Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms

Joe L. Blanton Jr. **Roger L. Wainwright**

Department of Mathematical and Computer Sciences
The University of Tulsa
600 South College Avenue
Tulsa, Oklahoma 74104-3189
email: rogerw@penguin.mcs.utulsa.edu

Abstract

In this paper we investigate genetic algorithms as a heuristic technique for obtaining optimal or near optimal solutions to the vehicle routing problem with time windows and capacity constraints (VRPTW). The traditional crossover operators for order-based genetic algorithms are not well suited for optimization problems with multiple constraints. We present two new crossover operators, Merge Cross #1 (MX1) and Merge Cross #2 (MX2). These new operators actually produce a family of new crossover operators. The Merge Cross operators are based upon the notion of a global precedence among genes independent of any chromosome. We compared the Merge Cross family of operators with several well known crossover operators: PMX, Cycle and Edge Recombination. Results show that the Merge Cross operators are clearly superior to the traditional operators.

1 INTRODUCTION

The vehicle routing problem (VRP) involves determining minimum cost vehicle routes for a fleet of vehicles originating and terminating from a central location. The fleet of vehicles service a set of customers with a known set of constraints. All customers must be assigned to vehicles such that each customer is serviced exactly once and each vehicle cannot exceed its capacity. The vehicle routing problem has been studied extensively. Bodin (Bodin *et al.* 1983) provides a comprehensive survey of VRP and several variations.

The vehicle routing problem with time windows (VRPTW) adds the additional constraint to the VRP where each customer provides a time window for servicing. Hence, in the presence of time windows, optimization of routing and scheduling of vehicles involves not only total distance traveled, but time costs for waiting when a vehicle has arrived too early to service a customer. Time windows arise frequently in business applications. Examples include bank deliveries, postal deliveries, school bus routing, industrial refuse collection, over night delivery services, passenger and freight operations such as airline, railway and bus routing and scheduling. Time windows also arise in most retail distribution systems. This is an extremely practical problem; efficient routing and scheduling can save industry and government millions of dollars each year. Solomon (Solomon 1987; Solomon *et al.* 1988) provides an excellent survey of the vehicle routing problem with time windows. Solomon makes a distinction between hard and soft time windows. In a hard time window, if a vehicle arrives at a customer too early, it must wait. Furthermore, a vehicle will not be allowed to service a customer if it arrives too late. In a soft time window, the early and late constraint can be violated at some cost. The examples cited above are all hard time windows. An example of a soft time window application is the dial-a-ride problems. We assume a hard time window in our model.

Savelsbergh (Savelsbergh 1985) has shown that the VRPTW is NP-hard. Hence research work in VRPTW involves developing some type of heuristic for effectively searching a vast search space for an optimal or near optimal solution. In this paper we investigate genetic algorithms (GA) as a heuristic technique for obtaining optimal or near optimal solutions to the vehicle routing problem with time and capacity constraints. The rest of the paper is organized as follows. In Section 2 we discuss previous GA research related to VRPTW. Our model for the vehicle routing problem with time and capacity constraints is

Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93), Stephanie Forrest, Editor; Morgan Kaufmann Publisher, 1993, pp. 452-459.

presented in Section 3. We developed a new family of crossover operators specifically for optimizing multiple constraints for a single vehicle. This is presented in Section 4. The Davis encoding for multiple vehicles is developed in Section 5. Results comparing various crossover operators for the VRPTW problem using several data sets are presented in Section 6. A summary is given in Section 7.

2 PREVIOUS VRPTW WORK USING GA

Thangiah (Thangiah *et al.* 1991) has investigated VRPTW using GAs. Their objective function was to minimize the cost of servicing a set of customers without being late or exceeding vehicle capacity or the travel time of a vehicle. They developed a GA, called GIDEON, for heuristically determining optimal or near optimal solutions to the vehicle routing problem with time windows. Their algorithm has three phases. Phase I is a global customer clustering module that assigns vehicles to customers. This phase uses a GA. Assigning vehicles to customers is accomplished by determining what they call seed angles. This is a heuristic way of partitioning a "pie" of customers, serviced from a central depot, into "slices" or clusters such that one vehicle is assigned to service all customers within its cluster. Phase II of their algorithm consists of a standard greedy heuristic technique for routing one vehicle among the customers in its assigned "pie". This is done for each cluster. Finally, phase III performs a standard 2-opting post processing of the customers to determine if some of the customers should be serviced by vehicles from neighboring clusters. Only Phase I makes use of a genetic algorithm.

3 OUR VRPTW MODEL

In our model, each vehicle departs from a central depot, services a set of customers and returns to the depot. Each customer has an *a priori* time window for servicing, and an *a priori* capacity of goods to receive. We simulate a working day such that each vehicle must return to the depot by the end of the day. To ensure this, we assign a time window to the depot and consider the depot as the last customer in all routes. When a vehicle arrives at each customer, a time delay occurs for unloading the capacity of goods. The time delay is proportional to the amount goods delivered. The vehicle then continues to the next customer on the route. In our VRPTW model, the goal is to determine the minimum number of vehicles required to service all of the customers within their allowed time windows subject to vehicle capacity constraints. Having satisfied these constraints, the objective is to minimize the total routing cost (distance). Our model searches for a single global solution to the VRPTW problem, rather than locally optimizing three separate phases of the problem as proposed by Thangiah.

There exists a global vector of time windows for each of the customers. For each customer we define the earliest arrival time allowed, time spent at each location, and latest arrival time allowed. Early arrivals are required to wait until the early arrival time for a particular customer in the tour. In our VRPTW, all customers are visited no later than their defined time windows.

4 NEW CROSSOVER OPERATORS

An order-based GA is when all chromosomes are a permutation of a list. That is, the solutions are coded as sequences. Order-based GAs can be applied to a wide range of combinatorial optimization problems. The VRPTW problem is one example. Other classical problems that can be solved using an order-based GA include the Traveling Salesman Problem, Bin Packing, Package Placement, Job Scheduling, Network Routing, Stock Cutting, various other layout problems, etc.

The traditional crossover operators for order-based GAs include Edge Recombination, Order Crossover #1, Order Crossover #2, PMX, Cycle Crossover and Position Crossover. These functions have been well studied and are described by Whitley (Whitley *et al.* 1990) and Starkweather (Starkweather *et al.* 1991). Each of these traditional crossover functions have two common factors: (1) they are based on a local precedence relationship among the genes in a chromosome and (2) the crossover operator is dependent only on the contents of the chromosomes involved.

Fox (Fox *et al.* 1991) examined genetic operators for solutions coded as sequences (ie. order-based GAs) in an attempt to improve on the traditional crossover operators. They presented several new crossover operators. The Intersection and Union crossover operators are two examples. These operators are based on the concept of a complete precedence matrix as defined by a chromosome. The genes in a chromosome by virtue of their position in the sequence define a complete precedence matrix. That is, each gene in the chromosome has a precedence relationship to every other gene. Thus each chromosome defines it own precedence matrix. They argue successfully that this gives more information for a crossover operator to work with compared to the traditional crossover operators. However, the crossover operators presented by Fox, like the traditional operators, are dependent only on the content of a specific chromosome.

We have found that the traditional crossover operators for order-based GAs are not very conducive for optimization problems with multiple constraints such as the VRPTW problem. We present two crossover operators Merge Cross #1 (MX1) and Merge Cross #2 (MX2) designed for multiple optimization order-based GA problems. The Merge Cross operators are based upon the notion of a global precedence among genes **independent** of any chromosome, rather than defining a local precedence among genes specific to a chromosome as the edge recombination operator (Whitley *et al.* 1989) and the intersection and union crossover operators (Jones *et al.* 1991).

In order for the merge cross operators to be effective a global precedence arrangement among the genes is required. That is, given a set of N possible genes, there must exist a definable precedence relationship such that there is a general desire that gene *i* occur before gene *j* within any chromosome, for all genes, *i* and *j*. In the vehicle routing problem with time windows each gene is a customer and has an associated service time window. Hence, there is a natural precedence relationship among all customers based on their time window. The crossover operators, MX1 and MX2 defined below are designed for a single vehicle, multiple optimization order-based GA problems. However, MX1 and MX2 can also be used (without modification) to apply to multiple vehicles.

The Merge Cross #1 (MX1) crossover operator on chromosomes A and B results in a single child, AB, as shown below:

Chromosome A: 2 5 6 1 0 7 3 8 4 9
Chromosome B: 4 1 6 9 3 8 2 0 5 7

Early Time Ordering: 0 1 2 3 4 5 6 7 8 9
(Global Precedence Vector)

The genes under consideration are underlined. The first gene of A is found to be earlier in precedence in the global precedence vector than the first gene of B. The gene with the earlier precedence is placed in AB. Note B's first gene is swapped to maintain validity. This is highlighted in bold face.

Chromosome A: 2 5 6 1 0 7 3 8 4 9
Chromosome B: **2** 1 6 9 3 8 4 0 5 7
Child AB: **2** X X X X X X X X X

The second gene of B is found to be earlier in precedence in the global precedence vector than the second gene of A. Again the gene with the earlier precedence is placed into the child, and in this case A's second gene is swapped to maintain validity.

Chromosome A: 2 1 6 5 0 7 3 8 4 9
Chromosome B: 2 1 6 9 3 8 4 0 5 7
Child AB: 2 1 X X X X X X X X

Continuing in this fashion, the resulting child chromosome, AB, is shown below.

Child AB: 2 1 6 5 0 7 3 4 8 9

Effectively, this produces a child which is closer to the order of the global precedence.

The Merge Cross #2 (MX2) crossover operator on chromosomes A and B results in a single child, AB, as shown below:

Chromosome A: 2 5 6 1 0 7 3 8 4 9
Chromosome B: 4 1 6 9 3 8 2 0 5 7

Early Time Ordering: 0 1 2 3 4 5 6 7 8 9
(Global Precedence Vector)

This operator works like a merge algorithm, merging two sorted vectors. The first gene of A is found to be earlier in precedence in the global precedence vector than the first gene of B. The gene with the earlier precedence is placed in AB. Now A's first gene is removed in both chromosomes.

Chromosome A: X 5 6 1 0 7 3 8 4 9
Chromosome B: 4 1 6 9 3 8 X 0 5 7
Child AB: 2 X X X X X X X X X

The next gene of each chromosome is compared and the first gene of B is found to be earlier in precedence. Again the gene with the earlier precedence is placed into the child, and removed from both chromosomes.

Chromosome A: X 5 6 1 0 7 3 8 X 9
Chromosome B: X 1 6 9 3 8 X 0 5 7
Child AB: 2 4 X X X X X X X X

The next iteration produces

Chromosome A: X 5 6 X 0 7 3 8 X 9
Chromosome B: X X 6 9 3 8 X 0 5 7
Child AB: 2 4 1 X X X X X X X

Continuing in this fashion, the resulting child chromosome, AB, is shown below.

Child AB: 2 4 1 5 6 0 7 3 8 9

Note when gene 9 is encountered in chromosome B, the rest of chromosome A is used. This operator has the side effect of moving the latest customer to the end of the chromosome.

The MX1 and MX2 operators described above are based on a global precedence relationship among the genes. In this instance it is based on time. There is no reason why these operators could not be based on other global precedence relationships, such as distance, capacity or any other optimization criteria. An example of MX1 based on distance is given below. Consider the original two Chromosomes A, B. The process begins with a time comparison to get started. This results in:

Chromosome A: 2 5 6 1 0 7 3 8 4 9
Chromosome B: 2 1 6 9 3 8 4 0 5 7
Child AB: 2 X X X X X X X X X

Next, the second gene of each parent is compared in distance to the first gene in the child. Assume distance(2,5) < distance(2,1) then the following results:

Chromosome A: 2 5 6 1 0 7 3 8 4 9
Chromosome B: 2 5 6 9 3 8 4 0 1 7
Child AB: 2 5 X X X X X X X X

At each instance the next genes of the parents are compared in distance with the previously generated gene of the child until the child is completely generated. The MX2 operator based on distance is performed in a similar manner.

To distinguish between the various operators, we define MX1T as the MX1 operator based on time, and MX1D as the MX1 operator based on distance. Based on the MX2 operator, we define MX2T and MX2D in the same manner. In fact it is possible to alternate between two or more precedence relationships while constructing a single child chromosome. The MX1T/D operator refers to the MX1 operator alternating between time and distance when selecting each new gene of the child. That is, time is used to select the first gene of the child, distance is used to select the second gene of the child, time is used to select the third gene of the child, and so forth. The MX1T/D/C operator alternates between time, distance and capacity when selecting each new gene in the child. The MX2T/D and MX2T/D/C operators are defined in the same way. We append an "*" after an operator such as MX1T/D* to indicate the first application of the operator behaves like MX1T/D and the second application behaves like MX1D/T. The operations alternate back and forth after that. Hence the basic MX1 and MX2 crossover operators produce a family of new operators, MX1T, MX1D, MX2T, MX2D, MX1T/D, MX2T/D, MX1T/D/C, and MX2T/D/C that produce a single child from two parents.

The MX1 and MX2 operators also produce a family of operators that yield two children from two parents by combining single child operators. For example, the MX1T-2T operator produces two children, one via the MX1T operator and the other via the MX2T operator. The MX1T-1T/D operator likewise produces two children based on the MX1T and MX1T/D operators. We define additional two child operators, MX1T-2T/D, MX2T-2T/D, and MX2T-1T/D in a similar manner. There are also many other combinations. All of the new crossover operators give versatility in solving order-based GA optimization problems with multiple constraints. In our previous, work we described some of the merge cross operators as they applied to the VRP for a single vehicle (Blanton *et al.* 1992).

5 ENCODING FOR MULTIPLE VEHICLES

Jones (Jones *et al.* 1991) investigated partitioning problems using genetic algorithms. To solve partitioning problems with GAs it is necessary to encode partitions in a way that allows manipulation by the genetic operators. The goal is to partition the permutation into k groups according to some criteria. They presented several methods for encoding a partition as a permutation of n objects. One of the methods is a greedy heuristic which they credit to L. Davis. The greedy heuristic takes the first k objects in the permutation to initialize (seed) the k groups. The remaining objects of the permutation are then added to the groups one at a time. A given object is added to the group that yields the best objective function. We will call this the Davis encoding method.

We adapted the Davis encoding method to the VRPTW problem where a chromosome represents a permutation of n customers to be partitioned into k vehicles. The number of vehicles and the capacity of each vehicle are assumed to be known by the evaluation (decoding) function. The evaluation function assumes the first k customers of a chromosome are placed (in order) into the k vehicles, one per vehicle as outlined by the Davis decoding method. Having made these initializations, the algorithm continues to build upon the customers allocated to each vehicle in a deterministic and repeatable fashion.

The remaining n - k customers are examined individually. As each new customer is selected, a possible subtour is evaluated for each vehicle and the best subtour is selected. That is, the subtour created by adding the new customer to a particular vehicle is evaluated, tracking the best subtour evaluation to date. Once all vehicles have been examined the customer can be added to the appropriate vehicle.

Thus, the evaluation of the subtours dictate the final evaluation of the overall chromosome.

Checking the effect of a customer within a subtour is accomplished by first verifying that each customer's time window is satisfied. If it is the best subtour evaluated so far, the value is saved and the vehicle noted. Upon subsequent subtour evaluations that result in the same value, the tie is resolved by checking which subtour adds the least amount of distance to the solution.

Capacity is maintained for each vehicle's subtour by adding the capacity values of each customer within the subtour. Exceeding the maximum allowable value results in an unserviced customer, just as though it had not satisfied the time window. Thus a customer is considered unserviced for either a time window violation or exceeding vehicle capacity at that point. Notice this strategy is exactly the classic bin packing problem, where the customers are the packages and the vehicles are the bins. Note further the evaluation function uses the Davis encoding method to partition the customers among k vehicles. However, the MX family of operators operates on the customers as they appear in the chromosome without reference to how they are partitioned among vehicles.

6 RESULTS

We tested our MX Crossover operators on four models consisting of 15, 30, 75, and 99 customers. In the 15, 30 and 75 models, the customers were randomly placed in a rectangular grid. The 30 and 75 customer problems come from standard Traveling Salesman Problem data found in (Whitley *et al.* 1989). For each customer we randomly added time window constraints and a random capacity constraint. We also assumed each vehicle had a fixed capacity. The 99 customer model comes from actual data obtained from a local retail distribution company. We used time windows and capacity data for their customers for a particular day. Figure 1 depicts the routes for 39 trucks servicing 99 customers for this company for this example day. For reference, an outline of the state of Oklahoma (without the panhandle) appears in the figure. The central depot is located in Tulsa. The units are in miles. We simulated an eight hour day for the 15, 30 and 75 customer problems, and a 13 hour day for the 99 customer problem. For each problem we tested the following crossover operators: Edge Recombination, Cycle, PMX, MX1T, MX2T, MX1T/D, MX1T/D/C, MX2T/D, MX1T/D*, MX1T-2T, MX1T-1T/D MX1T-1T/D*, MX2T-2T/D, MX2T-1T/D and MX2T-1T/D*. It is important to note the Merge Cross family of crossover operators are actually defined with one vehicle in mind. However, our results show these operators (without modification) work very well on chromosomes that encode multiple vehicles represented by the Davis encoding.

We used the GENITOR package (Whitley *et al.* 1988) modified for our needs. In all cases the mutation rate was .05. The mutation operator randomly exchanges two genes in a chromosome. The evaluation function is defined as follows: if there are customers that are not serviced (either because of time window or capacity violations) the evaluation function is the number of unserviced customers. These are infeasible solutions. If all of the customers are serviced within their time windows and capacities, then the evaluation function is -100,000 plus the sum of the Euclidian distances of all of the vehicle routes. These are feasible solutions. In our test cases, the sum of the distances for all of the tours is well under 100,000. Hence this ensures a negative value. In this way GENITOR will rank the chromosomes properly, since infeasible solutions evaluate to positive values. In general, we are not concerned about minimizing distance until we have located a feasible solution. We used a bias factor of 1.19 which reduces the effect of elitism as much as possible. This yields a more thorough search of the state space. The population size was 50 for the 15 and 30 customer problems, 100 for the 75 customer problem, and 250 for the 99 customer problem.

Tables I, II, III and IV depict the results of the 15, 30 75 and 99 customer problems, respectively. Each Table entry indicates the number of customers not serviced and the total distance of all of the routes. Results are shown for several vehicles, k. Only entries with zero customers not serviced are feasible solutions. A valid solution means all of the customers were serviced within their time windows and capacity constraints. For the 15 customer problem, Cycle and PMX did fairly well, but Edge Recombination did not. The best solutions ($k = 4$, distance = 456) resulted from using MX2T and MX2T/D. In the 30 customer problem, PMX did fairly well. Cycle and especially Edge Recombination did not. Most of our MX operators outperformed PMX. The best results came from MX1T for $k =$ 10 with a distance of 1101. In the 75 customer problem, Edge Recombination, Cycle and PMX failed to obtain a solution at $k = 21$. All of the MX operators, without exception, obtained valid solutions. The best solution came from MX1T-1T/D, for $k = 21$ at a distance of 1705.

At first, our best results using the MX operators for the 99 customer problem was around 50 vehicles. The traditional operators did much worse. However, the retail distribution company that supplied the data used only 39 vehicles. The problem is the first k customers are assigned to the first k vehicles in the Davis encoding scheme. If the first 5 customers, for example, are for stores in the same city 300 miles from the depot, the Davis encoding scheme will send

5 vehicles to each store individually, when only one is needed. The Davis encoding and the MX operators work fairly well for randomly generated points, as illustrated by the previous test cases, but work very poorly on non-random data, as shown in Figure 1. There are several simple solutions to this problem. First, the Davis encoding scheme can be modified to remove the seeding of the first *k* customers. A simple modification is all that is required, to develop similar heuristics to the First Fit, Next Fit or Best Fit heuristic strategies used in the classic Bin Packing Problem. Secondly, the researcher can initially seed the first *k* customers before processing. We elected to seed the first *k* vehicles and run the 99 customer problem over again. We also tested only those MX operators that performed well in the previous test cases. The results are shown in Table IV. All of the Merge Cross operators obtained feasible solutions using 30, 31 or 32 vehicles. Edge Recombination, Cycle and PMX did not yield feasible solutions even with 33 vehicles. The best solution resulted from MX1T-2T at 30 vehicles with a distance of 8973. The best 31 vehicle solution resulted from MX2T-1T/D and MX2T-1T/D* at a distance of 8803. In our test cases, as the problem size increases the MX family of operators increasingly out performed PMX, Cycle and Edge Recombination.

7. SUMMARY

The MX operators as a group outperformed the traditional order-based operators in all of the cases tested. In each instance one of our new operators was the best performer. No single MX operator was consistently better than any of the others. Cycle, PMX, and Edge Recombination showed signs of faltering for the larger customer problems, while the MX operators always found solutions. We have shown that all of the MX operators are excellent crossover operators for solving problems where there exists a global precedence relationship among the genes.

Acknowledgements

This research was supported by OCAST Grant ARO-038. The authors wish to acknowledge the support of Sun Microsystems Inc. The authors also wish to thank Bill Doyle and Hale-Halsell Company for providing data for this project from their retail distribution center in Tulsa.

References

Blanton, J.L. and Wainwright, R.L. (1992) "Vehicle Routing with Time Windows using Genetic Algorithms", Proceedings of the 6th Oklahoma Symposium on Artificial Intelligence, Tulsa, OK., pp. 242-251.

Bodin, L., Golden, B., Assad, A. and Ball, M. (1983) "Routing and Scheduling of Vehicles and Crews: The State of the Art," *Comput. Opns. Res.* vol. 10, pp. 62-212.

Fox, B.R. and McMahon, M.B. (1991) "Genetic Operators for Sequencing Problems", in *Foundations of Genetic Algorithms*, G.J.E. Rawlings, ed., Morgan Kaufmann

Jones, D.R. and Beltramo, M.A. (1991) "Solving Partitioning Problems with Genetic Algorithms", *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufmann Pub., pp. 442-449.

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

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

Solomon, M.M. and Desrosiers, J. (1988) "Time Window Constrained Routing and Scheduling Problems: A Survey. *Transportation Science* vol. 22 (1), pp. 1-11.

Starkweather, T., McDaniel S., Mathias K., Whitley D. and Whitley, C. (1991) "A Comparison of Genetic Sequencing Operators", *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufmann Pub.

Thangiah, S.R., Nygard, K.E. and Juell, P.L. (1991) "GIDEON: A Genetic Algorithm System for Vehicle Routing with Time Windows", Proceedings of the Seventh Conference on Artificial Intelligence Applications, Miami, Florida, pp. 322-325.

Whitley D. and Kauth J. (1988) GENITOR: A Different Genetic Algorithm, *Proceedings of the Rocky Mountain Conference on Artificial Intelligence*, Denver, Co., pp. 118-130.

Whitley, D., Starkweather T. and Fuquat D. (1989) "Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator", *Proceedings of the Third International Conference on Genetic Algorithms*,, Morgan Kaufmann Pub.

Whitley, D. and Starkweather, T. (1990) "GENITOR II: A Distributed Genetic Algorithm, *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 2, pp. 189-214.

15 Customers			
Crossover	k=2	k=3	k=4
edge	6/422	3/474	1/515
cycle	5/483	2/517	0/466
pmx	6/509	2/439	0/523
mx1T	6/492	3/559	0/483
mx2T	6/505	3/521	0/456
mx1T/D	7/459	3/432	0/524
mx1T/D/C	5/499	2/468	1/456
mx2T/D	6/505	3/521	0/456
mx1T/D*	5/473	2/525	0/469
mx1T-2T	6/473	4/462	0/489
mx1T-1T/D	6/434	3/485	0/462
mx1T-1T/D*	6/525	2/456	0/522
mx2T-2T/D	7/473	3/432	1/488
mx2T-1T/D	6/498	3/472	0/503
mx2T-1T/D*	6/498	3/472	0/503

Table I: Results of 15 Customers
(Customers Not Serviced/Best Distance)

30 Customers			
Crossover	k=9	k=10	k=11
edge	5/1212	2/1300	2/1101
cycle	2/1381	1/1407	0/1118
pmx	2/1279	0/1252	0/1084
mx1T	2/1234	0/1101	0/1072
mx2T	2/1246	0/1193	0/1189
mx1T/D	2/1254	0/1168	0/1150
mx1T/D/C	3/1246	1/1383	0/1106
mx2T/D	2/1246	0/1193	0/1189
mx1T/D*	2/1305	0/1117	0/1038
mx1T-2T	2/1238	0/1214	0/1083
mx1T-1T/D	2/1364	0/1120	0/1058
mx1T-1T/D*	2/1409	0/1161	0/1124
mx2T-2T/D	2/1211	0/1428	0/1049
mx2T-1T/D	3/1393	0/1156	0/1122
mx2T-1T/D*	3/1393	0/1156	0/1122

Table II: Results of 30 Customers
(Customers Not Serviced/Best Distance)

75 Customers			
Crossover	k=19	k=20	k=21
edge	11/2007	9/2022	6/2325
cycle	5/1935	3/2147	2/2076
pmx	7/2243	4/1914	3/1953
mx1T	3/1966	1/1807	0/1749
mx2T	3/1936	2/1966	0/1916
mx1T/D	3/2050	1/2206	0/1816
mx1T/D/C	5/2151	2/2056	0/2523
mx2T/D	3/1936	2/1966	0/1916
mx1T/D*	3/1932	1/2212	0/1743
mx1T-2T	2/1910	2/2181	0/1851
mx1T-1T/D	2/1904	1/2013	0/1705
mx1T-1T/D*	3/2025	1/1999	0/1722
mx2T-2T/D	3/2022	1/2026	0/2013
mx2T-1T/D	3/2246	1/2148	0/1782
mx2T-1T/D*	3/2246	1/2148	0/1782

Table III: Results of 75 Customers
(Customers Not Serviced/Best Distance)

99 Customers				
Crossover	k=30	k=31	k=32	k=33
edge	14/9299	13/8932	11/8954	12/9078
cycle	6/8916	3/9244	2/9714	2/9991
pmx	8/8298	8/8882	7/8737	6/9398
mx1T	0/8985	0/8933	0/8906	–
mx2T	1/8887	1/9293	0/9314	–
mx1T/D	4/8995	1/9231	0/9355	–
mx2T/D	1/8887	1/9293	0/9314	–
mx1T/D*	1/8889	0/9033	0/8748	–
mx1T-2T	0/8973	0/9001	–	–
mx1T-1T/D	1/9022	0/8865	–	–
mx1T-1T/D*	1/9031	0/8995	–	–
mx2T-1T/D	0/9276	0/8803	–	–
mx2T-1T/D*	0/9276	0/8803	–	–

Table IV: Results of 99 Customers
(Customers Not Serviced/Best Distance)

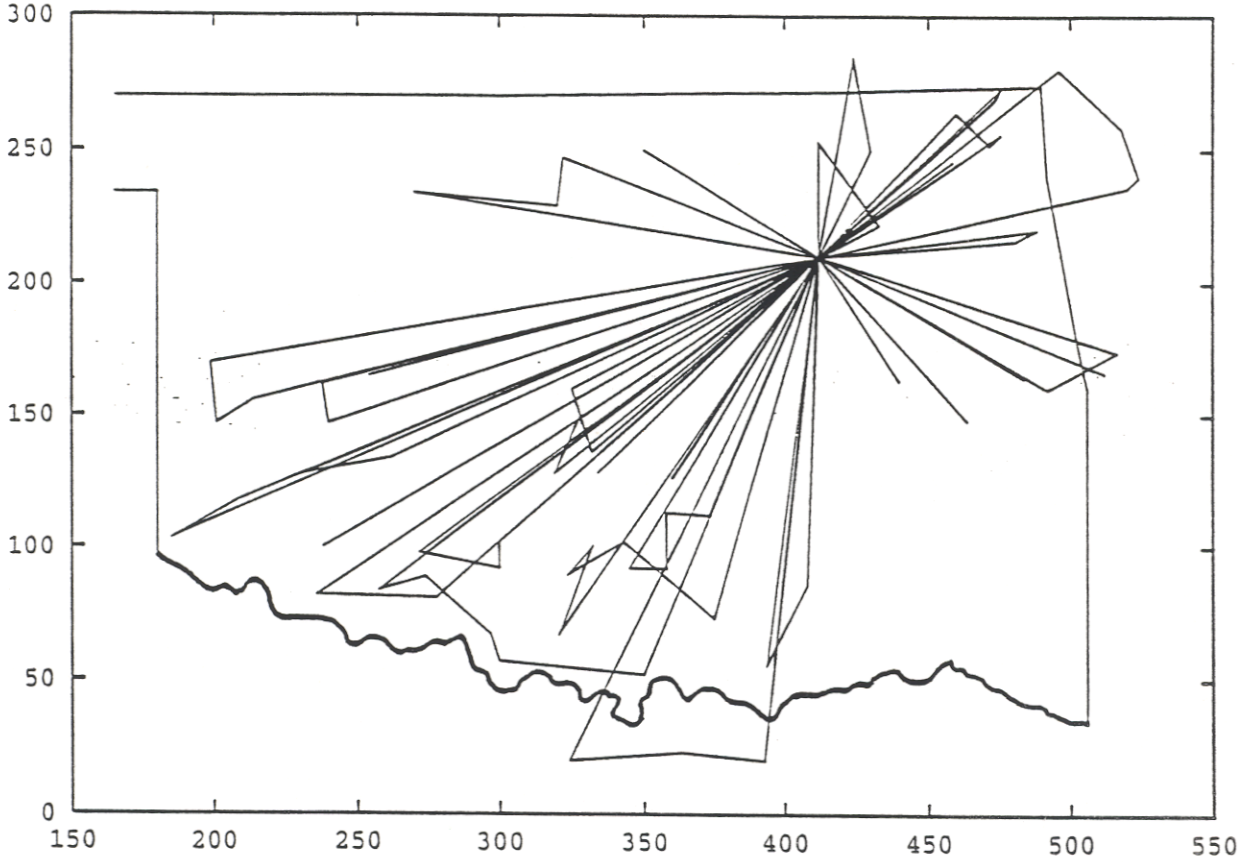

Figure 1. Data from a Local Retail Distribution Company
(99 Customers and 39 Vehicles)