GIDEON: A Genetic Algorithm System for Vehicle Routing with Time Windows

Sam R. Thangiah
thangiah@plains.nodak.edu

Kendall E. Nygard
nygard@plains.nodak.edu

Paul L. Juell
juell@plains.nodak.edu

Department of Computer Science and Operations Research
North Dakota State University

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 number of vehicles and the distance traveled for 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-hard, search methods based upon heuristics are most promising for problems of practical size. In this paper we describe GIDEON, a genetic algorithm system for solving the VRPTW. On a standard set of 56 VRPTW problems obtained from the literature, GIDEON did better than the alternate methods on 41 of them, with an average reduction of 3.9% in fleet size and 4.4% in distance traveled for the 56 problems.

AI TOPIC: Genetic Algorithms.
DOMAIN AREA: Vehicle Routing Problems with Time Windows.
LANGUAGE/TOOL: C Language/GENESIS.
STATUS: Implemented.
EFFORT: Two and a half person years.
IMPACT: Genetic algorithms used as a meta-level search strategy in routing and scheduling problems can obtain near optimal solutions for dynamic environments in real time.

1.0 Introduction

The problem we address is the Vehicle Routing Problem with Time Windows (VRPTW). The VRPTW involves routing a fleet of vehicles, of limited capacity and travel time, 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 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, airline and railway fleet routing and scheduling. Efficient routing and scheduling of vehicles can potentially save government and industry many millions of dollars a year. The current status of vehicle routing research is available in [3] and [1]. Solomon and Desrosiers[19] provide an excellent survey on vehicle routing problems with time windows.

Savelsbergh[16] has shown that finding a feasible solution for a VRPTW using a fixed fleet size is NP-hard. Due to the intrinsic difficulty of the problem, search methods based upon heuristics are most promising for solving practical size problems [2],[18] and [20].

In this paper we describe GIDEON, a genetic algorithm system to heuristically solve the VRPTW. GIDEON consists of two distinct modules, a global clustering module that assigns customers to vehicles by a process we call genetic sectoring (GENSECT) and a local route optimization module (SWITCH-OPT). The GENSECT module uses a genetic algorithm to adaptively search for sector rays that partition the customers into sectors or clusters. Customers within each of the clusters are serviced by a vehicle. The GENSECT solution ensures that each vehicle route begins and ends at the depot and that every customer is serviced by one vehicle. After the termination of the GENSECT module, the best set of clusters obtained from the module are passed on to the

tardy customers and reducing infeasibilities in capacity and travel time of vehicles.

The synergy between a global adaptive heuristic search combined with local optimization gives solutions superior to those of competing heuristic algorithms. On a standard set of 56 VRPTW problems obtained from the literature, GIDEON did better than the alternate methods on 41 of them, with an average reduction of 3.9% in fleet size and 4.4% in distance traveled for the 56 problems. GIDEON took an average of 127 cpu seconds to solve a problem on the SOLBOURNE 5/802 computer. In addition to performance and efficiency, the GIDEON system can be used to evaluate the tradeoff between fleet size, vehicle capacity, distance and route time by parametrically varying the values.

The paper is arranged in the following form. Section 2 gives an overview of genetic algorithms. Section 3 gives a description of the GENSECT and SWITCH-OPT modules. Section 4 describes the results of computational testing on a standard set of VRPTW problems obtained from the literature.

2.0 Genetic Algorithms

Genetic algorithms are a class of heuristic search algorithms based upon population genetics [14],[6] and [10]. As they are inherently adaptive, genetic algorithms can quickly converge to near optimal solutions in many applications. They have been used to heuristically solve traveling salesman problems [13] and [15], quadratic assignment problems [4], job-shop scheduling problems [5], bin-packing problems [17] and other NP-complete problems [7]. The flow of a genetic algorithm is shown in Figure 1.

```
Generation <- 0;
Initialize M(Generation);
Evaluate M(Generation);
Loop (until termination condition)
BEGIN
    Generation <- Generation + 1;
    Select M(Generation) from M(Generation-1);
    Crossover M(Generation);
    Mutate M(Generation);
    Evaluate M(Generation);
END
```

Figure 1. Flow of a Genetic Algorithm

The genetic algorithm (GA) is an iterative procedure that maintains a constant-size population, P, of candidate solutions. The population is a set of dynamic string entities of artificial chromosomes. The chromosomes are fixed length strings with a finite number of binary values (or alleles) at each position (or locus) (see Figure 2). The initial population P(0) is chosen randomly. Each chromosome is evaluated and a fitness value is assigned to it. The fitness value determines the relative ability of a chromosome to survive and produce offspring in the next generation.

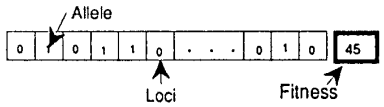

Figure 2. The Structure of a Chromosome

Let P be the constant size of the population at each generation. The selection, Select M(Generation), of a chromosome C_i, where i = 1 to P, into the next generation is dependent upon its fitness value f_i relative to the fitness value of other chromosomes in the population. At (Generation - 1) the relative fitness, R_i, of the chromosome C_i is as calculated follows:

$$R_i = \frac{f_i}{\sum f_i}$$

In carrying out the selection process Select M(Generation) we treat the relative fitness value as probabilities, and calculate the expected number of copies of chromosome C_i chosen for producing offspring to appear in the next generation as follows:

$$\textit{Expected number of copies of } C_i = R_i * P$$

For example, if the relative fitness value of chromosome C_i is 0.2 and P=100, then 2 identical copies of C_i will survive into the next generation. The expected copies are processed in descending order until the population size is reached.

Variation is introduced within the population chromosomes by genetic crossover and mutation operators. The most important genetic recombination operator is the crossover. After selecting high fitness valued chromosomes for the next generation, the operator, Crossover M(Generation), exchanges portions of the new chromosomes, allowing a sharing of information. Selection and crossover effectively search the space exploiting information present in population chromosomes by selecting and recombining primarily those offsprings that have high fitness values. These two processes eventually produce a population of chromosomes with high performance characteristics. The Mutate M(Generation) is a secondary operator that

prevents premature loss of important information by inverting binary values of alleles within a chromosome. For the GA to perform a rigorous search of the problem space, the differences between fitness values of chromosomes in a population should be relatively large to distinguish the high performance chromosomes from the low performance ones during the selection process.

GA's are adaptive in the sense that candidate offspring chromosomes generated at each generation reflect and exploit information obtained by chromosomes of earlier generations.

3.0 The GENSECT and SWITCH-OPT Modules

GIDEON consists of two distinct modules, GENSECT, a global clustering module and SWITCH-OPT, a local route optimization module. The GENSECT module assigns customers to vehicles using a process we call genetic sectoring. The clustering ideas in GENSECT are inspired by the work of Fisher and Jaikumar[8] and Gillett and Miller[9]. The GENESIS software [12], for genetic search, controls the search for sector rays that partition the customers. Once the clusters are formed, the vehicles are routed to serve the customers within a cluster.

The route with in the clusters generated by the GENSECT module could contain tardy customers and infeasibilities in capacity and travel time of vehicles. The SWITCH-OPT module uses local optimization procedures to move or exchange customers between clusters to remove the tardy customers and vehicle infeasibilities. The iteration between GENSECT and SWITCH-OPT can be performed any number of times. We have found that two iterations are sufficient to attain relatively good solutions.

3.1 The GENSECT Module

The GENSECT module initially divides customers into sectors or clusters using a sweep algorithm reminiscent of Gillett and Miller[9] with sector rays as the points of division.

In an N customer problem with the origin at the depot, the genetic search uses customer angles $C_1, C_2, \cdots, C_N$ expressed in what we call pseudo polar coordinate angles to perform genetic sectoring. The pseudo polar coordinate angles are obtained by fixing the largest angle, MaxAngle, then normalizing the angles between customers so that the angle difference between any two adjacent customers is equal. This is a contrivance that has the effect of allowing sector boundaries to freely fall between any pair of customers that have adjacent angles, whether the separation is small or large. The customers are divided into K clusters, where K is the fleet size, by planting a set of "seed" angles, $S_1, \ldots, S_K$, in the search space and drawing a ray from the origin to each seed angle. The genetic sectoring of customers using seed angles and sector rays is depicted in Figure 3. The initial seed angle S_1 is set at 0 degrees. A "sweep" algorithm assigns a customer C_i to a cluster A_j where j = 1 to K, based on the following condition:

$$C_i \text{ is assigned to } A_j \text{ if } S_j < C_i \leq S_{j+1}, \; j = 1 \text{ to } K$$

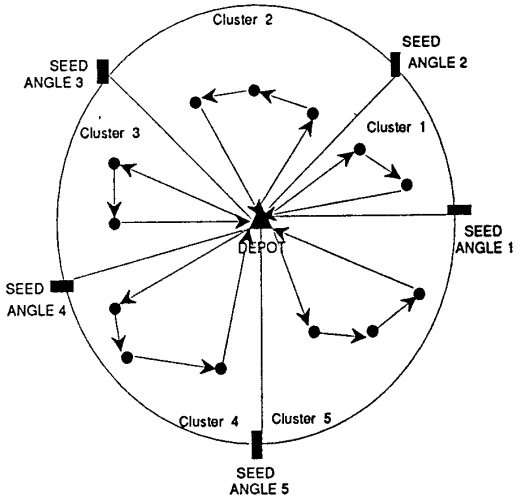

Figure 3. Division of Customers using Seed Angles

Once the clusters are formed, the vehicles are routed to serve the customers while minimizing the route cost calculated by the route cost function:

cost = (Tardiness * A1) + (Distance * A2)
+ (Route Time * A3) + (EX * Penalty); (3.1)

The constant values for equation (3.1) were set at A1=25, A2=0.5, A3=0.05 and Penalty=50. The fleet size were empirically determined. The cost function gives high priority to reducing tardy customers and penalizes vehicles not within their capacity and travel time in order to obtain a feasible solution. A positive EX term indicates a vehicle has exceeded its capacity or travel time. The distance and route time appear in the equation as they indirectly affect time violation constraints. The total route cost is the total of each vehicle's route cost.

The total route cost is sensitive to the clusters formed from the seed angles. The adaptive capability

of the genetic search is used to find a set of K-1 seed angles, as seed angle 1 is equal to 0, for sectoring customers into clusters with the lowest total route cost.

During the genetic sectoring process each chromosome represents a set of seed angles. Therefore, a population of P chromosomes has P sets of seed angles. Each seed angle in the chromosome is represented using 3 bits (see Figure 4). Initially, the bits for each chromosome in the population are chosen randomly. For a problem with K vehicles, K-1 seed angles are required, and the length of each chromosome in the population will be (K-1) * 3 bits. The seed angles, S_i, are derived from the chromosome by taking chromosome bits in sets of 3, denoted by B_i, and converting them to a polar coordinate angle using the following equation:

$$S_i = \left[i * \frac{MaxAngle}{K} \right] + INT\ (B_i) * C$$

MaxAngle is the maximum angle value of customers in C_i to C_N. INT is a function that converts a binary string to an integer value. The value of B_i ranges between 0 and 7 because of the 3 bit representation. The constant C is used to increase the value range of B_i. In our experiments, C is set at 1.5 to value range of B_i between 0 and 10.5. The fitness value of a chromosome is the total route cost of the clusters formed from the set of seed angles derived from that chromosome.

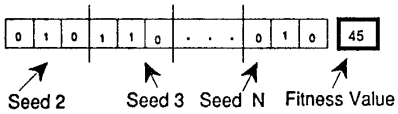

Figure 4. Representation of the Seed Angles in a Chromosome

The parameter values of the genetic search for population size, two point crossover rate and mutation rate were set at 50, 0.8 and 0.001 respectively. The maximum number of generations was set at 25. As the population of seed angles evolves through the generations, they eventually converge to a set of seed angles with a low total route cost relative to other seed angles evaluated over the generations. Convergence is obtained when the difference between the fitness values of the chromosomes in a population are negligible.

Once the clusters are formed, routing is done independently for each cluster. The routing process begins with a vehicle from the depot randomly selecting a customer from a cluster as the first to be visited. To illustrate, assume that the depot is denoted by 0 and for cluster 1 the first customer selected to be served is 3, giving an initial route 0-3. If the next customer to be served is 5, the customer can be inserted between the depot and customer 3, to form the route 0-5-3 or after customer 3 to form the route 0-3-5. The insertion location of customer 5 into the route 0-3 that yields the lowest route cost, using cost function (3.1), will be selected as the next insertion point. For a route that has L customers, to insert the next customer, the cost of L+1 locations are calculated and the location with the lowest route cost is chosen as the next insertion point. This is a selection/insertion algorithm with the cheapest insertion rule [11].

3.2 The SWITCH-OPT Module

Though the GENSECT module is effective in searching the entire search space for the set of seed angles with a low total route cost. There could still be tardy customers or vehicles that exceed vehicle capacity or travel time. For example, if customers C_1 and C_2 differ in their angles by 1 degree and have the same time window they will be assigned to the same cluster and result in tardiness. This inherent disadvantage can be removed by switching customers between clusters by applying local optimization procedures to the best set of clusters obtained from GENSECT. The customer switching method is similar to the cyclic transfer local optimization method of Thompson[20], that moves customers between clusters to reduce the total route cost.

The SWITCH-OPT module switches customers between clusters, by artificially changing their polar coordinate angles, if it leads to reduction in route cost. A customer is switched between clusters either by moving it into a different cluster or exchanging it with a customer from a different cluster. Four types of switches are used in the SWITCH-OPT module. They are called move 1 customer, move 2 customers, exchange 1 customer and exchange 2 customers.

In moving a customer from one cluster to another, the cost of inserting the customer to each of the other clusters is calculated using the (3.1) cost function. The customer is moved into a different cluster that has the lowest route cost with respect to the other clusters. This procedure is carried out for all the customers.

The exchange process begins by selecting a customer from one cluster and calculating the cost, using cost function (3.1), for exchanging it with each of the customers in clusters other than its own. An

exchange is performed if it leads to a reduction in the route cost. This process is carried out for each of the customers in all the clusters.

The local optimization is carried out in the order of move 1 customer, move 2 customers, exchange 1 customer and exchange 2 customers until switching customers does not result in the reduction of route cost. When no more optimization can be performed on the clusters, the GENSECT module can be invoked to form new clusters based upon the new locations of the customers. The iteration between GENSECT and SWITCH-OPT can be executed any number of times, but we have found that two iterations is normally sufficient to attain good solutions.

4.0 Computational Results

GIDEON was run on a set of 56 problems, in six data sets, developed and solved by Solomon[18]. The problems were also solved by Thompson[20]. GIDEON did better than the two alternatives on 41 of the 56 problems with an average reduction of 3.9% in fleet size and 4.4% in distance traveled by the vehicles for the 56 problems. Each problem took an average of 127 cpu seconds to be solved on a SOLBOURNE 5/802 machine.

The problems in these data sets have 100 customers and the fleet size to service them varies from 2 to 21 vehicles. They 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 and customer service times. The geographical distribution of the customers consists of randomly generated customers in data sets R1 and R2, clustered customers in data sets C1 and C2, and semi-clustered customers in data sets RC1 and RC2. The problem sets R1, C1 and RC1 have small vehicle capacities and travel time compared to those of R2, C2 and RC2.

Solomon and Thompson used different initialization criteria and heuristics to obtain solutions for the problem set. The solutions obtained by GIDEON are compared against the best solutions of Solomon[18], denoted by Method 1, and Thompson[20], denoted by Method 2, in Tables 1 through 6. In the tables, each problem number(PROB) shows the percent differences in fleet size(NV%) and distance(DIST%) of the GIDEON solution against the other two methods. Positive values indicate improvement over the other two methods.

The reduction of fleet size and distance obtained by GIDEON were larger for problem sets R1, R2, RC1 and RC2 than for problem sets C1 and C2. Due to the close geographical proximity of customers in problems sets C1 and C2 (see Figure 5), the genetic sectoring process converges upon a feasible solution relatively quickly.

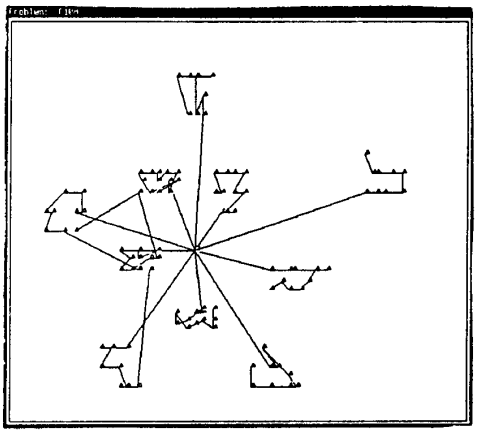

Figure 5. Example Routes in a C1 Problem Set

For randomly located customers the time violations and infeasibilities are greater during the genetic sectoring process (see Figure 6.), leading to a more rigorous adaptive search and better solutions relative to competing methods.

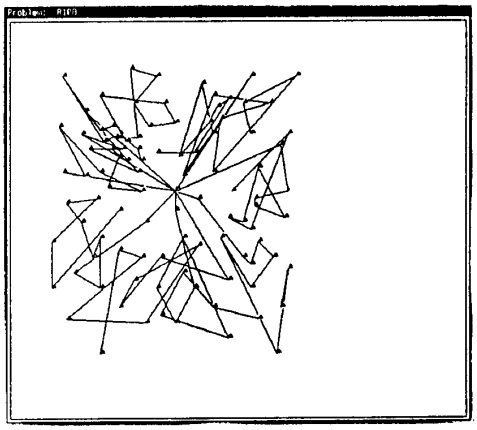

Figure 6. Example Routes in a R1 Problem Set

5.0 Conclusion

This research shows that genetic search can obtain high performance solutions to vehicle routing problems with time windows with a high degree of

efficiency. The adaptive nature of genetic algorithms is exploited by GIDEON to attain solutions that are superior to those of competing methods. This method is potentially useful for solving VRPTW's in real time for dynamically changing routing and scheduling problems.

Acknowledgment

We thank Marius Solomon and Paul Thompson for providing the test data. We gratefully acknowledge the help of Mary Krause who carefully read and edited earlier drafts of this paper.

REFERENCES

[1] Assad, A. and B. Golden, *Vehicle Routing: Methods and Studies*, North Holland, Amsterdam, 1988.

[2] Baker, E. K. and J. R. Schaffer, "Solution improvement heuristics for the vehicle routing problems with time window constraints," *Amer. Jour. of Math. and Mgt. Sciences*, vol. 6, nos. 3-4, pp. 261-300.

[3] 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*, vol. 10, pp. 63-211, 1983.

[4] Brown, D. E., C. L. Huntley and A. Spillane, "A parallel genetic heuristic for the quadratic assignment problem," *Proc. of the Third Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., New Jersey, pp. 416-421, 1989.

[5] Cleveland, G. A. and Stephen F. S., "Using genetic algorithms to schedule shop flow releases," *Proc. of the Third Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., New Jersey, pp. 160-169, 1989.

[6] DeJong, K., "Analysis of the behavior of a class of genetic adaptive systems," Ph.D. Dissertation, Univ. of Michigan, Ann Arbor, 1975.

[7] DeJong, K. and W. Spears, "Using genetic algorithms to solve NP-complete problems," *Proc. of the Third Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., New Jersey, pp. 124-132, 1989.

[8] Fisher, M. L. and R. Jaikumar, "Generalized assignment heuristic for vehicle routing," *NETWORKS*, vol. 11, pp. 109-124, 1981.

[9] Gillett, B. and L. Miller, "A heuristic algorithm for the vehicle dispatching problem," *Operations Research*, vol. 22, pp. 340-349, 1974.

[10] Goldberg D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Massachusetts, 1989.

[11] Golden, B. L. and W. Stewart, "Empirical analysis of heuristics," in *The Traveling Salesman*, Lawler, J. Lenstra, A. Rinooy Kan and D. Shmoys (Eds.), Wiley-Interscience, New York, 207-249, 1985.

[12] Grefenstette, J., "A users guide to GENESIS," Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington D.C. 20375-5000, 1987.

[13] Grefenstette, J., R. Gopal, B. Rosmaita and D. V. Gucht, "Genetic algorithms for the traveling salesman problem," *Proc. of the First Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., New Jersey, pp. 112-120, 1985.

[14] Holland, J. H., *Adaptation in natural and artificial systems*, Univ. of Michigan Press, Ann Arbor, 1985.

[15] Oliver, I. M., D. J. Smith and J. R. C. Holland, "A study of permutation crossover operators on the traveling salesman problem", *Proc. of the Second Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., pp. 224-230, 1987.

[16] Savelsbergh M. W. P., "Local search for routing problems with time windows," *Ann. of Oper. Res.*, vol. 4, pp. 285-305, 1985.

[17] Smith, Derek, "Bin packing with adaptive search," *Proc. of the First Int. Conf. on Genetic Algorithms*, Lawrence Erlbaum Assoc., New Jersey, pp. 202-207, 1985.

[18] Solomon, M. "Vehicle routing and scheduling with time window constraints: models and algorithms," Ph.D. Dissertation, Department of Decision Sciences, Univ. of Pennsylvania, 1983.

[19] Solomon, M. and J. Desrosiers, "Time Window Constrained Routing and Scheduling Problems: A Survey", *Transportation Science* vol. 22 no. 1, pp. 1-11, 1988.

[20] Thompson, P. M., "Local search algorithms for vehicle routing and other combinatorial problems," Ph.D. Dissertation, Massachusetts Institute of Technology, Massachusetts, 1988.

Table 1: % Improvement for data set R1

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
R101	4.7	9.2	-5.3	1.9
R102	10.5	15.9	0.0	17.7
R103	7.1	11.1	13.3	13.8
R104	9.1	8.3	0.0	1.0
R105	0.0	13.5	0.0	5.7
R106	7.1	7.6	0.0	2.1
R107	8.3	16.7	0.0	5.0
R108	0.0	7.8	0.0	-1.3
R109	7.7	4.6	0.0	-7.7
R110	8.3	11.4	8.3	-1.9
R111	8.3	-0.6	8.3	-1.9
R112	0.0	2.2	0.0	1.9
Avg	6.1	9.5	1.9	4.2

Table 2: % Improvement for data set R2

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
R201	0	15.1	0	17.3
R202	0	26.1	0	26.0
R203	0	26.1	0	10.9
R204	0	14.1	0	11.3
R205	0	13.4	0	8.5
R206	0	24.9	0	12.4
R207	0	22.0	0	5.3
R208	0	23.2	-50	4.2
R209	0	20.0	0	8.0
R210	25	17.7	0	0.0
R211	0	11.6	0	16.2
Avg	2.8	19.8	-2.9	11.9

Table 3: % Improvement for data set RC1

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
RC101	6.3	5.4	6.3	4.5
RC102	6.7	10.9	0.0	4.6
RC103	15.4	20.6	8.3	9.4
RC104	0.0	2.9	0.0	0.2
RC105	12.5	16.1	6.7	10.9
RC106	7.7	0.2	-	-
RC107	7.7	-0.8	0.0	4.3
RC108	0.0	0.2	0.0	1.8
Avg	7.4	7.7	3.9	2.7

Table 4: % Improvement for data set C1

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
C101	0	2.3	0	0.5
C102	0	14.1	0	10.9
C103	0	17.6	0	8.7
C104	0	29.5	0	20.0
C105	0	-1.5	0	-5.4
C106	0	-0.6	0	-3.9
C107	0	-2.4	0	0.0
C108	0	-8.5	0	7.1
C109	0	-7.7	0	-4.9
Avg	0	6.3	0	2.7

Table 5: % Improvement for data set C2

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
C201	0	-27.4	0	-27.6
C202	0	-3.4	0	-13.9
C203	0	-5.4	0	-30.9
C204	10	-5.9	0	-17.4
C205	0	-8.5	0	-6.2
C206	0	4.9	0	-8.3
C207	0	5.6	0	-16.4
C208	0	-19.5	0	-9.7
Avg	4.2	-8.1	0	-16.2

Table 6: % Improvement for data set RC2

	METHOD 1		METHOD 2	
PROB	NV%	DIST%	NV%	DIST%
RC201	0	13.3	0	6.9
RC202	0	18.9	0	21.5
RC203	25	18.6	25	13.0
RC204	0	15.5	0	10.7
RC205	20	25.3	0	19.8
RC206	25	3.3	0	-0.9
RC207	25	8.0	25	-3.0
RC208	0	24.4	-	-
Avg	12.9	16.1	8.9	14.3

Legend:

METHOD 1 : Solomon's Method[18]
METHOD 2 : Thompson's Method[19]
PROB : Problem number
NV% : % difference in fleet size
DIST% : % difference in distance