



University of Nebraska at Omaha  
DigitalCommons@UNO

---

Student Work

---

4-1-2003

## An evolutionary approach to vehicle routing problem with dynamic time and precedence relationships.

Darin C. Plum

Follow this and additional works at: <https://digitalcommons.unomaha.edu/studentwork>

---

### Recommended Citation

Plum, Darin C., "An evolutionary approach to vehicle routing problem with dynamic time and precedence relationships." (2003). *Student Work*. 3564.

<https://digitalcommons.unomaha.edu/studentwork/3564>

This Thesis is brought to you for free and open access by DigitalCommons@UNO. It has been accepted for inclusion in Student Work by an authorized administrator of DigitalCommons@UNO. For more information, please contact [unodigitalcommons@unomaha.edu](mailto:unodigitalcommons@unomaha.edu).



**An Evolutionary Approach  
to Vehicle Routing Problem  
with Dynamic Time and  
Precedence Relationships  
A Thesis Equivalent Project  
Presented to the  
Department of Computer Science  
and the  
Faculty of the Graduate College  
University of Nebraska  
In Partial Fulfillment  
of the Requirements for the Degree  
Masters of Science  
University of Nebraska at Omaha**

**by  
Darin Plum  
April 2003**

UMI Number: EP74762

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI EP74762

Published by ProQuest LLC (2015). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code



ProQuest LLC.  
789 East Eisenhower Parkway  
P.O. Box 1346  
Ann Arbor, MI 48106 - 1346

## Thesis Equivalent Project Acceptance

Acceptance for the faculty of the Graduate College,  
University of Nebraska, in partial fulfillment of the  
requirements for the degree Masters of Science,  
University of Nebraska at Omaha

### Committee

Dr. Deepak Khazanchi

Deepak Khazanchi

Professor Stanley Wileman

S. Wileman

Dr. Hesham Ali

Hesham Ali

Chairperson Dr. Hesham Ali

Hesham Ali

Date

April 23<sup>rd</sup>, 2003

# **An Evolutionary Approach to Vehicle Routing Problem with Dynamic Time and Precedence Relationships**

**Darin C. Plum, MS**

**University of Nebraska, 2003**

**Advisor: Dr. Hesham Ali**

The vehicle routing problem (VRP) deals with the allocation of vehicles to the customers that have requested products from a main depot. When the postal service, the bus system, or trucking industry plan for their everyday tasks of delivery goods or providing basic transportation needs to people, they are attempting to solve the VRP. However, this can be a daunting task because the VRP is computationally intensive. In this paper, we will address two areas of the VRP that have been relatively unexplored by previous research, yet play an important part in real-world applications of the VRP as well as define and create an evolutionary approach based on these ideas. The first area that we will address deals with the lack of real-world data sets when calculating the times between customers. This new implementation will allow the algorithm to calculate the actual time between customers at a given time of day, thus providing a final solution that is closer to a true optimal set of routes. The second addition to the VRP model will be contributed in the area of precedence relationships. These relationships often exist in real-world applications and are necessary in order to help the algorithm establish routes that are desired by the customers. The evolutionary approach provides global optimization insight to the problem.

## Table of Contents

<b>1.0 Introduction.....</b>	<b>1</b>
1.1 Basic Problem .....	1
1.2 Motivation of Work .....	4
1.3 Summary of Previous Work.....	5
1.4 Shortcomings .....	7
1.5 Project Goals .....	9
<b>2.0 Definition .....</b>	<b>12</b>
2.1 Vehicle Routing Problem Definition .....	12
2.2 Evolutionary Algorithm Definition.....	13
2.3 Proposed Additions to the Model .....	13
<b>3.0 Background and Previous Research .....</b>	<b>16</b>
3.1 Previous Research.....	16
3.1.1 Overview .....	16
3.1.2 Genetic Algorithm approach to VRP with Time Windows .....	18
3.1.3 Ant Colony approach to VRP with Time Windows .....	18
3.1.4 Set Covering Formulation for the VRPTW .....	19
3.1.5 Dynamic Programming approach to VRP with Split Pick-Ups.....	20
3.1.6 Minimum K-Trees optimal approach to the VRP.....	20
3.1.7 Tabu Search Heuristic.....	21
<b>4.0 Specification.....</b>	<b>22</b>
4.1 Problem Statement.....	22
4.2 Proposed Model .....	23
4.2.1 Basic VRP Model .....	23
4.2.2 Precedence Relationship VRP Model.....	25
4.2.3 Dynamic Travel Time VRP Model.....	27
4.3 Proposed Approach.....	30
4.3.1 Genetic Algorithm .....	30
4.3.1.1 VRP Basic Genetic Algorithm Structure .....	30
4.3.1.2 The Distance Matrix .....	33
4.3.1.3 The Precedence Relationship.....	33
The Chromosome.....	34
4.3.1.4 The Fitness Function.....	34
4.4 Greedy Algorithm Solution: An Alternative Approach.....	39
4.5 Random Distance Matrix Generation .....	42
4.6 The Graphical User Interface .....	44
4.7 The Database.....	47
<b>5.0 Obtained Results.....</b>	<b>50</b>
5.1 Example Test Results.....	50
5.1.1 Basic City Routing Example.....	50
5.1.2 Random Example Case Study.....	54
5.1.2.1 Small Population .....	55
5.1.2.2 Medium Population.....	58

5.1.2.3	Large Population.....	61
5.1.2.4	Comparison and Analysis .....	65
5.1.3	Graphical User Interface Example.....	67
5.1.3.1	GUI Example Software and GA Inputs .....	67
5.1.3.2	Results.....	68
<b>6.0</b>	<b>Conclusions and Future Research.....</b>	<b>71</b>
6.1	Conclusions.....	71
6.2	Ideas for Future Research .....	72
6.3	Lessons Learned.....	73
<b>7.0</b>	<b>References.....</b>	<b>75</b>

## Figures and Tables

Figure 1-1:	Diagram Illustrating How the Variations of the VRP Relate to Each Other ..	3
Figure 2-1:	Travel Time to Work.....	14
Figure 4-1:	Undirected Graph of Set V.....	24
Figure 4-2:	Distance Matrix $c$ Measured in Minutes .....	24
Figure 4-3:	Undirected Graph of Set V.....	25
Figure 4-4:	Distance Matrix $c$ Measured in Minutes .....	26
Figure 4-5:	Undirected Graph of Set V.....	28
Figure 4-6:	Distance Matrix $c_{ij}$ in Minutes.....	29
Figure 4-7:	Precedence Relationship .....	33
Figure 4-8:	The Chromosome.....	34
Figure 4-9:	VRP Architecture.....	45
Figure 4-10:	VRP Graphical User Interface .....	45
Figure 4-11:	VRP Input Interface .....	46
Figure 4-12:	VRP Route Output Table .....	47
Figure 4-13:	VRP Performance Data Output Table.....	47
Figure 4-14:	VRP Inputs Table.....	48
Figure 4-15:	VRP Distance Matrix Table.....	48
Figure 4-16:	VRP Precedence Matrix Table.....	49
Figure 4-17:	VRP GA Termination Table .....	49
Figure 5-1:	Basic City Example Software Inputs .....	51
Figure 5-2:	Basic City Example Dynamic Distance Matrix $e$ .....	51
Figure 5-3:	Basic City Example Genetic Algorithm inputs.....	52
Figure 5-4:	Fitness Function Vs. Total Distance Traveled. ....	53
Figure 5-5:	Random Example Software Inputs .....	55
Figure 5-6:	Small Population Genetic Algorithm Inputs .....	55
Figure 5-7:	Medium Population Genetic Algorithm Inputs.....	58
Figure 5-8:	Large Population Genetic Algorithm Inputs.....	61
Figure 5-9:	Comparison of Random Example Solutions.....	65
Figure 5-10:	The Inputs Used for the VRP GUI Example.....	68
Figure 5-11:	The Interface of the VRP GUI Example.....	70

## **1.0 Introduction**

Understanding the Vehicle Routing Problem (VRP) is essential when one attempts to understand how our society works. The basic VRP, which has several variations that will be discussed later in this section, can simply be defined as a fleet of vehicles with limited capacity that service customers in different locations with the objective of minimizing travel distance as well as the number of vehicles used. There may also be other constraints applied to this problem such as specific time windows for each location in which the vehicle must arrive, the weight of the goods the vehicle can transport, or the ability of the goods to be divided among multiple vehicles. Since the VRP problem is computationally intensive [7], it is often necessary to use a heuristic to find a near optimal solution in a relatively short amount of time.

This section will introduce the VRP and explain the motivation for the selection of the topic. Previous research will also be overviewed including both non-polynomial optimal-solution approaches and heuristic approaches. After these details have been presented, the proposed project goals will be stated.

### ***1.1 Basic Problem***

When the postal service, the bus system, or trucking industry plan for their every day tasks of delivery goods or providing basic transportation needs to people, they are attempting to solve a version of the VRP. The best optimal-solution approaches run in



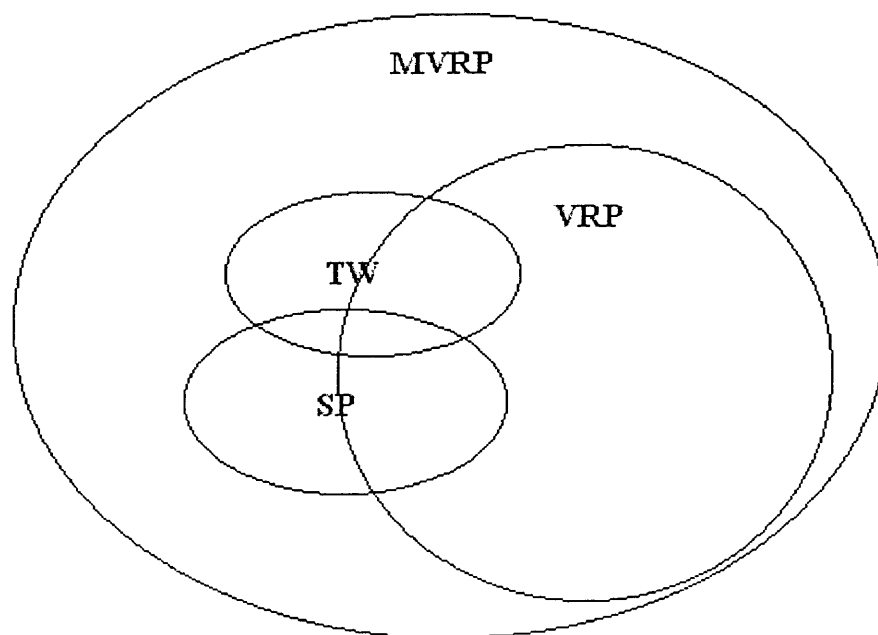
non-polynomial time and can only handle limited numbers of customers and then only for certain data sets. Even with a relatively small number of customers, these algorithms often have to be tailored to handle real-world data sets [2]. Alternatively, heuristic approaches can solve large real-world data sets in a reasonable amount of time with potentially optimal to near optimal solutions.

The VRP can be defined as the allocation of vehicles to customers that have requested goods or services. These vehicles are initially located at a depot, and after they complete their assigned routes they return to the central depot. The objective of the problem is to service all of the customers using the minimum number of vehicles and minimizing the total distance traveled by the vehicles. There are several common constraints placed on the problem:

- Vehicles must complete their routes within a given time frame or work day
- There are a limited number of vehicles that can be used to service the customers
- Each customer is visited once by a single vehicle
- Vehicles may be limited to a certain capacity

Other variations of the basic VRP include the Multiple Vehicle Routing Problem (MVRP), the Vehicle Routing Problem with Time Windows (VRPTW), the Multiple Vehicle Routing Problem with Time Windows (MVRPTW), and the Multiple Vehicle Routing Problem with Split Pickups. The MVRP differs from the standard VRP in that the same vehicle can be used on more than one route. This allows the vehicle to service

several customers on a route, then return to the depot, refill its payload, and service several new customers on a new route. Time windows are added to the basic problem for both the VRPTW and MVRPTW problems. This acts as an extra constraint that forces the vehicles to arrive at the customer within a predefined time window. If the vehicle arrives early, the vehicle must wait until it is within the time window before the customer will accept the goods. If the vehicle arrives after the time window, it is marked as tardy and is penalized. Another variation of the VRP allows for split pickups (VRPSP). This allows each customer to potentially be serviced by multiple vehicles. This type of variation is often found in industries where the size of goods being picked up or delivered is often large enough that one truck can not hold the capacity requested by a customer.



**Figure 1-1: Diagram Illustrating How the Variations of the VRP Relate to Each Other**

It is important to note that although the VRP is usually defined in terms of automobiles servicing customers, the ideas presented in this paper may be applied to a wide range of fields. The VRP could potentially be applied to packets sent over a network, the administrative processes within a company, or the complex allocation of data and products within an e-business framework.

## ***1.2 Motivation of Work***

This work was motivated by a real-world example of the VRP found in the armored transport industry. Armored transport companies drop off and pick up funds from banks on a daily basis. Routing the armored transports between the banks can be a complicated problem because the number of banks may change on a daily basis as well as the banks themselves. The armored transport companies must determine the most efficient routes using the minimum number of armored transports. In the case that inspired this work, the armored car company is using several people to determine the best route for the armored transports. This is quite expensive for the companies and could be replaced by a computer-based approach using a heuristic such as a genetic algorithm (GA) to find a near optimal solution. The computer-based approach could save the company money both in the human resources of the planners and the drivers, as well as the cost of using extra vehicles to do improperly, non-optimal routes that often arise when manual routing is done.

Although the problem is challenging, the benefits of solving the VRP can be great. A study in Oyster Bay New York showed the computerized schedule for the sanitation vehicles saved three vehicles out of 40 and \$750,000 per year [8]. The Montreal Urban Community Transit system also saved 3% of the drivers' salaries over the previous manual system or about 2 million dollars per year [8].

### ***1.3 Summary of Previous Work***

There has been a great deal of research in the past decade to improve the scalability of optimal-solution approaches to solving the VRP and its variations. In 1993, Fisher did one of the most successful exact approaches for the basic form of the VRP. He used the k-tree method to solve the problem for 71 customers [10]. There are also several noteworthy exact methods for the VRPTW. In 1991, Desrochers, Desrosiers, and Solomon were able to optimally solve problems containing 100 customers. They used a method of set partitioning in which the solution space is narrowed by a subprogram that generates feasible columns (routes) and only these columns are considered for set partitioning [7]. In 1997, Kohl et al. produced similar results with an approach that was able to solve the VRPTW with 100 customers [6]. It is also worthy to note, however, that the VRPTW has the potential of reducing the solution space by using small time windows for the customers that do not overlap; thus allowing for larger customer sets to be solved. Overall, the robustness and scalability of the optimal-solution approaches often falls short of real-world applications and data sets.

With the limitation of the optimal-solution approaches, the use of heuristics is justified when solving real-world vehicle routing problems. Several heuristic techniques have been used to solve the VRP and its variations. These techniques include genetic algorithms, ant colony algorithms, and tabu searches. Genetic algorithms were used in 1999 by Louis, Yin, and Yaun to solve the VRPTW. Their work emphasized the merge crossover technique to enhance the performance of genetic algorithms on clustered customer locations. This approach performed optimally or within 0.23% of optimal in their test benchmarks [1]. Ant colony research was also done in 1999 by Gambardella, Taillard, and Agazzi to solve the VRPTW using a Multiple Ant Colony System (MACS). The basic idea of Ant Colony Optimization is that a large number of simple, goal-oriented agents can produce good solutions to hard combinatorial optimization problems via low-level communications. MACS solves the VRPTW by using two ACO populations: one to minimize the travel time and one to minimize the number of vehicles. These two colonies then communicate via low-level communications using shared memory similar in idea to the pheromone trails left by ants. This approach produced results that were comparable to the genetic algorithm approach [6]. Another method that is recognized as one of the best heuristics for solving the VRP [6] is the tabu search approach. In 1995, Taillard, Laporte, and Gendreau expanded the work of Rochat and Taillard. The Rochat and Taillard algorithm using tabu search works in a similar manner to a genetic algorithm, allowing a population of VRP routes to be optimized over several generations. A tabu search uses penalties to avoid cycles in the search path. When a

move is made that will take the path to a point in the solution space that has already been visited, that move is penalized and possibly forbidden, therefore avoiding cycles [4].

## **1.4 Shortcomings**

Our research has shown that there are numerous heuristic as well as non-polynomial, optimal algorithms for solving the VRP. The shortcomings of the non-polynomial, optimal algorithms are fairly straightforward:

- They can only handle a limited number of customers.
- They run in non-polynomial time.
- They are often tailored to a specific data set. When the data set changes, which often occurs in a real-world situations, the algorithm can not handle the new data set and must be modified.

Heuristic algorithms lack many of these shortcomings, but of course do not guarantee an optimal solution. We noticed two significant areas of weakness during our research:

1. The ability of the algorithms to handle real-world data sets
2. The narrowness of the previous research.

While examining the past research done on the VRP, we discovered several relatively unexplored areas. The first area that we found during our research was that algorithms for the VRP were not applied to a data set that reflected real-world situations such as variations in time between customers based on the time of day. For example, businesses in metropolitan areas face the difficult task of routing vehicles between customers

throughout the city, but the time between customer A and customer B may be 20 minutes at 11am and 45 minutes at 4pm. Algorithms that do not take this variation into account could provide “optimal” solutions, but in the end be unfeasible unless they use the maximum time between the two customers for calculations within the algorithm.

Although this approach may indeed provide a feasible solution, it will not take advantage of the cost-savings gained by considering the lesser time of 20 minutes at 11am. At first this may seem trivial, but in reality, during the 25 minutes saved by using the lesser time, another customer might have been served by that vehicle. Over the period of a day, this could possibly save the use of several vehicles. This would amount to large cost savings for the company and provide a more “optimal” solution.

The second area of weakness has to deal with the narrowness of the scope of the previous research. Although many different types of algorithms were proposed and implemented in the previous research, the model does not include things such as precedence relationships and criticality of customers. Precedence relationships can occur in many real-world situations. The armored car routing problem presented early in this paper can be expanded to require precedence relationships. For example, it may be required that banks that have funds to be picked up by the armored cars be serviced prior to the banks that require funds to be dropped off so that the armored cars contain the correct amount of funds to be delivered.

## **1.5 Project Goals**

In this project we will address two of the shortcomings that were identified in the previous research. The first shortcoming that we will address deals with the lack of real-world data sets when calculating the times between customers. This new implementation will allow the genetic algorithm to calculate the actual time between customers at a given time of day, thus providing a final solution that is closer to a true optimal set of routes.

The second addition to the VRP model will be contributed in the area of precedence relationships. These relationships often exist in real-world data sets and are necessary in order to help the genetic algorithm to establish routes that are feasible and desired. These contributions to the model will be obtained in three main steps:

- 1. Define a new, innovative VRP model powerful enough and flexible enough to incorporate new features that allow the VRP model to handle real-world data
2. Implement a genetic algorithm to solve the new VRP model
3. Provide a graphical user interface for interaction with the genetic algorithm and a database to store the information entered by the user as well as the completed routes

The following more detailed steps will be completed in order to achieve our main objectives:



1. Research the VRP. Describe in detail what the VRP is, what research has been done in the past, and what direction the vehicle routing problem may take in the future.
2. Define a new, real-world based method for solving the VRP that allows for different travel times based on the time of day. This contributes both to the VRP model and to the database model for this problem. The solution generated by this approach will be closer to the “optimal” solution than earlier approaches by allowing the travel time to more accurately reflect the actual travel time of the trip at different times of day.
3. Expand the VRP model to provide the ability to establish and enforce precedence relationships when creating the routes.
4. Provide a solution to the newly defined VRP model using a genetic algorithm approach.
5. Implement the genetic algorithm solution using Java. This implementation will place an emphasis on performance as well as its ability to produce near optimal results.
6. Provide a user interface for interaction with the genetic algorithm. The user interface shall also allow the user to input data concerning the travel time for routes at specific times of day. The travel time data will then be stored in a database for use by the genetic algorithm.
7. Compare and contrast the genetic algorithm’s results from the new model where the travel time is dynamic based on the time of day to the previous models where

the travel time is the maximum time between two customers. This comparison will involve the number of vehicles used, the total time traveled by each vehicle, and the number of customers served by each vehicle. This testing can be done by reducing the problem from dynamic time to one set maximum route time by simply providing one time for each route in the distance matrix. The overall performance of the GA will also be measured on speed of resolution, the quality of the results, and the optimality of the routes.

8. Present the results in an informative manner that will reflect the genetic algorithm with dynamic travel time model used for solving the VRP used in this paper.

## 2.0 Definition

When the postal service, the bus system, or trucking industry plan for their every day tasks of delivery goods or providing basic transportation needs to people, they are attempting to solve the VRP. In this section, we will define the basic VRP, the basic concepts of the evolutionary approach, and our additions to the VRP model. These additions to the model include the use of dynamic travel time and precedence relationships.

### 2.1 Vehicle Routing Problem Definition

The basic version of the VRP can be formally defined as follows: Let  $G = (V, E)$  be an undirected graph where  $V = \{v_0, v_1, \dots, v_n\}$  is a set of vertices where  $v_0$  represents the main depot and  $V \setminus \{v_0\}$  represents the customers, and  $E = \{(v_i, v_j) \mid v_i, v_j \in V, i < j\}$  is an edge set representing the non-negative edges between the vertices. Identical vehicles,  $m$ , are located at the depot,  $v_0$ , each with a capacity of  $Q$ . Each customer,  $V \setminus \{v_0\}$ , has a non-negative demand,  $q_i$ , and non-negative service time,  $s_i$ . A distance matrix,  $c_{ij}$ , is defined on  $E$  [4]. The objective of the VRP is to minimize the total distance for each vehicle while visiting all of the customers. The following constraints are generally applied to the VRP: the capacity of an individual vehicle can not exceed  $Q$ , at most  $m$  vehicles can be used, routes (which may contain multiple customers) must start and end at the main depot, the total time must not exceed a preset limit, and each customer is visited only once.

## ***2.2 Evolutionary Algorithm Definition***

The genetic algorithm approach essentially emulates the same process of real-world natural evolution that was first explained by Charles Darwin. It establishes an initial population, then uses natural selection by means of a fitness function to determine the quality of the individuals in the population. Individuals are then chosen from the population using a selection function that takes into account the individuals fitness.

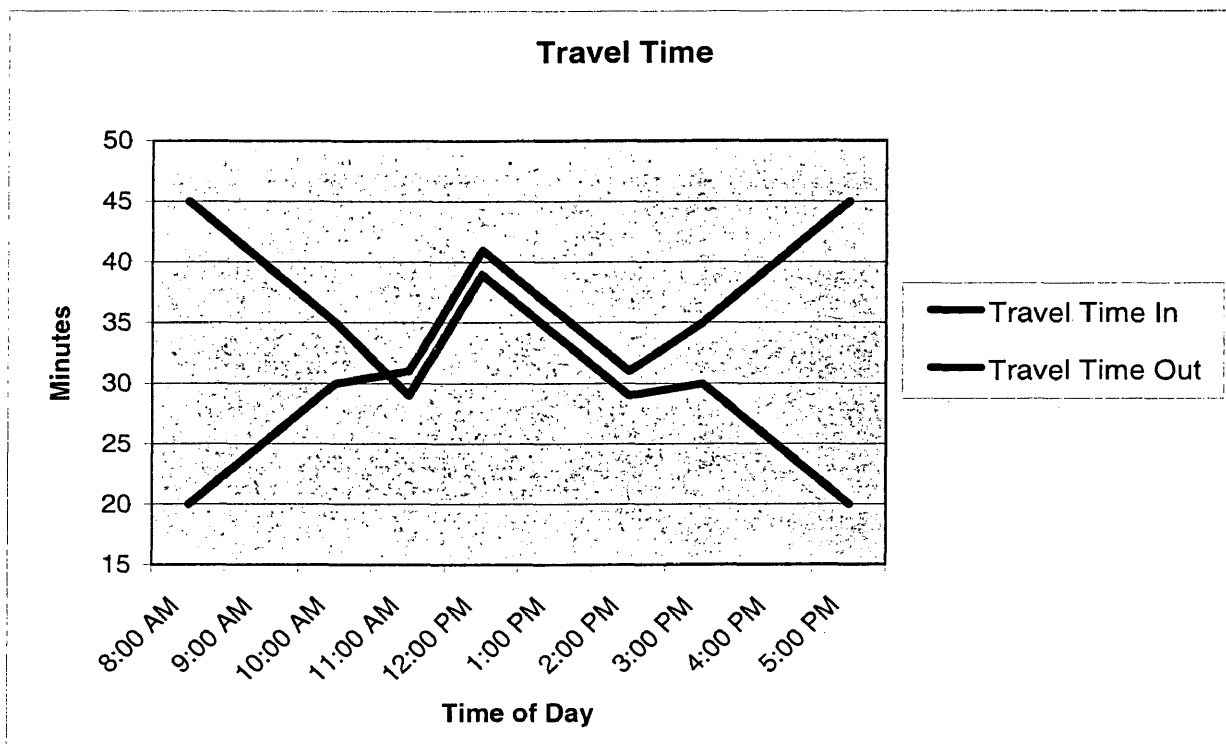
These individuals are then bred using a cross over function, then move on to the next generation. The other important part of the genetic algorithm is a mutation function that introduces new chromosomes into the population so that it does not become too homogeneous, thus narrowing the solution space too quickly. Our approach will be discussed in greater detail in section 4.3.1

## ***2.3 Proposed Additions to the Model***

To illustrate our first improvement to the VRP which adds realistic data sets by means of using dynamic travel time between vertices, one might consider a common problem: driving to work. At one time or another, anyone that works in a metropolitan area has experienced the problem that you are driving the same way as everyone else going to work, and when you are leaving work, you are once again traveling the same direction as the majority of the people. For example, going into the city at 8am may take 45 minutes; going into the city at 5pm may only take 20 minutes. Conversely, going out of the city at

8am may take 20 minutes; going out of the city at 5pm may take 45 minutes.

Graphically, the problem may look something like this:



**Figure 2-1: Travel Time to Work**

This new approach to solving the VRP with travel time dynamically based on the time of day will use this type of scenario to its advantage. It will be able to plan routes so that the minimal travel time is taken into account. In essence, the main problem is that the more congested a road, the longer it takes to travel on that road. This new model will favor traveling on roads when the traffic is minimal.

Furthermore, using travel time data based on the time of day has implications for the entire model: the algorithm, the user interface, and the database will all be affected. The algorithm might be faced with scheduling a segment that may fit into a route in the morning hours but not in the afternoon, as the travel time may have increased, therefore no longer allowing that segment to fit in. This solution will essentially change the structure of the distance matrix  $E$  used in the standard VRP. This will create a complicated scheduling problem which should be able to be handled by a genetic algorithm. The user interface and database are also affected by the addition of several new features that allow the entry and management of data related to time.

The addition of precedence relationships to the VRP model will allow customer priority to influence route creation. The precedence relationships will be enforced in a manner that allows each customer to be serviced by any vehicle as long as the precedence relationships hold. For example, if there is a precedence relationship between A and B ( $A \rightarrow B$ ) and customer A is serviced by vehicle 1 at 10:30 am, then customer B can be serviced by any vehicle after 10:30 am. This is applicable to many real-world companies that use a priority basis when servicing their customers.

## **3.0 Background and Previous Research**

Several aspects of the Vehicle Routing Problem (VRP) have been researched in the past. This section will provide information concerning the previous research and algorithms used to solve the VRP as well as the research's place in the overall map of the VRP.

### ***3.1 Previous Research***

#### **3.1.1 Overview**

The VRP has existed for some time, but in the past two decades more research has been given to this non-trivial problem. Bertsiman and Simchi-Levi [2] address the new developments in the VRP over the time span of the mid-nineteen-eighties to the mid-nineteen-nineties. The main objective of their work was to outline and define the developments during this time frame with an emphasis on the new insight gained and the new algorithms that have been developed.

One of the main areas of research was the robustness of the VRP heuristics. The algorithms often have to be tailored to the specific applications because these real-world situations often have cases that the algorithms have trouble solving. This leads to heuristic algorithms with ever-increasing complexities that are extremely sensitive to changes in the data set. This has led to research on what causes the complexity within the VRP, which in turn has led to more robust heuristics. The second area of study is the role

uncertainty plays in the VRP. The uncertainty often occurs in real-world situations concerning the customer's demands, location, or time schedule.

Bodin [8] summarizes his twenty years of personal experiences in the routing and scheduling field as well as the algorithmic, computational, graphical, and geographical areas. His experience provides a diverse set of examples such as scheduling street sweepers, barges, and hoist compactors for New York City, scheduling household refuse collection vehicles, school buses, planes and crews, and meter readers for public utilities. He also provides an excellent example of how computer based algorithms can save companies substantial amounts of money. In the solid-waste collection problems communities must decide on the number and location of disposal facilities and determine the best set of routes to use during collection. A study in Oyster Bay, New York showed the computerized schedule for the sanitation vehicles saved three vehicles out of 40 and \$750,000 per year. A Montreal Urban Community Transit system also saved 3% of the drivers' salaries over the previous manual system or about 2 million dollars per year. Bodin also highlights the history and major advances of the VRP from 1970 to 1990. Possibly the most dramatic improvement was the computation power of the machines that the algorithms run on. The computers went from batch algorithms operating on large computers with no graphical interfaces to visualize the routes to extremely fast microcomputers with interactive graphical displays. Geographic Information Systems (GIS) and widely available positional data have also advanced the VRP and helped move it from theoretical applications to practical, real-world applications.



### **3.1.2 Genetic Algorithm approach to VRP with Time Windows**

The VRP with Time Windows (VRPTW) has been researched and solved using genetic algorithms. The paper by Louis, Yin, and Yaun [1] expands on previous work that shows the merged-crossover works better than the traditional crossover, except when presented with non-random customer locations. They modified the merged crossover operation to perform better against clustered customer locations in order to solve the non-random customer location problem. The merged crossover operations are based on the idea that there exists a “global precedence,” regardless of the chromosome, of some alleles to occur before others in within the same chromosome. In the VRPTW problem this translates to each customer being represented by an allele with an associated time window; there is a natural precedence relationship among all customers based on their time window. The results of their modifications showed optimal solutions to three out of the six benchmark problems and was very close to the optimal solution in the other three cases.

### **3.1.3 Ant Colony approach to VRP with Time Windows**

Gambardella, Taillard, and Agazzi [6] approached the VRPTW using a Multiple Ant Colony System (MACS). The basic idea of Ant Colony Optimization (ACO) is that a large number of simple, goal-oriented agents can produce good solutions to hard combinatorial optimization problems via low-level communications. MACS solves the VRPTW by using two ACO populations: one to minimize the travel time and one to minimize the number of vehicles. These two colonies then communicate via low-level

communications using shared memory similar in idea to the pheromone trails left by ants. This approach produced results that were comparable to the genetic algorithm approach.

### **3.1.4 Set Covering Formulation for the VRPTW**

In 1991, Desrochers, Desrosiers, and Solomon [7] published a paper reporting their effort to design and implement a new algorithm capable of optimally solving larger VRPTW data sets. The algorithm uses column generation for set partitioning formulation and is capable of optimally solving problems up to six times larger than previous published research. Because the solution space is too large to allow set partitioning to be applied directly, a dynamic subprogram was used to generate feasible columns. Set partitioning was then applied only considering these feasible solutions.

In 1993, Bramel and Simchi-Levi [9] studied the effectiveness of the model used by Desrochers, Desrosiers, and Solomon [7] with the objective of analytically defining why the set-covering approach using column generation was so effective. They found that the gap between fractional and integer solutions to the set-covering problem tends to converge as the number of customers in the VRP increases; thus allowing for larger data sets to be solved.

### **3.1.5 Dynamic Programming approach to VRP with Split Pick-Ups**

The VRP with Split Pick-Ups is a variation of the traditional VRP that allows multiple vehicles to pick up parts from a supplier. This situation is applicable in industries where the payload constraints play a large factor in the shipments. It also can help maximize the payload of a truck by allowing it to pick up partial shipments of different suppliers that are on a similar route. In the paper by Chi-Guhn lee, Marina Epelman, Chelsea C. White III, and Yavuz A. Bozer [3], they formulate a dynamic program with infinite states and action space. This dynamic program is then reduced to a finite dynamic program for any given initial condition. A best-fit shortest path search is then used to solve the problem.

### **3.1.6 Minimum K-Trees optimal approach to the VRP**

Fisher [10] uses minimal K-trees to solve moderately sized VRP. Given a graph with  $n + 1$  edges, Fisher defines a K-tree to be a set of  $n + K$  edges that span the graph, where  $n$  is the number of customers and  $K$  is the number of vehicles. He then models the VRP as a minimal K-tree with degree  $2K$  at the depot and each customer node having the degree of 2. This approach is then used to solve a variety of benchmark and real-world data sets. These data sets are provided in the appendix and may provide a good benchmark for the research of this paper.

### 3.1.7 Tabu Search Heuristic

The paper by Taillard, Laporte, and Gendreau [4] expands the work of Rochat and Taillard. The Rochat and Taillard algorithm using tabu search works in a similar to a genetic algorithm, allowing a population of VRP routes to be optimized over several generations. A tabu search uses penalties to avoid cycles in the search path. When a move is made that will take the path to a point in the solution space that has already been visited, that move is penalized and possibly forbidden, therefore avoiding cycles. The Rochat and Taillard algorithm first produces several viable VRP routes using tabu search. These routes are then selected and combined with other routes to form the next generation and the process is repeated. Some of the routes are then selected from the final generation as candidates for the final VRP solution.

The Taillard, Laporte, and Gendreau algorithm consists of three parts. It first generates a large set of viable routes using the Rochat and Taillard algorithm. The second step is to select a subset of these routes based on the total distance and the frequency that they appear in the population. These routes are then placed in a search tree and all viable routes that can be formed by combining these selected routes are generated. The third step is the application of a bin packing heuristic to sort the routes into feasible working day solutions.

## 4.0 Specification

### 4.1 *Problem Statement*

In this project we will address two of the shortcomings that were identified in the previous research. The first shortcoming that we will address deals with the lack of real-world data sets when calculating the times between customers. The new implementation developed in this project will allow the genetic algorithm to calculate the actual time between customers at a given time of day, thus providing a final solution that is closer to a true optimal set of routes. The second addition to the VRP model will be contributed in the area of precedence relationships. These relationships often exist in real-world data sets and are necessary in order to help the genetic algorithm to establish routes that desired by the customers. These contributions to the model will be obtained in three main steps:

1. Define a new, innovative VRP model powerful enough and flexible enough to incorporate new features that allow the VRP model to handle real-world data
2. Implement a genetic algorithm to solve the new VRP model
3. Provide a graphical user interface for interaction with the genetic algorithm and a database to store the information entered by the user as well as the completed routes

## **4.2 Proposed Model**

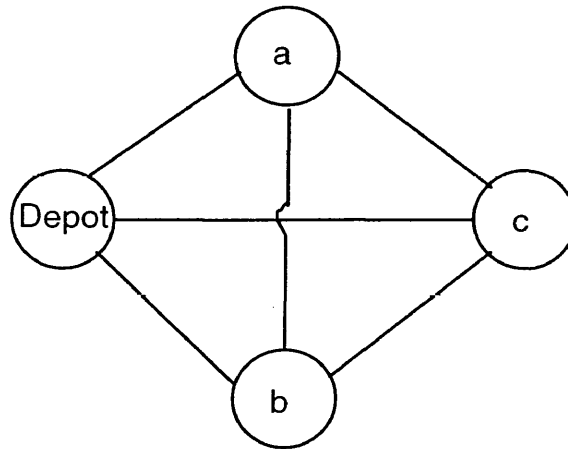
To illustrate the innovative improvements made to the VRP, it is first necessary to describe a basic VRP model example. This example will then be re-illustrated using the enhanced the precedence relationship model and the dynamic time VRP model.

### **4.2.1 Basic VRP Model**

The basic VRP model consists of an undirected graph  $G = (V, E)$  where  $V$  is a set of vertices  $\{v_0, v_1, \dots, v_n\}$  and  $E$  is an edge set representing the non-negative edges between the vertices. The depot is represented by  $v_0$ , and  $V \setminus \{v_0\}$  represents the customers.

Identical vehicles  $m$  are initially located at the depot  $v_0$ . A distance matrix,  $c_{ij}$ , is defined on  $E$ . The objective of the VRP is to minimize the total distance for each vehicle while visiting all of the customers.

For this example, we will assume that the number of vehicles  $m$  equals two, and the set of vertices,  $V$ , contains  $\{\text{depot}, a, b, c\}$ .



**Figure 4-1: Undirected Graph of Set V**

	Depot	a	b	c
Depot		5	10	20
a	5		8	12
b	10	8		30
c	20	12	30	

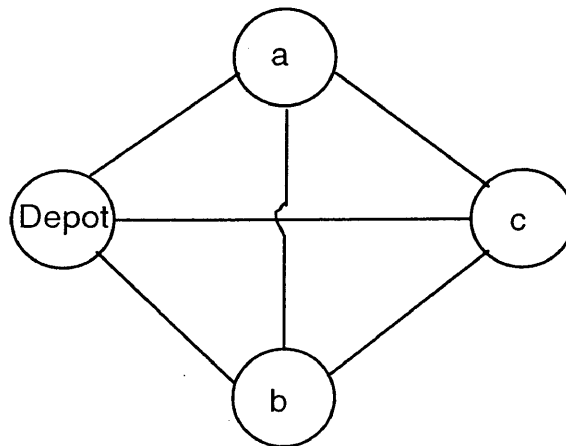
**Figure 4-2: Distance Matrix  $c$  Measured in Minutes**

This would allow the first vehicle to travel a route of {depot, b, depot} and vehicle 2 to travel a route of {depot, a, c, depot}. Vehicle 1 would finish in 20 minutes and vehicle 2 would finish in 37 minutes, thus minimizing the total time to visit each location with two vehicles while not repeating any nodes.

### 4.2.2 Precedence Relationship VRP Model

Precedence relationships can provide a useful means of ensuring that one customer is served before another customer. The main expansions to the model will occur in the fitness and crossover functions. We will go in to more detail on these expansions in section 4.3 detailing our approach. Of course, the major constraint for precedence relationships is that the precedence relationship graph must not contain any cycles.

A simple example of the precedence relationships effect on the model is shown by expanding on the basic example. We will still assume that the number of vehicles  $m$  equals two and set of vertices,  $V$ , contains {depot, a, b, c}. However, in this example there is an overall precedence relationship between vertex b and vertex a where vertex b must precede vertex a.



**Figure 4-3: Undirected Graph of Set V**



	Depot	a	b	c
Depot		5	10	20
a	5		8	12
b	10	8		30
c	20	12	30	

**Figure 4-4: Distance Matrix c Measured in Minutes**

This would allow the first vehicle ( $m_1$ ) to travel a route of {depot, b, a, depot} and vehicle 2 ( $m_2$ ) to travel a route of {depot, c, depot}. Vehicle 1 would finish in 23 minutes and vehicle 2 would finish in 40 minutes

There are several methods of defining the precedence relationships between the customers. One method might be to force the customers associated in the precedence relationship to be serviced by the same vehicle. With this type of constraint, you could ensure that customer B is always served before customer A. In real-world situations, one of the main uses for precedence relationships may stem from the need to transfer goods between two customers. For example, in the armored car example, customer B may contain the payroll for customer A, therefore the same vehicle must service customer B first to pick up the payroll, then service customer A to drop the payroll off. Another method of enforcing the precedence relationship is to start vehicle 1 at 8:00am and send it to customer B where in this example it would arrive at 8:10am. Vehicle 2 could then be delayed and depart for customer A at 8:06am where it would arrive at 8:11am, thus

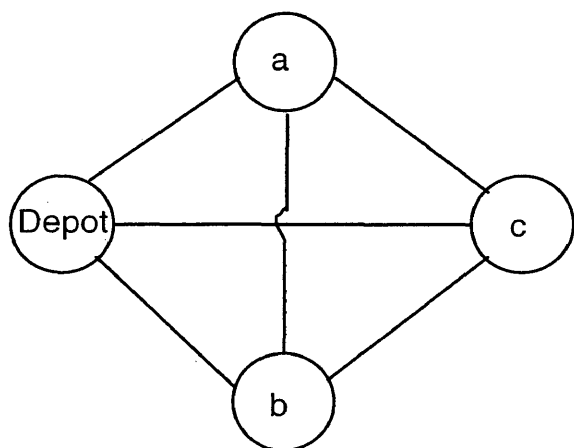
enforcing the precedence relationship, but using two vehicles. A third method is to enforce the precedence relationships in a manner that allows each customer to be serviced by any vehicle as long as the precedence relationships hold. For example, if there is a precedence relationship between A and B ( $A \rightarrow B$ ) and customer A is serviced by vehicle 1 at 10:30 am, then customer B can be serviced by any vehicle after 10:30 am. This is applicable to many real-world companies that use a priority basis when servicing their customers.

For our model, we believe that the third method is the preferred choice. This allows the genetic algorithm (GA) to work in a broader solution space that does not have many of its vehicle's routes predefined by these precedence relationships. For example, if you were to input the precedence relationships of  $A \rightarrow B$ ,  $B \rightarrow C$ ,  $C \rightarrow D$ ,  $D \rightarrow E$  in the first situation, one vehicle must have a route of (A, B, C, D, E). However, using the third approach would allow the customers to be serviced any number of ways, as long as the precedence relationships hold, thus expanding the GA's solution space.

### **4.2.3 Dynamic Travel Time VRP Model**

The dynamic travel time model will allow for a more optimal solution to the overall vehicle routes by allowing the genetic algorithm to measure the time each route takes using actual travel times for the applicable time-of-day the route will be completed.

The major difference between the basic VRP and the Dynamic Travel Time VRP is the distance matrix  $c_{ij}$  is expanded to contain the time. The new distance matrix, which can be denoted  $c_{ij}^t$  where  $t$  represents the desired time the route will take place, is shown in Figure 4-6.



**Figure 4-5: Undirected Graph of Set V**

	Time	Depo t	a	b	c
<b>Depo</b> <b>t</b>	8:00a m		5	1 0	2 0
<b>a</b>	8:00a m	5		8	1 2
<b>b</b>	8:00a m	10	8		3 0
<b>c</b>	8:00a m	20	1 2	3 0	
<b>Depo</b> <b>t</b>	8:10a m		1 1	2 5	1 0
<b>a</b>	8:10a m	11		1 2	5
<b>b</b>	8:10a m	25	1 2		1 5
<b>c</b>	8:10a m	10	5	1 5	
<b>Depo</b> <b>t</b>	8:20a m		1 5	3 5	7

<b>a</b>	8:20a m	15		1 6	5
<b>b</b>	8:20a m	35	1 6		1 5
<b>c</b>	8:20a	7	5	1	

	m			5	
--	---	--	--	---	--

**Figure 4-6: Distance Matrix  $c_{ij}$  in Minutes**

Using the new distance matrix with the includes the desired time of travel, one possible result of the VRP using 2 vehicles would look as follows:

The first vehicle could travel a route of {depot, c, depot} and vehicle 2 could travel a route of {depot, a, b, depot}. Vehicle 1 would finish in 27 minutes and vehicle 2 would finish in 38 minutes, thus minimizing the total time to visit each location with 2 vehicles while not repeating any nodes.

One thing to note is that in these examples, the dynamic travel time approach did not actually improve upon the result of the basic VRP model. But, as discussed in section 1.4, in order to create the distance matrix ( $c_{ij}$ ) for the basic VRP from the distance matrix with time of travel ( $c_{ij}$ ), one must take the maximum distance from  $i$  to  $j$  regardless of the time of travel  $t$ . If this was not the case, then the distances/times used to calculate the basic VRP may be too short and it would create routes that are infeasible. Using the maximum distance regardless of time ensures that the basic VRP will be forced to use the travel time for the busiest time of day, thus creating a conservative VRP solution. It is our contention that this solution is drastically over-

constrictive, and that is why the dynamic vehicle travel time VRP model will create a better and still feasible solution.

### **4.3 Proposed Approach**

In this section, we will introduce the genetic algorithm approach we have selected to solve the VRP with precedence relationships and dynamic travel time. The essential parts of the genetic algorithm approach, such as the make up of the chromosome and the fitness function, will also be defined.

#### **4.3.1 Genetic Algorithm**

A genetic algorithm was chosen to solve this problem because of the computationally intense nature of this problem. The section defines the approach used for the genetic algorithm.

##### **4.3.1.1 VRP Basic Genetic Algorithm Structure**

The algorithm for the VRP GA will follow the basic structure used for most genetic algorithms. The majority of the enhancements to the model will occur when calculating the fitness of the individuals.

Inputs to the VRP GA will consist of:

Read inputs

- Graph  $G$  that defines the set of vertices  $V$

consisting of the customer locations as well as the main depot

- 3D distance matrix  $e_{ij}$  that consists of the times between vertices over the time of day
- The size of the initial population  $P$
- The maximum number of vehicles
- The amount of time for all routes to be competed (e.g. 8am to 5pm workday)

generation=0

Set initial generation number to zero

Initialize\_Population  $P(\text{generation})$  Initialize a random population of individuals

Evaluate  $P(\text{generation})$

Evaluate fitness of all initial individuals of population

While (not Terminate) do

Continue evolving generations until the termination condition

    generation = generation + 1

    Increment the generation number

    Select  $P(\text{generation})$

    Select a subset of the current population to be used to create the next generation. This will

    From  $P(\text{generation} - 1)$

    typically be the fittest members of the current population, although some less fit members will be included in order to keep a more diverse gene

pool.

Crossover P (generation) Crossover the genes of selected parents to form the next generation

Mutate P (generation) Randomly mutate a small percent of the population

Fitness P (generation) Evaluate its new fitness

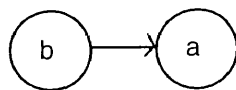
End While

### 4.3.1.2 The Distance Matrix

The distance matrix will be expanded from the original VRP to include the time the route will begin to be traveled. When accessing the new distance matrix you will need to specify not only the origin vertex and the destination vertex, but also the time in which the route will begin to be traveled. If no time exists for the specified time of day, the next earliest time of day will be used as the travel time. The ultimate goal will be to provide a means for the end user to record the actual time traveled between the two vertices after each route is executed. This will enable the model to use more accurate times in the future, thus increasing the performance of the model as the end user uses it.

### 4.3.1.3 The Precedence Relationship

The precedence relationships will be defined by a precedence graph (potentially disconnected) that will in essence overlay the main graph used by the VRP. It will be predefined by the user and used by the fitness function when evaluating individuals of the population. It is important to note that not all customers may require a precedence relationship with another customer. For example, the precedence graph from the example in Section 4.2.2 would look like this:



**Figure 4-7:Precedence Relationship**



## The Chromosome

The chromosome for this implementation of the VRP contains the order in which the customers were visited as well as which vehicle was used.

Using the example from Section 4.2.1, it can be represented as:

	1	2	3
Vehicle	1	2	2
Customer Visited	b	a	c

**Figure 4-8: The Chromosome**

The Vehicle row may contain the values 1 through  $m$  where  $m$  is the maximum number of vehicles available, and the Customer Visited row contains the values 1 through  $|\mathcal{V} \setminus \{v_0\}|$ .

It is important to note that because each vehicle must begin and end its route at the depot, the depot itself does not need to be encoded in the chromosome. However, the distance from the depot to the first customer, as well as the distance from the last customer to the depot, will be included when calculating the overall time of the route.

### 4.3.1.4 The Fitness Function

The fitness function evaluates the quality of each individual in the population. The fitness value that it assigns to each individual decreases as the individual improves in

quality. It will evaluate the individuals in the populations based on the following attributes:

- The total time for all vehicles to visit all of the customers and return to the depot
  - This is calculated based on the dynamic travel time distance matrix based on the current time of day along the route.
  - The total time includes the “Time Per Stop” provided by the user.
  - The dynamic distance calculation uses the travel time for the vehicle for the current time of day along the route. If no travel time exists for the given time of day, the next earliest time of day is used.
- Must service all customers
  - Individuals are penalized for not servicing all customers.
  - The algorithm simply penalizes the individual by increasing its fitness 10 times.
- The precedence relationships are enforced
  - Individuals are penalized if the relationships are not present in their solution.
  - Like the customer rule, the algorithm simply penalizes the individual by increasing its fitness 10 times. This forces the algorithm to take the precedence relationships into account regardless of how much extra it costs the fitness value in terms of actual distance.
- The number of vehicles used

- Individuals are rewarded for using fewer vehicles.
- This is done by using a ratio that approaches 1 as fewer vehicles are used.
- The algorithm, however, still has to take into account the maximum workday, so it is almost always impossible for one vehicle to service all of the customers in larger examples. But the GA will try to service as many customers as it can with each vehicle.
- The “maximum number of vehicles” is supplied to the user, but this does not mean that the algorithm must use all of the vehicles. The fitness of any individual will improve with fewer vehicles used.
- The following equation is used to determine the penalty applied to the fitness function:

$$\left( \text{numVehiclesUsed}^2 \left( \frac{N^2}{\sum_{i=0}^m (\text{vehUsed}[i])^2} \right) \right)$$

This equation was selected in order to reward the individual for servicing as many customers as possible with each vehicle. For example, if individual A used 3 vehicles to service 25 customers (15 with vehicle 1, 7 with vehicle 2, and 4 with vehicle 3) the equation would look like this:

$$\left( 3^2 \left( \frac{25^2}{15^2 + 7^2 + 3^2} \right) \right) = \left( 9 \left( \frac{625}{225 + 49 + 9} \right) \right) = \left( 9 \left( \frac{625}{283} \right) \right) \cong 19.9.$$

However, if individual B used 3 vehicles to service 25 customers (16 with vehicle 1, 8 with vehicle 2, and 2 with vehicle 3) the equation would look like this:

$$\left(3^2 \left( \frac{25^2}{16^2 + 8^2 + 2^2} \right) \right) = \left(9 \left( \frac{625}{256 + 64 + 4} \right) \right) = \left(9 \left( \frac{625}{324} \right) \right) \cong 17.4 .$$

However, if individual C used 2 vehicles to service 25 customers (15 with vehicle 1 and 10 with vehicle 2) the equation would look like this:

$$\left(2^2 \left( \frac{25^2}{15^2 + 10^2} \right) \right) = \left(4 \left( \frac{625}{225 + 100} \right) \right) = \left(4 \left( \frac{625}{325} \right) \right) \cong 7.7 .$$

However, if individual D used 1 vehicle to service all 25 customers (25 with vehicle 1) the equation would look like this:

$$\left(1^2 \left( \frac{25^2}{25^2} \right) \right) = \left(1 \left( \frac{625}{625} \right) \right) = 1 .$$

As you can see, the fitness penalty is reduced to 1 as the each vehicle service more customers. The difference between individual A and individual B was that individual B serviced more customers with two of the vehicles and less with the third, thus bringing it closer to the goal of reducing the total number of vehicles used. The effect of reducing the number of vehicles by one, such as the change between individual B and individual C, was quite dramatic in that the penalty was reduced by more than half. This reward strongly encourages the

individuals to use fewer vehicles. Keep in mind though that the other parts of the fitness function may increase their penalty if their objectives are not met. For example, if the vehicle finishes past the maximum workday value or any of the precedence relationships are violated it will be penalized.

- The time the last vehicle returns to the depot
  - Individuals are penalized if their routes exceed the maximum work time.
  - This is done using a ratio that penalizes the individual the more they exceed the maximum workday.
  - For example, an individual whose last vehicle returned to the depot 10 minutes late would not be penalized as much as one that returns 90 minutes late.

The actual fitness function looks something like this:

$$\begin{aligned}
 & \textit{fitness} = (\textit{totalDistAllVehicles}) * \\
 & (1 + \textit{numMissedCustomers}) * \\
 & (1 + \textit{numPrecedenceViolations}) * \\
 & (\textit{numVehiclesUsed}^2 \left( \frac{N^2}{\sum_{i=0}^m (\textit{vehUsed}[i])^2} \right) ) * \\
 & (1 + \textit{numMinutesOverWorkDay} / 60)
 \end{aligned}$$

#### **4.4 Greedy Algorithm Solution: An Alternative Approach**

In order to validate that the genetic algorithm method was producing high quality valid results, two versions of greedy algorithms were written for comparison purposes: a relatively simple greedy algorithm and a more complex greedy algorithm.

The simple greedy algorithm allocates each vehicle to the closest customer, and then from the customers left, allocates each vehicle to the next closest unvisited customer. This continues until no customers are left. At that point, the vehicles return to the main depot. This ensures that it is a valid solution. This greedy algorithm solution is a fairly simple one in the fact that it uses the maximum number of vehicles allowed and does not try to reduce that number like the GA does. However, in order to present a fair comparison between the GA and the simple greedy algorithms, the maximum number of vehicles must be set to an optimal value so they will both use the same number of vehicles. For example, if the initial problem consist of 50 customers, and it is shown through the GA solution that it is possible to service all of the customers in the allotted time (work day) with two vehicles, then the maximum number of vehicles will be set to two. If this is not done, the simple greedy algorithm will use all of the vehicles that it can. The other drawback of the simple greedy algorithm is that it does not use dynamic time as the GA does. This too can be overcome by using a distance matrix with only one time between nodes, essentially reducing the problem to the basic VRP. When this is done, both the GA and the

simple greedy algorithm will use the same time between the node regardless of the time of day.

In order to overcome the limitations of the simple greedy algorithm, a more complex greedy algorithm was written. This algorithm routes one vehicle at a time until the vehicle can service no more customers in the allotted workday. Then that vehicle returns to the depot, and the next vehicle is routed. It uses the same general principle as the simple greedy algorithm to pick the next customer to service, however it uses dynamic time to determine the closest customer based on the time of day. This version of the greedy algorithm is easier to compare to the GA than the simple version. By planning the vehicles one at a time, it can minimize the number of vehicles used and also attempt to minimize the distance traveled by each vehicle. The more complex greedy algorithm also takes into account the precedence relationships. This is more difficult for the greedy algorithm than it is for the GA because the GA can see the entire route that is traveled by the individual, and then if precedence relationships are violated, it is penalized in its fitness value. The greedy algorithm, however, only knows the route as it progresses and must pick the best customer at that point in time. This lack of knowledge makes resolving the precedence relationships more difficult. For example, if there was a precedence violation between customer A and D ( $A \rightarrow D$ ) and customers D and B ( $D \rightarrow B$ ), then as the greedy algorithm creates the route, it may find that A is the closest customer and customer D and customer B have not yet been planned, so it adds A to the route. Then

B may be the next closest customer, but it can not be added because of the precedence relationship  $D \rightarrow B$ , so the next closest customer is D, and it is added to the route. At this point, the vehicle does not have enough time to service any more customers, so it returns to the depot. Then the next vehicle is started and it finds that B is the closest customer; the problem is that the second vehicle is started at the beginning of the day (because this is how the route will be executed by the vehicles). So when customer B is serviced, it ends up being before D, thus violating the precedence relationship  $D \rightarrow B$ . With that said, the greedy algorithm's precedence relationships are valid for each vehicle, just not between the vehicles.

With the exception of the precedence relationships, the more complex greedy algorithm compares quite favorably to the GA when random data is generated for the distance matrix. This is due to the fact that normal traffic patterns are not taken into account when generating the random data. By using the fitness function, the GA is able to see the whole route, and avoid situations where going a short distance at the beginning of the route means that you must take a very long distance later in the route. This is often the case of traffic patterns in cities as illustrated in section 1.4. Example of these comparisons will be presented in section 5.1.3. When using more complex, realistic travel time data, the GA tends to out-perform the greedy algorithm.



## **4.5 Random Distance Matrix Generation**

A random distance matrix generator was written to assist in the testing of the algorithm. This was necessary in order to test a wide range of data with a large number of customers. The three-dimensional distance matrix grows rapidly, and it can be quite time consuming to input a large number of nodes as well as the time between these nodes over the course of the day. For example, 100 customers with travel times for five different times of day would require 50,000 entries.

In order to simulate realistic data for testing purposes with both the greedy algorithm and the GA, it was necessary to build a certain amount of intelligence into the random distance matrix generator. The generator takes as inputs the number of customers (N), the length of the work day, a seed to start the random generator so the results can be reproduced, the number of times you wish to allocate throughout the day, and the dimensions of a graph to map the customers and depot onto. Next, it uses these inputs to create coordinate points on a graph of the customers and depot. It then calculates the Pythagorean distance between the points and uses this for a starting point to generate the times between the customers. It then randomly selects a method for each customer to use when defining the distance between the customers throughout the day. These methods are as follows where  $c$  is the original distance between two points:

- Time is constant over the day
  - Each allocated time between those two customers is set to  $c$ .

- Time increases over the day
  - Time starts at  $c$ , then each subsequent time increases by 10%.
- Time decreases over the day
  - Time starts at  $c$ , then each subsequent time decreases by 10%.
- Time starts high, decreases, then increases
  - Time is set to  $c + c * (\sin ([currentTime]/(workday/\pi)))$ .
  - This uses a sine function with a value that changes from 1 to 0 then back to 1.
  - When this is added to the  $c$ , it gives the effect of the time between the two nodes decreasing and then increasing as the day goes on.
- Time starts low, increases, then decreases
  - Time is set to  $c + c * (1 + (\pi/2) + \cos ([currentTime]/(workday/\pi)))$ .
  - This uses a cosine function with a value that changes from 0 to 1 then back to 0.
  - When this is added to  $c$ , it gives the effect of the time between the two nodes increasing and then decreasing as the day goes on.

These times are then placed in the three-dimensional distance matrix and used by both the GA and the greedy algorithm.

## ***4.6 The Graphical User Interface***

The graphical user interface (GUI) for the VRP project allows the user to input the desired settings to be used by the genetic algorithm as well as watch the VRP genetic algorithms progress as it is running. The interface provides graphs of the current route, the fitness for each generation, and the total distance traveled by the vehicles. It also displays the current generation and information that is used by the fitness function to decide the quality of the results. This includes the total number of vehicles used, the number of precedence violations, the time the last vehicle returns to the depot, and the number of customers not served. The GUI also has a feature that allows the user to stop the GA at the current generation. This feature is useful when the GA results as viewed by the fitness function graph are not making improvements from one generation to the next.

The GUI works independently from the GA. The GUI first writes the inputs to the database and then starts the GA. The GA writes its output to the database after some set number of generations. The output from the GA includes the best route so far and the performance data of the individual using that route. The GA also looks for termination information that is written to the database when the “Stop” button is pushed on the GUI. The GUI then reads the GA output from the database and graphs some of the information, including the current best route, the fitness over all generations, and the total distance traveled over all generations. The GUI also displays some other information from the current best individual including the

number of vehicles used, the number of precedence violations, the number of customers not served, and the time that the last vehicle returns to the depot.

The basic architecture looks as follows:

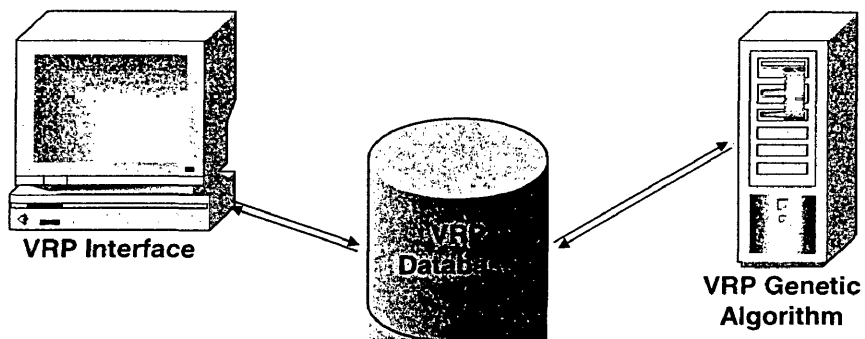


Figure 4-9:VRP Architecture

Here is an example of the GUI for the VRP:

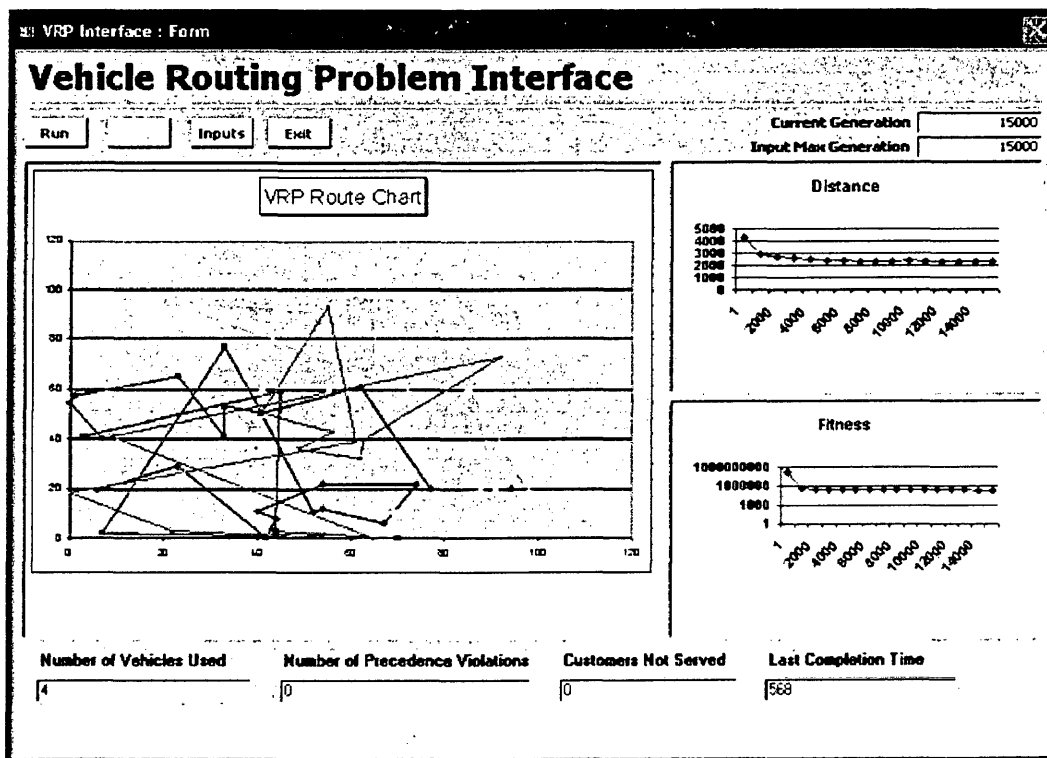
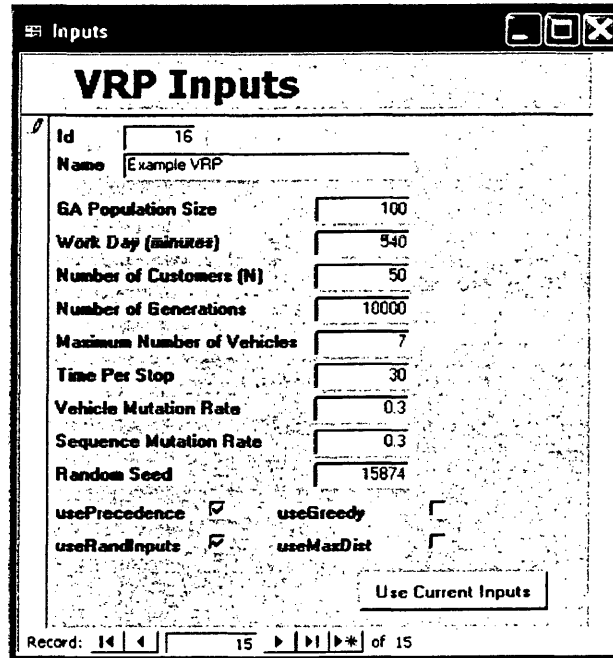


Figure 4-10: VRP Graphical User Interface

The Interface for the VRP inputs look as follows:



The screenshot shows a window titled "Inputs" with a sub-header "VRP Inputs". The interface contains several input fields and checkboxes. The "Id" field is set to 16, and the "Name" field is set to "Example VRP". The "GA Population Size" is 100, "Work Day (minutes)" is 540, "Number of Customers (N)" is 50, "Number of Generations" is 10000, "Maximum Number of Vehicles" is 7, "Time Per Stop" is 30, "Vehicle Mutation Rate" is 0.3, and "Sequence Mutation Rate" is 0.3. The "Random Seed" is 15874. There are four checkboxes: "usePrecedence" (checked), "useGreedy" (unchecked), "useRandInputs" (checked), and "useMaxDist" (unchecked). A "Use Current Inputs" button is located at the bottom right. At the bottom of the window, there is a record navigation bar showing "Record: 14 of 15" with navigation icons.

Parameter	Value
Id	16
Name	Example VRP
GA Population Size	100
Work Day (minutes)	540
Number of Customers (N)	50
Number of Generations	10000
Maximum Number of Vehicles	7
Time Per Stop	30
Vehicle Mutation Rate	0.3
Sequence Mutation Rate	0.3
Random Seed	15874

usePrecedence     useGreedy   
useRandInputs     useMaxDist

Use Current Inputs

Record: 14 of 15

Figure 4-11: VRP Input Interface

## 4.7 The Database

The database used for this project contains tables that hold the information both for the inputs to the VRP as well as the outputs of the VRP. The database acts as the central location for communication of inputs and outputs from both the GA and the GUI. The tables contained in the VRP database are as follows:

Field Name	Data Type	Description
generation	Long Integer	The route generation
customer	Integer	The customer served
x	Integer	The x coordinate of the customer served
y	Integer	The y coordinate of the customer served
vehicle	Integer	The vehicle used to served the customer
time	Integer	The time of day the customer was served

**Figure 4-12: VRP Route Output Table**

Field Name	Data Type	Description
generation	Long Integer	The generation of the performance information
fitness	Integer	The fitness of the best individual
distance	Integer	The total distance traveled of the best individual
numVehUsed	Integer	The total distance traveled of the best individual
numPrecViolations	Integer	The number of precedence violations committed by the best individual
customersNotServed	Integer	The number of customers not served
lastCompletionTime	Integer	The time the last vehicle returns to the depot

**Figure 4-13: VRP Performance Data Output Table**

Field Name	Data Type	Description
inputId	Long Integer	The unique identifier of the input set
inputSetName	String	The name of the input set
numGenerations	Integer	The number of desired generations
timePerStop	Integer	The time in minutes that each vehicle spends at each customer they visit in minutes
vehMutationRate	Double	The percent of individuals in the population that will have their vehicle mutated
seqMutationRate	Double	The percent of individuals in the population that will have their sequence mutated
workDay	Integer	The time that is allotted for all customers to be served in minutes
N	Integer	The number of customers to be served
randSeed	Long Integer	The seed that will be used for the random number generator
usePrecedence	Boolean	A flag indicating if the precedence relationships should be used
useRandInputs	Boolean	A flag indicating if random inputs should be used
useGreedy	Boolean	A flag indicating if the greedy algorithm should be used
useMaxDist	Boolean	A flag indicating if the traditional VRP using the maximum travel time should be used when computing the distance between the customers
currentInputs	Boolean	A flag indicating that this is the current inputs with which to run the VRP GA

**Figure 4-14: VRP Inputs Table**

Field Name	Data Type	Description
i	Integer	The vertex from which the route begins
j	Integer	The vertex from which the route ends
time_of_day	Integer	The time the route will be traveled
travel_time	Integer	The time between the two vertices

**Figure 4-15: VRP Distance Matrix Table**

<b>Field Name</b>	<b>Data Type</b>	<b>Description</b>
primary	Integer	The customer that must be served before the secondary customer
secondary	Integer	The customer that must be served after the primary customer

**Figure 4-16: VRP Precedence Matrix Table**

<b>Field Name</b>	<b>Data Type</b>	<b>Description</b>
generation	Long Integer	The generation that the GA will stop. This is initially set to the number of generations that is inputted, but is updated when the user presses the “Stop” button on the GUI

**Figure 4-17: VRP GA Termination Table**



## **5.0 Obtained Results**

### ***5.1 Example Test Results***

#### **5.1.1 Basic City Routing Example**

This example illustrates how the software operates when defining the vehicle routes in a city. The city environment presents dynamic traffic conditions where the travel time from one customer to another may vary throughout the day. The “dynamic distance” used by this software will allow for a more optimal solution in this environment. The precedence relationships will also be used to establish a relative importance of certain customers.

In this example, there are 13 customers to be serviced from one central depot. There are a maximum of 5 vehicles that can be used to service these customers, but the software places an emphasis on using only as many vehicles as necessary to complete the routes in the specified working hours (in this case 8am-2pm). Each vehicle will leave the depot at 8am, visit the designated customers, and return to the depot.

#### **Software Inputs**

The inputs that are required for this software are described in the figure below. This information is read from the database using a JDBC connection. The following values were used for this example:

Software Inputs	Value
n	13
length of workday (min)	360 (8am to 2pm)
time per stop (min)	20
max number of vehicles	5
p	2 → 5; 6 → 9; 11 → 10
c (see below)	

Figure 5-1: Basic City Example Software Inputs

The distance matrix e represents the time between the nodes throughout the day:

		8:00 AM	8:30 AM	9:00 AM	9:30 AM	10:00 AM	3	1	23	21	19	17	15	8	7	11	31	51	71	91
from	to						3	1	23	21	19	17	15	8	7	11	31	51	71	91
depot	1	15	17	19	21	23	3	2	7	8	9	10	11	9depot	20	22	24	26	28	
depot	4	23	21	19	17	15	3	4	5	6	7	8	9	9	6	37	35	33	31	29
depot	5	30	32	34	36	38	4	5	9	8	7	6	5	9	7	52	56	58	56	52
depot	6	12	16	20	24	28	4	2	3	13	23	13	3	10depot	55	60	65	60	55	
depot	8	48	45	42	39	36	4	3	5	10	15	20	25	10	11	3	9	15	9	3
depot	9	20	22	24	26	28	4	5	9	8	7	6	5	10	12	40	35	30	25	20
depot	10	55	50	45	40	35	4	1	2	4	6	8	10	10	13	12	13	14	15	16
depot	11	42	46	50	54	58	5	2	15	19	40	12	32	11depot	42	43	44	45	46	
1	depot	15	14	13	12	11	5	4	9	11	13	15	17	11	10	3	7	11	7	3
1	3	23	25	27	29	31	5	2	15	19	40	12	32	11	12	31	37	43	49	55
1	5	2	3	4	5	6	5	4	9	11	13	15	17	11	12	31	37	43	49	55
2	3	7	8	9	10	11	6	7	42	50	58	50	42	11	13	7	8	9	8	7
2	4	3	5	7	9	11	6	9	37	35	33	31	29	11	13	7	8	9	8	7
2	5	11	9	7	5	3	7	6	42	46	50	54	58	11	13	7	8	9	8	7
							7	8	11	10	9	8	7	12	10	40	37	34	31	28
							7	9	52	48	44	48	52	12	11	31	41	21	11	31
							8depot	51	41	31	21	11		12	13	19	22	25	28	31
														13	10	12	10	8	6	4
														13	11	7	9	11	13	15
														13	12	19	29	39	29	19

Figure 5-2: Basic City Example Dynamic Distance Matrix e

### The Genetic Algorithm Inputs

The genetic algorithm also uses several inputs. These consist of the size of the genetic algorithm population, the mutation rate for the vehicle sequences, and the mutation rate for the vehicle used to service the customers.

GA Inputs	Value
Population Size	100
Maximum number of generations	500
Sequence Mutation Rate	30%
Vehicle Used Mutation Rate	30%

**Figure 5-3: Basic City Example Genetic Algorithm inputs**

### Results

\*\*\*\*\* Gen 349 \*\*\*\*\*

cust	veh	time
6	0	8:32
9	0	9:27
7	0	10:45
8	0	11:12
4	1	8:43
3	1	9:13
2	1	9:42
5	1	10:07
1	1	10:37
11	4	9:02
13	4	9:31
12	4	10:20
10	4	11:08

Total distance traveled by all vehicles = 644

Total customers not served = 0

Total number of precedence violations = 0

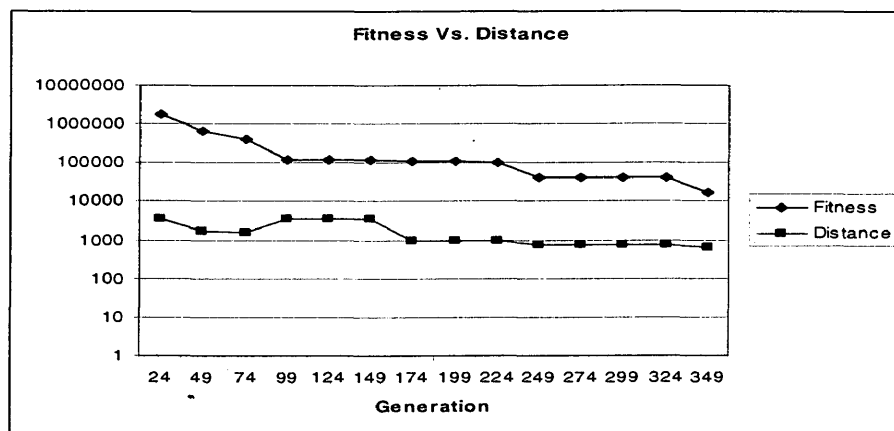
Total number of vehicles used = 3

Last vehicle arrives back at depot at 12:03

Fitness 17184.631578947367

### Analysis of Basic City Example

This example shows how the GA uses dynamic time to find the shortest route between all of the customers. It also shows the effects of the precedence relationships. The algorithm found a route that enforced all of the precedence relationships ( $2 \rightarrow 5$ ;  $6 \rightarrow 9$ ;  $11 \rightarrow 10$ ). The following chart shows the total distance traveled as it relates to the fitness value of the GA and the current generation of the GA.



**Figure 5-4: Fitness Function Vs. Total Distance Traveled.**

### 5.1.2 Random Example Case Study

In order to test the software on larger examples that have a wide range of inputs, the random distance matrix generator described in section 4.3.3 was used. The following sections illustrate the behavior of the GA, the Greedy algorithm, and the GA using the maximum distance instead of the dynamic distance. Using the maximum distance provides a good comparison with what algorithms that do not take advantage of dynamic distance would have to use for their times in order to ensure a feasible solution.

In this example, the distance matrix is generated with ten time entries for the distance between customers for the hours of 8:00am to 5:00pm. Precedence relationships are also created between some of the customers.

#### Software Inputs

The following values were used for all of the random examples in this section:

Software Inputs	Value
n	50
Length of workday (min)	540 (8am to 5pm)
Time per stop (min)	10
Max number of vehicles	5
p	6-->3 3-->26 36-->40 47-->11 27-->32; 27-->9 50-->11 7-->35 11-->31 22-->20

	13-->14 5-->4
e	Ten time entries between each customer. The matrix itself is too large to display.

**Figure 5-5: Random Example Software Inputs**

### 5.1.2.1 Small Population

#### Genetic Algorithm Inputs

The following inputs were used for the genetic algorithm:

GA Inputs	Value
Population size	80
Maximum number of generations	15000
Sequence mutation rate	35%
Vehicle used mutation rate	25%
Random seed	13675

**Figure 5-6: Small Population Genetic Algorithm Inputs**

#### GA Results

custx	y	veh	time						
44	89	27	2	14:32					
6	87	28	1	9:11	16	78	27	2	14:55
45	86	34	1	9:28	33	66	63	2	15:16
50	88	34	1	9:40	46	20	90	2	15:36
36	97	66	1	10:19	1	36	64	2	15:52
34	34	57	1	11:19	12	55	93	2	16:08
10	34	61	1	11:36	38	25	70	2	16:21
9	24	46	1	11:56	17	8	43	2	16:34
11	1	7	1	12:33	26	15	25	2	16:45
42	9	27	2	8:34	19	3	1	2	16:57
28	31	7	2	9:13	2	30	23	3	8:22
8	69	6	2	9:57	18	51	14	3	8:54
49	65	79	2	11:05	43	51	27	3	9:21
15	61	91	2	11:23	5	62	35	3	9:44
27	70	91	2	11:39	7	58	39	3	9:59
30	64	81	2	11:55	21	96	21	3	10:42
48	79	63	2	12:37	47	92	67	3	11:24
20	87	46	2	13:05	23	57	52	3	12:00
24	67	18	2	13:32	3	13	53	3	12:36
4	43	14	2	13:51	22	12	76	3	12:57
35	87	12	2	14:18	40	7	87	3	13:25

13	30	90	3	13:44
31	21	88	3	14:03
32	22	89	3	14:15
29	31	67	3	14:34
14	89	69	3	15:01
41	79	63	3	15:14
37	77	62	3	15:26
25	49	0	3	15:49

39	26	14	3	16:41
----	----	----	---	-------

Total distance traveled by all vehicles = 1366  
Total customers not served = 0  
Total number of precedence violations = 0  
Total number of vehicles used = 3  
Last vehicle arrives back at depot at 16:59  
Fitness 32420.88607594937

### Greedy Results

custx	y	veh	time	
28	31	7	0	8:15
4	43	14	0	8:38
18	51	14	0	8:56
25	49	0	0	9:20
43	51	27	0	9:57
5	62	35	0	10:20
7	58	39	0	10:35
23	57	52	0	10:58
33	66	63	0	11:24
41	79	63	0	11:49
48	79	63	0	11:59
37	77	62	0	12:11
49	65	79	0	12:33
30	64	81	0	12:45
15	61	91	0	13:10
27	70	91	0	13:24
47	92	67	0	13:47
36	97	66	0	13:59
20	87	46	0	14:31
24	67	18	0	14:51
39	26	14	1	8:17
2	30	23	1	8:36
26	15	25	1	9:01
42	9	27	1	9:17
17	8	43	1	9:43
3	13	53	1	10:05
9	24	46	1	10:33
29	31	67	1	11:05

38	25	70	1	11:19
10	34	61	1	11:41
34	34	57	1	11:56
46	20	90	1	12:27
31	21	88	1	12:39
22	12	76	1	12:56
13	30	90	1	13:17
40	7	87	1	13:50
11	1	7	1	14:32
19	3	1	1	14:48
8	69	6	1	15:17
35	87	12	1	15:45
6	87	28	2	9:11
44	89	27	2	9:23
21	96	21	2	9:41
45	86	34	2	10:07
50	88	34	2	10:20
16	78	27	2	10:44
12	55	93	2	11:42
32	22	89	2	12:38
14	89	69	2	13:57
1	36	64	2	14:28

Total distance traveled by all vehicles = 1543  
Total customers not served = 0  
Total number of precedence violations = 5  
Total number of vehicles used = 3  
Last vehicle arrives back at depot at 16:02  
Fitness 1967325.0

### GA Max Time Results

cust	x	y	veh	time
2	30	23	0	8:22
5	62	35	0	9:06
33	66	63	0	10:09
37	77	62	0	10:41
47	92	67	0	11:21
36	97	66	0	11:36

14	89	69	0	11:54
32	22	89	0	13:13
46	20	90	0	13:25
31	21	88	0	13:37
15	61	91	0	14:27
12	55	93	0	14:43
7	58	39	1	8:51

6	87	28	1	9:32	11	1	7	3	13:27
50	88	34	1	9:48	19	3	1	3	13:43
45	86	34	1	10:02	28	31	7	3	14:21
16	78	27	1	10:22	42	9	27	4	8:59
24	67	18	1	10:46	3	13	53	4	9:35
35	87	12	1	11:37	27	70	91	4	10:53
44	89	27	1	12:02	30	64	81	4	11:14
21	96	21	1	12:21	23	57	52	4	11:53
8	69	6	1	13:01	1	36	64	4	12:27
20	87	46	1	13:54	38	25	70	4	12:49
49	65	79	1	14:43	10	34	61	4	13:11
34	34	57	1	15:31	43	51	27	4	13:59
9	24	46	1	15:55	4	43	14	4	14:24
26	15	25	1	16:27	18	51	14	4	14:42
39	26	14	1	16:52	25	49	0	4	15:06
48	79	63	3	9:22	Total distance traveled by all vehicles = 1866				
41	79	63	3	9:32	Total customers not served = 0				
29	31	67	3	10:30	Total number of precedence violations = 0				
13	30	90	3	11:03	Total number of vehicles used = 4				
40	7	87	3	11:36	Last vehicle arrives back at depot at 16:55				
22	12	76	3	11:58	Fitness 115900.62111801242				
17	8	43	3	12:41					

## Analysis

The GA using dynamic time out-performed the greedy algorithm and the GA max time. The GA using dynamic time and the greedy algorithm used three vehicles to service all of the customers, while the GA max time took four vehicles to service the same set of customers. The GA using dynamic distance also found the lowest travel time between customers (1366 minutes for all vehicles.) All three algorithms were able to service all of the customers within the workday (8:00am to 5:00pm). The greedy algorithm did have trouble solving the precedence relationships, but the other two algorithms were able to satisfy all of the precedence relationships. Overall, the GA using dynamic distance out-performed the other two methods in all categories.



### 5.1.2.2 Medium Population

#### Genetic Algorithm Inputs

The following inputs were used for the genetic algorithm:

GA Inputs	Value
Population size	160
Maximum number of generations	15000
Sequence mutation rate	35%
Vehicle used mutation rate	25%
Random seed	13675

Figure 5-7: Medium Population Genetic Algorithm Inputs

#### GA Results

<b>custx</b>	<b>y</b>	<b>veh</b>	<b>time</b>	31	21	88	1	15:56	
15	61	91	0	9:36	1	36	64	1	16:11
27	70	91	0	9:54	12	55	93	1	16:24
41	79	63	0	10:27	37	77	62	1	16:37
47	92	67	0	10:47	39	26	14	1	16:54
36	97	66	0	11:00	17	8	43	4	8:47
48	79	63	0	11:33	29	31	67	4	9:30
5	62	35	0	12:15	10	34	61	4	9:48
9	24	46	0	13:06	23	57	52	4	10:17
14	89	69	0	13:50	7	58	39	4	10:47
2	30	23	0	14:29	34	34	57	4	11:27
28	31	7	1	8:15	3	13	53	4	11:51
25	49	0	1	8:44	22	12	76	4	12:14
50	88	34	1	9:45	40	7	87	4	12:40
45	86	34	1	9:58	13	30	90	4	13:01
21	96	21	1	10:27	30	64	81	4	13:28
35	87	12	1	10:54	49	65	79	4	13:41
44	89	27	1	11:14	26	15	25	4	14:20
6	87	28	1	11:26	42	9	27	4	14:36
16	78	27	1	11:42	19	3	1	4	14:54
43	51	27	1	12:08	38	25	70	4	15:25
11	1	7	1	12:50	4	43	14	4	15:46
18	51	14	1	13:25	Total distance traveled by all vehicles = 1404				
24	67	18	1	13:51	Total customers not served = 0				
20	87	46	1	14:14	Total number of precedence violations = 0				
8	69	6	1	14:41	Total number of vehicles used = 3				
33	66	63	1	15:08	Last vehicle arrives back at depot at 16:57				
46	20	90	1	15:34	Fitness 34411.76470588235				
32	22	89	1	15:45	Stats: parent = 0.34375 child = 0.65625				

### Greedy Results

custx	y	veh	time	38	25	70	1	11:19	
28	31	7	0	8:15	10	34	61	1	11:41
4	43	14	0	8:38	34	34	57	1	11:56
18	51	14	0	8:56	46	20	90	1	12:27
25	49	0	0	9:20	31	21	88	1	12:39
43	51	27	0	9:57	22	12	76	1	12:56
5	62	35	0	10:20	13	30	90	1	13:17
7	58	39	0	10:35	40	7	87	1	13:50
23	57	52	0	10:58	11	1	7	1	14:32
33	66	63	0	11:24	19	3	1	1	14:48
41	79	63	0	11:49	8	69	6	1	15:17
48	79	63	0	11:59	35	87	12	1	15:45
37	77	62	0	12:11	6	87	28	2	9:11
49	65	79	0	12:33	44	89	27	2	9:23
30	64	81	0	12:45	21	96	21	2	9:41
15	61	91	0	13:10	45	86	34	2	10:07
27	70	91	0	13:24	50	88	34	2	10:20
47	92	67	0	13:47	16	78	27	2	10:44
36	97	66	0	13:59	12	55	93	2	11:42
20	87	46	0	14:31	32	22	89	2	12:38
24	67	18	0	14:51	14	89	69	2	13:57
39	26	14	1	8:17	1	36	64	2	14:28
2	30	23	1	8:36	Total distance traveled by all vehicles = 1543				
26	15	25	1	9:01	Total customers not served = 0				
42	9	27	1	9:17	Total number of precedence violations = 5				
17	8	43	1	9:43	Total number of vehicles used = 3				
3	13	53	1	10:05	Last vehicle arrives back at depot at 16:02				
9	24	46	1	10:33	Fitness 1967325.0				
29	31	67	1	11:05					

### GA Max Time Results

cust	x	y	veh	time	28	31	7	0	16:30
43	51	27	0	8:38	27	70	91	2	9:40
5	62	35	0	9:01	48	79	63	2	10:19
44	89	27	0	9:39	36	97	66	2	10:47
21	96	21	0	9:58	30	64	81	2	11:33
6	87	28	0	10:19	49	65	79	2	11:47
16	78	27	0	10:38	15	61	91	2	12:09
23	57	52	0	11:20	12	55	93	2	12:25
33	66	63	0	11:58	40	7	87	2	13:23
29	31	67	0	12:43	22	12	76	2	13:45
1	36	64	0	13:04	3	13	53	2	14:18
13	30	90	0	13:40	17	8	43	2	14:39
46	20	90	0	14:00	39	26	14	2	15:23
32	22	89	0	14:12	7	58	39	3	8:51
31	21	88	0	14:23	50	88	34	3	9:31
38	25	70	0	14:51	45	86	34	3	9:45
10	34	61	0	15:13	35	87	12	3	10:17
9	24	46	0	15:41	18	51	14	3	11:03

25	49	0	3	11:27	20	87	46	4	14:00
8	69	6	3	12:18	41	79	63	4	14:28
24	67	18	3	12:51	14	89	69	4	14:49
4	43	14	3	13:25	34	34	57	4	15:55
42	9	27	3	14:11	Total distance traveled by all vehicles = 1938				
26	15	25	3	14:33	Total customers not served = 0				
2	30	23	3	15:11	Total number of precedence violations = 0				
47	92	67	4	9:35	Total number of vehicles used = 4				
37	77	62	4	10:16	Last vehicle arrives back at depot at 16:41				
11	1	7	4	11:59	Fitness 114674.55621301776				
19	3	1	4	12:15					

### Analysis

The GA using dynamic time out-performed the greedy Algorithm and the GA max time. The GA using dynamic time and the greedy algorithm used three vehicles to service all of the customers, while the GA max time took four vehicles to service the same set of customers. The GA using dynamic distance also found the lowest travel time between customers (1404 minutes for all vehicles.) All three algorithms were able to service all of the customers within the workday (8:00am to 5:00pm). The Greedy algorithm did have trouble solving the precedence relationships, but the other two algorithms were able to satisfy all of the precedence relationships. Overall, the GA using dynamic distance out performed the other two methods in all categories.

### 5.1.2.3 Large Population

#### Genetic Algorithm Inputs

The following inputs were used for the genetic algorithm:

GA Inputs	Value
Population size	320
Maximum number of generations	15000
Sequence mutation rate	35%
Vehicle used mutation rate	25%
Random seed	13675

Figure 5-8: Large Population Genetic Algorithm Inputs

#### GA Results

<b>cust</b>	<b>x</b>	<b>y</b>	<b>veh</b>	<b>time</b>	25	49	0	4	8:33
38	25	70	0	9:09	43	51	27	4	9:10
34	34	57	0	9:36	18	51	14	4	9:31
10	34	61	0	9:51	7	58	39	4	10:06
12	55	93	0	10:31	6	87	28	4	10:40
15	61	91	0	10:46	50	88	34	4	10:54
27	70	91	0	11:02	45	86	34	4	11:06
30	64	81	0	11:20	16	78	27	4	11:23
47	92	67	0	12:01	24	67	18	4	11:47
14	89	69	0	12:14	5	62	35	4	12:15
20	87	46	0	12:47	2	30	23	4	12:45
44	89	27	0	13:16	48	79	63	4	13:58
21	96	21	0	13:30	36	97	66	4	14:26
8	69	6	0	13:52	23	57	52	4	14:48
35	87	12	0	14:20	1	36	64	4	15:05
37	77	62	0	14:45	29	31	67	4	15:25
41	79	63	0	14:56	32	22	89	4	15:39
49	65	79	0	15:12	31	21	88	4	15:50
9	24	46	0	15:32	46	20	90	4	16:03
33	66	63	0	15:51	40	7	87	4	16:15
39	26	14	0	16:13	28	31	7	4	16:33
13	30	90	2	9:29	4	43	14	4	16:44
22	12	76	2	10:17	Total distance traveled by all vehicles = 1340				
3	13	53	2	10:45	Total customers not served = 0				
17	8	43	2	11:02	Total number of precedence violations = 0				
26	15	25	2	11:25	Total number of vehicles used = 3				
42	9	27	2	11:41	Last vehicle arrives back at depot at 16:45				
11	1	7	2	12:21	Fitness 31803.79746835443				
19	3	1	2	12:37					

### Greedy Results

cust	x	y	veh	time	38	25	70	1	11:19
78	31	7	0	8:15	10	34	61	1	11:41
4	43	14	0	8:38	34	34	57	1	11:56
18	51	14	0	8:56	46	20	90	1	12:27
25	49	0	0	9:20	31	21	88	1	12:39
43	51	27	0	9:57	22	12	76	1	12:56
5	62	35	0	10:20	13	30	90	1	13:17
7	58	39	0	10:35	40	7	87	1	13:50
23	57	52	0	10:58	11	1	7	1	14:32
33	66	63	0	11:24	19	3	1	1	14:48
41	79	63	0	11:49	8	69	6	1	15:17
48	79	63	0	11:59	35	87	12	1	15:45
37	77	62	0	12:11	6	87	28	2	9:11
49	65	79	0	12:33	44	89	27	2	9:23
30	64	81	0	12:45	21	96	21	2	9:41
15	61	91	0	13:10	45	86	34	2	10:07
27	70	91	0	13:24	50	88	34	2	10:20
47	92	67	0	13:47	16	78	27	2	10:44
36	97	66	0	13:59	12	55	93	2	11:42
20	87	46	0	14:31	32	22	89	2	12:38
24	67	18	0	14:51	14	89	69	2	13:57
39	26	14	1	8:17	1	36	64	2	14:28
2	30	23	1	8:36	Total distance traveled by all vehicles = 1543				
26	15	25	1	9:01	Total customers not served = 0				
42	9	27	1	9:17	Total number of precedence violations = 5				
17	8	43	1	9:43	Total number of vehicles used = 3				
3	13	53	1	10:05	Last vehicle arrives back at depot at 16:02				
9	24	46	1	10:33	Fitness 1967325.0				
29	31	67	1	11:05					

### GA Max Time Results

cust	x	y	veh	time	26	15	25	0	15:49
43	51	27	0	8:38	39	26	14	0	16:14
24	67	18	0	9:06	25	49	0	1	8:57
44	89	27	0	9:39	7	58	39	1	9:47
6	87	28	0	9:53	37	77	62	1	10:26
50	88	34	0	10:09	11	1	7	1	12:09
45	86	34	0	10:23	19	3	1	1	12:25
16	78	27	0	10:43	22	12	76	2	9:16
4	43	14	0	11:30	13	30	90	2	9:48
18	51	14	0	11:48	46	20	90	2	10:08
5	62	35	0	12:21	15	61	91	2	10:59
1	36	64	0	13:09	12	55	93	2	11:15
38	25	70	0	13:31	49	65	79	2	11:42
10	34	61	0	13:53	33	66	63	2	12:08
34	34	57	0	14:10	29	31	67	2	12:53
3	13	53	0	14:41	32	22	89	2	13:26
9	24	46	0	15:17	31	21	88	2	13:37

40	7	87	2	14:15
42	9	27	2	15:25
28	31	7	2	16:04
17	8	43	4	8:47
20	87	46	4	10:16
41	79	63	4	10:44
48	79	63	4	10:54
47	92	67	4	11:17
36	97	66	4	11:32
14	89	69	4	11:50
27	70	91	4	12:29
30	64	81	4	12:50

23	57	52	4	13:29
21	96	21	4	14:28
35	87	12	4	15:03
8	69	6	4	15:50
2	30	23	4	16:42

Total distance traveled by all vehicles = 1824  
 Total customers not served = 0  
 Total number of precedence violations = 0  
 Total number of vehicles used = 4  
 Last vehicle arrives back at depot at 16:54  
 Fitness 102184.87394957984

## **Analysis**

The GA using dynamic time out-performed the greedy algorithm and the GA max time. The GA using dynamic time and the greedy algorithm used three vehicles to service all of the customers, while the GA max time took four vehicles to service the same set of customers. The GA using dynamic distance also found the lowest travel time between customers (1340 minutes for all vehicles.) All three algorithms were able to service all of the customers within the workday (8:00am to 5:00pm). The greedy algorithm did have trouble solving the precedence relationships, but the other two algorithms were able to satisfy all of the precedence relationships. Overall, the GA using dynamic distance out-performed the other two methods in all categories.

### 5.1.2.4 Comparison and Analysis

These examples shed light on several areas of interest in showing the advantage of the GA using dynamic distance. The results show that the GA using dynamic distance can solve a relatively large example containing 50 customers while not violating any of the precedence relationships and servicing all of the customers. It also shows that the GA using dynamic distance is superior to the greedy algorithm and the GA using maximum travel time. Thirdly, the results show the effects of using different population sizes for the GA. The table below shows a summary of the results from this example.

<b>Pop Size</b>	<b>Method</b>	<b>#Vehicles Used</b>	<b>Distance</b>	<b>#Precedence Violations</b>	<b>#Customers Not Served</b>
Small	GA Dynamic Dist	3	1366	0	0
	Greedy	3	1543	5	0
	GA Max Dist	4	1866	0	0
Medium	GA Dynamic Dist	3	1404	0	0
	Greedy	3	1543	5	0
	GA Max Dist	4	1938	0	0
Large	GA Dynamic Dist	3	1340	0	0
	Greedy	3	1543	5	0
	GA Max Dist	4	1824	0	0

**Figure 5-9: Comparison of Random Example Solutions**



It can be seen from the chart that the GA using dynamic time and a large population produced the best results using only three vehicles with a combined travel time of 1340. This was only slightly better than the GA using dynamic distance with a small population, 1366. While the medium sized population's distance was 1404, it is still a great deal better than the other two algorithms. These results in respect to the population size tend to show that the overall answer is not necessarily improved when population size is increased. This is due to the random nature of the GA.

The results also show the great advantage that using dynamic time has over an algorithm that just uses the max time between two customers. It was able to solve the problem in approximately a quarter less time and using one less vehicle.

The comparison with the greedy algorithm was done to show how another heuristic might solve the problem. As you can see from the results the GA, using dynamic distance outperforms the greedy algorithm in the area of overall distance and in solving the precedence relationships. The greedy algorithm's difficulty solving the precedence relationships was discussed in section 4.4.

Overall, the results of the random example show that the genetic algorithm using dynamic distance is a great improvement over the traditional way of defining the travel time between customers (the maximum time approach.).

### **5.1.3 Graphical User Interface Example**

This example demonstrates the use of the VRP GA using the graphical user interface on a relatively large VRP. When running the VRP interface, the software and genetic algorithm inputs are first set up, once “Use Current Inputs” is selected the main VRP interface window is redisplayed and the “Run” button is clicked to start the GA. The GA then begins to run and periodically updating the database with the route and performance data of the best individual to that point. The main VRP Interface then reads from the database and displays the current data in fields or on graphs.

#### **5.1.3.1 GUI Example Software and GA Inputs**

This contains 60 customers and uses randomly generated customer locations and times between those customers. The all routes must be completed in a workday of 8am to 5pm with each customer requiring a 15-minute stop for the items to be delivered before the vehicle can continue on its route. The vehicle mutation rate is set to 30% while the route sequence mutation rate is set to 35%. There are a maximum of eight vehicles that may be used to complete the routes. The Inputs GUI is displayed in the following figure:

**VRP Inputs**

Id: 17  
 Name: VRP GUI Example

GA Population Size: 200  
 Work Day (minutes): 540  
 Number of Customers (N): 60  
 Number of Generations: 20000  
 Maximum Number of Vehicles: 8  
 Time Per Stop: 15  
 Vehicle Mutation Rate: 0.3  
 Sequence Mutation Rate: 0.35  
 Random Seed: 18657

usePrecedence     useGreedy   
 useRandInputs     useMaxDist

Use Current Inputs

Record: 16 of 16

Figure 5-10: The Inputs Used for the VRP GUI Example

### 5.1.3.2 Results

The results are displayed on the main Interface GUI as they are available from the GA via the database. The “VRP Route Chart” displays the best solution so far. It is important to note that this graph is based on the physical location of customers based on an x-y coordinate system, however the VRP defined in this paper uses time not physical distance to create the routes between customers. The random data generator, as discussed in section 4.3.3, attempts to take in to account the physical locations of the nodes when generating the times, however it is still possible, both in this example and real life, for a customer to physically

located on the opposite side of the graph, yet be the closest customer in terms of travel time at any given time of day. This is while the graph does not display the four vehicles that were used to accomplish this solution simply partitioning the customers into four clusters containing customers in close physical vicinity.

The VRP interface also has the useful feature that the fitness of the best individual as well as the distance travel of the best individual (a main component of the fitness function) are graphed over the generations. This allows the user to receive feedback on the progress of the GA and potentially use the “Stop” button to terminate the GA if the solution is either acceptable to the user or is not making any noticeable improvements.

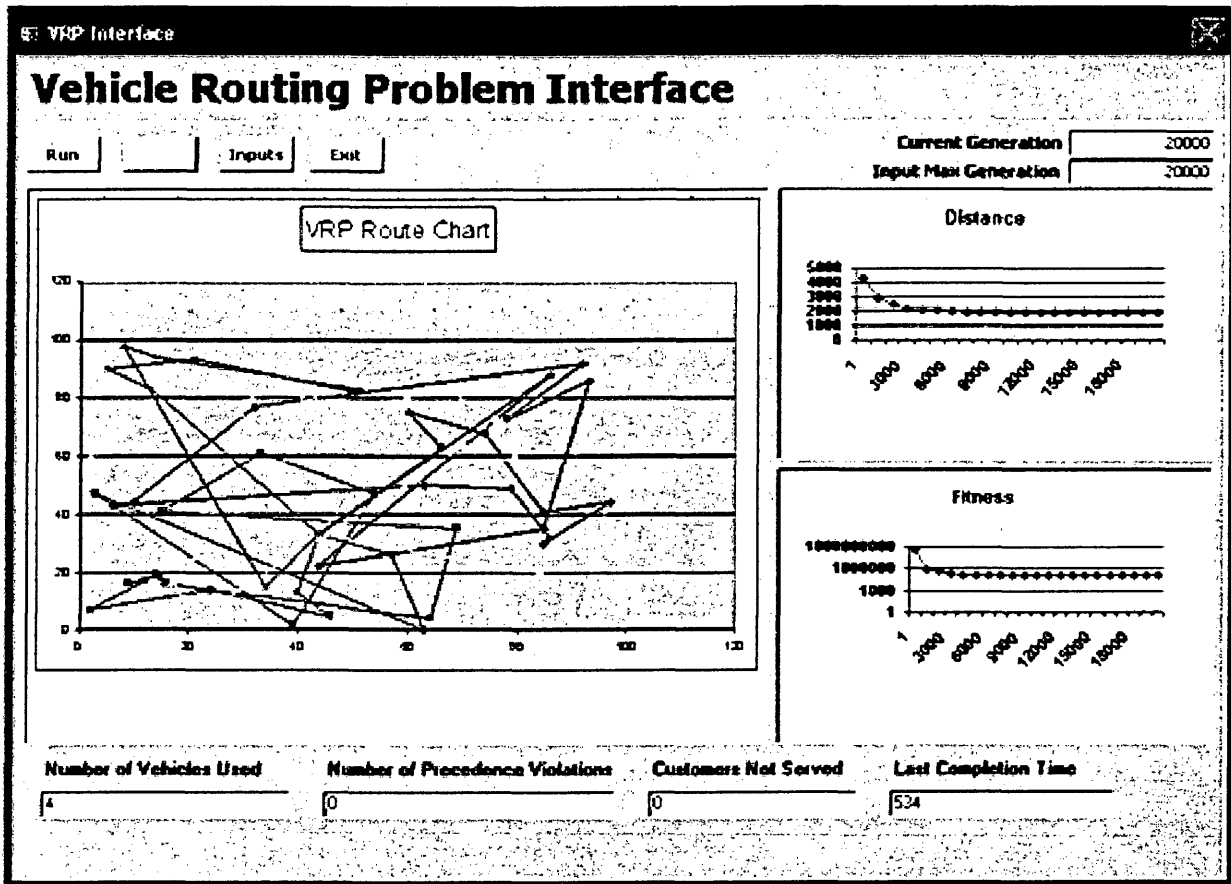


Figure 5-11: The Interface of the VRP GUI Example

This example solved the problem using only four vehicles within the scheduled workday.

The VRP Interface worked well in its ability to display the current progress of the GA as well as the current planned route.

## **6.0 Conclusions and Future Research**

The purpose of this project was to research the current VRP and provide meaningful additions to the VRP model. With the additions of dynamic travel time and precedence relationships, the VRP model is now closer to modeling the real world and therefore providing better solutions. This chapter will cover our conclusions, ideas for future research, and the numerous benefits that have come from working on this project.

### ***6.1 Conclusions***

The numerous applications of the VRP as it relates to real-world scenarios make it an invaluable part of our society. The fact that this problem is computationally intensive gives way to the use of heuristics such as genetic algorithms to solve it. With these two facts in mind, it is very important that we research the use of heuristics to solve the VRP with an emphasis placed on modeling the real world. Both enhancements to the VRP model discussed in this paper, dynamic time and precedence relationships, help us to achieve this goal. Dynamic time allows for an overall better solution by allowing the algorithm to use a more accurate travel time based on the time of day. The use of precedence relationships allows the VRP to utilize constraints that the end solution must obey for it to correctly model the real-world situation for which the routes will be used. The GA created in this paper also demonstrates its benefits over other heuristics, such as the greedy algorithm, which is limited by its inability to see the whole route at once and thus can not fully take advantage of the

changes in distance between customers as the GA does. The VRP is a complex and common problem in our society, and dynamic time and precedence relationships bring us closer to managing it.

## ***6.2 Ideas for Future Research***

There is a great deal of potential for commercial applications of the VRP using dynamic time and precedence relationships, and many of these applications may also facilitate future research into this subject matter.

One potential way to expand on the research and additions to the VRP presented in this paper is to add other variations of the VRP to our new model. These additions may include adding a capacity to each vehicle and then using a model such as the vehicle routing problem with split pick up method described earlier in this paper. This method would essentially allow each customer to request a certain amount of a product. If this amount was greater than the capacity of the vehicle, then it may require two or more vehicles to service the customer. There is also the potential to have vehicles with differing capacities. This may provide situations where a customer requires 100 units of a product, and it could either be serviced by two vehicles, one carrying 60 units and the other carrying 40 units, or the customer could simply be serviced by one vehicle carrying 100 units. These vehicles would then also be constrained as to the number of customers they could serve based on their capacity. This expansion also allows for different goods to be requested by the customers. These goods may have varying sizes that would also be constrained by the capacity of the vehicles. The

addition of time windows to the model would also provide expanded capability to our model, allowing the customers to request when their products are delivered.

Although some of these additions have been previously researched, they have not been fully integrated together. If such models as split pick-ups, time windows, limited vehicle capacity, different product requests by customers, as well as our additions of dynamic time and precedence relationships, were integrated into one model, it would have a great ability to simulate real-world data sets.

### ***6.3 Lessons Learned***

This project has allowed a great deal of education in the areas of research, genetic algorithms, the vehicle routing problem, and Java. One of the main reasons that I proposed this project to Professor Ali was to gain experience in these areas that were of great interest to me. Although I have had numerous research projects in the past, the research that was done at the beginning of this project allowed me to gain experience researching a relatively specific topic with a wide variety previous research done on it. I also analyzed that information to find the shortcomings of the previous work in order to propose new research that is interesting as well as applicable to real-world situations. The second area that interested me was the use of genetic algorithms to solve this complex problem. Although I have studied GAs in several of my computer science classes in the past, I have never had the opportunity to implement one. This aspect of the project provided me with a great learning opportunity. We also chose Java to write the GA, in part, because of my interest to become



more familiar with the language. Although I have a great deal of experience with other languages, I have never had the opportunity to write an application in it.

## 7.0 References

- [1] Sushel J. Louis, Xiangying Yin, Zhen Ya Yuan. (January 29, 1999): Multiple Vehicle Routing With Time Windows Using Genetic Algorithms. Genetic Adaptive Systems LAB.
- [2] Dimitris Bertsiman and David Simchi-Levi. (March 1993, revised December 1993): A New Generation of Vehicle Routing Research – Robust algorithms, Addressing Uncertainty.
- [3] Chi-Guhn lee, Marina Epelman, Chelsea C. White III, and Yavuz A. Bozer. (April 23, 2001): A Shortest Path Approach to the Multiple-Vehicle Routing Problem with Split Pick-Ups. Department of Industrial and Operations Engineering, University of Michigan.
- [4] Eric D. Taillard, Gilbert Laporte, and Michel Gendreau. (March 1995): Vehicle Routing with Multiple Use of Vehicles. University of Montreal.
- [5] Kim Kristensen and Thomas Randers Jensen. (May 1999): Solving a Real life Scheduling Problem Using Evolutionary Algorithms, Chapter 4.
- [6] Gambardella, Taillard, and Agazzi. (1999): MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing problems with Time Windows. Lugano, Switzerland. Swiss national Science Foundation project “Adaptive Memory Programming for Dynamic Optimization Problems”.
- [7] Martin Desrochers, Jacques Desrosiers, and Marius Solomon (Revised June 1990, Accepted January 1991): A new Optimization Algorithm for the Vehicle Routing

- Problem with Time Windows. *Operations Research* Vol. 40, No. 2, March-April 1992.
- [8] Lawrence D. Bodin (Received November 1989, accepted January 1990): Twenty Years of Routing and Scheduling. *Operations Research*, Vol. 38, No. 4, July-August 1990.
- [9] Julien Bramel and David Simchi-Levi (Received October 1993, accepted February 1995): On the Effectiveness of Set Covering Formulation for the Vehicle Routing Problem with Time Windows. *Operations Research*, Vol. 45, No. 2, March-April 1997.
- [10] Marshall L. Fisher (Received February 1990, accepted November 1993): Optimal Solution of the Vehicle Routing Problems Using Minimum K-Trees. University of Pennsylvania, *operations Research*, Vol. 42, No. 4, July-August 1994.
- [11] Julien Bramel, Edward G. Coffman Jr., Peter W. Shor, and David Simchi-Levi (Received June 1991, Accepted October 1991): Probabilistic Analysis of the Capacitated Vehicle Routing Problem with Unsplit Demands.