Project Report

Mustafa Gökhan Oyanık

1.Introduction

VRP is a is an optimization problem where a fleet of vehicles must service a set of customers from a depot while minimizing the total cost, measured in distance or time. This report represents the result of three parts of the project using various construction and improvement heuristics, as well as Gurobi. We compared the quality in terms of cost and time.

2.Mathematical Model For VRP model

Parameters:

- $N = \{0, 1, 2, ..., n\}$ the set of nodes (0 is the depot)
- d_{ij}: distance between node i and j
- q_i: demand of customer i
- Q: vehicle capacity
- K: number of vehicles

Decision Variables:

- $x_{ij} = 1$ if the route from i to j is used; 0 otherwise
- u_i: cumulative demand at node i

Objective Function:

Minimize the total distance:

$$\min \sum\sum d_{ij} * x_{ij}$$

Visit each customer exactly once:

$$\sum x_{ij} = 1 \text{ for all } i \neq 0$$

$$\sum x_{ij} = 1 \text{ for all } j \neq 0$$

- Vehicle count from/ to depot (Use k-means to cluster nodes):

$$\sum x_{0j} = K \text{ and } \sum x_{i0} = K$$

3. Subtour elimination (MTZ):

- $u_i - u_j + Q\, x_{ij} \leq Q - q_j$ for all $i \neq j$, $i \neq 0$, $j \neq 0$

4. Capacity constraints:

- $q_i \leq u_i \leq Q$ for all $i \neq 0$

3. Heuristic Algorithms

CFRS-Cluster First, Route Second:

- Use k-means to cluster nodes
- Solve TSP within each cluster

RFCS-Route First, Cluster Second:

- Solve TSP across all customers
- Split the route into feasible tours

    $\sum q_i \le Q$ for each route r

Christofides Algorithm:

- Build Minimum Spanning Tree (MST)
- Find perfect matching on odd-degree nodes
- Merge edges to create Eulerian tour
- Shortcut to obtain a Hamiltonian tour
- Approximation ratio: $\le 1.5 \times$ OPT

2-Opt Algorithm:

- Iteratively swap two edges
- Accept only if distance is reduced

Relocate Algorithm:

- Move one customer from one position to another
- Accept only if feasible and reduces cost

4. Part A – CFRS + Improvement

Combination	**Cost**	**% Improve**	**Time (s)**
CFRS + Nearest Neighbor	111,919,861.75	0.00%	0.4222
CFRS + NN + 2-opt	111,566,922.53	0.32%	0.4559
CFRS + NN + Relocation	106,243,949.25	5.07%	0.4284

5. Part B – RFCS + Improvement

Combination	Cost	% Improve	Time (s)
RFCS + Christofides	106,118,215.32	0.00%	0.0135
RFCS + Christofides + Relocate	103,949,916.89	2.04%	0.0159
RFCS + Nearest Neighbor	101,142,323.85	0.00%	0.0020
RFCS + Nearest Neighbor + 2-opt	100,962,383.69	0.18%	0.0039

6. Part C – Gurobi Optimization

Method	Cost	Time (s)
Gurobi Optimal VRP Solution	86,527,607.39	900.0464

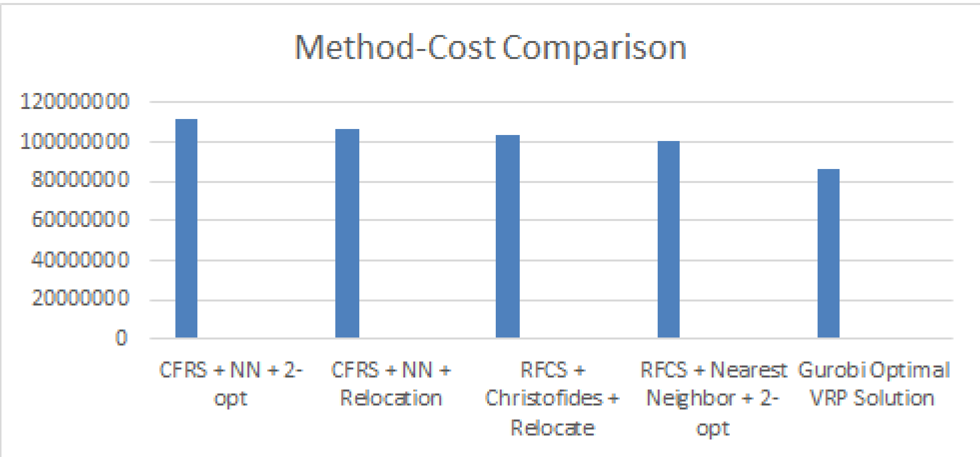
7. Overall Comparison

Method	Cost	% Improve	Time (s)	% vs Gurobi
CFRS + NN + 2-opt	111566922.5	0.32	0.4559	22.44331436
CFRS + NN + Relocation	106243949.3	5.07	0.4284	18.55761387
RFCS + Christofides + Relocate	103949916.9	2.04	0.0159	16.7602919
RFCS + Nearest Neighbor + 2-opt	100962383.7	0.18	0.0039	14.29718255
Gurobi Optimal VRP Solution	86527607.39		900.0464	0

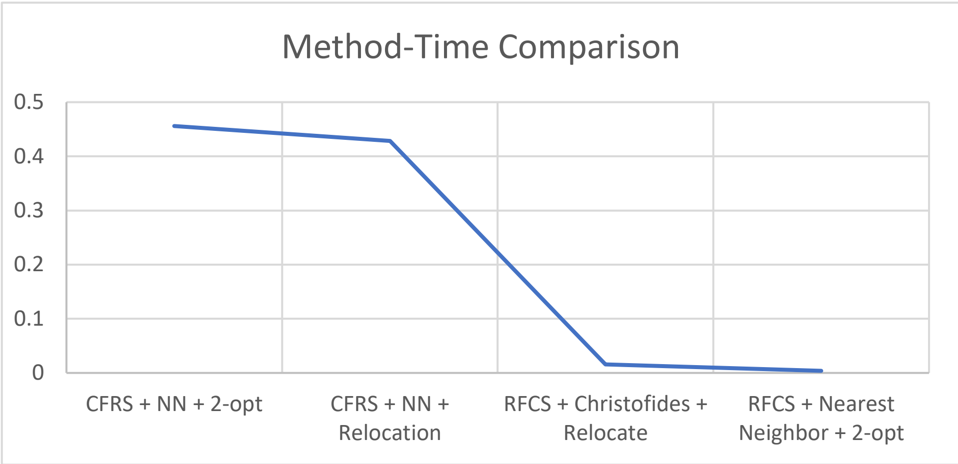
Note: "% vs Gurobi" measures how much more costly a method is compared to Gurobi.

According to the data we collected:

- The gurobi provides the most cost-efficient solution. But heuristics, the RFCS + Nearest Neighbor + 2-opt offers the lowest cost (100.96M), showing the closest result to Gurobi with only a **14.3%** deviation.



- CFRS with Relocation gives a larger improvement (5.07%) than CFRS with 2-opt (0.32%). This proves that Relocation is more impactful in CFRS.
- However RFCS based methods outperform CFRS in overall cost.



- Heuristic methods are faster compared to Gurobi.
- Even the most time consuming heuristic (CFRS+NN+2-opt) only takes **0.456** seconds
- Gurobi takes 15 minutes (900 seconds) but gives a 100% optimal solution

Finaly,

- If accuracy is important and time is not a concern, then we must use gurobi.
- If time is important with near-optimal solution, we must use RFCS+Nearest Neighbor+2-opt because it is the best trade-off between runtime and cost.
- RFCS+NN+2-opt is the best choice because:
  - It has the lowest cost among heuristics (100.96M)
  - It is only **14.3%** away from the optimal gurobi solution
  - It is the fastest.