Vehicle Routing Problem Optimization

JAY TURAKHIA, Northeastern University
SHLOK GANDHI, Northeastern University
CHIRAYU DESAI, Northeastern University

This report covers a very interesting problem that has been in research since a very long time. The problem is formally known as the "Vehicle Routing Problem". The problem came in limelight in Oct. 1959 when Dantzig and Ramser published their paper **The Truck Dispatching Problem** [1] in which they talked about finding an optimal route for delivering products across a number of stations with minimum fleet. Although a solution based on linear programming formulation was made but since then many solutions have been presented for this problem where each have some improvement over others.

KEYWORDS

Vehicle Routing Problem, Sweep Algorithm, Savings Algorithm, Genetic Algorithm

1 INTRODUCTION

With the boom in e-commerce, fast moving consumer goods and increased production in agricultural products, there is a dire need to improve the logistics so that inventory can be moved quickly. This holds great importance since there are a few factors that affect each organization concerned with moving products like

- Longevity – Some perishable products may need optimized routes to reach customers faster
- Customer Demand – Businesses these days use fast delivery as a marketing strategy
- Environmental concerns – Optimizing delivery helps reduce fleet vehicles and hence reduced emissions
- Reduced Costs – Businesses save a lot when goods delivery is optimized

There are many benefits that may be specific to businesses and thus developing an optimal solution to this problem becomes necessary. This paper explores various approaches that have been taken to solve this problem and discusses the ways that were tried to improve the current results.

Vehicle Routing Problem is a generalization of the Travelling Salesman problem. There are nodes which represent customers/delivery spots/end points and a depot. Each customer position is defined on a planar surface and the distance to this point is known. A formulation of this problem would help the reader identify the key components involved and determine what are the necessary restrictions that need to be taken care of while devising a solution for the problem.

Constants

K = number of vehicles

n = number of customers to which a delivery must be made. Customers are indexed from 1 to n and index 0 denotes the central depot.

b_k = capacity (weight or volume) of vehicle k.

a_i = size of the delivery to customer i.

c_{ij} = cost of direct travel from customer i to customer j.

Variables

Y_{ik} = 1, if the order from customer i is delivered by vehicle k

0, otherwise

X_{ijk} = 1, if vehicle k travels directly from customer i to customer j

0, otherwise

Formulation of VRP

$$min \sum_{ijk} c_{ij}\, x_{ijk} \qquad \dots (1)$$

$$\sum_{k} y_{ik} = \begin{cases} K, & i = 0 \\ 1, & i = 1 \dots k \end{cases} \qquad \dots (2)$$

$$\sum y_{ik} = 1 \; or \; 0 \qquad i = 0 \dots n, \quad k = 0 \dots K \qquad \dots (3)$$

[6]

Since this problem came to the forefront there have been many variants of this problem. These include the following:

- Capacitated VRP
- Multiple Depot VRP
- Periodic VRP
- Split Delivery VRP
- Stochastic VRP

The focus of this paper is on Capacitated Vehicle Routing Problem (CVRP). The extra parameter that has been added to the original VRP is the capacity variable. The capacity here refers to the carrying capacity of the delivery unit. This seems to be practical since with varying delivery units like airplanes for international cargo, trucks for inter-state delivery and mid-size carriers for intercity deliveries, this limit helps us formalize the load bearing capacity of each vehicle. With known customer demands we can determine how much will each vehicle be able to carry, and which customers can be served by a particular customer. Adding the following restriction to our earlier formulation describes CVRP.

$$\sum_{i} a_i y_{ik} \leq b_k, \qquad k = 1,2 \dots K$$

[6]

2 Brief on Approaches and challenges

CVRP, being a generalization of Travelling Salesman Problem is NP Hard. With all such problems, solutions must be extremely efficient or else all computing resources are utilized and in the end no solution is reached.

We had initially assumed that since **Genetic Algorithms** (explained further) are highly exploratory in nature, they would find the optimal solution. They did work well for small set of customers since the computation was limited. But once we started with higher number of customers, the algorithms started displaying their shortcomings. Computations increased exponentially, and we had to look to other solutions. We then moved onto heuristics and search. We found **Savings** [2] to be a promising candidate out of a few approaches. However, savings in itself wasn't performing well. A few improvements were needed. Through our readings we landed on an **Improved Savings** [3] algorithm. We took a few ideas from there and created our own algorithm which gave us satisfactory solutions. The heuristic algorithms didn't take into consideration the distribution of customer nodes across the search space. This was achieved through a class of approaches known as **2-Phase (Cluster-first, Route-second)**

algorithms. Here we implemented Sweep [4] to cluster the customer nodes. Each cluster satisfies the capacity constraint set by the capacity of the vehicle. Following this, the route was determined by two approaches, (1) Solving as a Travelling Salesman Problem[5] (2) Modified Genetic Approach. Our approaches are further discussed in detail.

Our approaches managed to perform very close to the best-known solutions. We have included a comparative analysis in this report. The evaluation section of this report should give the reader an understanding of how different algorithms performed under different customer distributions and possible scope of improvements. Apart from the solutions we implemented, there are some that we considered but didn't implement. Following is a list of such solutions –

- Fisher and Jaikumar[6] for clustering
- Granular Tabu search[7] as a meta-heuristics approach
- Constaint Programming (CP) [8]

3 Approach

The following approaches were taken to solve the vehicular routing problem.

- Heuristics:
  - Improved savings: built on the idea of Clark and Wright
- Two-Phase (Cluster first, solve later)
  - Sweep (Cluster) and TSP (solution)
- Meta Heuristics
  - Genetic Algorithm Approach 1
  - Genetic Algorithm Approach 2

3.1 Improved Savings

Traditionally, heuristics have assisted search methodologies in many problems. Heuristics may not help achieve the best solution but provide a useful measure to reach to an acceptable solution. For example, a game of Pacman (applicable to many map-based real-world searches) may use manhattan distance to other elements in the maze as a heuristic to find an optimal solution.

Since the vehicle routing problem is NP Hard, heuristics would provide a quick way to reach an optimal solution. After researching through various heuristics, we decided to pursue the "savings" heuristics. This was originally proposed by Clarke and Wright in 1964[2]. They suggested that savings was a measure of cost saved by taking a route between any two nodes p and q as opposed to taking an individual route like $depot - p - depot$ and $depot - q - depot$. This measure helps determine that which node pair, when included in a route would provide the maximum savings in terms of distance travelled and hence cost. A detailed explanation has been given below.

Start by generating savings pairs between nodes. In order to generate savings pair, first calculate the distance between every two nodes and between each node to the depot using the following equation:

$$d_{pq} = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$$

where p and q are the nodes and d_{pq} is the Euclidean distance between them. Then calculate all savings between every two nodes. Rank the savings in decreasing order:

$$s_{pq} = d_{p0} + d_{q0} - d_{pq}$$

where d_{p0} is the Euclidean distance between the node p and the depot.

Start with the first savings pair, add it to the solution route. Look for pairs with one node connected to the savings pair in route, if found add it to the route. Continue this until for one pass thereby utilizing the trucks capacity to its maximum. If pairs are left add a new truck and continue the process.

This approach proves to be optimal and quick. It can be applied if the application requires to calculate new routes frequently based on changing requirements. There is however one shortcoming to this approach. It doesn't take into consideration how p and q are spatially related. Although there are few cases where even if nodes p and q are farther from each other they are selected to give a higher savings. But if the ordering of nodes is improved and modifications are made as to how new nodes are added to existing routes we might achieve better results.

3.2 Two-Phase Cluster and Solve

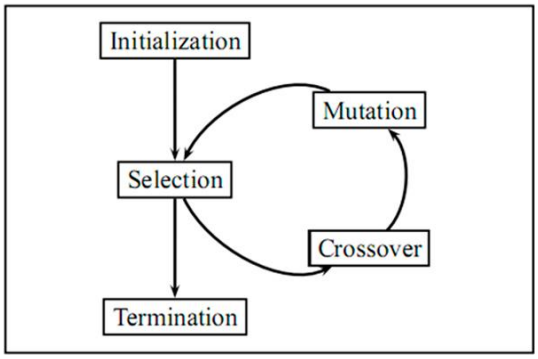
Two-phase methods primarily solve the problem by finding the routes and clustering the nodes, which can be done in either order. In Route-First Cluster-Second method, a single route through all the nodes is found considering the problem as Travelling Salesman Problem (TSP) and then the found route is broken into multiple sub-routes considering the capacity of the trucks available. First and the last customer of each subroute are connected to the depot. The final set of routes forms the solution to the vehicle routing problem. In contrast, in Cluster-First Route-Second method, all nodes are clustered into sets which in turn are solved as an individual problem of TSP. Sweep Algorithm solves the problem with latter approach – Cluster-First Route-Second.

In the first phase of Sweep algorithm, all nodes are divided into clusters such that the total demand of all the nodes in the cluster totals to less than the capacity of a single truck. Initially, all the nodes are ordered with increasing polar angle through a reference axis perpendicular to the depot. Once all the nodes are radially ordered, clusters are formed on the condition that the total demand in the cluster is less than or equal to the capacity of the vehicle. In the second phase of the algorithm, clusters created are solved as individual TSP. Google OR tools library (which internally uses Constraint Satisfaction) has been used to solve those individual subproblems. Once a solution for each cluster is generated, we create a final solution route from those solutions.

3.3 Genetic Algorithm

3.3.1 Genetic Approach 1

Genetic algorithms are metaheuristics that are inspired by evolution theory. Below is the basic flow of genetic algorithms:



In the initialization step a population to undergo evolution is generated. A population shall comprise of individuals. Each individual is made of chromosomes which comprise of genes, just like human genome. Fitness assignment is responsible for assigning a fitness value to each individual based on a fitness function. The fitness function assigns a quantifiable weight decided features of the individual and produces a cumulative value. Selection refers to picking individuals with the best fitness values for recombination as parents. Crossover and mutation are the operators that will be applied on parents to generate offspring. These offspring then become a part of the population. If the stopping criterion is met, the solution is returned. Stopping criterion refers to achieving an individual with a desired fitness or completing evolution over a required number of generations.

In this approach each solution (collection of valid routes) to CVRP will be represented by a chromosome, on which genetic operators (crossover and mutation) will be applied. In this approach each chromosome is represented by the following structure:

….. {DEPOT – NODE1 – NODE2 – DEPOT – NODE3 – NODE4 – NODE5 - DEPOT} ……

Each gene in a chromosome is node. Each Chromosome will be a collection of routes. This approach of solving the vehicle routing problem by genetic algorithm consists of the following steps:

First an initial population with solutions complaint to the above structure of chromosome are generated randomly. A fitness function will be used to compute the fitness of each solution in the population. Solutions with lesser route cost will have higher fitness. Based on the fitness, two best parents are selected for recombination. The parents undergo crossover using partially matched crossover technique to generate offspring with a certain probability. The offspring then undergo mutation through random node exchange with a certain low probability. The resulting offspring are added to the population. We repeat the genetic algorithm from the selection step until

the stopping condition is met. The stopping condition refers to completing a pre-stated number of generations with valid solutions.

This approach takes a long time to reach a solution, so it is not feasible for problems in which we need to change route dynamically. Since the generation of initial population is done randomly, we are not able to achieve an optimal solution.

3.3.2 Genetic Approach 2

This approach tries to work out the initial population generation shortcoming of the first genetic algorithm approach. The idea is to generate a robust population initially for genetic algorithm to solve the sub-problems generated by clustering.

Valid sub-solutions can be formed using clusters made by ordering nodes in radial order, keeping a constraint to not exceed a vehicle's maximum capacity. If adding a new node to the cluster leads to the vehicle exceeding its capacity, a new vehicle is added. This leads to a set of sub-solutions that make up a valid solution in which all required nodes would have been classified into clusters using one or more vehicles. Now genetic algorithm can be applied to these sub-solutions. Each sub-solution will be represented by a chromosome, where each node will be a gene. Genetic operators (crossover and mutation) will be applied on the chromosome. Chromosome is represented by the following structure:

….. {DEPOT – NODE1 – NODE2– NODE3– NODE6 – NODE4– NODE5 – DEPOT} ……

This approach of solving the sub-problems (clusters) by genetic algorithm consists of the following steps:
For each sub-solution generate an initial population by uniformly mutating it with a 50% mutation probability, until the population reaches a pre-decided size. Select fittest parents, i.e. the ones with the lowest sub-solution cost. Apply partially matched crossover and uniform mutation with a pre-decided probability and add children to the population for a given number of generations. Update the sub-solution if and only if the cost of the new sub-solution is lower than the current sub-solution cost. Repeat the process for all sub-solutions.

On an average this approach generates lower cost solutions in comparison to other approaches taken above. However due to an evolutionary step involved, it takes more time than improved savings and sweep-TSP approach, consequently is less feasible to use in problems which require dynamic path updates

4 Data and Results

4.1 DATA

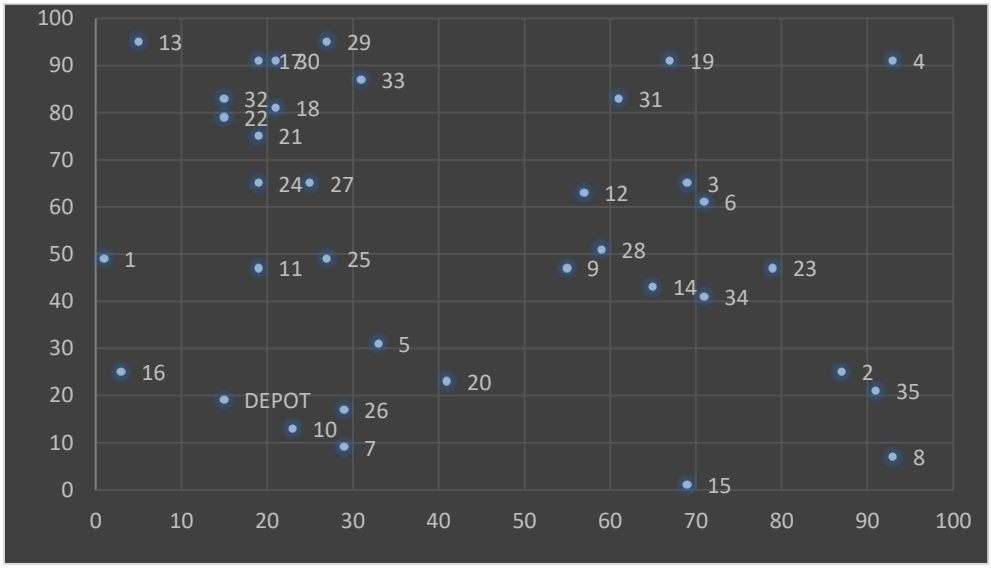
To compare the results of algorithms, computational tests were performed on a total of five datasets. For robust testing, all datasets differed from each other in number of nodes, node locations and node density, depot position.
For representation, one of those datasets is selected, and analyzed by comparing results of different approaches on this dataset.
The nodes, coordinates and demands of the selected dataset are as follows:

Node	Coordinate	Demand
DEPOT	(15,19)	0
1	(1,49)	1
2	(87,25)	14
3	(69,65)	15
4	(93,91)	11
5	(33,31)	18
6	(71,61)	2
7	(29,9)	22
8	(93,7)	7
9	(55,47)	18
10	(23,13)	23
11	(19,47)	12

Node	Co-ordinate	Demand
12	(57,63)	21
13	(5,95)	2
14	(65,43)	14
15	(69,1)	9
16	(3,25)	10
17	(19,91)	4
18	(21,81)	19
19	(67,91)	2
20	(41,23)	20
21	(19,75)	15
22	(15,79)	11
23	(79,47)	6

Node	Co-ordinate	Demand
24	(19,65)	13
25	(27,49)	19
26	(29,17)	13
27	(25,65)	8
28	(59,51)	15
29	(27,95)	18
30	(21,91)	11
31	(61,83)	21
32	(15,83)	12
33	(31,87)	2
34	(71,41)	23
35	(91,21)	11

The representation of the nodes in the table 1 on a XY plane is as follows:



There are some assumptions made in solving the problem. Firstly, the capacity of all trucks is considered the same. In practical scenarios, the trucks may have different capacities, however that is outside the scope of this project implementation and might be included in future scope for the project. Furthermore, the second assumption Is there is no limit on number of trucks. The algorithms would still try to minimize the number of trucks used. All the datasets are planar i.e. all nodes are in the same plane and cost is the Euclidian distance between those nodes. Finally, the last assumption is that all nodes have their demand ready beforehand so that every node will be visited only once.

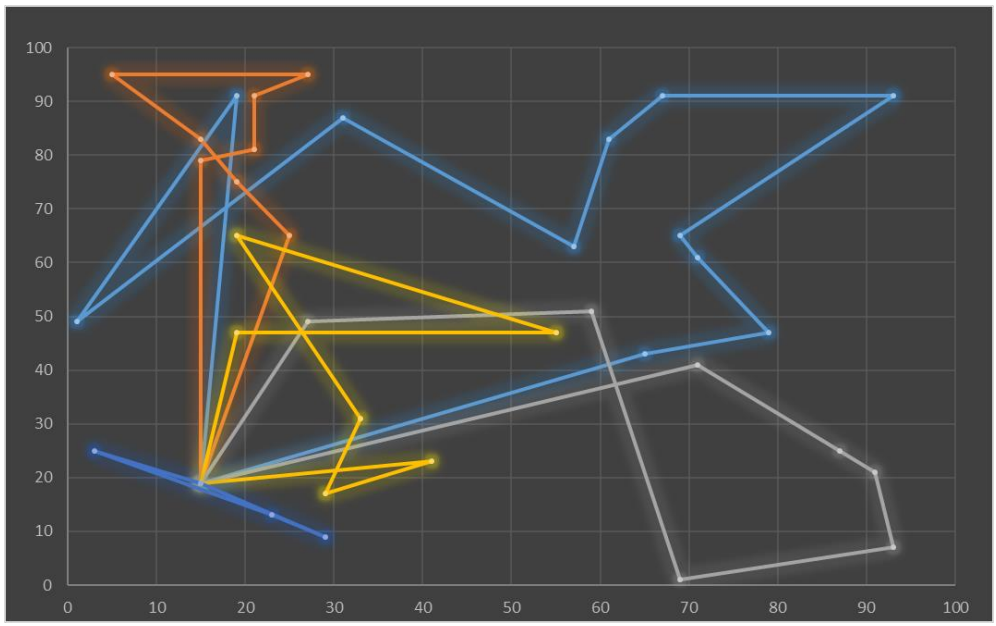
4.2 RESULTS

The table below shows the results of all 4 algorithms, Google OR tools and best-known solution for the dataset under consideration with cost taken as Euclidian distance rounded to the closest integer are as follows:

Algorithm	Cost
Savings	1072
Sweep	962
Genetic1	1550
Genetic2	960
Google OR Tool	1065
Best known solution	799

Improved Savings

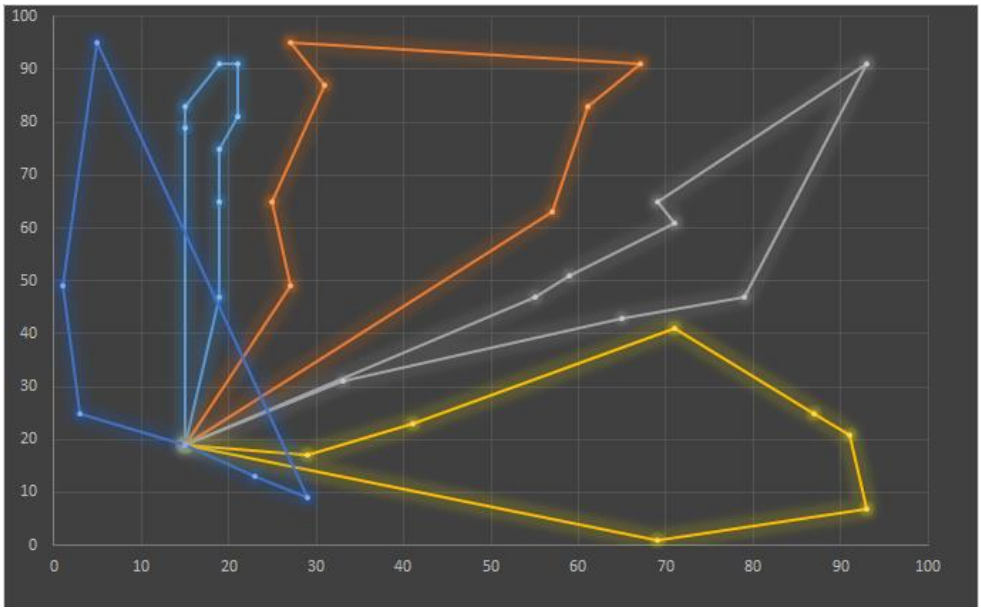
The solution by improved savings algorithm used 5 trucks with their routes totaling to distance of 1072 units. The routes can be plotted on a planar graph as shown below.



The routes can crossover each other which results in sub-optimal route length. Savings algorithm proposed by Clarke and Wright produced better results in beginning and results did not improve significantly while the problem converged. This problem can be eliminated by using the improved savings approach.

Sweep

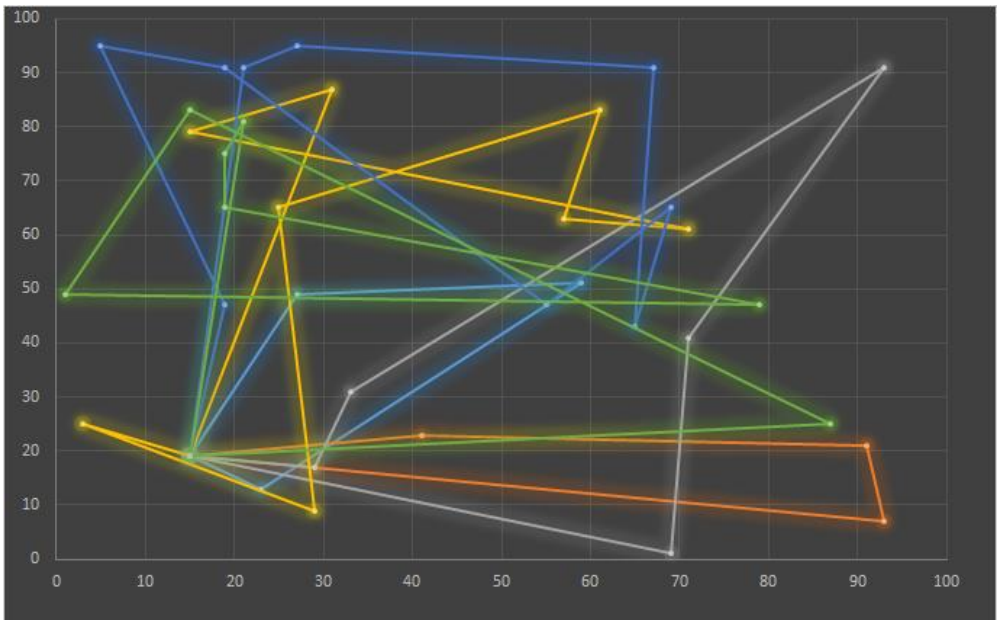
The solution by sweep algorithm used 5 trucks with their routes totaling to distance of 962 units. The routes can be plotted on a planar graph as shown below.



The clustering in the first phase of sweep algorithm helped in eliminating the crossovers in routes as seen in improved savings algorithm. As a result, the cost for same dataset can result in improvement by around 10%.

Genetic1

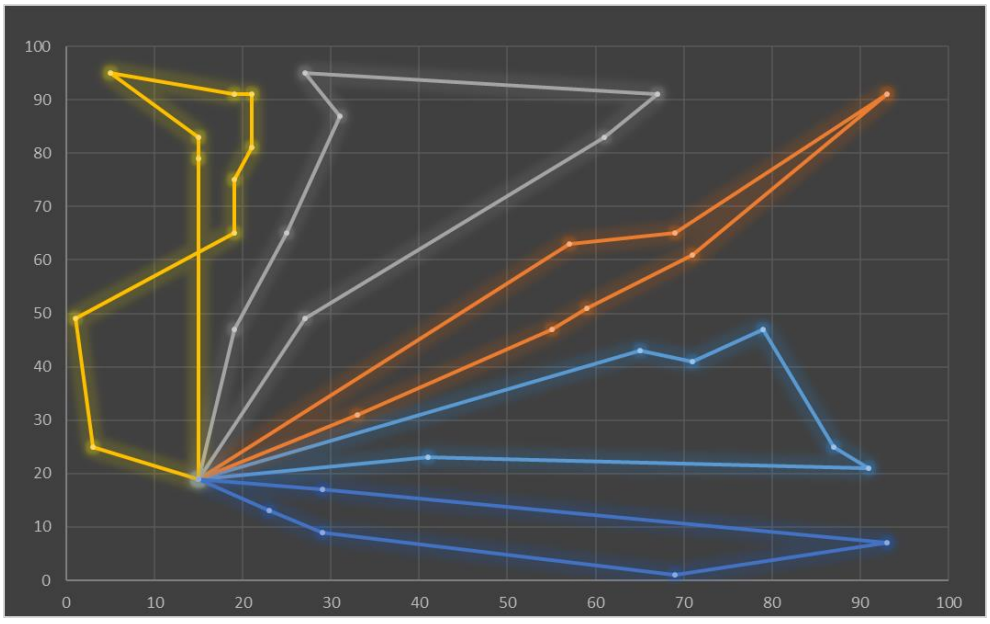
The solution by the generic Genetic algorithm used 6 trucks with their routes totaling to distance of 1550 units. The routes can be plotted on a planar graph as shown below.



As the initial population generation is random, the chromosomes do not have any regular behavior. As a result, the running time is high, and the routes generated contains the nodes that are not close to each other and cost is relatively higher than improved savings and sweep algorithms. The result is averaged by running the algorithm 5 times over the same dataset for all datasets.

Genetic2

The solution by the improved Genetic algorithm used 5 trucks with their routes totaling to distance of 960 units. The routes can be plotted on a planar graph as shown below.



As an improvement over generic Genetic algorithm, the initial population was generated using clustering. Here the routes do not crossover each other and save cost relative to the former approach. The running time is similar to the former Genetic approach, but the results are significantly better. Hence, this approach has benefits of orderliness of clustering and scaling of genetic algorithm.

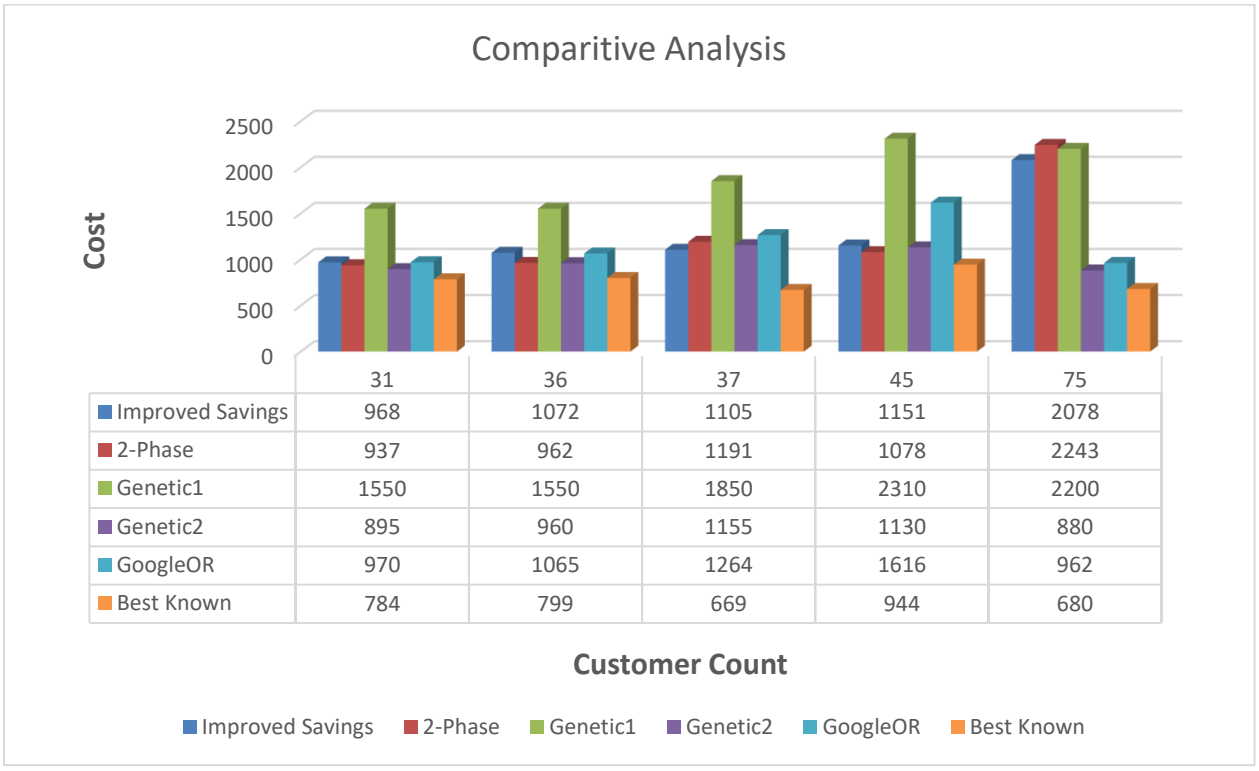
Google OR Tools

The solution by Google OR Tools' library used 5 trucks with their routes totaling to distance of 1065 units. Internally, the problem is solved using Constrain Satisfaction Problem. The algorithm produces results in short time and scales well.

Best-known Solution

The best-known solution for the given dataset used 5 trucks with their routes totaling to distance of 799 units.

The figure below is a comparative analysis of all algorithms across 5 datasets



	31	36	37	45	75
Improved Savings	968	1072	1105	1151	2078
2-Phase	937	962	1191	1078	2243
Genetic1	1550	1550	1850	2310	2200
Genetic2	895	960	1155	1130	880
GoogleOR	970	1065	1264	1616	962
Best Known	784	799	669	944	680

Several observations can be made from the above comparison.

- Improved Savings and Sweep algorithms give near better solutions for smaller datasets while both the algorithms fail to scale efficiently for larger datasets.
- The generic approach of Genetic algorithm gave the least optimal route due to randomness in the population but as genetic algorithm scales well, the cost for larger dataset is around the results of improved savings and sweep.
- The second approach of genetic algorithm, which also uses clustering gives better results to all former approaches and scales efficiently.
- None of the approaches gave optimal results which shows that a single algorithm cannot give optimal results for all the datasets

5. CONCLUSIONS

Vehicle Routing Problem enabled us to apply Artificially Intelligent algorithms to NP-Hard problems. We learned that even if algorithms appear random in nature, they succeed in finding a feasible solution that can be applied in real world scenarios.

A single solution couldn't be devised that can be applied to all scenarios and that guaranteed an optimal solution every time. The distribution of nodes / customers, or the search space in AI terminology governed how well would the algorithm perform.

Heuristics and search are some of the very primitive approaches to solving problem. All approaches had these components in some form. Without these, either the algorithm became extremely complex or a feasible solution was not reachable. Choosing the right heuristic is also of great value. It can significantly increase or decrease the accuracy of your solution. The Time taken to solve a problem drastically changes across our implementations. This might matter when we want to incorporate the solution in dynamic environments where the routes have to updated constantly.

6. FUTURE SCOPE

We will try to incorporate the good aspects of all four approaches and implement a hybrid solution to the capacitated vehicle routing problem. We will also explore methods like Simulated Annealing and Tabu search to solve the problem. Our approaches so far assume a homogeneous fleet with a constraint on the vehicle capacity. In future, we will incorporate the concept of a heterogeneous fleet in which different vehicles have different capacities. We can further add constraints on the time window of delivery, the priority of each node, etc. We will also modify our solutions to handle multiple depots and the aspect of picking up goods on the way for a delivery. After achieving a satisfactory solution, we will further look into using it to answer dynamic routing queries.

7. ACKNOWLEDGMENTS

This project wouldn't have been possible without the constant assistance from Professor Stacy Marsella and his teaching assistants Dan Feng and Navya Kuchibhotla. The concepts taught in class were extremely useful in our implementations.

REFERENCES

[1] - G. B. Dantzig and R.H. Ramser. "The Truck Dispatching Problem". Management Science 6, 80â91. 1959

[2] - G. Clarke and J. Wright "Scheduling of vehicles from a central depot to a number of delivery points", Operations Research, 12 #4, 568-581, 1964.

[3] - Wang Xing, Zhao Shu-Zhi, Wang Xing, Chu Hao and Li Yan "An improved savings method for vehicle routing problem", 2016

[4] - B. E. Gillet and L. R. Miller. "A Heuristic Algorithm for the Vehicle Dispatch Problem". Operations Research, 22:340-349, 1974.

[5] - Google OR tools

[6] - M. L. Fisher and R. Jaikumar. "A Generalized Assignment Heuristic for Vehicle Routing". Networks, 11:109-124, 1981.

[7] - P. Toth and D. Vigo. "The Granular Tabu Search (and its Application to the Vehicle Routing Problem)". Working Paper, DEIS, University of Bologna. 1998.

[8] - P. Shaw. "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems", Proceedings of the Fourth International Conference on Principles and Practice of Constraint Programming (CP '98), M. Maher and J.-F. Puget (eds.), Springer-Verlag, 417-431. 1998