DMS639:

ANALYTICS IN TRANSPORT AND TELECOM

Project Report

Site-Dependent Multi-Trip Periodic Vehicle Routing Problem

Group Members:

201036 Sweta Kumari
200108 Amit Jain
201090 Varun Gangwar
200315 Devang Kumawat
200351 Divyansh Gupta

Instructor:

Dr. Faiz Hamid

Acknowledgement:

We extend our sincere gratitude to Professor Dr. Faiz Hamid from the Department of Industrial and Management Engineering at the Indian Institute of Technology, Kanpur, for his invaluable support and guidance during the course of our project. His mentorship not only enriched our project but also broadened our understanding of the subject matter. We acknowledge and appreciate the guidance and support of all individuals who contributed to the realisation of this project. Their collective efforts have been instrumental in our academic and personal growth.

Summary:

This project report addresses the Site-Dependent Multi-Trip Periodic Vehicle Routing Problem, a complex routing scenario involving periodic deliveries and multiple trips per vehicle. In addition to traditional constraints, such as vehicle capacity and route optimization, the problem incorporates considerations of vehicle accessibility restrictions, where not all vehicles can access every customer. Through this report, we provide insights into the problem formulation, challenges. We then give the CPLEX code for the problem. Then we propose another heuristic algorithm called the tabu-search algorithm to solve the problem which can perform better than the branch-and-bound of CPLEX.

Introduction:

We went through various variations of the Vehicle Routing Problem (VRP) encountered in distribution systems. The classic VRP is where the objective is to minimise total distance travelled by a fleet of vehicles servicing a set of customers with known demands. It then extends to the Periodic Vehicle Routing Problem (PVRP), where deliveries must be made periodically over a fixed time frame, and the Site-Dependent Vehicle Routing Problem (SDVRP), where vehicles have compatibility constraints with certain customers. We then encountered the Multi-Trip Vehicle Routing Problem (MTVRP) where vehicles can be used for multiple routes over the planning period. Our problem is the Site-Dependent Multi-Trip Periodic Vehicle Routing Problem (SDMTPVRP), which combines characteristics of the aforementioned problems. It combines aspects of various VRP variants:

Site-Dependent Constraints:

- Not all vehicles can visit all customers due to compatibility restrictions.
- Vehicles must be assigned to customers based on compatibility relationships.

Multi-Trip Requirement:

- Vehicles can make multiple trips over a fixed planning period.
- Optimization must consider the potential for vehicles to be assigned to multiple routes.

Periodic Deliveries:

- Goods must be delivered to customers periodically within a set timeframe.

Objective:

- Minimise total distance travelled by the fleet of vehicles.
- Ensure all customer demand is met while respecting operational constraints.

Background:

The SDMTPVRP has direct relevance in industries where complex distribution patterns are commonplace. Examples include:

- Specialised Product Delivery: Deliveries of hazardous materials, oversized goods, or items requiring refrigeration often necessitate vehicles with specific capabilities, thus introducing the site-dependency aspect.
- Distribution in Urban and Rural Areas: Diverse accessibility constraints might restrict certain vehicles from servicing customers in congested urban centres with narrow streets or in rural areas with rough terrain.
- Last-Mile Delivery: Optimising last-mile delivery operations often involves maximising the number of deliveries completed by each vehicle through multi-trip planning.

Cordeau and Laporte (2001) were among the first to address a problem with similar characteristics to the SDMTPVRP within their work on the SDVRP. Existing metaheuristic algorithms designed for the PVRP, SDVRP, or related problems often serve as the basis for tackling the SDMTPVRP. The main challenge lies in adapting procedures to handle the combined complexities.

Work on metaheuristics, particularly tabu search, for VRPs has provided powerful tools adaptable to complex scenarios like the SDMTPVRP. The focus on practical constraints within the PVRP, SDVRP, and MTVRP has highlighted the need for solution methods addressing real-world distribution complexities. Benchmark test sets developed for the component problems lay the groundwork for the creation of standardised SDMTPVRP test instances to compare solution algorithms.

Methodology:

Notations:

n	Total number of customers
t	Number of days of the period
m	Number of available vehicles
w	Maximum number of daily routes per vehicle
C	Set of available delivery-day patterns
P	Set of available vehicle types
K	Set of available vehicles
$C_i \subseteq C$	Set of feasible delivery-day patterns for customer i
$P_i \subseteq P$	Set of feasible vehicle types for customer i
$K_i \subseteq K$	Set of feasible vehicles for customer i
q_i	Demand per service for customer i
Q_k	Capacity of vehicle k
D_k	Maximum operation time for vehicle k
c_k	Operation cost per unit of distance travelled for vehicle k
d_{ij}	Travel distance between customer i and customer j
t_{ij}	Travel time between customer i and customer j
ϕ_i	Unload time for customer i
μ_i	Load time at the depot for customer i

The SDMTPVRP is characterised by a multigraph G=(V,A), where:

- V=$\{v_0, v_1, \ldots, v_n\}$ represents the vertex set with v_0 as the depot and vertices V\$\{v_0\}$ as the customers .
- A= $\{(v_i, v_j)^{k,r,l}\}$ is the arc set where, k, r and l represent the vehicle, the route of that vehicle and the delivery day, respectively.

Each arc $(v_i, v_j)^{k,r,l}$ has a cost associated with it depending on arc length and vehicle.

Decision Variables:

x_{ij}^{krl}: Binary variable. Equals 1 if vehicle k, on route r, on day l, visits customer j immediately after customer i. Otherwise, it equals 0.

y_{ih}: Binary variable. Equals 1 if delivery pattern h is assigned to customer i. Otherwise, it equals 0 (where h belongs to the set Ci of feasible delivery patterns for customer i).

a_{hl}:Binary variable. Equals 1 if day l belongs to the delivery pattern h (zero otherwise).

The primary decision variables required:

Objective Function:

Minimise: Σ (Cost of travelling from i to j * Cost factor of vehicle k) * x_i_j_k_r
for all i, j (customers), k (vehicles), r (routes)

$$\text{minimize} \quad \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k\in K_i\cap K_j}\sum_{r=1}^{w}\sum_{l=1}^{t} d_{ij}c_k x_{ij}^{krl}$$

Constraints:

1. Delivery Pattern Assignment

$$\sum_{h\in C_i} y_{ih} = 1 \qquad \text{(for all customers i)}$$

Ensures exactly one feasible delivery pattern is assigned to each customer.

2. Delivery Day Compatibility

$$\sum_{j=0}^{n}\sum_{k\in K_i\cap K_j}\sum_{r=1}^{w} x_{ij}^{krl} - \sum_{h\in C_i} a_{hl}\, y_{ih} = 0 \qquad \text{(for all i, l)}$$

Guarantees that a customer is visited only on days belonging to their assigned delivery pattern (a_{hl} is 1 if day l is in pattern h).

3. Route Flow

$$\sum_{i=0}^{n} x_{ie}^{krl} - \sum_{j=0}^{n} x_{ej}^{krl} = 0$$ (for all vehicles k, routes r, days l, and nodes e)

Enforces that if a vehicle enters a customer node, it must also leave that node on the same route and day.

4. Depot Departure

$$\sum_{i=0}^{n} x_{0j}^{krl} \leqslant 1$$ (for all vehicles k, routes r, days l)

Limits each vehicle to at most one departure from the depot per route per day.

5. Vehicle Capacity

$$\sum_{i=0}^{n} \sum_{j=0}^{n} q_i x_{ij}^{krl} \leqslant Q_k$$ (for all vehicles k, routes r, days l)

Ensures the total demand served on a route doesn't exceed the vehicle's capacity.

6. Route Duration

$$\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{r=1}^{w} (\mu_i + t_{ij} + \phi_i) x_{ij}^{krl} \leqslant D_k$$ (for all vehicles k, routes r, days l)

Limits the total duration of a route (including loading, travel, and unloading times) to not exceed the vehicle's maximum operation time.

7. Subtour Elimination

$$\sum_{i,v_i \in S} \sum_{j,v_j \in S} x_{ij}^{krl} \leqslant |S| - 1$$ (for all vehicles k, routes r, days l, and subsets S of customers, where S does not include the depot and has at least two customers)

Prevents the formation of disconnected subtours.

8. Variable Domains

$x_{ij}^{krl} \in \{0, 1\}$ (for all i, j, k, r, l)

$y_{ih} \in \{0, 1\}$ (for all i, h ∈ Ci)

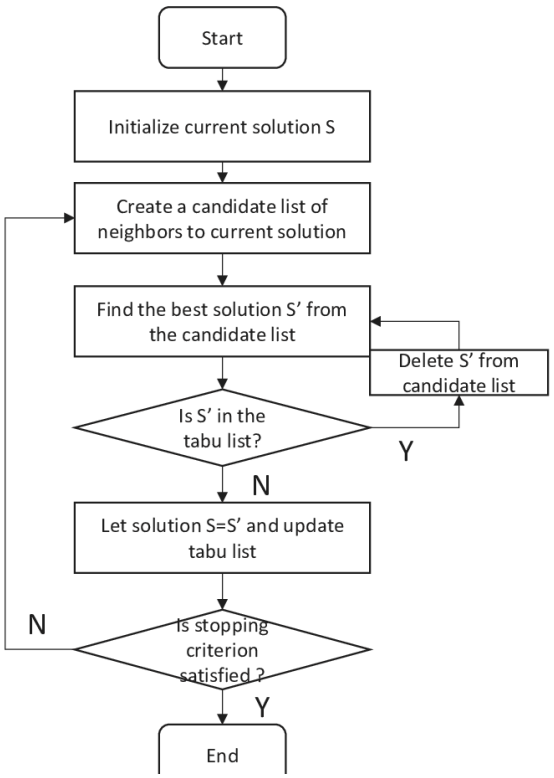
Enforces binary restrictions on the decision variables.

Efficient Algorithms:

The tabu-search algorithm:

Tabu search tackles the complex SDMTPVRP by strategically exploring different delivery plans. It examines neighbouring solutions (those with slight changes) while keeping a "tabu

list" of recent modifications to avoid getting stuck revisiting the same solutions. This memory mechanism, along with occasional overrides of restrictions, helps guide the search towards better routes and delivery schedules.



Initialization (Steps 1-3)

1. **Initial Solution:** Begin with a feasible initial solution s0. If no feasible solution is readily available, you might temporarily relax some constraints.

2. **Parameters and Data Structures:**
   - Notation:
     s : Current solution
     s* : Best solution found so far
     N(s): The neighbourhood of a solution (set of solutions reachable by modifying s)
     B(s): Set of attributes (customer, vehicle, route, day combinations) in a solution
     c(s): Cost of solution s
   - Initialize parameters:
     - α: Iteration counter
     - β: Counter for adaptive penalties
     - δ,γ,θ,η,p (parameters influencing tabu search behaviour)
   - Create data structures:
     - τikrl: Tabu tenure for attributes.
     - σikrl: Aspiration level (cost) for attributes.

3. **Current Solution:** Set the current solution to the initial solution s0.

Iterative Process (Step 4)

1. **Neighborhood Generation (a-b):**
   - **Empty Neighbourhood:** Initialise an empty set M(s) to store potential neighbouring solutions.
   - **Explore Neighbours:** Examine the neighbourhood N(s) of the current solution. Solutions in the neighbourhood are created by removing an attribute (customer, vehicle, route, day) from s and adding an attribute to s, ensuring feasibility.
   - **Filtering:** Add a neighbouring solution s to M(s) if it meets one of these criteria:

     Tabu Override: The attribute $\tau ikrl < \lambda$ (it hasn't been tabu for long enough).
     Aspiration Override: s is feasible and has a cost less than the stored aspiration level $\sigma ikrl$.

2. **Evaluation (c):**
   - For each solution s in the filtered neighbourhood M(s):

     Calculate objective function f(s). Consider these components:

     Total routing cost and penalty for violating capacity or route duration constraints (adaptive penalties explained later)

     Apply a weighting factor to favour solutions with either:

     Better objective function values than the current solution or lesser constraint violations than the current solution.

3. **Selection (d):** Identify the solution $s' \in M(s)$ with the best (lowest) weighted score calculated in the previous step.

4. **Route Adjustment (e):**

   Remove Attributes: For attributes present in s but absent in s' :
   - Use the GENIUS heuristic to remove the customer from the corresponding route.
   - Mark the attribute tabu ($\tau ikrl = \lambda + \theta$).

   Insert Attributes: For attributes present in s' but absent in s:
   - Use the GENIUS heuristic to insert the customer into the appropriate route.
   - Increment the frequency counter $\rho ikrl$.

5. **Update (f-h):**
   - **New Solution:** Let R be the updated set of routes. The solution associated with R is our new current solution s.
   - **Best Solution:** If s is feasible and better than the stored best solution s*, replace s* with s.
   - **Aspiration Levels:** For each attribute in s, update the aspiration level $\sigma ikrl$ (consider keeping the lower cost).
   - **Adaptive Penalties:** Adjust penalty factors (for capacity and time violations) based on the feasibility of the current solution s.

Repeat Steps 4-5 for η iterations

Parameter Explanation

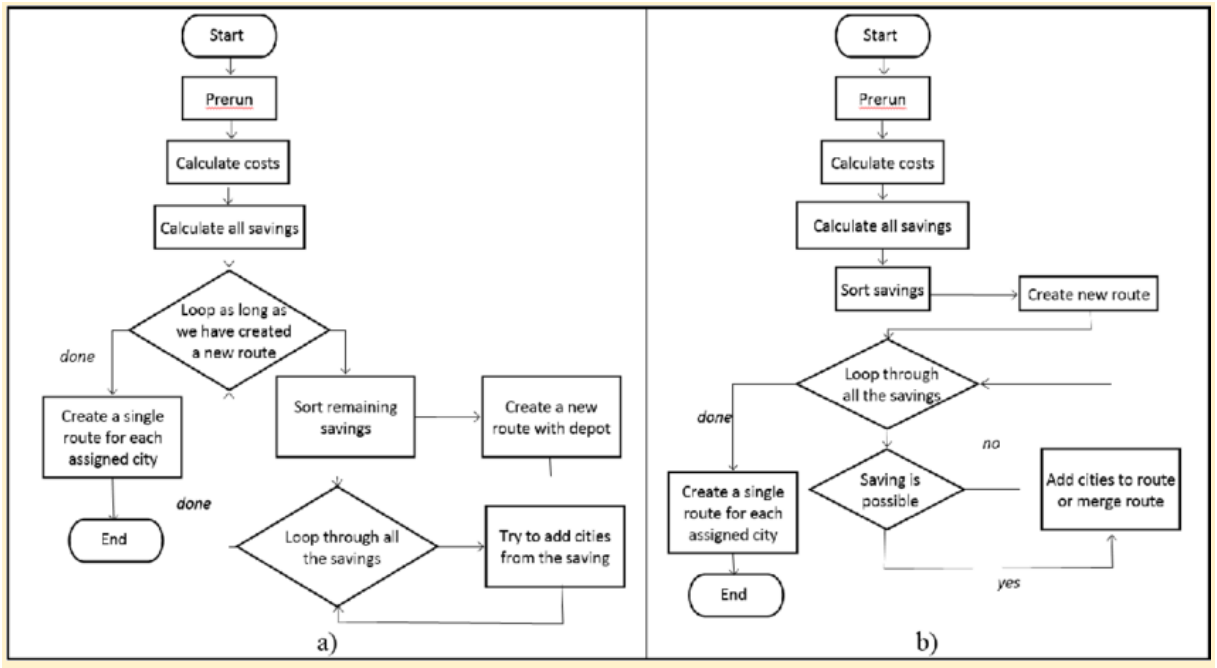
δ, γ, θ: Control tabu restrictions and aspiration criteria.
η: Determines the number of iterations.

p: Influences the neighbourhood size in the GENIUS heuristic.

Computational Complexity

- Dominated by route modification in the GENIUS heuristic.
- Influenced by problem size, feasible patterns, and the GENIUS neighbourhood size..

Clarke and Wright Savings Algorithm (Modified):



Core Modifications:

- Feasibility Checks: The standard Clarke and Wright algorithm primarily checks if merging routes violates capacity constraints. For the SDMTPVRP, we'll need to introduce additional checks:
- Delivery Patterns: Ensure that the combined route satisfies the delivery day requirements of both customers.
- Site-Dependency: Verify that the vehicle type used on the route is compatible with both of the customers being merged.
- Multi-Trip Capacity: If merging routes allows a vehicle to make multiple trips within the planning period, recalculate capacity constraints accordingly.

Savings Calculation:

We could refine the savings calculation to further align it with the SDMTPVRP's complexities:

- Delivery Frequency: If customers have different delivery frequencies, factor that into cost savings. Merging routes that align in frequency might lead to longer-term savings.
- Vehicle Compatibility: If there are significant cost differences between vehicle types, consider favouring merges where customers are compatible with the same, less

expensive vehicle type.

Modified Algorithm Steps:

Compute Savings:

Same as in the classic algorithm, ensuring distance/cost calculations are compatible with your SDMTPVRP instance's parameters. For each pair of customers (i,j)∈V where i!=j and i!=0, and j!=0, calculate the savings: s(i,j)=$c_{i0k} + c_{0jk} - t * c_{ijk}$.

Select a vehicle type k∈Pi ∩ Pj in a way that considers compatibility and potential cost differences across vehicle types (more on this later).

Sort Savings in descending order

Route Merging:

- Start with each customer on an individual route.
- Iteratively examine potential merges from the sorted savings list.
- Perform feasibility checks:
    - Capacity and Time Constraints: As in the basic algorithm.
    - Delivery Patterns: Can both customers be serviced on the required days on the combined route?
    - Site-Dependency: Is there a vehicle type compatible with both customers that can be assigned to the route?
    - Multi-Trip Capacity: Recalculate if multiple trips become feasible.
- If merging is feasible, add the customers to a new route or modify an existing one.
- If the merge requires a more expensive vehicle type or changes delivery patterns, consider adjusting the savings value associated with that merge.

Termination: Stop when no more feasible merges exist.

The modified Clarke & Wright Savings Algorithm might best serve to generate a good initial solution for further optimization. We can pair it with metaheuristics (like Tabu Search) to explore a wider search space.

Route-First Cluster-Second Heuristics:

These approaches have two distinct phases:

- Route-First: Design routes that might temporarily violate capacity or time constraints.
- Cluster-Second: Assign customers to routes while respecting feasibility. Compatibility restrictions and delivery patterns are crucial here.

Metaheuristic Approaches:

Simulated Annealing: Inspired by metallurgical processes, it allows temporary acceptance of worse solutions to escape local optima. Parameters require careful tuning for SDMTPVRP's complexity.

Genetic Algorithms: Model solutions as chromosomes and evolve them over iterations. Ensure your solution representation and crossover/mutation operators consider SDMTPVRP's unique constraints.

Hybrid Approaches:

Combining Construction Heuristics and Metaheuristics:

- Use a construction heuristic to generate a good initial solution.
- Apply a metaheuristic to iteratively refine the solution.

Large Neighbourhood Search (LNS):

- Iteratively destroy a part of the current solution using a removal heuristic.
- Rebuild the destroyed section using a construction heuristic.
- LNS can be very effective in exploring diverse areas of the search space.

Conclusion:

The Site-Dependent Multi-Trip Periodic Vehicle Routing Problem (SDMTPVRP) models a complex but highly practical distribution challenge involving customer-specific delivery patterns, vehicle compatibility restrictions, and the potential for multi-trip routes. As finding optimal solutions can be computationally demanding, heuristic algorithms play a vital role. We discussed the basic CPLEX implementation and construction heuristics like the Clarke & Wright Savings Algorithm, powerful metaheuristics such as Tabu Search, and the potential benefits of hybrid approaches. Selecting the best heuristic depends on the SDMTPVRP instance's size, complexity, and your time and solution quality goals.

References:

Home Page J E Beasley (brunel.ac.uk)
https://www.researchgate.net/publication/237899195_A_tabu_search_algorithm_for_the_periodic_vehicle_routing_problem_with_multiple_vehicle_trips_and_accessibility_restrictions
https://link.springer.com/article/10.1057/jors.2009.176
https://www.sciencedirect.com/science/article/pii/S0305054823000539
https://pubsonline.informs.org/doi/abs/10.1287/inte.20.4.74
https://link.springer.com/chapter/10.1007/0-387-28356-0_6
Clark-Wright Algorithm