Optimizing Freight Transportation Routes using Vehicle Routing Problem with Time Windows

Team

Group - 7 | Operations Research Laboratory: IM29204

Sanchay Baranwal - 22IM10032
Aman Sinha - 22IM10004
Jishnu Mukherjee - 22IM10019
Sandipan Ghosh-22IM10045

Abstract
The report looks at the attempt to present an optimized method of freight transportation over a large scale. Several assumptions have been made which are enlisted in further sections. The mathematical formulation is given, and the code has been executed through the IBM CPLEX Solver. The implementation of Modified-Vehicle Routing Problem with Time Windows (MVRP-TW) methodology in optimizing freight transportation routes within Indian Railways. Through strategic planning and algorithmic application, this approach orchestrates the movement of multiple vehicles to serve customers efficiently while adhering to time constraints and optimizing route efficiency. The mathematical model has been thoroughly discussed along with the assumptions and notations used. The report mentions the references and code snippets towards the end for the viewers' reference.

Introduction
Indian Railways, a vast railway network, faces significant hurdles in optimizing transportation operations, particularly in route and schedule management for efficient goods delivery. This project delves into solving the Modified Vehicle Routing Problem (MVRP), customized for Indian Railways' freight logistics, with a key modification reflecting real-life railway scenarios.

Traditionally, in vehicle routing problems, the input and output nodes are typically the same. However, to better mimic real-world railway operations, we've introduced a distinction between the initial depot, where goods are loaded onto trains, and the final depot, where trains conclude their service after completing deliveries across the network. This adjustment mirrors the operational reality of railways, enhancing the problem's relevance to freight logistics.

Leveraging IBM CPLEX Solver, we've implemented an algorithm to tackle this intricate problem within Indian Railways' expansive network.

This report offers a detailed analysis of the MVRP for Indian Railways, encompassing problem formulation and algorithm implementation using IBM CPLEX. By confronting these challenges, Indian Railways can explore avenues to enhance operational efficiency and cost-effectiveness in managing freight logistics.

Problem Statement

For Indian Railways' freight operations, we address the Modified Vehicle Routing Problem (MVRP) to optimize freight transportation. The task involves visiting $n + 2$ nodes: the first and last nodes represent the initial and final depots, respectively, while the other n nodes are delivery destinations with specific demands and time windows

Each delivery destination requires a certain quantity of goods and specifies a preferable time window for delivery. Importantly, each station should be visited by only one train to meet its demand adequately.

Our objective is to devise an efficient routing and scheduling strategy for freight trains, ensuring timely delivery of goods and minimizing transportation costs. Through algorithmic solutions tailored to these constraints, we aim to enhance the efficiency of Indian Railways' freight logistics management.

Motivation

The Vehicle Routing Problem poses a crucial logistical challenge, optimizing vehicle allocation for multiple locations while considering constraints like time windows, capacities, and costs. Real-life examples, such as Indian Railways' vast network, underscore the significance of efficient routing. Indian Railways must navigate complex routing dilemmas, balancing passenger and freight needs while adapting to dynamic circumstances like festivals or disasters. Addressing the VRP is essential for operational efficiency and customer satisfaction, making this project's contributions to transportation logistics pivotal for optimizing routes and schedules and fostering more sustainable transportation systems.

Industrial Relevance

In a railway freight transportation setup, locomotives play a vital role in transporting goods from an initial point to a designated depot, with intermittent stops for loading or unloading cargo. The optimization of locomotive routes holds paramount importance in ensuring the prompt delivery of goods while keeping operational expenses in check. This optimization is indispensable for maintaining the stability and efficiency of the entire supply chain.

Model

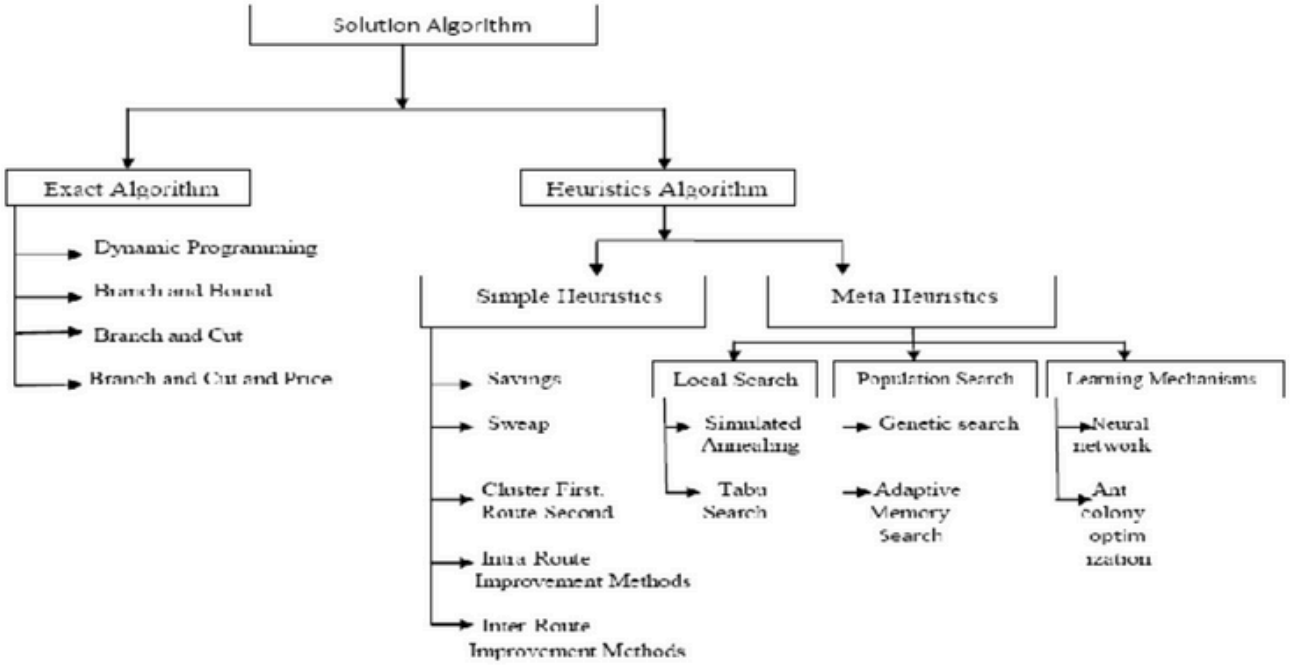
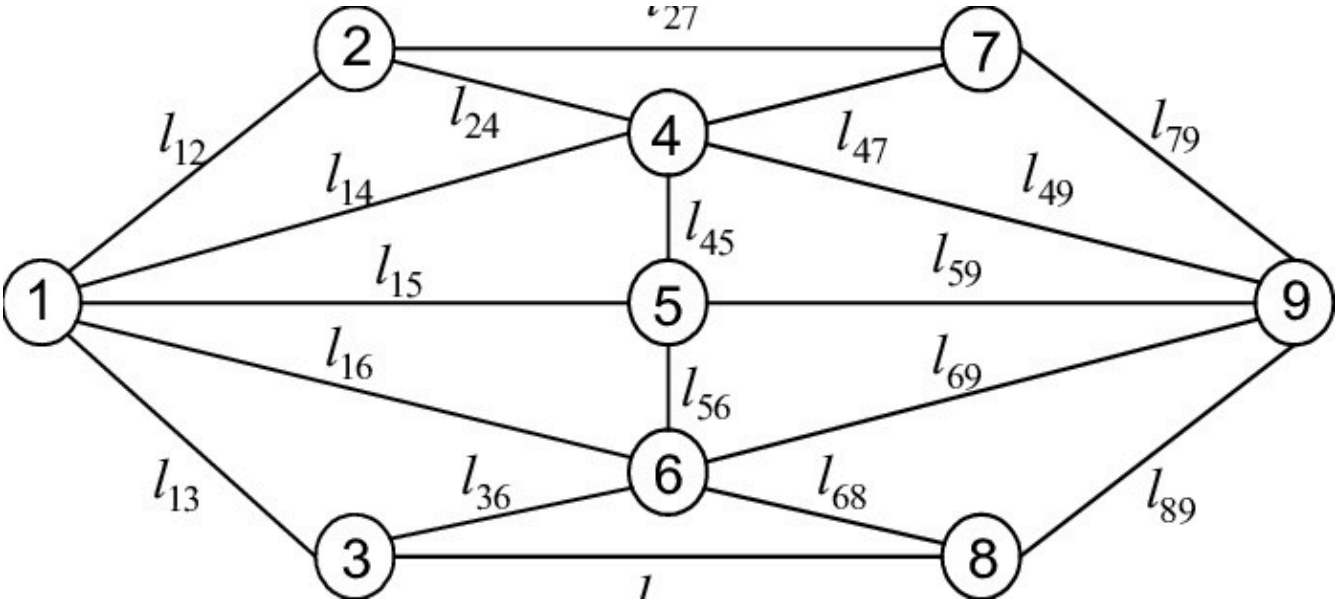


Fig 3: Solution algorithms for VRP and its variant

Ref: Solution Algorithm Strategies for VRP and its variants



Ref: Schematic Diagram for better understanding the mapping and freight routes

Assumptions

- Homogeneous Fleet: The vehicle fleet considered in this model is homogeneous in nature, i.e., the vehicles are identical in capacity, speed, mileage, etc.
- Customer Locations: Customers are assumed to be located at certain fixed locations or specified points in this model.
- Time Windows: It is considered that the customers can only be served during the specified time intervals.
- Travel Time: The time taken to travel from one location to another is fixed and constant.

Notations

- n -> Number of Dealers
- m -> Number of Transportation Vehicles
- $Cost_{ij}$ -> Cost Matrix
- $demand_i$ -> Demand Vector
- time_window -> Time Window
- $trav\_time_{ij}$ -> Travel Time between the Nodes i and j
- Vehicle_Capacity -> The Capacity of the Vehicles for Transportation

Model Parameters

- Number of Dealers
- Number of Transportation Vehicles
- Cost Matrix
- Demand Vector
- Time Window
- Travel Time between the Nodes
- Capacity of the Vehicles for Transportation

Decision Variable

The model has 2 decision variables:

- x_{ijk} = 1 if vehicle k drives directly from vertex i to vertex j
  = 0 otherwise
- u_{ik} denotes the time vehicle k starts to service customer i.

Mathematical Formulation

Objective Function

$$\text{Minimize} \quad \sum_{k=1}^{m} \sum_{i=0}^{n+1} \sum_{j=0}^{n+1} \text{Cost}_{ij} \cdot x_{ijk}$$

Constraints

Subject to

1) $\sum_{j=1}^{n+1} \sum_{k=1}^{m} x_{ijk} = 1$ $\quad \forall$ i = 1, 2,…., n i ≠ j

2) $u_{0k} = 0$ $\quad \forall$ k = 1, 2,...., m

3) $\sum_{i=1}^{n+1} x_{0ik} = 1$ $\quad \forall$ k = 1, 2,...., m

4) $\sum_{j=0}^{n} x_{j,\, n+1,\, k} = 1$ $\quad \forall$ k = 1, 2,...., m

5) $\sum_{j=0}^{n} x_{j0k} = 1$ $\quad \forall$ k = 1, 2,...., m

6) $\sum_{j=0}^{n} x_{n+1,\, j,\, k} = 1$ $\quad \forall$ k = 1, 2,...., m

7) $\sum_{i=0}^{n+1} (\text{demand}_i \cdot \sum_{j=0}^{n+1} x_{ijk}) \leq \text{Vehicle\_Capacity}$ $\quad \forall$ k = 1, 2,...., m

8) $x_{ijk} \cdot (u_{ik} + \text{trav\_time}_{ij} - u_{jk}) \leq 0$ $\quad \forall$ i = 1, 2,...., n+1; $\forall$ j = 1, 2,...., n+1; $\forall$ k = 1, 2,...., m

9) $\sum_{i=0}^{n} x_{ihk} = \sum_{i=1}^{n+1} x_{hik}$ $\quad \forall$ h = 1, 2,...., n; $\forall$ k = 1, 2,...., m

10) $\text{time\_window}_{i1} \leq u_{ik}$ $\quad \forall$ i = 0, 1, 2,...., n+1; $\forall$ k = 1, 2,...., m

11) $u_{ik} \leq \text{time\_window}_{i2}$ $\quad \forall$ i = 0, 1, 2,...., n+1; $\forall$ k = 1, 2,...., m

12) $x_{iik} = 0$ $\quad \forall$ i = 0, 1, 2,...., n+1; $\forall$ k = 1, 2,...., m

13) $x_{ijk} \geq 0$ $\quad \forall$ i = 0, 1, 2,...., n+1; $\forall$ j = 0, 1, 2,...., n+1

$\forall$ k = 1, 2,...., m

14) $x_{ijk} \leq 1$ $\forall$ i = 0, 1, 2,...., n+1
$\forall$ j = 0, 1, 2,...., n+1
$\forall$ k = 1, 2,...., m

System Configuration

- DEVICE SPECIFICATION
  - Processor: AMD Ryzen 7 5852U
  - RAM: 16 GB
  - System Type: 64-bit OS, x64 based processor

- WINDOWS SPECIFICATION
  - Edition: Windows 11 Home
  - Version: 23H2
  - OS Build: 22631.3296

Test case:

n=3;
m=3;
demand=[0,20,30,40,0];
cost=[[0,1,2,5,7],[1,0,9,4,11][2,9,0,1,1],[5,4,1,0,1],[0,0,0,0,0]];
time_window=[[0,10],[5,10],[10,40],[30,90],[60,85]];
trav_time=[[0,0,0,0,0],[1,0,1,3,3],[1,3,0,1,3],[1,1,3,0,1],[0,0,0,0,0]];
Vehicle_capacity=50;

Result

x = [[[0 0 0] [0 1 0] [0 0 1] [1 0 0] [0 0 0]] [[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 1 0]] [[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 1]] [[0 0 0] [0 0 0] [0 0 0] [0 0 0] [1 0 0]] [[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]]];
u = [[0 0 0] [5 10 5] [10 27 10] [71 67 30] [85 60 61]];

x[i][j][k] = 1 if vehicle k drives from vertex i to vertex j
u[i][k]: denotes the time vehicle k starts to service vertex i.

Discussion

- *Solution Approach*

  Constraint Programming was used through IBM CPLEX Solver, in which we formulated different constraints according to the problem statement and optimized the answer.

- *Conflict Resolution*

  Many different kinds of conflict arose during the formulation and execution of the code. Most of them were resolved by thorough debugging. However, some caused the solution to be infeasible, so we had to optimize and make certain assumptions.

Conclusion

In conclusion, the application of Modified-Vehicle Routing Problem with Time Windows (MVRP-TW) methodology for optimizing freight transportation routes signifies a significant advancement in enhancing efficiency and effectiveness within our logistics operations. Through strategic planning and implementation of MVRP-TW algorithms, we have successfully orchestrated the movement of multiple vehicles to serve our customers while adhering to time constraints and optimizing route efficiency.

The Report has successfully tackled the problem statement with future scope in increasing the model and enlarging the dataset under purview. The references mentioned below have added significant value to the process by suggesting optimizing methods and suitable test cases. The differentiating factor here is the presence of two depots, namely, the Initial Depot and the Destination Depot from the research papers which have considered single depots only. Similarly, all the codes and mathematical formulations have been adapted to suit the current situation of multiple depots, which makes the task at hand simpler.

This optimization endeavour has yielded substantial benefits, including minimized travel times and improved overall operational performance. By efficiently allocating resources and optimizing routes for multiple vehicles, we have streamlined our logistics processes, ensuring timely deliveries and enhancing customer satisfaction. In summary, the optimization of freight transportation routes using MVRP-TW represents a strategic investment in our company's future, ensuring that we remain agile, efficient, and sustainable in a rapidly evolving marketplace.

References

- **Research Papers**
  1. https://ieeexplore.ieee.org/abstract/document/5773510
  2. https://www.sciencedirect.com/science/article/pii/S0377221711010605
  3. https://www.researchgate.net/publication/282689785_Issues_in_Solving_Vehicle_Routing_Problem_with_Time_Window_and_its_Variants_using_Meta_heuristics_-_A_Survey

Appendix

Code Snippets

```
dexpr int z = sum(k in 1..m, i in 0..n+1, j in 0..n+1) (cost[i][j]*x[i][j][k]);
minimize z;         //The objective function minimizes the total cost of travel
```

```
/*********************************************
 * OPL 22.1.1.0 Model
 * Author: Aman Sinha
 * Creation Date: 01-Apr-2024 at 6:29:27 AM
 *********************************************/
using CP;

int n = ...;                        //  number of dealer's
int m = ...;                        //  number of transportation vehicles
int cost[0..n+1][0..n+1] = ...;
int demand[0..n+1] = ...;
int time_window[0..n+1][1..2] = ...;
int trav_time[0..n+1][0..n+1] = ...;
int Vehicle_capacity = ...;

dvar int x[0..n+1][0..n+1][1..m];
dvar int u[0..n+1][1..m];

dexpr int z = sum(k in 1..m, i in 0..n+1, j in 0..n+1) (cost[i][j]*x[i][j][k]);
minimize z;         //The objective function minimizes the total cost of travel

```

```
subject to{
  //This constraint ensures that each customer is visited exactly once
  constraint_1:
    forall(i in 1..n){
      (sum(j in 1..n+1,k in 1..m: i!=j)(x[i][j][k]))==1;
    }

  //This initializes the time of each vehicle with zero at the depot (node 0)
  constraint_2:
    forall(k in 1..m)
        u[0][k] == 0;


   constraint_3:
    forall(k in 1..m){
      sum(i in 1..n+1)(x[0][i][k]) == 1;
      sum(j in 0..n)(x[j][n+1][k]) == 1;
      sum(j in 0..n+1)(x[j][0][k])==0;
      sum(j in 0..n+1)(x[n+1][j][k])==0;
    }
```

```
    // This ensures that each vehicle is loaded
    constraint_4:
        forall(k in 1..m){
          (sum(i in 0..n+1)(demand[i])*((sum(j in 0..n+1)(x[i][j][k])))) <= Vehicle_capacity;
        }

    //establish the relationship between the vehicle departure time from a customer and its immediate
    constraint_5:
        forall(i in 0..n+1,j in 0..n+1, k in 1..m)
          x[i][j][k]*(u[i][k] + trav_time[i][j]-u[j][k]) <= 0;

    //after a vehicle arrives at a customer it has to leave for another destination
    constraint_6:
        forall(h in 1..n,k in 1..m){
          (sum(i in 0..n)(x[i][h][k])) == (sum(i in 1..n+1)(x[h][i][k]));
        }

    constraint_7:
        forall(i in 0..n+1, k in 1..m){
            time_window[i][1] <= u[i][k] <= time_window[i][2];
        }
```