Hybrid Genetic Algorithm, Simulated Annealing and Tabu Search Methods for Vehicle Routing Problems with Time Windows

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

Ibrahim H. Osman
Institute of Mathematics and Statistics, University of Kent
Canterbury, Kent CT2 7NF, U.K.

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

ABSTRACT

The Vehicle Routing Problem with Time Windows (VRPTW) involves servicing a set of customers, with earliest and latest time deadlines, with varying demands using capacitated vehicles with limited travel times. The objective of the problem is to service all customers while minimizing the number of vehicles and travel distance without violating the capacity and travel time of the vehicles and customer time constraints. In this paper we describe a λ-interchange mechanism that moves customers between routes to generate neighborhood solutions for the VRPTW. The λ-interchange neighborhood is searched using Simulated Annealing and Tabu Search strategies. The initial solutions to the VRPTW are obtained using the Push-Forward Insertion heuristic and a Genetic Algorithm based sectoring heuristic. The hybrid combination of the implemented heuristics, collectively known as the GenSAT system, were used to solve 60 problems from the literature with customer sizes varying from 100 to 417 customers. The computational results of GenSAT obtained new best solutions for 40 test problems. For the remaining 20 test problems, 11 solutions obtained by the GenSAT system equal previously known best solutions. The average performance of GenSAT is significantly better than known competing heuristics. For known optimal solutions to the VRPTW problems, the GenSAT system obtained the optimal number of vehicles.

Keywords: Vehicle routing with time windows, Local search, Simulated Annealing, Tabu Search, Genetic Algorithms, Heuristics.

1. Introduction

The vehicle routing problem with time windows (VRPTW) is an extension of the vehicle routing problem (VRP) with latest, earliest and service times for customers. The VRPTW routes a set of vehicles to service customers having earliest, latest service times. The objective of the problem is to minimize the number of vehicles and the distance travelled to service the customers. The constraints of the problem are to service all the customers within the earliest and latest service time of the customer without exceeding the route time of the vehicle and overloading the vehicle. The route time of the vehicle is the sum total of the waiting time, the service time and distance travelled by the vehicle. A vehicle that reaches a customer before the earliest service time incurs waiting time. The service time is the time taken by a vehicle to service a customer. A vehicle is said to be overloaded if the sum total of the customer demands exceed the total capacity of the vehicle.

Applications of routing and scheduling models arise in a wide range of practical decision making problems. Efficient routing and scheduling of vehicles can save the public and private sectors millions of dollars per year. The VRPTW arises 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. Surveys on classifications and applications of VRP can be found in Osman [Osman, 1993a], Laporte [Laporte, 1992], Fisher [Fisher, 1993] and Bodin, Golden and Assad and Ball [Bodin et al. 1983].

Savelsbergh [Savelsbergh, 1985] has shown that finding a feasible solution to the traveling salesman problem with time windows (TSPTW) is a NP-complete problem. Therefore VRPTW is more complex as it involves servicing customers with time windows using multiple vehicles that vary with respect to the problem. VRPTW has been the focus of intensive research and special purpose surveys can be found in: Desrosier et al. [Desrosier, Dumas, Solomon, Soumis, 1993], Desrochers, Desrosiers and Solomon [Desrochers, Desrosiers and Solomon, 1992], Desrochers et al. [Desrochers, Lenstra, Savelsbergh and Soumis, 1988], Golden and Assad [Golden and Assad, 1988], Solomon and Desrosiers [Solomon and Desrosiers, 1988], Solomon [Solomon, 1987], and Golden and Assad [Golden and Assad, 1986]. Though optimal solutions to VRPTW can be obtained using exact methods, the computational time required to solve a VRPTW to optimality is prohibitive [Desrochers, Desrosiers and Solomon, 1992; Fisher, Jornsten and Madsen, 1992]. Heuristic methods often produce optimal or near optimal solutions in a reasonable amount of computer time. Thus, there is still a considerable interest in the design of new heuristics for solving large-sized practical VRPTW.

Heuristic approaches for the VRPTW use route construction, route improvement or methods that integrate both route construction and route improvement. Solomon [Solomon, 1987] designed and analyzed a number of route construction heuristics, namely: the savings, time-oriented nearest neighbor insertion and a time oriented sweep heuristic for solving the VRPTW. In his study, the time-oriented nearest neighbor insertion heuristic was found to be very successful. Other route construction procedures that have been employed to solve VRPTW are the parallel insertion method [Potvin and Rousseau, 1993], the greedy randomized adaptive procedure [Kontoravadis and Bard, 1992] and the Generalized Assignment heuristic [Koskosidis, Powell and Solomon, 1992]. A number of route improvement heuristics have been implemented for the VRPTW, namely branch exchange procedures [Solomon, Baker and Schaffer, 88; Baker and Schaffer, 1986] and cyclic-transfer algorithms [Thompson and Psaraftis, 1989]. Heuristic search strategies based on Genetic Algorithms, Simulated Annealing and Tabu Search have also been explored for solving the VRPTW.

Potvin and Bengio [Potvin and Bengio, 1993] implemented a genetic based algorithm to solve the VRPTW, in which a population of solutions evolve from one generation to another by mating with parent solutions resulting in offsprings that exhibit characteristics acquired from the parents. A search strategy based on the Tabu Search [Potvin, Kervahut, Garcia and Rousseau, 1992] was implemented that uses a branch exchange procedure to improve VRPTW solutions. Recently a search method based on Simulated Annealing and Tabu Search [Chiang and Russell, 1993] was used to solve the VRPTW.

A heuristic search strategy based on Genetic Algorithms was implemented by Thangiah [Thangiah, 1993; Thangiah, Nygard and Juell, 1991]. This paradigm uses a cluster-first route-second approach to solve the VRPTW. The Genetic Algorithm is used to find a set of customer clusters that are routed using the cheapest insertion procedure and the routes are improved using a post-optimization procedure which resulted in good feasible solutions. The work of Osman [Osman, 1993b] in using Simulated Annealing and Tabu Search for VRP showed that customer interchange methods guided by Simulated Annealing and Tabu Search obtained solutions that were significantly better than solutions obtained by competing heuristic methods.

In this paper we investigate the use of a customer interchange method to improve solutions using a descent algorithm, Simulated Annealing and Tabu Search. The initial solutions for the problem are obtained using a Push-Forward Insertion Heuristic [Solomon, 1987] and the Genetic Sectoring Heuristic [Thangiah, 1993]. The heuristics implemented will be collectively called as the GenSAT system. The GenSAT system was used to solve 60 VRPTW problems obtained from the literature

with customer sizes varying from 100 to 417. The GenSAT system obtained 40 new best known solutions. In addition 11 solutions obtained by GenSAT equal previously known best solutions. The average performance of the GenSAT system is better than other known competing heuristic methods.

The paper is arranged as follows. Section 2 describes the structure of the local search methods. The λ-interchange generation mechanism is the core strategy for generating neighbors. The Push-Forward Insertion heuristic to obtain an initial solution to the VRPTW and the λ-interchange generation mechanism to further improve the solution is also described. Section 3 describes the Simulated Annealing and hybrid Simulated Annealing/Tabu Search implementations that guides the descent algorithm using the λ-interchange generation mechanism. The Genetic Sectoring heuristic that is used to obtain initial solutions for the VRPTW is also described. Section 4 reports the computational experience on 60 VRPTW problems obtained from the literature and the analysis of the results with other competing methods and known optimal solutions. Section 5 gives the summary and concluding remarks.

2. Structure of Local Search Methods

This section describes the various notations and features that are common to each of the implemented local search strategies. A λ-interchange local search descent method is used to improve initial solutions from a route construction method. This section describes the route construction heuristic for obtaining an initial solution to the VRPTW, the λ-interchange mechanism to generate neighboring solutions for improving the initial solution, the evaluation method for computing cost of changes, and the two different strategies for selecting neighbors.

2. 1. Notation

The following notations will help in the description of the methods used for solving VRPTW.

q_k = total capacity of vehicle k, where $k = 1,...,N$

K	=	total number of vehicles.
N	=	total number of customers.
C_i	=	customer i, where $i=1,...,N$.
C_0	=	central depot.
d_{ij}	=	Euclidean distance (proportional to the travel time) from customer i to j,

		where $i,j = 0,...,N$. 0 is the central depot.
e_i	=	earliest arrival time at customer i, where $i = 1,...,N$.
l_i	=	latest arrival time at customer i, where $i = 1,...,N$.
t_i	=	total travel time to reach customer i, where $i = 1,...,N$.
u_{ij}	=	urgency of the customer j, i.e. $u_{ij} = l_j - (t_i + d_{ij})$, where $i, j = 1,...,N$.
a_i	=	service time for customer i, where $i = 1,...,N$.
b_{ij}	=	waiting time for customer j, i.e. b_{ij} = Max $[e_j - (t_i + d_{ij}), 0]$ where $i, j = 1,...,N$.
p_i	=	polar coordinate angle of customer i, where $i = 1,...,N$.
R_k	=	vehicle route k, where $k=1,...,K$.
O_k	=	total overload for vehicle route k, where $k=1,...,K$.
T_k	=	total tardiness for vehicle route k, where $k=1,...,K$.
D_k	=	total distance for a vehicle route k, where $k=1,...,K$.
W_k	=	total travel time (total distance+total waiting time+total service time) for a vehicle route k, where $k=1,...,K$.
$C(R_k)$	=	cost of the route R_k based on a cost function.
$C(S)$	=	sum total cost of individual routes $C(R_k)$.
α	=	weight factor for the total distance travelled by a vehicle.
β	=	weight factor for the urgency of a customer.
γ	=	weight factor for the polar coordinate angle of a customer.
ϕ	=	weight factor for the travel total time of a vehicle.
η	=	penalty weight factor for an overloaded vehicle.
κ	=	penalty weight factor for the total tardy time in a vehicle route.

2. 2. Push-Forward Insertion Heuristic (PFIH)

The Push-Forward Insertion method for inserting customers into a route for the VRPTW was introduced by Solomon [Solomon, 1987]. It is an efficient method for computing the cost of inserting a new customer into the current route. Let us assume a route $R_p = \{C_1,...,C_m\}$ where C_1 is the first customer and C_m is the last customer with their earliest arrival and latest arrival time defined as e_1, l_1 and e_m, l_m respectively. The feasibility of inserting a customer into route R_p is checked by inserting the customer between all the edges in the current route and selecting the edge that has the lowest travel cost. For a customer C_i to be inserted between C_0 and customer C_1, the insertion feasibility is checked by computing the amount of time that the arrival time of t_1 is pushed forward. A change in the arrival time of t_1 could affect the arrival time of all the successor customers of C_1 in the current route. Therefore, the insertion feasibility for C_i needs to be computed by sequentially checking the Push-Forward values of all the successor customers C_j of C_i. The Push-Forward value for a customer C_j is 0 if the time propagated by the predecessor customer of C_j, by the insertion of C_i into the route, does not affect the arrival time t_j. The sequential checking for feasibility is continued until the Push-Forward value of a customer is 0 or a customer is pushed into being tardy. In the worst case, all customers are checked for feasibility.

The Push-Forward Insertion Heuristic (PFIH) starts a new route by selecting an initial customer and then inserting customers into the current route until either the capacity of the vehicle is exceeded or it is not time feasible to insert another customer into the current route. The cost function for selecting the first customer C_i is calculated using the following formula:

$$\text{Cost of } C_i = -\alpha\, d_{0i} + \beta\, l_i + \gamma\,((p_i/360)d_{0i}) \qquad (1)$$

The unrouted customer with the lowest cost is selected as the first customer to be visited. The weights for the three criteria were derived empirically and were set to $\alpha = 0.7$, $\beta = 0.1$ and $\gamma = 0.2$. The priority rule in (1) for the selection of the customer depends on the distance, polar coordinate angle and latest time. The polar coordinate angle of the customer with respect to the depot in (1) is normalized in terms of the distance. This normalization allows comparison of the distance, latest deadline and angular value of the customer in terms of a common unit.

Once the first customer is selected for the current route, the heuristic selects from the set of unrouted customers the customer j^* which minimizes the total insertion cost between every edge $\{k, l\}$ in the current route without violating the time and capacity constraints. The customer j^* is inserted in the least cost position between $\{k^*, l^*\}$ in the current route and the selection process is repeated until no further customers can be inserted. At this stage, a new route is created and the above is repeated until all customers are routed. It is assumed that there is an unlimited number of vehicles, K, which is large and determined by the heuristic to route all the customers. The flow of the PFIH is described below.

Step PFIH-1: *Begin with an empty route starting from the depot.*
Set r=1.

Step PFIH-2: *If {all customers have been routed} then*
go to step PFIH-8.
For all unrouted customers j: Compute the cost according to (1), and sort them in ascending order of their costs.

Step PFIH-3: *Select the first customer, j*, from the ordered list with the least cost and feasible in terms of time and capacity constraints.*

Step PFIH-4: *Append j* to the current route r and update the capacity of the route.*

Step PFIH-5: *For all unrouted customers j: For all edges {k, l} in the current route, compute the cost of inserting each of the unrouted customers between k and l.*

Step PFIH-6: *Select an unrouted customer j* at edge {k*, l*} that has the least cost.*
If {insertion of customer j between k* and l* is feasible in terms of time*

and capacity constraints} then
insert customer j between k* and l*,*
update the capacity of the current route r, and
go to Step PFIH-5,
else
go to Step PFIH-7.

Step PFIH-7: *Begin a new route from the depot.*
Set r = r + 1.
Go to Step PFIH-2.

Step PFIH-8: *All Customers have been routed.*
Stop with a PFIH solution.

2. 3. λ–interchange Generation Mechanism

The effectiveness of any iterative local search method is determined by the efficiency of the generation mechanism and the way the neighborhood is searched. A λ-interchange generation mechanism was introduced by Osman & Christofides [1989, 1994] for the capacitated clustering problem. It is based on customer interchange between sets of vehicle routes and has been successfully implemented with a special data structure to other problems in [Osman 1993b], [Osman 1993c], [Osman & Salhi 1994] and [Thangiah, Osman, Vinayagamoorthy and Sun 1994]. The λ-interchange generation mechanism for the VRPTW is described as follows.

Given a solution for the VRPTW represented by $S = \{R_1,...,R_p,...,R_q,...,R_k\}$ where R_p is a set of customers serviced by a vehicle route p, a λ–interchange between pair of routes R_p and R_q is a replacement of subsets $S_1 \subseteq R_p$ of size $|S_1| \leq \lambda$ by another subset $S_2 \subseteq R_q$ of size $|S_2| \leq \lambda$, to get two new route sets $R'_p = (R_p - S_1) \cup S_2$, $R'_q = (R_q - S_2) \cup S_1$ 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 by the λ–interchange method for a given integer λ.

The order in which the neighbors are searched is specified as follows. Let the permutation σ be the order of vehicle indices in a given solution $S = \{R_1,...,R_p,...,R_q,...,R_k\}$ (say, $\sigma(p) = p, \forall p \in K$). An ordered search selects all possible combination of pairs (R_p,R_q) according to (2) and σ without repetition. A total number of $\frac{K \times (K-1)}{2}$ different pair of routes (R_p, R_q) are examined to define a cycle of search in the following order:

$$(R_{\sigma(1)}, R_{\sigma(2)}),..., (R_{\sigma(1)}, R_{\sigma(k)}), (R_{\sigma(2)}, R_{\sigma(k)}),... (R_{\sigma(k-1)}, R_{\sigma(k)}) \qquad (2)$$

For heuristics based on a descent algorithm and Tabu Search the same permutation σ is used after each cycle of search is completed. Furthermore for a given pair (R_p, R_q) we must also define the

search order for the customers to be exchanged. We consider the case of λ=1 and λ=2 for the neighboring search. The λ-interchange method between two routes results in customers either being shifted from one route to another, or customers being exchanged with other customers. 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 shifting of customers between two routes. 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) indicate exchange of customers between vehicle routes.

The customers in a given pair of routes are searched sequentially and systematically for improved solutions by the shift and exchange process. The order of search we implemented uses the following order of operators (0,1), (1,0), (1,1), (0,2), (2,0), (2,1), (1,2) and (2,2) on any given pairs to generate neighbors. After a solution is generated a criterion is required for accepting or rejecting a move. An acceptance criterion may consider many solutions as potential candidates. Two selection strategies are proposed to select between candidate solutions.

(i) The First-Best (FB) strategy will select the first solution in S' in $N_\lambda(S)$ in the neighborhood of S that results in a decrease in cost with respect to a cost function.

(ii) The Global-Best (BG) strategy will search all solutions S' *in* $N_\lambda(S)$ in the neighborhood of S and select the one which will result in the maximum decrease in cost with respect to a given cost function.

2. 4. Evaluation of a Cost Move

A move which is a transition from one solution to another in its neighborhood may cause a change in the objective function values measured by $\Delta = C(S') - C(S)$. As the λ-interchange move involves insertion of customers into routes, the following cost function is used to compute the cost of inserting customer C_i into route R_k:

$$\text{insertion cost of } C_i = D_k + \phi W_k + \eta O_k + \kappa T_k \qquad (3)$$

The insertion cost function (3) 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 (3). The weight factor for total travel time ϕ was set to one percent of the total distance D_k. When calculating the penalty weight factors η and κ in (3), η was set to ten percent of D_k and κ to one percent of D_k. The penalty values were chosen in this manner to allow penalization relative to the total distance travelled by the

vehicle. This cost function (3) can be similarly generalized for other cases and values of λ.

2. 5. A λ-Interchange Local Search Descent Method(LSD)

In this section, we describe the λ-interchange local search descent method (LSD) that starts from an initial feasible solution obtained by the Push-Forward Insertion Heuristic. The PFIH is further improved using the λ-interchange mechanism. The steps of the λ-interchange LSD are as follows:

Step LSD-1: *Obtain a feasible solution S for the VRPTW using the PFIH.*
Step LSD-2: *Select a solution $S' \in N_\lambda(S)$ in the order indicated by (2).*
Step LSD-3: *If {$C(S') < C(S)$}, then*
accept S' and go to Step LSD-2,
else go to Step LSD-4.
Step LSD-4: *If {neighborhood of $N_\lambda(S)$ has been completely searched (there are no moves that will result in a lower cost} then*
go to Step LSD-5
else go to Step LSD-2.
Step LSD-5: *Stop with the LSD solution.*

The LSD solution is dependent upon the initial feasible solution. The LSD uses two different selection strategies for selection of neighbors. The two strategies are First-Best(FB) and Global-Best(GB). The LSD with the GB search strategy (LSD-GB) is computationally more expensive than the LSD with FB strategy (LSD-FB), as LSD-GB has to keep track of all the improving moves while the LSD-FB is a blind search that accepts the first improving move.

As the LSD-FB and LSD-GB strategies accept only improving moves, the disadvantage of using such methods is that they could get stuck in a local optima and never have the means to get out of it. Simulated Annealing is a search strategy that allows non-improving moves to be accepted with a probability in order to escape the local optima while improving moves are always accepted as in the local search descent method.

3. Meta-heuristic Methods

The past decade saw the rise of several general heuristic search schemes that can be adopted with a variable degree of effort to a wide range of combinatorial optimization problems. These search schemes are often referred to as “meta-heuristics”, “meta-strategies”, or “modern heuristics”. We refer for further details to the recent bibliography on these techniques by Reeves [Reeves, 1993]

and Osman and Laporte [Osman and Laporte, 1994]. The following sections describe the meta-heuristic search strategies that were implemented to solve the VRPTW. The structure of each search strategy and its parametric values are described individually.

3. 1. Simulated Annealing

Simulated Annealing (SA) is a stochastic relaxation technique which has its origin in statistical mechanics [Metropolis et al., 1953; Kirkpatrick, Gelatt and Vecchi, 1983]. The Simulated Annealing methodology draws its analogy from the annealing process of solids. In the annealing process, a solid is heated to a high temperature and gradually cooled in order for it to crystallize. As the heating process allows the atoms to move randomly, if the cooling is done too rapidly it prevents the atoms from reaching thermal equilibrium. If the solid is cooled slowly, it gives the atoms enough time to align themselves in order to reach a minimum energy state. This analogy can be used in combinatorial optimizations with the states of the solid corresponding to the feasible solution, the energy at each state corresponding to the improvement in objective function and the minimum energy being the optimal solution.

The interest in SA to solve combinatorial optimization problems began with the work of Kirkpatrick et al. [1983]. Simulated Annealing (SA) uses a stochastic approach to direct the search. It allows the search to proceed to a neighboring state even if the move causes the value of the objective function to become worse. Simulated annealing guides the original local search method in the following way. If a move to a neighbor S' in the neighborhood $N_\lambda(S)$ decreases the objective function value, or leaves it unchanged then the move always accepted. More precisely, the solution S' is accepted as the new current solution if $\Delta \leq 0$, where $\Delta = C(S') - C(S)$. To allow the search to escape a local optimum, moves that increases the objective function value are accepted with a probability $e^{(-\Delta/T)}$ if $\Delta > 0$, where T is a parameter called the "temperature". The value of T varies from a relatively large value to a small value close to zero. These values are controlled by a cooling schedule which specifies the initial and temperature values at each stage of the algorithm.

We use a non-monotonic cooling schedule similar to that outlined in [Osman 1993b, Osman and Christofides, 1994]. The non-monotonic reduction scheme reduces the temperature after each generated move (one iteration) with occasional temperature increases (or higher temperature resets) after the occurrence of a special neighborhood without accepting any moves. The design of the non-monotonic cooling schedule in Osman [1993b] to induce an oscillation behavior in the temperature consequently in the objective function is a kind of strategic oscillation concept borrowed from tabu search [Glover 1986; Glover 1993]. The first hybrid combination of the non-monotonic search with simulated annealing concept was pioneered by Osman [Osman 1991;

Osman 1993b]. This combination has been shown to yield better and improved performance over other standard SA approaches on a number of problems [Osman, 1993c; Osman and Christofides, 1994; Hasan nad Osman 1994].

The SA implementation starts from an initial solution which is generated by the Push-Forward Insertion Heuristic. It searches systematically the 2-interchange generation mechanism and allows infeasible moves to be to be accepted/rejected according to a SA criterion using the cost function (3) for move evaluations. The following notation will help in the explanation of the SA algorithm.

T_s = *Starting temperature of the SA method.*
T_f = *Final temperature of the SA method.*
T_b = *Temperature at which the best current solution was found.*
T_r = *Reset temperature of the SA method.*
S = *Current solution.*
S_b = *Best current route found so far in the search.*
R = *Number of resets to be done.*
τ = *a decrement constant in the range of* $0 < \tau < 1$.

The flow of the Simulated Annealing method can be described as follows:

Step SA-1: *Obtain a feasible solutions for the VRPTW using the PFIH heuristic.*
Step SA-2: *Improve S using the 2-interchange local search descent method with the First-Best selection strategy.*
Step SA-3: *Set the cooling parameters.*
Step SA-4: *Generate systematically a* $S' \in N_\lambda(S)$ *and compute* $\Delta = C(S') - C(S)$.
Step SA-5: *If* $\{(\Delta \leq 0)$ *or* $(\Delta > 0$ *and* $e^{(-\Delta/T_k)} \geq \theta)$, *where* θ *is a random number between* $[0,1]\}$ *then*
set $S = S'$.
if $\{C(S') < C(S_b)\}$ *then*
improve S' *using the local search procedure in Step-SA2,*
update $S_b = S'$ *and* $T_b = T_k$.
Step SA-6: *Set* $k = k + 1$.
Update the temperature using: $T_{k+1} = \frac{T_k}{(1 + \tau\ T_k)}$

If $\{N_2(S)$ *is searched without any accepted move}* *then*
set $T_r = maximum\ \{T_r / 2, T_b\}$, *and set* $T_k = T_r$,
Step SA-7: *If* $\{R$ *resets were made since the last* S_b *was found}* *then*
go to Step SA-8,
else go to Step SA4.
Step SA-8: *Terminate the SA algorithm and print the* S_b *routes.*

The above hybrid procedure combines the SA acceptance criteria with the local search descent algorithm and the strategic oscillation approach embedded in the cooling schedule. In our

implementation, and after experimentation, the initial parameters for the cooling schedule is set at T_s=50, T_f =0, T_r=T_s, R=3, S_b= S *and* k =1. After each iteration k, the temperature is decreased according to a parameter τ which was set to 0.5. Initial experiments using only feasible moves resulted in solutions that were not competitive with solutions obtained from competing heuristics. The LSD-FB selection strategy was used by the SA method to select candidate moves. The Simulated Annealing method, at times, could get caught in a succession of moves that could result in a move being made in state S that is reversed in state S'. In order to avoid moves that result in cycles and also force the search to explore other regions a hybrid combination of the Tabu Search and Simulated Annealing method was implemented.

3. 2. Tabu Search

Tabu search (TS) is a memory based search strategy to guide the local search descent method to continue its search beyond local optimality [Glover, 1989; Glover 1990]. When a local optimum is encountered, a move to the best neighbor is made to explore the solution space, even though this may cause a deterioration in the objective function value. The TS seeks the best available move that can be determined in a reasonable amount of time. If the neighborhood is large or its elements are expensive to evaluate, candidate list strategies are used to help restrict the number of solutions examined on a given iteration.

Tabu search uses memory structures to record attributes of the recent moves when going from one solution to another in a Tabu List. Attributes of recently visited neighborhood moves are designated as Tabu and such moves are not permitted by the search strategy for the duration that it is considered to be Tabu. The duration that an attribute remains on a tabu list is determined by the Tabu List Size (TLS). A special degree of freedom is introduced by means of an aspiration concept and the tabu status of a move can be overruled if certain aspiration conditions are met. Since TS is a heuristic method the rule for execution is generally expressed as a pre-specified limit on the number of iterations or on the number of iterations since the last improvement was found. More details on the recent developments and applications can be found in Glover, Taillard and De Werra (1993).

A new modification to the TS was implemented to solve the VRPTW. The TS algorithm was combined with the SA acceptance criterion to decide which moves to be accepted from the candidate list. The combined algorithm uses a special data structure which identifies the exact candidate list of moves rather than using a sampling approach of the neighborhood. The hybrid algorithm using the TS elements was implemented as follows. Given a solution $S = \{R_1,...,R_k\}$

with K vehicles, there are $\frac{K\times(K-1)}{2}$ pairs of $\{R_p,R_q\}$ routes considered by the 2-interchange mechanism to generate the whole neighborhood $N_2(S)$. The best solution in terms of the objective function value generated from each $\{R_p,R_q\}$ is recorded. The set of all these best solutions form the candidate list of solutions. Two matrices are used to record the $\frac{K\times(K-1)}{2}$ best solutions. *BestCost* and *BClist* are matrices with dimensions $K\times K$, and $\frac{K\times(K-1)}{2}\times 8$. The top triangular part of the matrix *BestCost (p, q)*, $1 \le p < q \le K$, stores the objective value Δ_{pq} associated the best move between (R_p,R_q).

The lower triangular part of *BestCost(q,p)* stores an index *l* indicating the row in *BClist* where the information (customers exchanged, route indices, etc.) associated with *BestCost(q,p)* are recorded for faster retrieval. The minimum attributes associated with a move are the indices of the maximum number of customers to be interchanged and of the two routes leading to six column entries in *BClist*. However, more information can be stored in *BClist* by adding more columns if necessary. The importance of *BestCost* and *BClist* is that they can be updated by evaluating only moves in the $2\times K$ pairs of routes rather than looking at $\frac{K\times(K-1)}{2}$ pairs in $N_2(S)$. After a move involving $\{R_p,R_q\}$ is done, the pairs of routes $N_2(S)$ which do not involve either R_p or R_q remain intact. The pairs of routes which need evaluations are $\{R_p,R_t\},\{R_t,R_q\}\ \forall t$ in K.

The tabu list structure is represented by a matrix *TabuList* with dimension $K\times N$. Each *TabuList (p,i)* records the iteration number at which a customer *i* is removed from route R_p plus the value of the tabu list size TSL. As the neighborhood size generated by the 2-interchange mechanism is large, a value for TSL was set to 10 and found to be sufficient. When a move is done that involves four customers exchanged between routes, the appropriate elements are added to the *TabuLis*t. The above structure can easily be checked to identify the tabu status of the current move. For example, at iteration *m*, if *TabuList (p, i)* is greater than *m*, then a move which returns customer *i* to route R_p is considered Tabu. A tabu status of a move S' is overruled if its objective function value $C(S')$is smaller than the objective value of the best solution $C(S_b)$ found so far.

The hybrid Tabu Search and Simulated Annealing (TSSA) algorithm steps can be described as follows:

Step TSSA-1: *Obtain an initial PFIH solution S.*
Step TSSA-2: *Improve S using the 2-interchange local search descent algorithm with the First-Best selection strategy.*
Step TSSA-3: *Initialize TabuList to zeros, set S_b = S and set the iteration counter m=0.*
Step TSSA-4: *Set the Simulated Annealing cooling schedule parameters.*
Step TSSA-5: *If (m = 0) then*

update the BestCost and BClist matrices from information in Step TSSA
else
update the matrices of the candidate list as necessary to reflect the changes due to the performed move.

Step TSSA-6: *Select $S' \in N_2(S)$ with the largest improvement or least non-improvement from the candidate list of moves in BestCost.*

Step TSSA-7: *If {S' is tabu} then*
if {$C(S') < C(S_b)$} then
go to Step TSSA-9,
else
go to Step TSSA-6 to select the next best solution $S' \in$ BestCost.
else if {S' is not tabu} then
go to Step TSSA-8.

Step TSSA-8 *Accept or Reject S' according to the Simulated Annealing criterion.*
If {S' is accepted} then
go to Step TSSA-9.
else
go to Step TSSA-6 to select the next best solution $S' \in$ BestCost.

Step TSSA-9: *Update $S = S'$, and TabuList and other SA parameters.*
If {$C(S) < C(S_b)$} then
set $S_b = S$,
set $m = m + 1$, and
go to Step TSSA-10.

Step TSSA-10: *If {m is greater than a given number of iterations $K \times N$} then*
go to Step TSSA-11.
else go to Step TSSA-5.

Step TSSA-11: *Terminate the hybrid TSSA algorithm and print the S_b routes.*

Instead of using the PFIH for obtaining an initial solution, the Genetic Sectoring Heuristic, a clustering algorithm based on Genetic Algorithms, can be used for obtaining customer clusters. The next section explains the structure of the Genetic Sectoring heuristic.

3. 3. Genetic Algorithms

The Genetic Algorithm (GA) is an adaptive heuristic search method based on population genetics. The basic concepts of a GA were primarily developed by Holland [Holland, 1975]. Holland's study produced the beginning of the theory of genetic adaptive search [DeJong, 1980; Grefenstette, 1986; Goldberg, 1989].

The GA is an iterative procedure that maintains a population of *P* candidate members over many simulated generations. The population members are string entities of artificial chromosomes. The

chromosomes are fixed length strings with binary values (or alleles) at each position (or locus). Allele is the 0 or 1 value in the bit string, and the Locus is the position at which the 0 or 1 value is present in each location of the chromosome. Each chromosome has a fitness value associated with it. The chromosomes from one generation are selected for the next generation based on their fitness value. The fitness value of a chromosome is the payoff value that is associated with a chromosome. For searching other points in the search space, variation is introduced into the population chromosomes by using crossover and mutation genetic operators. Crossover is the most important genetic recombination operator. After the selection process, a randomly selected proportion of the chromosomes undergo a two point crossover operation and produce offsprings for the next generation.

Selection and crossover effectively search the problem space exploring and exploiting information present in the chromosome population by selecting and recombining primarily the offsprings that have high fitness values. These two genetic operations generally produce a population of chromosomes with high performance characteristics. Mutation is a secondary operator that prevents premature loss of important information by randomly mutating alleles within a chromosome. The adaptations in a GA are achieved by exploiting similarities present in the coding of the chromosomes. The termination criteria of a GA are convergence within a given tolerance or realization of the maximum number of generations to be simulated.

A clustering method using the GA has been highly successful in solving vehicle routing problems with time constraints, multiple depots and multiple commodities [Thangiah, 1993; Thangiah, Vinayagamoorthy and Gubbi, 1993; Thangiah and Nygard, 1993; Thangiah and Nygard, 1992a,1992b; Thangiah, Nygard and Juell, 1991]. We investigate the use of the genetic clustering method and show how it can be combined with other meta-heuristics for solving VRPTW for a large number of customers.

The GA clustering method is based on the cluster-first route-second approach. That is, given a set of customers and a central depot, the heuristic clusters the customers using the GA, and the customers within each sector are routed using the cheapest insertion method [Golden and Stewart, 1985]. The GA solution can be improved using the LSD-FB, LSD-GB, SA, or TSSA heuristics. The clustering of customers using a GA is referred to as Genetic Sectoring. The Genetic Sectoring Heuristic (GSH) 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 local search meta-heuristics. The following notations will help in the description of the Genetic Sectoring heuristic.

s_i	=	pseudo polar coordinate angle of customer i, where $i = 1,...,N$.
F	=	fixed angle for Genetic Sectoring, $\text{Max}[s_i,...,s_n]/2K$, where $n = 1,...,N$.
M	=	maximum offset of a sector in Genetic Sectoring, $M= 3F$.
B	=	length of the bit string in a chromosome representing an offset, $B=5$.
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, where $k=1,...,K-1$.
S_k	=	seed angle for sector k, where $k=1,...,K-1$.
S_0	=	initial seed angle for Genetic Sectoring, $S_0=0$.

The GENESIS [Grefenstette, 1987] genetic algorithm software was used in the implementation of the GSH. 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 total cost of serving all the customers computed with respect to the sector divisions derived from it. The GSH is an extension of the clustering method of Fisher and Jaikumar [Fisher and Jaikumar, 1981] and Gillett and Miller [Gillett and Miller, 1974].

In a N customers problem with the origin at the depot, the GSH replaces the customer angles $p_1,...,p_N$ with pseudo polar coordinate angles. 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 the PFIH. 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, R_k, based on the following equation:

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

Customer C_i is assigned to vehicle R_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. The maximum offset, *M*, was set to three times the fixed angle to allow for large variations in the size of the sectors during the genetic search. 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 GSH is invoked, is 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 derived empirically and was set at 5 bits. The decimal conversion of 5 bits results in a range of integer values between 0 and 31. The offsets are derived proportionately from the decimal conversion of the bit values using the decimal value 0 as a 0^{o} offset and the bit value 31 as the maximum offset. Figure 1 describes the chromosome mapping used to obtain the offsets.

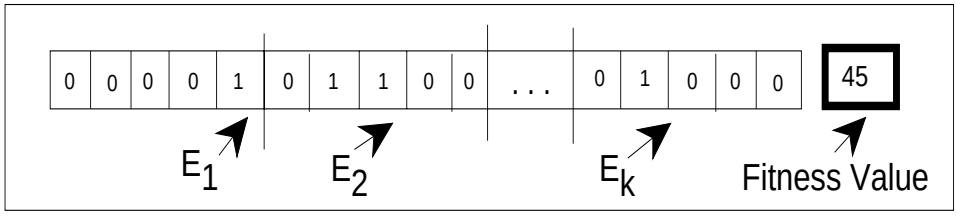

Figure 1: Representation of the offsets using a chromosome. Each offset is represented by five bits in the chromosome. The fitness value of the chromosome is the total route cost obtained using the offsets obtained from the chromosome.

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

$$S_i = S_{i-1} + F + \left(3^{E_i \left(\frac{\log M}{\log 3} \right)} \right) \left(\frac{M}{2^B} \right) \qquad (5)$$

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 cost function (5) for calculating the seed angles uses an exponential function. The exponential function allows for large fluctuations in the seed angles with respect to the offsets derived from the chromosomes during the Genetic Sectoring process. The customers within the sectors, obtained from the chromosomes, are routed using the cheapest insertion method described in Section 2.4.

In the GSH a chromosome represents a set of offsets for the 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 GSH 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 VRPTD. The GSH would utilize more computer time than the PFIH for obtaining a solution because it has to evaluate $P \times G$ vehicle routes, where P is the population size and G is the number of generations to be simulated.

The parameter values for the number of generations, population size, crossover and mutation rates for the Genetic Sectoring process were derived empirically and were set at 1000, 50, 0.6 and 0.001. During the simulation of the generations, the GSH 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 infeasibility in a solution arises because of overloading or tardiness in a vehicle route. The solution obtained from the GA is improved using the LSD-FG, LSD-GB, SA and TSSA methods. The GSH method can be described as follows.

Step GSH-1: *Set the bit string size for the offset: Bsize=5.*
Set the variable, NumVeh, to the number of vehicles required by the PFIH to obtain a feasible solution.

Step GSH-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 GSH-3: *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 {cost of the current set of sectors is lower than l} then
set l to the current route cost, and
save the set of sectors in lr.
If {cost of the current set of sectors is lower than g} then
set g to the current route cost, and
save the set of sectors in gr.
If {all the chromosomes have not been processed} then
go to Step GSH-3.
else go to Step GSH-4.

Step GSH-4: *Do Selection, Crossover and Mutation on the chromosomes.*
Go to Step GSH-3.

Step GSH-5: *Improve the routes using LSD-FB, LSD-GB, SA and TSSA methods.*

Step GSH-6: *Terminate the GSH and print the best solution found.*

4. Computational Results

The heuristic methods LSD-FB, LSD-GB, SA and TSSA are collectively referred to as the GenSAT system. The heuristics in the GenSAT system were applied to the six data VRPTW sets R1, C1, RC1, R2, C2, and RC2 generated by Solomon [Solomon, 1987] consisting of 100 customers with Euclidean distance. In these problems, the travel time between the customers are equal to the corresponding Euclidean distances. The data consist of geographical and temporal differences in addition to differences in demands for the customers. Each of the problems in these data sets has 100 customers. The fleet size to service them varied between 2 and 21 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.Problem sets R1, C1 and RC1 have narrow scheduling horizon. Hence, only a few customers can be served by the same vehicle. Conversely, problem sets R2, C2 and RC2 have large scheduling horizon, and more customers can be served by the same vehicle.

In addition to the problem sets generated by Solomon, the heuristics were applied to four real

world problems consisting of 249 and 417 customers. The first problem consisting of 249 customers is reported by Baker and Schaffer [Baker and Schaffer, 1986]. Two problems, D249 and E249, are generated from this problem by setting the capacity of the vehicles to 50 and 100. The second problem consisting of 417 customers is based on a route for a fast food industry located in southeastern United States. The two problems, D417 and E417, consisting of 417 customers differ in that E417 has a larger number of tight time windows than D417.

Comparison of heuristics within the GenSAT system

Table 1 lists the average number of vehicles and distance travelled for the VRPTW problems using the PFIH and GSH to obtain initial solutions and the LSD-FB, LSD-GB, SA and TSSA methods to improve the solutions.

A detailed listing of the initial solutions obtained using the PFIH and GSH and improved using the LSD-FB, LSD-GB, SA and TSSA methods are listed in Appendices A and B. For problems in which the customers are clustered and have long time windows, that is data sets C1 and C2, the PFIH+TSSA obtains better average solutions in comparison to the other heuristics. The GS+LSD-FB obtains the best average solutions for problems that have uniformly distributed customers with vehicles that have large capacity and routing time. The GS+TSSA heuristic does well for problems in which the customers are uniformly distributed and have tight time windows. The best average solutions in terms of the number of vehicles and total distances are obtained by the heuristics with respect to the data sets are highlighted in bold. The highlighted entries compare the combined average of the minimum number of vehicles followed by the total distance travelled obtained by the heuristics for the data sets. The comparison is done in this manner as it is possible to reduce the total distance travelled by increasing the number of vehicles. For example in Table 1 when comparing only the distance the PFIH+SA heuristic obtains a average value of 1249 for the total distance travelled for the RC2 data set. The GSH+LSD-FB has an average total distance of 1254 leading to the conclusion that the PFIH-SA does better than GSH+LSD-FB. This is misleading because the average number of vehicles required by the PFIH+SA is 3.8 while GSH+LSD-FB requires 3.4.

Table 1: Average number of vehicles and distance obtained for the six data sets using two heuristics, Push-Forward Insertion and Genetic Algorithm, for obtaining initial solutions and four heuristics LSD-FB, LSD-GB, SA and TSSA for improving the solution.

Prob.	Push-Forward Insertion Heuristic				Genetic Sectoring Heuristic			
	LSD-FB	LSD-GB	SA	TSSA	LSD-FB	LSD-GB	SA	TSSA
R1	14.8 1326	14.0 1307	13.7 1252	13.3 1242	12.4 1289	12.5 1289	12.4 1287	**12.4** **1242**
C1	10.0 850	10.0 847	10.0 883	**10.0** **831**	10.0 969	10.0 993	10.0 937	10.0 874
RC1	13.8 1459	13.8 1483	13.4 1454	13 1413	12.3.0 1511	12.2 1499	12.1 1471	**12.0** **1447**
R2	3.4 1233	3.6 1178	3.2 1169	3.2 1113	**3.0** **1023**	3.1 1056	3.2 1052	3.2 1031
C2	3.1 708	3.4 703	3.0 687	**3.0** **663**	3.0 673	3.0 709	3.0 684	3.0 676
RC2	4.0 1323	3.9 1262	3.8 1249	3.9 1257	**3.4** **1254**	3.4 1255	3.4 1307	3.4 1261

Legend:

Prob.: VRPTW data set.
LSD-GB::Local descent with Global-Best search strategy.
TSSA: Tabu Search Simulated Annealing with Global Best.
LSD-FB: Local descent with First-Best strategy.
SA: Simulated Annealing+ local descent with First Best Strategy.

Comparison of the GenSAT system with Competing Heuristics

Table 2 compares the average number of vehicle and distance obtained by eight different heuristics and the GenSAT system. The eight different heuristics are diverse in their approach to solving VRPTW. The GenSAT system obtains the best average solutions for data sets R1, C1, RC1 and RC2 in comparison to the eight competing heuristics. The Chiang-Russell heuristic obtains the best average solution for data set R2 and the PTABU heuristic for data set C2. The GenSAT system obtains the best average number of vehicles and total distance travelled for data sets R1, C1, RC1 and R2. For data set R2 the Chiang-Russell algorithm, in comparison to the GenSAT system, obtains better average number of vehicles but has a larger average total distance. For data set C2, the GENEROUS algorithm obtains the best average number of vehicles and total distance in comparison to all the competing algorithms.

Table 2: Average number of vehicles and distance obtained for the six data sets by the GenSAT system and six other competing heuristics.

Prob.	Solomon	CTA	PARIS	GIDEON	GRASP	PTABU	GENEROUS	Chiang/ Russell	GenSAT
R1	13.6 1437	13.0 1357	13.3 1696	12.8 1300	13.1 1427	12.6 1378	12.6 1297	12.4 1290	**12.3** **1238**
C1	10.0 952	10.0 917	10.7 1610	10.0 892	10.6 1401	10.0 861	10.0 838	10.0 886	**10.0** **832**
RC1	13.5 1722	13.0 1514	13.4 1877	12.5 1474	12.8 1603	12.6 1473	12.1 1446	12.5 1446	**12.0** **1284**
R2	3.3 1402	3.2 1276	3.1 1513	3.2 1125	3.2 1539	3.1 1171	3.0 1118	**2.9** **1144**	3.0 1005
C2	3.1 693	3.0 645	3.4 1478	3.0 749	3.4 858	3.0 604	**3.0** **590**	3.0 659	3.0 650
RC2	3.9 1682	3.7 1634	3.6 1807	3.4 1411	3.6 1782	3.4 1470	3.4 1368	3.4 1361	**3.4** **1229**

Legend:

Prob.:	VRPTW data set.		
Solomon:	Solomon, 1987.	CTA:	Thompson and Psaraftis, 1993.
PARIS:	Potvin and Rousseau, 1993.	GIDEON:	Thangiah, 1993.
GRASP:	Kontoravdis and Bard, 1992.	PTABU:	Potvin et al., 1993.
GENEROUS:	Potvin and Bengio, 1994.	Chiang/Russell:	Chiang and Russell, 1994.

Table 3 compares each of the solutions obtained by the GenSAT system against six competing heuristics that have reported individual solutions obtained for the 60 problems. In Table 3 “NV” indicates the total number of vehicles and “TD” the total distance required to obtain a feasible solution. The column “Best Known” lists the previously best known solution from six different heuristics that have reported individual solutions for each of the problems. The “GenSAT” column contains the best solution obtained by using two different heuristics to obtain an initial solution, Push-Forward Insertion and Genetic Sectoring, and four different local heuristics LSD-FB, LSD-GB, SA, TAAS for improving the solution. The solutions obtained by GenSAT are better than previously known best solutions, in terms of the number of vehicles and total distance, are highlighted in bold.

Table 3: Comparison of the best known solutions with the GenSAT solutions for data sets R1, C1, RC1, R2, C2, RC2 and problems D249, E249, D417 and E417.

Prob.	Best Known		GenSAT		Prob.	Best Known		GenSAT		Prob.	Best Known		GenSAT	
	NV	TD	NV	TD		NV	TD	NV	TD		NV	TD	NV	TD
R101	19	1704[e]	**18**	**1644**	RC101	15	1676[d]	**14**	**1669**	C201	3	590[a]	3	591
R102	17	1549[b]	**17**	**1493**	RC102	14	1569[b]	**13**	**1557**	C202	3	591[c]	3	707
R103	13	1319[b]	**13**	**1207**	RC103	11	1138[b]	**11**	**1110**	C203	3	592[d]	3	791
R104	10	1065[d]	**10**	**1048**	RC104	10	1204[d]	10	1226	C204	3	591[d]	3	685
R105	14	1421[d]	14	1442	RC105	14	1612[b]	**14**	**1602**	C205	3	589[c]	3	589
R106	12	1353[c]	**12**	**1350**	RC106	12	1486[e]	13	1420	C206	3	588[c]	3	588
R107	11	1185[c]	**11**	**1146**	RC107	11	1275[d]	**11**	**1264**	C207	3	588[c]	3	588
R108	10	1033[e]	**10**	**898**	RC108	11	1187[d]	**10**	**1281**	C208	3	588[c]	3	588
R109	12	1205[d]	12	1226										
R110	11	1136[c]	**11**	**1105**										
R111	11	1184[c]	**10**	**1151**										
R112	10	1003[f]	**10**	**992**										
C101	10	829[a]	10	829	R201	4	1478[b]	**4**	**1354**	RC201	4	1734[e]	**4**	**1294**
C102	10	829[d]	10	829	R202	4	1279[b]	**4**	**1176**	RC202	4	1459[e]	**4**	**1291**
C103	10	873[b]	**10**	**835**	R203	3	1167[b]	**3**	**1126**	RC203	3	1253[d]	**3**	**1203**
C104	10	865[d]	**10**	**835**	R204	2	904[d]	**3**	**803**	RC204	3	1002[d]	**3**	**897**
C105	10	829[a]	10	829	R205	3	1159[d]	**3**	**1128**	RC205	4	1594[d]	**4**	**1389**
C106	10	829[d]	10	829	R206	3	1066[d]	**3**	**8338**	RC206	3	1298[e]	**3**	**1213**
C107	10	829[d]	10	829	R207	3	954[d]	**3**	**904**	RC207	3	1194[e]	**3**	**1181**
C108	10	829[d]	10	829	R208	2	759[d]	2	823	RC208	3	1038[b]	**3**	**919**
C109	10	829[d]	10	829	R209	3	1108[d]	**2**	**855**					
					R210	3	1146[d]	**3**	**1052**					
					R211	3	898[b]	**3**	**816**					
D249	4	477[e]	**4**	**457**										
E249	5	506[e]	**5**	**495**										
D417	55	4235[e]	**54**	**4866**										
E417	55	4397[e]	**55**	**4149**										

Legend:

Prob.: VRPTW data set.
NV: Number of vehicles
TD: Total Distance
[a]: CTA [Thompson and Psaraftis, 1993]
[c]: TABU [Potvin et al., 1993].
[e]: Chiang/Russell [Chiang and Russell, 1994].

Best Known: Best known solution among competing heuristics.
GenSAT: Best solution obtained using the GenSAT system.

[b]: GIDEON [Thangiah, 1993.]
[d]: GENEROUS [Potvin and Bengio, 1994].

The GenSAT system obtained 40 new best known solutions for the VRPTW problems. In addition it attained previously known best solutions for 11 problems and failed in the other 9 problems.The average performance of GenSAT system is better than other known competing methods. A detailed comparison of solutions obtained by GenSAT with the competing heuristics for each of the VRPTW problems are listed in Appendix C.

Comparison of GenSAT with optimal solutions

In Table 4 the optimal solutions for some of the VRPTW problems reported in [Desrochers, Desrosiers and Solomon, 1992] are compared with the best known solutions and those obtained by the GenSAT system.

Table 4: Comparison of the best known, GenSAT and optimal solution for problems R101, R102, C101, C102, C106 and C108.

Prob.	Optimal	Best Known	GenSAT	% over optimal for best known solution.	% over optimal for GenSAT solution
R101	18 1608	19[d] 1704	18 1677	6% 6%	0% 4%
R102	17 1434	17[b] 1549	17 1505	0% 8%	0% 5%
C101	10 827	10[a] 829	10 829	0% 0.2%	0% 0.2%
C102	10 827	10[c] 829	10 829	0% 0.2%	0% 0.2%
C106	10 827	10[c] 829	10 829	0% 0.2%	0% 0.2%
C108	10 827	10[c] 829	10 829	0% 0.2%	0% 0.2%

Legend:

Prob.: VRPTW data set.
NV: Number of vehicles
TD: Total Distance
[b]: GIDEON [Thangiah, 1993.]
[d]: Chiang/Russell [Chiang and Russell, 1994].
Best Known: Best known among competing heuristics.
GenSAT: Best solution obtained using the GenSAT system.
[a]: CTA [Thompson and Psaraftis, 1993.]
[c]: GENEROUS [Potvin and Bengio, 1994].

For problems C1, C2, C6, and C8 the best known solutions and the GenSAT solutions are near optimal. It is worth noting that the optimal solutions were obtained by truncating the total distance to the first decimal place as reported by [Desrochers, Desrosiers and Solomon, 1992]. For the

R101 problem the best known solution is 6% over the optimal solution in terms of the number of vehicles and the total distance travelled, and GenSAT obtains the optimal number of vehicles and is 4% away from the optimal distance travelled. For R102 problem both of the best known solution and the GenSAT system obtain the optimal number of vehicles required for the customers, but the best known solution is 8% over the optimal distance travelled while the GenSAT solution is only 5% away from the optimal distance.

Computational Time for the GenSAT system

The algorithms in the GenSAT system were written in the C language and implemented on a NeXT 68040 (25Mhz) computer system. The average CPU time required to solve the VRPTW varied with the problems. Table 5 gives the average CPU time taken to solve the problems for the data sets. For problems that have a large number of customers, such as D249, E249, D417 and D417, it took an average CPU time of 26 minutes to solve the problems.

Table 5: Average CPU time for obtaining feasible solutions to the VRPTW problems in min:sec on a Next 68040 (25Mhz) machine.

Prob.	Push-Forward Insertion Heuristic				Genetic Sectoring Heuristic			
	LSD-FB	LSD-GB	SA	TSSA	LSD-FB	LSD-GB	SA	TSSA
R1	0:3	0:4	1:6	9.9	0:2	0:2	0:9	16:6
C1	0:2	0:2	0:4	4:8	0:9	0:1	1:2	0:7
RC1	0:4	0:31	1:8	8:5	0:4	0:2	0:6	8:6
R2	3:3	2:5	20:9	10.7	3:3	1:9	10:5	13:2
C2	0:3	0:3	11.9	4.7	1:2	1:4	2:2	1:8
RC2	3:1	0:6	8:5	17:8	1:2	1:3	7:7	3.4

Legend:

Prob.: VRPTW data set.
LSD-GB: Local Descent with Global-Best search strategy.
TSSA: Tabu Search Simulated Annealing with Global Best.
LSD-FB: Local Descent with First-Best strategy.
SA: Simulated Annealing+ Local Descent with First Best Strategy.

It is very difficult to compare the CPU times for the GenSAT system with those of the competing methods due to differences in the language of implementation, the architecture of the system and the amount of memory available in the system. The Chiang/Russell algorithm was implemented on a 486DX/66 machine and required an average of 2 minutes of CPU time for the 100 customer problems. The GENEROUS system was implemented on a Silicon Graphics workstation and

required an average of 15 minutes of CPU time. The GIDEON system was implemented on a SOLBOURNE 5/802 and the PTABU on a SPARC 10 workstation.

5. Summary and Conclusion

In this paper, we have developed a number of meta-heuristics to successfully solve the vehicle routing problem with time windows (VRPTW). These algorithms are based on a two-phase approach. In the first phase, an initial construction solution is obtained by either the Push-Forward-Insertion or a sectoring heuristic based on Genetic Algorithms. The second phase applies one the following local search procedures which use the λ-interchange mechanism: a local search descent procedure using either a first-best or a global best strategy to select candidate solutions; a hybrid Simulated Annealing algorithm with non-monotonic cooling schedule and a local search descent procedure after each improved solution; a hybrid Simulated Annealing and Tabu Search with a special data structure for fast update of the candidate list of solutions. The main feature of the local search procedures is that infeasible solutions with penalties are allowed if considered attractive.

An extensive computational study on a set of 60 test problems with size varying from 100 to 417 customers is conducted comparing the above procedures with the best known competing procedures from the literature. The GenSAT system obtained new best know solutions for 40 test problems, and were equal in 11 other problems and a very close in the other 9 cases. For example, the GenSAT system obtained different combinations of best solutions using a total of 13 vehicles with 1420 for distance and 3 vehicles with 803 for distance for problems R204, RC106, respectively. However, the best known solutions for the two problems are 12 vehicles with 1486 for distance and 2 vehicles with 914 for distance respectively. It was found that the Genetic based algorithms uses more computational time as compared to the PFIH methods.

Finally, our results indicate that hybrid combinations of the best features of Simulated Annealing, Tabu Search and Genetic Algorithm provided a good algorithm for the solution of the VRPTW. Further research, along this direction to derive a unified hybrid algorithm should be encouraged.

Acknowledgments

The authors wish to thank R. Russell for providing the 200 and 400 customer problem sets. This material is based upon work partly supported by the Slippery Rock University Faculty Development Grant No. FD2E-030, State System of Higher Education Faculty Development Grant No. FD3I-051-1810 and the National Science Foundation Grant No. USE-9250435. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and not necessarily reflect the views of Slippery Rock University, State System of Higher Education or the National Science Foundation.

References

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

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

[3] Chiang, Wen-Chyuan and R. Russell (1993). Simulated Annealing Metaheuristics for the Vehicle Routing Problem with Time Windows, Working Paper, Department of Quantitative Methods, University of Tulsa, Tulsa, OK 74104.

[4] Davis, Lawrence (1991). Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York.

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

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

[7] Desrochers, M., J. K. Lenstra, M. W. P. Savelsbergh and F. Soumis (1988). *Vehicle Routing with Time Windows: Optimization and Approximation*. Vehicle Routing: Methods and Studies, B. Golden and A. Assad (eds.), North Holland.

[8] Desrosier J., Y. Dumas, M. Solomon, F. Soumis (1993). Time Constrained Routing and Scheduling. Forthcoming in Handbooks on Operations Research and Management Science. Volume on Networks, North-Holland, Amsterdam.

[9] Fisher M. (1993). Vehicle Routing. Forthcoming as a chapter in Handbooks on

Operations Research and Management Science. Volume on Networks, North-Holland, Amsterdam.

[10] Fisher, Marshall, K. O. Jornsten and O. B. G. Madsen (1992). Vehicle Routing with Time Windows, Research Report, The Institute of Mathematical Statistics and Operations Research, The Technical University of Denmark, DK-2800, Lyngby.

[11] Fisher, M and R. Jaikumar (1981). A Generalized Assignment Heuristic for the Vehicle Routing Problem. *Networks*, 11, 109-124.

[12] Gillett, B. and L. Miller (1974). A Heuristic Algorithm for the Vehicle Dispatching Problem. *Operations Research* 22, 340-349.

[13] Glover, F., E. Taillard and D. de Werra (1993). A User's Guide to Tabu Search, *Annals of Operations Research*, 41, 3-28.

[14] Glover, Fred (1993). Tabu Thresholding: Improved Search by Non-monotonic Trajectories, Working Paper, Graduate School of Business, University of Colorado, Boulder, Colorado 80309-0419.

[15] Glover, Fred (1989). Tabu Search-Part I, *ORSA Journal on Computing*, 1(3), 190-206.

[16] Glover, Fred (1990). Tabu Search-Part II, *ORSA Journal on Computing*, 2(1), 4-32.

[17] Glover, Fred (1986). Future Paths for Integer Programming and Link to Artificial Intelligence. *Computers and Operations Research*, 13, 533-554.

[18] Goldberg D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, Inc.

[19] Golden B. and A. Assad (eds.) (1988). Vehicle Routing: Methods and Studies. North Holland, Amsterdam.

[20] Golden, B and A. Assad (eds.) (1986). Vehicle Routing with Time Window Constraints: Algorithmic Solutions. American Journal of Mathematical and Management Sciences 15, American Sciences Press, Columbus, Ohio.

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

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

[23] Grefenstette, John J. (1986). Optimization of Control Parameters for Genetic

Algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1), 122-128.

[24] Hasan, Merza and I. H. Osman (1994). Local Search Strategies for the Maximal Planar Layout Problem. *International Transactions in Operations Research*, 1(4).

[25] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

[26] Kirkpatrick, S., Gelatt, C. D. and Vecchi, P. M.(1983), Optimization by Simulated Annealing. *Science* 220, 671-680.

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

[28] Kontoravadis, G. and J. F. Bard (1992). Improved Heuristics for the Vehicle Routing Problem with Time Windows, Working Paper, Operations Research Group, Department of Mechanical Engineering, The University of Texas, Austin, TX 78712-1063.

[29] Laporte, G. (1992). The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms. *European Journal of Operational Research*, 59, 345-358.

[30] Lin, S and B. W. Kernighan (1973). An Effective Heuristic Algorithm for the Travelling Salesman Problem. *Operations Research*, 21, 2245-2269.

[31] Lundy, M and A. Mees (1986). Convergence of an Annealing Algorithm, *Mathematical Programming*, 34, 111-124.

[32] Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller (1953). Equation of State Calculation by Fast Computing Machines. *Journal of Physical Chemistry* 21, 1087-1092.

[33] Osman, I. H. and G. Laporte (1994). Modern Heuristics for Combinatorial Optimization Problems: An Annotated Bibliography. Forthcoming in the Annals of Operations Research on Metaheuristics.

[34] Osman, I. H. & S. Salhi (1994). Heuristics for the Mix Fleet Vehicle Routing Problem. Report no UKC/IMS/OR94/2b, University of Kent, U.K. Forthcoming in Proceedings of the TRISTAN II conference, CAPRI, Italy June 1994.

[35] Osman, I. H. and N. Christofides (1994). Capacitated Clustering Problems by Hybrid Simulated Annealing and Tabu Search. *International Transactions in Operational Research*,1 (3).

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

France.

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

[38] Osman I. H.(1993c). Heuristics for the Generalized Assignment Problem: Simulated Annealing and Tabu Search. Report no. UKC/IMS/OR93/10b, University of Kent, Canterbury, Submitted to OR Spektrum.

[39] Osman, I. H. and N. Christofides (1989). Simulated Annealing and Descent Algorithms for Capacitated Clustering Problem. Research Report, Imperial College, University of London, Presented at EURO-X Conference, Beograd, Yugoslavia, 1989.

[40] Potvin, Jean-Yves and S. Bengio (1993). A Genetic Approach to the Vehicle Routing Problem with Time Windows, Technical Report, CRT-953, Centre de Recherche sur les Transports, Universite de Montreal, C.P. 6128, Succ. A, Montreal, Canada H3C 3J7.

[41] Potvin, Jean-Yves, Tanguy Kervahut, Bruno-Laurent Garcia and Jean-Marc Rousseau (1992). A Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. Technical Report, CRT-855, Centre de Recherche sur les Transports, Universite de Montreal, C.P. 6128, Succ. A, Montreal, Canada H3C 3J7.

[42] Potvin, Jean-Yves and Jean-Marc Rousseau (1993). A Parallel Route Building Algorithm for the Vehicle Routing and Scheduling Problem with Time Windows. *European Journal of Operational Research*, 66, 331-340.

[43] Savelsbergh M. W. P. (1985). Local Search for Routing Problems with Time Windows. *Annals of Operations Research*, 4, 285-305.

[44] Solomon, Marius M., Edward K. Baker, and Joanne R. Schaffer (1988). *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.

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

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

[47] Thangiah, Sam R., Ibrahim H. Osman, Rajini Vinayagamoorthy and Tong Sun (1994). Algorithms for Vehicle Routing Problems with Time Deadlines. *American Journal of Mathematical and Management Sciences*, 13 (3&4).

[48] Thangiah, Sam R. (1993). Vehicle Routing with Time Windows using Genetic Algorithms. Submitted to the International Journal on Computers and Operations Research.

[49] Thangiah, Sam R., Rajini Vinayagamoorthy and Ananda Gubbi (1993). 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.

[50] Thangiah, Sam R. and Kendall E. Nygard (1993). Dynamic Trajectory Routing using an Adaptive Search Strategy. Proceedings of Association for Computing Machinery's Symposium on Applied Computing, Indianapolis, 1993.

[51] Thangiah, Sam R. and Kendall Nygard (1992). School Bus Routing using Genetic Algorithms. Proceedings of the Applications of Artificial Intelligence X: Knowledge Based Systems, Orlando.

[52] Thangiah, Sam R. and Kendall Nygard (1992). MICAH: A Genetic Algorithm System for Multi-Commodity Networks. Proceedings of the Eight IEEE Conference on Applications of Artificial Intelligence, Monterey.

[53] Thangiah, Sam R., Kendall E. Nygard and Paul L. Juell (1991). GIDEON: A Genetic Algorithm System for Vehicle Routing Problems with Time Windows. Proceedings of the Seventh IEEE Conference on Artificial Intelligence Applications, Miami, Florida.

[54] Thompson, Paul M. and H. Psaraftis (1989). Cyclic Transfer Algorithms for Multi-Vehicle Routing and Scheduling Problems. *Operations Research*, 41, 935-946.

Appendix A

Table 6: Comparison of solutions obtained by LSD-FB, LSD-BG, SA and TSSA with initial solutions obtained from the PFIH on data sets R1, C1, and RC1.

Prob.	LSD-FB			LSD-GB			SA			TSSA		
	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU
R101	20	1600	0:3	20	1666	0:03	19	1665	0:36	**19**	**1648**	**4:01**
R102	19	1549	0:3	18	1522	0:3	18	1499	0:47	**18**	**1496**	**3:45**
R103	17	1418	0:3	16	1403	0:3	15	1371	2:17	**15**	**1300**	**9:39**
R104	13	1116	0:4	12	1147	0:4	**11**	**1032**	**5:57**	11	1053	17:49
R105	16	1552	0:3	16	1446	0.3	**15**	**1407**	**0:32**	15	1451	6:20
R106	15	1437	1:4	14	1330	0:4	**14**	**1326**	**1:0**	14	1336	10:08
R107	13	1166	1:5	13	1297	0:4	**12**	**1140**	**1:10**	12	1147	10:59
R108	12	1081	0:4	11	1047	0:6	10	1001	3:16	**10**	**989**	**15:30**
R109	14	1371	0:4	13	1303	0:3	13	1226	0:59	**12**	**1218**	**8:20**
R110	13	1334	0:3	13	1206	0:4	13	1187	1:06	**12**	**1158**	**11:49**
R111	14	1220	0:3	13	1257	0:4	13	1158	2:26	**12**	**1120**	**12:49**
R112	11	1063	0:3	11	1064	0:4	11	1016	1:55	**10**	**993**	**10:15**
C101	**10**	**829**	**0:2**	**10**	**829**	**0:1**	**10**	**829**	**0:19**	**10**	**829**	**0:31**
C102	**10**	**829**	**0:3**	10	832	0.:1	10	930	0:30	10	832	3:59
C103	10	890	0:3	10	861	0:2	10	890	0:22	**10**	**835**	**11:31**
C104	10	964	0:3	10	956	0:2	10	961	1:12	**10**	**840**	**11:32**
C105	10	829	0:2	**10**	**829**	**0:1**	10	880	0:10	**10**	**829**	**1:29**
C106	**10**	**829**	**0:2**	**10**	**829**	**0:1**	10	870	0:22	**10**	**829**	**0:31**
C107	**10**	**829**	**0:2**	**10**	**829**	**0:1**	**10**	**829**	**0:16**	**10**	**829**	**2:14**
C108	**10**	**829**	**0:3**	**10**	**829**	**0:2**	10	907	0:28	**10**	**829**	**5:15**
C109	**10**	**829**	**0:3**	**10**	**829**	**0:3**	10	854	1:30	**10**	**829**	**7:48**
RC101	16	1778	0:3	16	1741	0:4	16	1695	0:49	**16**	**1670**	**8:18**
RC102	15	1458	0:4	15	1618	0:3	**14**	**1532**	**2:01**	14	1536	7:07
RC103	13	1350	0:5	13	1383	0:3	12	1441	2:41	**12**	**1409**	**7:17**
RC104	**11**	**1213**	**0:3**	**11**	1216	0:3	**11**	**1213**	**2:08**	11	1222	11:07
RC105	15	1645	0:6	16	1690	0:3	14	1731	2:09	**14**	**1602**	**7:30**
RC106	14	1577	0:3	**13**	**1506**	**0:3**	14	1510	0:48	14	1430	6:39
RC107	13	1377	0:3	13	1356	0:3	14	1294	3:37	**12**	**1264**	**9:50**
RC108	13	1277	0:9	13	1357	0:3	12	1218	1:13	**11**	**1169**	**11:32**

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles.
TD:	Total Distance travelled.	CPU:	CPU time on a NeXT machine in min:sec.
LSD-FB:	Local Descent Method with First-Best strategy.	LSD-GB:	Local Descent Method with Global-Best strategy.
SA:	Simulated Annealing + First Best strategy.		
TSSA	Tabu Search+ Simulated Annealing+ LSD with Global Best Strategy.		

Appendix A (cont.)

Table 7: Comparison of solutions obtained by LSD-FB, LSD-GB, SA and TSSA with initial solutions obtained by PFIH on data sets R2, C2, RC2 and problems D249, E249, D417 and E417.

Prob	LSD-FB			LSD-GB			SA			TSSA		
	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU
R201	5	1466	2:02	5	1466	0:45	4	1431	10:40	**4**	**1363**	**11:20**
R202	4	1433	3:01	4	1456	1:19	4	1372	13:57	**4**	**1285**	**8:34**
R203	4	1212	2:04	4	1245	2:31	3	1147	16:12	**3**	**1126**	**22:03**
R204	3	1158	4:13	**4**	**897**	**3:31**	3	998	34:12	3	959	17:05
R205	3	1477	5:17	4	1255	1:31	**3**	**1319**	**30:41**	3	1468	8:0
R206	3	1322	2:07	4	1434	0:36	3	1247	27:50	**3**	**833**	**16:22**
R207	3	1234	3:05	**3**	**1099**	**5:22**	**3**	**1099**	**20:31**	**3**	**1099**	**8:03**
R208	3	980	6:06	**3**	**926**	**8:12**	3	979	31:42	**3**	**926**	**12:03**
R209	3	1015	1:47	3	1089	1:49	**3**	**997**	**18:22**	3	1022	4:05
R210	3	1243	1:12	4	1152	1:27	**3**	**1211**	**17:26**	3	1237	3:02
R211	3	936	4:21	3	939	2:40	3	1059	11.37	**3**	**927**	**8:12**
C201	**3**	**591**	**0:04**	**3**	**591**	**0:5**	**3**	**591**	**0:10**	**3**	**591**	**0:12**
C202	3	708	0:32	4	849	0:29	3	708	1:07	**3**	**707**	**1:0**
C203	4	829	0:23	4	840	0:26	3	796	3:04	**3**	**791**	**5:0**
C204	3	787	0:37	4	853	0:33	**3**	**685**	**7:07**	3	772	11:05
C205	3	589	0:30	3	589	0:29	**3**	**588**	**30:75**	3	589	1:37
C206	3	670	0:26	3	665	0:24	3	651	32:23	**3**	**608**	**6:03**
C207	3	665	0:26	3	651	0:28	3	665	12:07	**3**	**647**	**8:13**
C208	3	836	0:43	**3**	**588**	**0:44**	3	815	8:9	8	601	5:17
RC201	5	1707	1:59	5	1505	0:21	**4**	**1294**	**3:10**	5	1589	3:06
RC202	4	1581	1:27	4	1478	0:27	4	1382	62:49	**4**	**1338**	**17:28**
RC203	4	1220	1:37	4	1219	1:10	4	1220	25:32	**4**	**1205**	**13:07**
RC204	3	994	1:22	3	994	1:40	**4**	**948**	**11:42**	3	971	11:04
RC205	5	1642	4:37	4	1540	0:32	4	1542	21:42	**4**	**1431**	**32:38**
RC206	4	1227	3:43	**4**	**1213**	**0:34**	4	1249	1:30	4	1412	5:06
RC207	4	1249	7:01	4	1200	0:53	**3**	**1231**	**12:20**	4	1187	16:08
RC208	3	965	3:53	3	944	1:33	3	1125	11:06	**3**	**919**	**44:05**
D249	**4**	**703**	**1:01**	**4**	**703**	**1:03**	**4**	**703**	**2:00**	**4**	**703**	**2:00**
E249	5	550	12:29	**5**	**495**	**11:08**	5	555	17:50	**5**	**495**	**23:02**
D417	56	4447	2:36	56	3968	2:40	**55**	**4053**	**32:07**	55	4100	12:59
E417	56	4785	2:33	56	4730	3:04	**56**	**4608**	**22:07**	56	4660	12:07

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles.
TD:	Total Distance travelled.	CPU:	CPU time on a NeXT machine in min:sec.
LSD-FB:	Local Descent Method with First-Best strategy.	LSD-GB:	Local Descent Method with Global-Best strategy.
SA:	Simulated Annealing + First Best strategy.		
TSSA	Tabu Search+ Simulated Annealing+ LSD with Global Best Strategy.		

Appendix B

Table 8: Comparison of solutions obtained by hybrid LSD-FB, LSD-GBD, SA, and TSSA with initial solutions obtained from the GSH method on data sets R1, C1, and RC1.

Prob.	LSD-FB			LSD-GB			SA			TSSA		
	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU
R101	18	1725	0:9	19	1644	0:14	18	1750	0:46	**18**	**1677**	**9:53**
R102	17	1528	0:11	17	1547	0:14	**17**	**1493**	**1:02**	17	1505	12:09
R103	13	1321	0:09	13	1297	0:12	13	1275	0:59	**13**	**1207**	**16:22**
R104	10	1081	0:09	10	1097	0:10	**10**	**1048**	**5:07**	10	1056	14:31
R105	14	1484	0:09	14	1491	0:15	**14**	**1442**	**0:36**	14	1477	10:52
R106	13	1306	0:11	13	1376	0:15	12	1412	0:22	**12**	**1350**	**17:59**
R107	11	1273	0:10	11	1204	0:27	11	1233	0:32	**11**	**1146**	**20:29**
R108	10	1074	0:10	10	1204	0:10	10	1069	1:09	**10**	**1013**	**26:38**
R109	12	1304	0:31	12	1304	0:31	12	1280	0:32	**12**	**1228**	**16:05**
R110	11	1165	0:40	**11**	**1105**	**0:40**	11	1148	0:24	11	1118	19:58
R111	**10**	**1151**	**0:44**	**10**	**1151**	**0:44**	11	1175	0:32	10	1223	23:28
R112	10	1049	0:25	10	1049	0:25	10	1114	1:09	**10**	**992**	**14:46**
C101	10	884	0:54	**10**	**829**	**0:19**	**10**	**829**	**0:28**	**10**	**829**	**0:58**
C102	10	1001	0:38	**10**	**829**	**0:16**	10	991	2:06	**10**	**829**	**1:09**
C103	10	1014	0:17	10	1020	0:10	**10**	**886**	**1:09**	10	968	1:20
C104	10	943	0:12	10	960	0:09	**10**	**908**	**1:56**	10	926	0:57
C105	**10**	**829**	**0:09**	**10**	**829**	**0:09**	10	955	1:45	10	862	0:40
C106	**10**	**829**	**0:10**	**10**	**829**	**0:12**	10	1008	1:34	**10**	**829**	**0:36**
C107	10	1005	0:12	10	1063	0:11	10	1060	1:29	**10**	**965**	**0:37**
C108	10	1123	0:17	10	1060	0:13	10	939	0:45	**10**	**829**	**0:57**
C109	10	992	0:19	10	980	0:16	10	858	1:14	**10**	**829**	**0:56**
RC101	15	1725	0:12	15	1782	0:13	15	1731	0:24	**14**	**1669**	**3:04**
RC102	13	1713	0:8	13	1580	0:13	13	1656	0:17	**13**	**1557**	**9:34**
RC103	11	1356	0:10	11	1388	0:14	**11**	**1110**	**2:34**	11	1310	12:36
RC104	10	1313	0:9	10	1314	0:15	10	1295	0:50	**10**	**1226**	**14:38**
RC105	14	1697	0:10	14	1712	0:14	14	1698	0:34	**14**	**1670**	**13:21**
RC106	13	1459	0:8	14	1482	0:11	13	1491	0:32	**13**	**1420**	**13:29**
RC107	12	1547	0:11	11	1415	0:19	11	1500	0:21	**11**	**1410**	**2:25**
RC108	**10**	**1281**	**0:10**	10	1324	0;15	10	1290	1:01	10	1310	1:06

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles.
TD:	Total Distance travelled.	CPU:	CPU time on a NeXT machine in min:sec.
LSD-FB:	Local Descent Method with First-Best strategy.	LSD-GB:	Local Descent Method with Global-Best strategy.
SA:	Simulated Annealing + First Best strategy.		
TSSA	Tabu Search+ Simulated Annealing+ LSD with Global Best Strategy.		

Appendix B (cont.)

Table 9: Comparison of solutions obtained by hybrid LSD-FB, LSD-GB, SA, and TSSA with initial solutions obtained from GSH on data sets R2, C2, RC2, and problems D249, E249, D417 and E417.

Prob	LSD-FB			LSD-GB			LSD-SA			TSSA		
	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU	NV	TD	CPU
R201	**4**	**1354**	**0:58**	4	1431	0:48	4	1404	3:33	4	1358	1:47
R202	4	1187	2:12	4	1180	1:02	4	1222	10:00	**4**	**1176**	**1:17**
R203	**3**	**1170**	**1:57**	3	1171	2:07	3	1181	9:58	3	1195	6:45
R204	3	812	2:40	3	812	2:18	3	820	4.32	**3**	**803**	**4:38**
R205	**3**	**1128**	**1:03**	3	1228	1:14	3	1167	3:59	3	1176	1:41
R206	**3**	**1018**	**2:19**	3	1019	2:10	3	1029	10:03	**3**	**1018**	**2:09**
R207	3	938	3:04	3	937	2:53	3	932	10:05	**3**	**904**	**6:32**
R208	**2**	**823**	**5:45**	**2**	**823**	**6:14**	3	851	5:43	2	826	4:15
R209	**2**	**855**	**14:40**	3	1057	1:10	3	1061	9:45	3	993	2:19
R210	3	1112	1:10	3	1112	0:56	**3**	**1052**	**8:26**	3	1071	2:49
R211	3	851	2:02	3	850	2:08	3	851	10:50	**3**	**816**	**3:51**
C201	**3**	**592**	**1:37**	**3**	**592**	**0:41**	**3**	**592**	**4:31**	**3**	**592**	**1:04**
C202	3	784	1:50	3	784	3:05	3	794	3:42	**3**	**781**	**1:59**
C203	**3**	**831**	**0:59**	**3**	**831**	**1:53**	3	870	1:49	3	836	1:45
C204	3	818	1:41	3	851	1:41	3	847	1:14	**3**	**817**	**1:03**
C205	**3**	**589**	**1:20**	3	851	1:04	**3**	**589**	**2:02**	**3**	**589**	**3:05**
C206	**3**	**588**	**1:40**	**3**	**588**	**1:22**	**3**	**588**	**1:53**	3	618	1:46
C207	3	592	1:49	3	591	1:25	3	592	2:26	**3**	**588**	**3:35**
C208	**3**	**588**	**0:59**	**3**	**588**	**1:04**	**3**	**588**	**1:42**	**3**	**588**	**1:29**
RC201	**4**	**1605**	**0:56**	4	1633	1:02	4	1631	2:12	4	1638	3:21
RC202	4	1394	0:51	4	1394	0:49	**4**	**1291**	**10:40**	4	1368	1:13
RC203	3	1226	1:42	3	1225	1:06	**3**	**1203**	**9:41**	3	1211	5:04
RC204	3	903	1:06	3	903	1:04:	3	917	8:53	**3**	**897**	**2:03**
RC205	**4**	**1389**	**1:26**	4	1403	1:18	4	1413	4:28	4	1418	1:43
RC206	3	1349	1:18	3	1349	2:20	3	1351	9:41	**3**	**1348**	**4:32**
RC207	3	1209	1:12	**3**	**1181**	**1:31**	3	1223	8:55	**3**	**1181**	**1:40**
RC208	**3**	**957**	**2:48**	3	958	2:34	3	1431	9:49	3	1027	6:28
D249	4	516	20.5	4	518	19.7	**4**	**457**	**26.5**	**4**	**457**	**23.5**
E249	6	500	21.2	6	503	18.3	**6**	**516**	**34.56**	6	514	33.23
D417	55	6352	31.4	**54**	**4866**	**30.8**	55	4726	29.8	55	4783	33.3
E417	55	7267	33.654	55	4560	37.3	**55**	**4149**	**27.4**	55	4730	35.7

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles.
TD:	Total Distance travelled.	CPU:	CPU time on a NeXT machine in min:sec.
LSD-FB:	Local Descent Method with First-Best strategy.	LSD-GB:	Local Descent Method with Global-Best strategy.
SA:	Simulated Annealing + First Best strategy.		
TSSA	Tabu Search+ Simulated Annealing+ LSD with Global Best Strategy.		

Appendix C

Table 10: Best known solutions for data sets R1, C1 and RC1 using seven different heuristics.

Prob.	Solomon		CTA		GIDEON		PTABU		GENEROUS		Chia/Russ		GenSAT	
	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
R101	21	1873	19	1734	20	1700	19	1753	19	1784	19	1704	**18**	**1677**
R102	19	1843	17	1881	17	1549	17	1736	17	17777	17	1610	**17**	**1493**
R103	14	1484	15	1530	13	1319	14	1410	14	1433	13	1467	**13**	**1207**
R104	11	1188	10	1101	10	1090	10	1094	10	1065	10	1079	**10**	**1048**
R105	15	1673	15	1535	15	1448	14	1432	**14**	**1421**	15	1465	14	1442
R106	14	1475	13	1392	13	1363	12	1353	12	1353	12	1353	**12**	**1350**
R107	12	1425	11	1250	11	1187	11	1185	11	1191	10	1215	**11**	**1140**
R108	10	1137	10	1035	10	1048	10	1020	10	993	10	1003	**10**	**898**
R109	13	1412	12	1249	12	1345	12	1222	**12**	**1205**	12	1239	12	1226
R110	12	1393	12	1258	11	1234	11	1136	11	1136	11	1152	**11**	**1105**
R111	12	1231	12	1215	11	1238	11	1184	11	1184	10	1188	**10**	**1151**
R112	10	1106	10	1103	10	1082	10	2016	10	1020	10	1003	**10**	**992**
C101	10	853	**10**	**829**	10	833	10	829	10	829	10	829	10	829
C102	10	968	10	934	10	832	10	846	**10**	**829**	10	1082	10	829
C103	10	1059	10	956	10	873	10	951	10	875	10	982	**10**	**835**
C104	10	1282	10	1130	10	904	10	978	10	865	10	936	**10**	**840**
C105	10	861	**10**	**829**	10	874	10	829	10	829	10	829	10	829
C106	10	897	10	868	10	902	**10**	**829**	10	829	10	829	10	829
C107	10	904	10	926	10	926	**10**	**829**	10	829	10	829	10	829
C108	10	855	10	866	10	928	**10**	**829**	10	829	10	829	10	829
C109	10	888	10	912	10	957	**10**	**829**	10	829	10	829	10	829
RC101	16	1867	16	1851	15	1767	15	1751	15	1676	15	1681	**14**	**1669**
RC102	15	1760	14	1644	14	1569	14	1685	13	1671	14	1598	**13**	**1557**
RC103	13	1673	12	1465	11	1328	12	1430	11	1401	11	1381	**11**	**1110**
RC104	11	1301	11	1265	11	1263	10	1224	**10**	**1204**	10	1210	10	1226
RC105	16	1922	15	1809	14	1612	14	1697	14	1670	**14**	**1691**	**14**	**1602**
RC106	13	1611	-	-	12	1608	13	1461	**12**	**1486**	13	1501	13	1420
RC107	13	1385	12	1338	12	1396	12	1328	11	1275	12	1390	**11**	**1264**
RC108	11	1253	11	1228	11	1250	11	1210	11	1187	11	1192	**10**	**1281**

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles
TD:	Total distance travelled.	Solomon:	Solomon, 1987.
CTA:	Thompson and Psaraftis, 1993.	GIDEON:	Thangiah, 1993.
PTABU:	Potvin et al., 1993.	GENEROUS:	Potvin and Bengio, 1994.
Chia/Russ:	Chiang and Russell, 1994.		

Appendix C (cont.)

Table 11: Best known solutions for data sets R2, C2 and RC2 and problems D249, E249, D417 and E417 using seven different heuristics.

Prob	Solomon		CTA		GIDEON		PTABU		GENEROUS		Chia/Russ		GenSAT	
	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
R201	4	1741	4	1786	4	1478	4	1669	4	1554	4	1494	**4**	**1354**
R202	4	1730	4	1736	4	1279	3	1530	3	1530	3	1562	**4**	**1176**
R203	3	1578	3	1309	3	1167	3	1237	3	1201	3	1173	**3**	**1126**
R204	3	1059	3	1025	3	909	3	952	3	904	**2**	**914**	3	803
R205	3	1471	3	1392	3	1274	3	1200	3	1159	3	1212	**3**	**1128**
R206	3	1463	3	1254	3	1098	3	1153	3	1066	3	1074	**3**	**8338**
R207	3	1302	3	1072	3	1015	3	987	3	954	3	999	**3**	**904**
R208	3	1076	3	862	3	826	3	806	**2**	**759**	2	805	2	823
R209	3	1449	3	1260	3	1159	3	1164	3	1108	3	1132	**2**	**855**
R210	4	1542	3	1269	3	1269	3	1200	3	1146	3	1245	**3**	**1052**
R211	3	1016	3	1071	3	898	3	984	3	913	3	977	**3**	**816**
C201	3	591	3	590	3	753	**3**	**590**	3	592	3	592	3	591
C202	3	731	3	664	3	756	**3**	**591**	3	592	3	648	3	707
C203	3	811	3	653	3	855	3	631	**3**	**592**	3	751	3	791
C204	4	758	3	684	3	803	3	665	**3**	**591**	3	927	3	685
C205	3	615	3	628	3	667	**3**	**589**	3	589	3	589	3	589
C206	3	730	3	641	3	694	**3**	**588**	3	588	3	588	3	588
C207	3	691	3	627	3	730	**3**	**588**	3	588	3	588	3	588
C208	3	615	3	670	3	735	**3**	**588**	3	588	3	588	3	588
RC201	4	2103	4	1959	4	1823	4	1985	4	1799	4	1734	**4**	**1294**
RC202	4	1799	4	1858	4	1459	4	1742	4	1603	4	1564	**4**	**1291**
RC203	4	1626	4	1521	3	1323	3	1309	3	1253	3	1284	**3**	**1203**
RC204	3	1208	3	1143	3	1021	3	1034	3	1002	3	1012	**3**	**897**
RC205	5	2134	4	1988	4	1594	4	1888	4	1693	4	1751	**4**	**1389**
RC206	4	1582	3	1515	3	1530	3	1440	3	1324	3	1298	**3**	**1213**
RC207	4	1632	4	1457	3	1501	3	1321	3	1222	3	1194	**3**	**1181**
RC208	3	1373	-	-	3	1038	3	1041	3	1049	3	1053	**3**	**919**
D249											4	477	**4**	**457**
E249											5	506	**5**	**495**
D417											55	4235	**54**	**4866**
E417											55	4397	**55**	**4149**

Legend:

Prob.:	VRPTW problems from the literature.	NV:	Number of vehicles
TD:	Total distance travelled.	Solomon:	Solomon, 1987.
CTA:	Thompson and Psaraftis, 1993.	GIDEON:	Thangiah, 1993.
. PTABU:	Potvin et al., 1993.	GENEROUS:	Potvin and Bengio, 1994.
Chia/Russ:	Chiang and Russell, 1994		