units of service time and the vehicles had a capacity of 200 units and a maximum route time of 1236 units. The optimal solution for this problem class requires 10 vehicles and has a distance of 827 units[9]. The RC1 data set was created using data sets, R1 and C1. The vehicle capacity for this problem was set at 200 units with a maximum route time of 240 units. Each of the customers in this problem required 10 units of service time.

The R2 data set was a modification of the R1 data set to allow for servicing of many customers by one vehicle. The maximum route time of the vehicles was set at 1000 units and each vehicle had a capacity of 1000 units. Two vehicles are enough to satisfy the customer demands if no time windows are present. In the C2 data set, customers from the C1 data set were relocated to create a structured problem with three large clusters of customers. The vehicles for this data set had a maximum route time of 3390 units and a capacity of 700 units with each customer requiring 90 units of service time. For the RC2 data set, the customer demands and service times are the same as for RC1. The vehicles for this data set have a maximum route time of 960 units and a capacity of 1000 units. Without time windows, a fleet of two vehicles was enough to satisfy the demands.

The data sets, R1, C1, and RC1, had short horizons while the data sets, R2, C2, and RC2, had long horizon problems. Short horizon problems have vehicles that have small capacities and short route times and cannot service many customers at one time. Long horizon problems use vehicles that have large capacities and long travel times, and are able to service many customers with fewer vehicles. The VRPTW problems generated by Solomon incorporate many distinguishing features

of vehicle routing with two-sided time windows. The problems vary in fleet size, vehicle capacity, travel time of vehicles, spatial and temporal distribution of customers, time window density (the number of demands with time windows), time window width, percentage of time constrained customers and customer service times.

Solutions to each of the 56 VRPTW were obtained by Solomon [28] and Thompson [30]. Solomon tested a number of algorithms and heuristics and reported that the overall best performance were obtained using a sequential insertion procedure that used a weighted combination of time and distance in its cost function. The best solutions using the heuristic insertion procedure were obtained using eight different combinations of parameters and three different initialization criteria. Thompson's solutions use local post-optimization procedures, based on cyclical transfers, to obtain feasible solutions. The solutions reported are the best of eight different combinations of parameters and two different initialization criteria. For comparison purposes the heuristic used to obtain the best solution by Solomon will be referred to as Heuristic 1 and by Thompson as Heuristic 2.

Koskosidis et al. [17] used a "soft" time approach based on the Generalized Assignment Heuristic for solving the VRPTW. This approach allowed time windows to be violated at a cost which results in final solution that could infeasible. This method was used to solve only some Solomon's time window problems name some problems from the R1 and RC1 data set and all of the problems in the C1 data set.

In GIDEON the solution quality is based on minimizing the number of routes followed by the distance and route time. That is, a solution with M number of routes is better than M+1 routes, even if the distance and route time for the M routes is greater than M+1 routes. In VRPTW it is possible to get distance and route time for M+1 routes, that is less than the distance and route time for M+1 routes. The GIDEON system was used to solve the 56 VRPTW problems using two types of initial placement of customers. The first method initially sorted the customers by the polar coordinate

angles before assigning the customers the pseudo polar coordinate angles. The second method assigned pseudo polar coordinate angles to the customers in a random manner. The solutions obtained by GIDEON using the two methods are tabulated in Tables 1 and 2. The best of the solutions obtained from these two methods were compared against the best solutions obtained Solomon's and Thompson's heuristics.

The comparison between the solutions obtained by GIDEON and other heuristic algorithms were done in the following form. As Solomon[28] and Thompson [30] report the results for each of the problems in the literature, the solutions obtained by GIDEON were compared with each of their solutions. In addition the average number of vehicles and distance obtained by the GIDEON system are compared against the solutions obtained by Potvin's Tabu search heuristic[23].

The best solutions obtained by GIDEON did better than both Heuristic 1 and Heuristic 2 on 41 of the 56 problems as indicated in Tables 3 and 4 in bold. In comparison to the best solutions obtained by Heuristic 1 and Heuristic 2, the solutions obtained by GIDEON resulted in an average reduction of 3.9% in fleet size and 4.4% in distance traveled by the vehicles. Table 5 is a summary of the average improvement in vehicle fleet size and distance obtained by GIDEON with respect to Heuristic 1 and Heuristic 2 for the six different data sets. The GIDEON system was written in C language and the experiments were conducted on a SOLBOURNE 5/802 system. The solution to the VRPTW using the GIDEON system required an average of 127 CPU seconds to be solved on a SOLBOURNE 5/802 computer. The SOLBOURNE 5/802 computer is about 10 times faster than a personal computer. On the average, the Genetic Sectoring process took about 27 seconds to form the sectors and the local post-optimization process took 100 seconds to improve the solution.

Table 1: Solutions for data Sets R1, C1 and RC1 using the three different heuristics.

	Heuristic 1			Heuristic 2			GIDEON		
Problem Number	Number of Vehicles	Total Distance	CPU[1]	Number of Vehicles	Total Distance	CPU[2]	Number of Vehicles	Total Distance	CPU[3]
R101	21	1873	21.8	19	1734	1394	20	1700	88.2
R102	19	1843	22.9	17	1881	3209	**17**	**1549**	**100.5**
R103	14	1484	24.5	15	1530	3337	**13**	**1319**	**102.9**
R104	11	1188	27.3	10	1101	2327	**10**	**1090**	**50.4**
R105	15	1673	22.0	15	1535	2359	**15**	**1448**	**95.9**
R106	14	1475	23.5	13	1392	1575	**13**	**1363**	**105.2**
R107	12	1425	25.0	11	1250	3261	**11**	**1187**	**103.4**
R108	10	1137	28.0	10	1035	1575	10	1048	91.0
R109	13	1412	23.4	12	1249	2236	12	1345	96.5
R110	12	1393	25.0	12	1258	1514	**11**	**1234**	**103.1**
R111	12	1231	25.0	12	1215	3046	**11**	**1238**	**109.4**
R112	10	1106	28.2	10	1103	2168	**10**	**1082**	**121.9**
C101	10	853	22.4	10	829	464	10	833	87.5
C102	10	968	23.7	10	934	1360	**10**	**832**	**88.7**
C103	10	1059	26.7	10	956	2404	**10**	**873**	**81.6**
C104	10	1282	30.7	10	1130	3602	**10**	**904**	**95.3**
C105	10	861	22.8	10	829	449	10	874	91.8
C106	10	897	23.2	10	868	716	10	902	91.2
C107	10	904	24.1	10	926	757	10	926	93.1
C108	10	855	25.2	10	866	987	10	928	89.9
C109	10	888	28.8	10	912	1277	10	957	87.8
RC101	16	1867	21.9	16	1851	2282	**15**	**1767**	**104.7**
RC102	15	1760	22.8	14	1644	2957	**14**	**1569**	**105.5**
RC103	13	1673	24.1	12	1465	3661	**11**	**1328**	**116.5**
RC104	11	1301	26.1	11	1265	2438	**11**	**1263**	**108.4**
RC105	16	1922	23.0	15	1809	2417	**14**	**1612**	**111.6**
RC106	13	1611	22.7	-	-	-	**12**	**1608**	**109.2**
RC107	13	1385	24.2	12	1338	2295	**12**	**1396**	**122.8**
RC108	11	1253	25.6	11	1228	2297	**11**	**1250**	**115.9**

Legend:

Heuristic 1	:	Best solution from Solomon's Heuristic[28].
Heuristic 2	:	Best solution from Thompson Heuristic[30].
CPU[1]	:	CPU time in seconds to obtain a solution on an DEC-10.
CPU[2]	:	CPU time in seconds to obtain a solution on an IBM PC-XT.
CPU[3]	:	CPU time in seconds to obtain a solution on a SOLBOURNE 5/802.

Table 2: Solutions for data sets R2, C2 and RC2 using the three different heuristics.

	Heuristic 1			Heuristic 2			GIDEON		
Problem Number	Number of Vehicles	Total Distance	CPU[1]	Number of Vehicles	Total Distance	CPU[2]	Number of Vehicles	Total Distance	CPU[3]
R201	4	1741	32.9	4	1786	3603	**4**	**1478**	**127.7**
R202	4	1730	42.2	4	1736	2514	**4**	**1279**	**128.7**
R203	3	1578	60.1	3	1309	12225	**3**	**1167**	**251.3**
R204	3	1059	90.6	3	1025	22834	**3**	**909**	**137.5**
R205	3	1471	42.9	3	1392	3039	**3**	**1274**	**128.4**
R206	3	1463	53.3	3	1254	2598	**3**	**1098**	**315.4**
R207	3	1302	71.9	3	1072	2598	**3**	**1015**	**183.9**
R208	3	1076	108.6	3	862	12992	**3**	**826**	**119.1**
R209	3	1449	52.5	3	1260	7069	**3**	**1159**	**140.6**
R210	4	1542	51.2	3	1269	11652	3	1269	215.3
R211	3	1016	82.7	3	1071	9464	**3**	**898**	**267.7**
C201	3	591	31.2	3	590	240	3	753	123.1
C202	3	731	39.7	3	664	1644	3	756	124.0
C203	3	811	48.0	3	653	2757	3	855	162.2
C204	4	758	61.0	3	684	2211	3	803	140.0
C205	3	615	36.0	3	628	1723	3	667	119.0
C206	3	730	40.3	3	641	1429	3	694	139.0
C207	3	691	41.4	3	627	722	3	730	156.0
C208	3	615	46.6	3	670	1103	3	735	174.0
RC201	4	2103	31.1	4	1959	1140	**4**	**1823**	**135.9**
RC202	4	1799	39.1	4	1858	4164	**4**	**1459**	**155.3**
RC203	4	1626	53.7	4	1521	6109	**3**	**1323**	**156.0**
RC204	3	1208	85.5	3	1143	5015	**3**	**1021**	**192.9**
RC205	5	2134	36.5	4	1988	5906	**4**	**1594**	**183.3**
RC206	4	1582	39.9	3	1515	4833	**3**	**1530**	**180.3**
RC207	4	1632	30.3	4	1457	13340	**3**	**1501**	**156.1**
RC208	3	1373	77.6	-	-	-	**3**	**1038**	**115.7**

Legend:

Heuristic 1	:	Best solution from Solomon's Heuristic[28].
Heuristic 2	:	Best solution from Thompson's Heuristic[30].
CPU[1]	:	CPU time in seconds to obtain a solution on an DEC-10.
CPU[2]	:	CPU time in seconds to obtain a solution on an IBM PC-XT.
CPU[3]	:	CPU time in seconds to obtain a solution on a SOLBOURNE 5/802.

The quality of the solutions obtained by GIDEON for the VRPTW measured in fleet size and total distance traveled vary considerably with geographical clustering and time window tightness of the customers. For example, for a problem from the C1 data set, the Genetic Sectoring method quickly clusters the data in the natural fashion and finds a feasible solution in a short period of time. The Genetic Sectoring for clusters is much more extensive for an unclustered problem in data set R1. For an unclustered problem, the assignment of customers to vehicles does not follow radial clustering, but rather strongly utilizes the local search process to form pseudo clusters for the Genetic Sectoring process. As expected, for problems from data sets RC1 and RC2, in which the customers are not all naturally clustered, GIDEON produced good solutions. For problems in data sets R2, C2 and RC2 the Genetic Sectoring process is reliant upon the local optimization process to obtain good solutions due to the small number of clusters involved. GIDEON consistently produces higher performance solutions relative to competing heuristics on problems that has large number of vehicles, tight windows and customers that are not clustered. Further computational analysis was performed to analyze the significance of the solutions obtained by GIDEON against Heuristic 1 and Heuristic 2.

Table 3: Comparison of the average% differences between GIDEON and Heuristic 1 and Heuristic 2.

	Heuristic 1		Heuristic 2	
Problem group	Average% difference in number of Vehicles	Average% difference in Total Distance	Average% difference in number of Vehicles	Average% difference in Total Distance
R1	6.1	9.5	1.9	4.2
C1	0.0	6.2	0.0	2.7
RC1	7.4	7.6	3.9	2.7
R2	4.5	19.8	-2.9	11.7
C2	4.5	-8.1	0.0	-27.4
RC2	12.9	16.1	8.0	14.2

Legend:

Heuristic 1 : Best solution from Solomon's Heuristic[28].
Heuristic 2 : Best solution from Thompson's Heuristic[30].

5. Summary and Conclusions

GIDEON performs uniformly better than both the heuristics used by Solomon and Thompson with the exception of the problem group C2. GIDEON does not tend to perform well for problems in which the customers are geographically clustered together and have a small number of vehicles. In comparison to the Potvin's Tabu heuristic for solving the VRPTW, GIDEON obtains solutions that are as good as those of the Tabu search. For data sets in which the customers are clustered GIDEON does not obtain good solutions. This is to be expected as the genetic algorithm requires large differences in the fitness values of the chromosomes to exploit the search space.

This research shows that genetic search can obtain good solutions to vehicle routing problems with time windows compared to traditional heuristics for problems that have tight time windows and a large number of vehicles.with a high degree of efficiency. The adaptive nature of the genetic algorithms are exploited by GIDEON to attain solutions that are of high performance relative to those of competing heuristics. This methodology is potentially useful for solving VRPTW's in real time for routing and scheduling in dynamic environments.

Acknowledgment

We thank Marius Solomon and Paul Thompson for providing the test problems used in this study.

REFERENCES

1. Baker, Edward K. and Joanne R. Schaffer, Solution Improvement Heuristics for the Vehicle Routing Problem with Time Window Constraints. *American Journal of Mathematical and Management Sciences* (Special Issue) 6, 261-300, 1986.

2. Bodin, L., B. Golden, A. Assad and M. Ball, The State of the Art in the Routing and Scheduling of Vehicles and Crews. *Computers and Operations Research* 10 (2), 63-211, 1983.

3. Christofides, N., A. Mingozzi and P. Toth, The Vehicle Routing Problem. In

Combinatorial Optimization, P. Toth, N. Christofides, R. Mingozzi and C. Sandi (Eds.), John Wiley, New York, 315-338, 1989.

4. Davis, Lawrence, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.

5. DeJong, Kenneth and William Spears, Using Genetic Algorithms to Solve NP-Complete Problems. Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufman Publishers, California, 124-132, 1989.

6. DeJong, Kenneth, Adaptive System Design: A Genetic Approach. *IEEE Transactions on Systems, Man and Cybernetics* 10 (9), 566-574, 1980.

7. DeJong, Kenneth, Analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph.D. Dissertation, University Michigan, Ann Arbor, 1975.

8. Desrochers, M., J. Desrociers and Marius Solomon. A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows, *Operations Research* 40(2), 1992.

9. Desrochers, M. et al., Vehicle Routing with Time Windows: Optimization and Approximation. Vehicle Routing: Methods and Studies, B. Golden and A. Assad (eds.), North Holland, 1988.

10. Gillett, B. and L. Miller, A Heuristic Algorithm for the Vehicle Dispatching Problem. *Operations Research* 22, 340-349, 1974.

11. Goldberg D. E., Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, Inc., 1989.

12. Golden B. and A. Assad (Eds.), Vehicle Routing: Methods and Studies. North Holland, Amsterdam, 1988.

13. Golden B. and W. Stewart, Empirical Analysis of Heuristics. In The Traveling Salesman Problem, E. Lawler, J. Lenstra, A. Rinnooy and D. Shmoys (Eds.), Wiley-Interscience, New York, 1985.

14. Grefenstette, John J., A Users Guide to GENESIS. Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington D.C. 20375-5000, 1987.

15. Grefenstette, John, Rajeev Gopal, Brian Rosmaita and Dirk Van Gucht, Genetic Algorithms for the Traveling Salesman Problem. Proceedings of the First International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum Associates, New Jersey, 112-120, 1985.

16. Holland, J. H., Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

17. Koskosidis, Yiannis, Warren B. Powell and Marius M. Solomon, An Optimization Based Heuristic for Vehicle Routing and Scheduling with Time Window Constraints. *Transportation Science* 26 (2), 69-85, 1992.

18. Lenstra, J. and Rinnooy Kan, Complexity of the Vehicle Routing and Scheduling Problems. NETWORKS 11 (2), 221-228, 1981.

19. Michalewicz, Zbigniew, Genetic Algorithms + Data Structures = Evolution Programs. Springer-Velarg, New York, 1992.

20. Osman, I. H. and Christofides N (1994). Capacitated Clustering Problems by Hybrid Simulated Annealing and Tabu Search. *International Transactions in Operational Research*, Forthcoming.

21. Osman, I. H. (1993a). Vehicle Routing and Scheduling: Applications, Algorithms and Developments. Proceedings of the International Conference on Industrial Logistics, Rennes, France.

22. Osman, I. H. (1993b). Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problems. *Annals of Operations Research*, 41, 421-451.

23. Savelsbergh M. W. P., Local Search for Constrained Routing Problems. Report OS-R8711, Department of Operations Research and System Theory, Center for Mathematics and Computer Science, Amsterdam, Holland, 1987.

24. Savelsbergh M. W. P., Local Search for Routing Problems with Time Windows. Annals of Operations Research 4, 285-305, 1985.

25. Solomon, Marius M., Edward K. Baker, and Joanne R. Schaffer, Vehicle Routing and Scheduling Problems with Time Window Constraints: Efficient Implementations of Solution Improvement Procedures. In Vehicle Routing: Methods and Studies, B. L. Golden and A. Assad (Eds.), Elsiver Science Publishers B. V. (North-Holland), 85-90, 1988.

26. Solomon, Marius M. and Jacques Desrosiers, Time Window Constrained Routing and Scheduling Problems: A Survey. *Transportation Science* 22 (1), 1-11, 1986.

27. Solomon, Marius M., Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research* 35 (2), 254-265, 1987.

28 Thangiah, Sam R., Ibrahim Osman, Rajini Vinayagamoorthy and Tong Sun, Algorithms for Genetic Algorithm for Vehicle Routing with Time Deadlines. Forthcoming in the *American Journal of Mathematical and Management Sciences*, 1994.

29 Thangiah, Sam R., Rajini Vinayagamoorthy and Ananda Gubbi, Vehicle Routing with Time Deadlines using Genetic and Local Algorithms. Proceedings of the Fifth International Conference on Genetic Algorithms, 506-513, Morgan Kaufman, New York, 1993.

30. Thompson, Paul M., Local Search Algorithms for Vehicle Routing and Other Combinatorial Problems. Ph.D. Dissertation, Massachusetts Institute of Technology, Massachusetts, 1988.