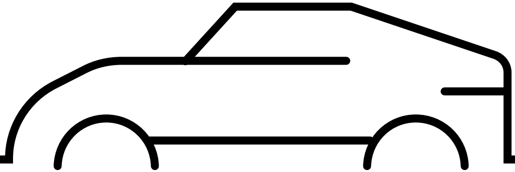
Vehicle Routing Problem

Efficiently optimizing the path to deliver goods



João Marinho
Rodrigo Tuna
Constraint Logic Programming
M.EIC - FEUP

Problem Recap (VRP)

- The VRP is a classical optimization problem, problem that seeks to find the optimal set of routes for a fleet of vehicles to visit a set of locations (clients) minimizing the distance travelled.

- Introduced by Dantzig et al. in 1959 [1] as a generalization of the Travelling Salesman Problem.

- It is a NP hard problem and so several techniques can be used to solve it, optimization methods, heuristic methods.

- Several variants of the VRP exist 2 of which will be explored.

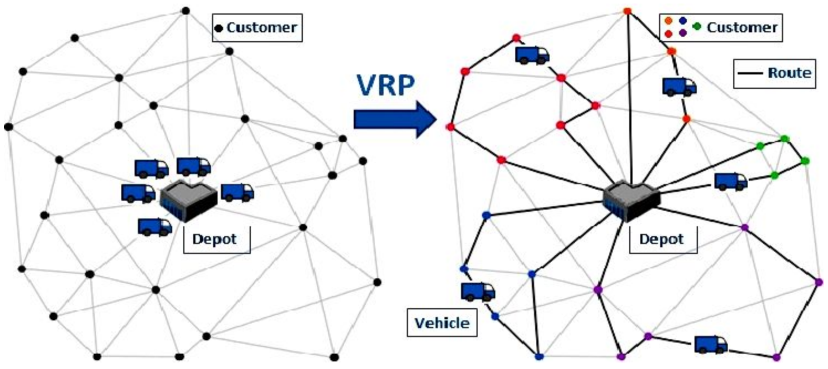


Figure 1 - Classical VRP [2]

Problem Recap (MDVRPTW)

1 The vehicles can depart from any of the depots.

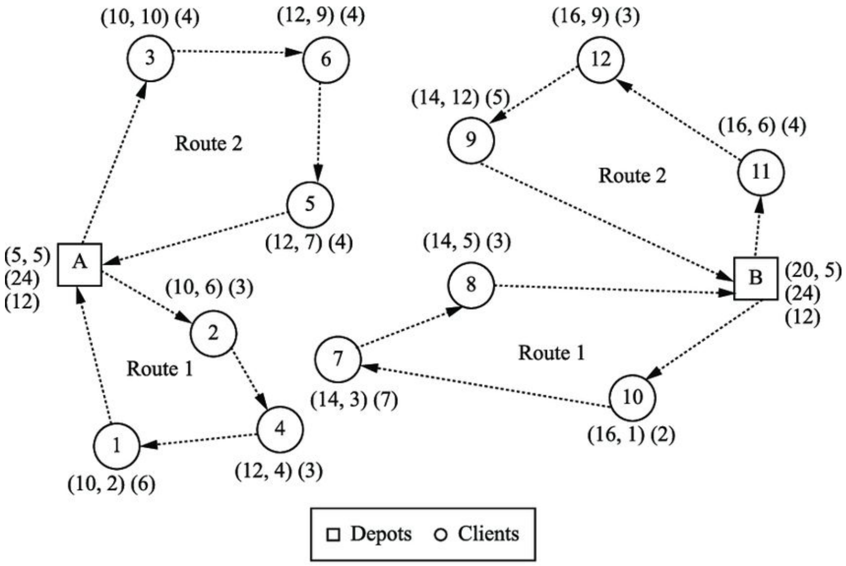


Figure 2 - Example of MDVRP [3]

2 Each client i, is associated with an interval $[a_i, b_i]$ (*time window*) that specifies that a vehicle should not arrive at i after b_i.

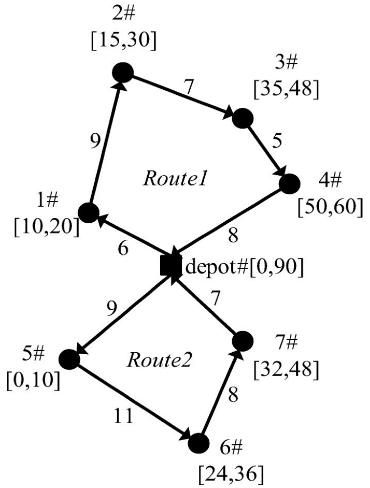


Figure 3 - Example of VRPTW[4]

Problem Definition

Vehicle Capacity

Vehicle Route Duration

Homogeneous Fleet

Time Windows

No Traffic Congestion

Optimal Solution

$\text{Cost} = \Sigma \; \text{route\_cost}_i \,, \; i \in \text{list of vehicles}$

CP Model - Domain Variables

Add new depot set D’ containing a copy of each depot for each vehicle in the depot.

Notation	Explanation
$$Next_i \in D' \cup C,\ \forall i \in D' \cup C \quad (1)$$	- (1) Next node to visit
$$A_i \in [0, T] \quad (2)$$	- (2) Arrival Time
$$W_i \in [0, T] \quad (3)$$	- (3) Waiting Time (Only for TW)

CP Model - Constraints

Notation		Explanation
$Next_i \in \{i\} \cup C, \ \forall i \in D'$	(4)	- (4) No inter-depot routing
$AllDifferent(Next_i \mid i \in D' \cup C)$	(5)	- (5) Not visit node more than once
$A_{Next_i} = Time(i, Next_i), \ \forall i \in D'$	(6)	- (6) Define Arrival time for first node
$A_{Next_i} = A_i + W_i + ServTime(i) + Time(i, Next_i), \ \forall i \in C$	(7)	- (7) Define Arrival time for next nodes
$A_i + W_i \geq Open(i)$	(8)	- (8,9) Time Window constraints
$A_i \leq Close(i)$	(9)	

CP Model - Target

Minimize costs of all trips.

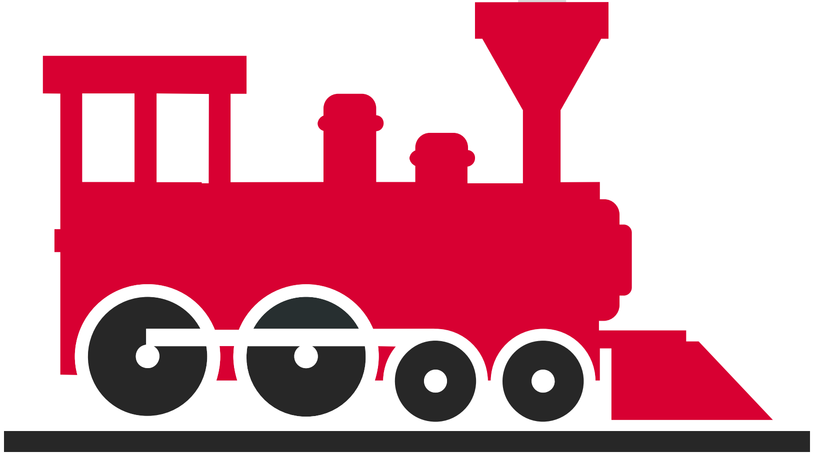
$$\sum_{i \in D' \cup C} Cost(i, Next_i)$$

Datasets

MDVRP Dataset [5]

VRPTW Dataset [5]

MDVRPTW Dataset [5]

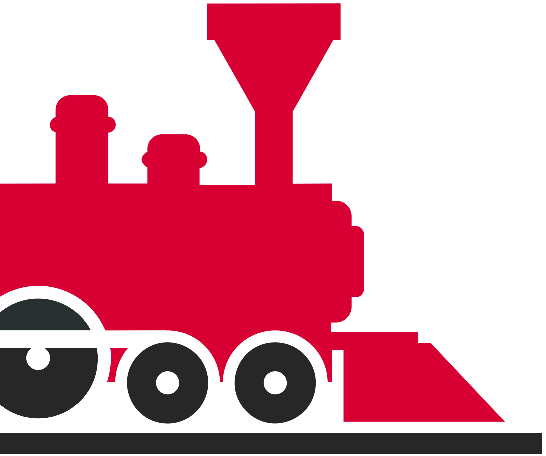


Datasets - Structure

1. type #vehicles #customers #depots
2. vehicle_details (max duration & load)

#customers + #depots lines:

3. id x y service_time demand (+ discarded information)



Python Implementation

Dimensions

- Time (Our Cost)

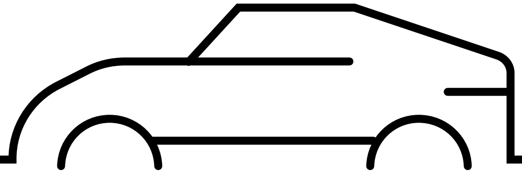
    - Distance(i,j) / Speed + ServiceTime(i)
    - 0 to maxRouteTime
    - Slack time = maxRouteTime for Time Window problems

- Capacity
    - Demand(i)
    - 0 to MaxCapacity
    - No slack

Constraints

- Start and End Depots

- Add Time Window to each Client



Prolog Implementation

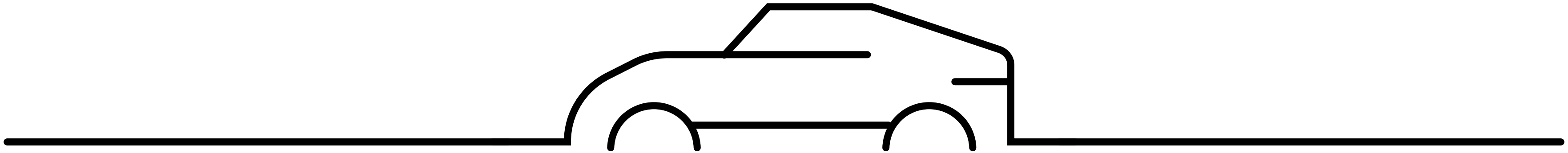
Variables

- Routes
  - MaterializedRoutes
- LeaveTimes
- TotalRouteTimes
  - TotalTime

Constraints

Depot, customer, demand, time window and total time

- subcircuit(R_i), $\forall i \in$ Routes
- lex_chain(R_Set$_i$), $\forall i \in$ Routes Per Depot
- Next_Start_Work #= max(Arrive_Time, Open_Time)
- Close_Time #>= Arrive_Time
- Next_Leave_Time #= Next_Start_Work + Service_Time
- sum(Materialized_Customer, #=, 1)
- scalar_product(Demands, Materialized_Route, #=<, Max_Demand)
- Total_Route_Time #=< Max_Route_Time



Or-Tools Experimental Setup

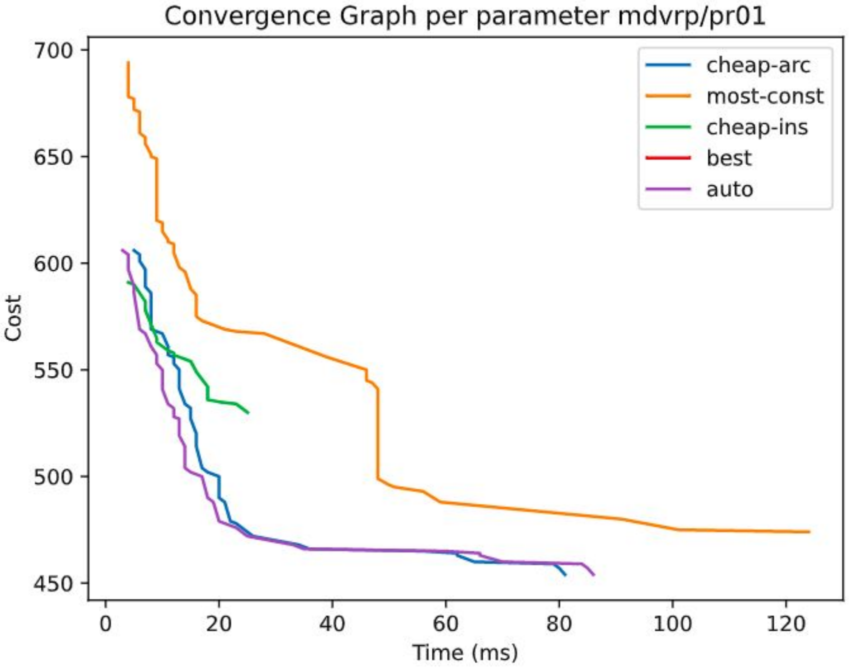
Parametres

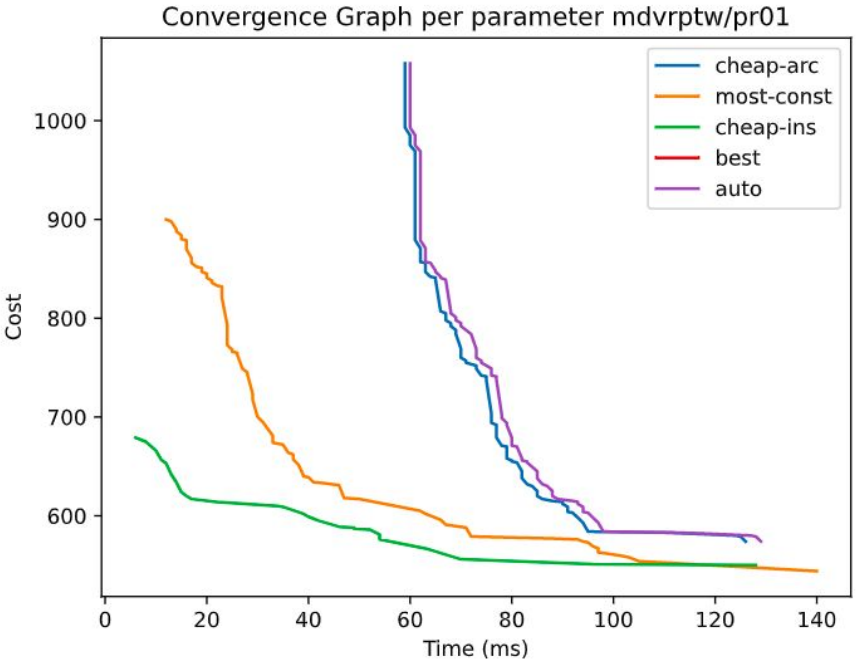
- AUTOMATIC
- PATH_CHEAPEST_ARC
- PATH_MOST_CONSTRAINED_ARC
- BEST_INSERTION
- LOCAL_ChEAPEST_INSERTION

Time

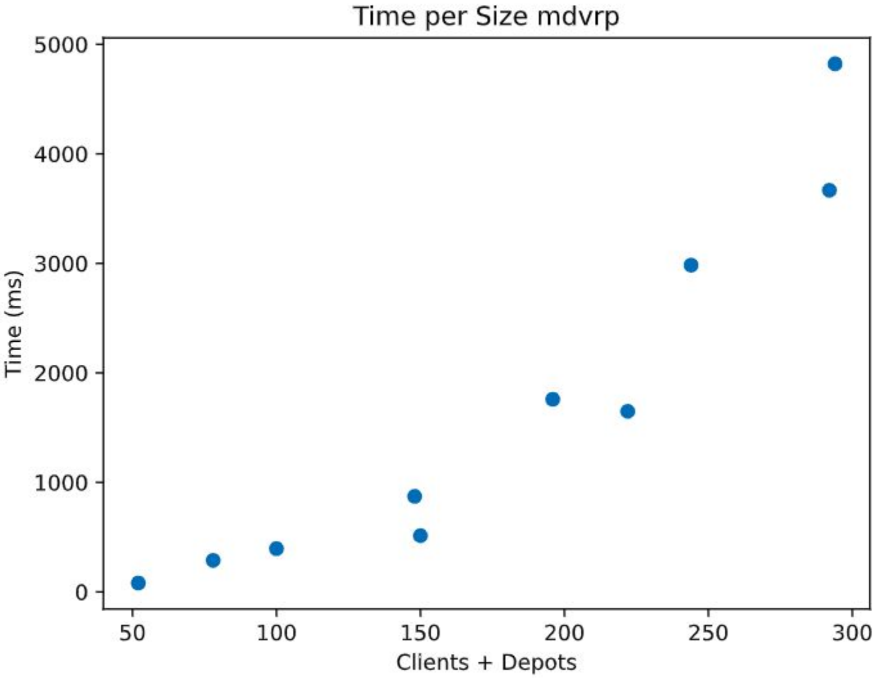
- Timeout of 20 minutes
- Collect all solutions

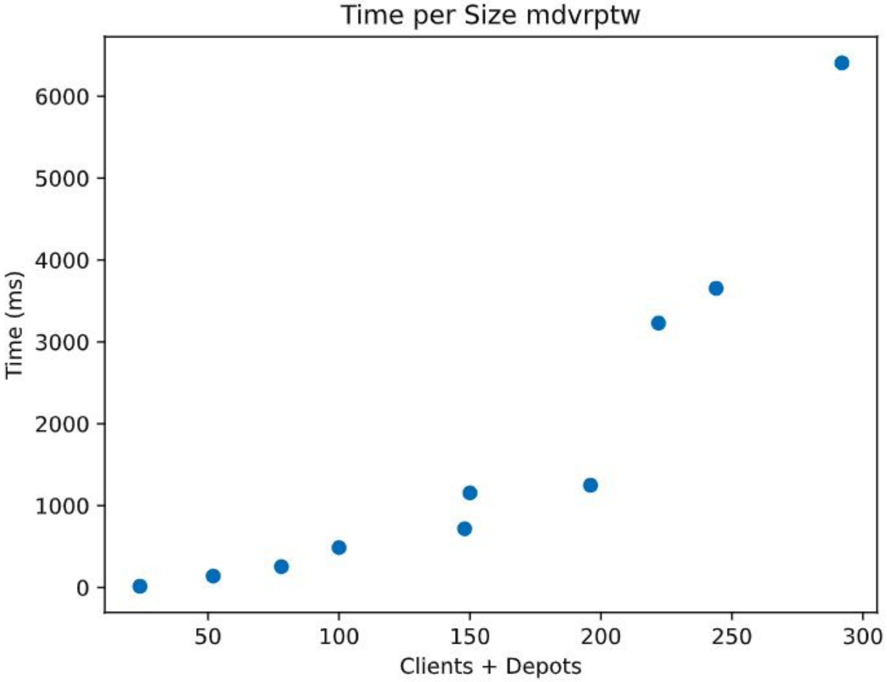
Results - Or-Tools - Parameters





Results - Or-Tools - Size





Prolog Experimental Setup

01 Search Variation

Test all possible variations of variable ordering, value selection and ordering …

02 Reduced Variations

Variable ord. - leftmost, ffc.
Value select. - step, bisect.
Value ord. - up, down.

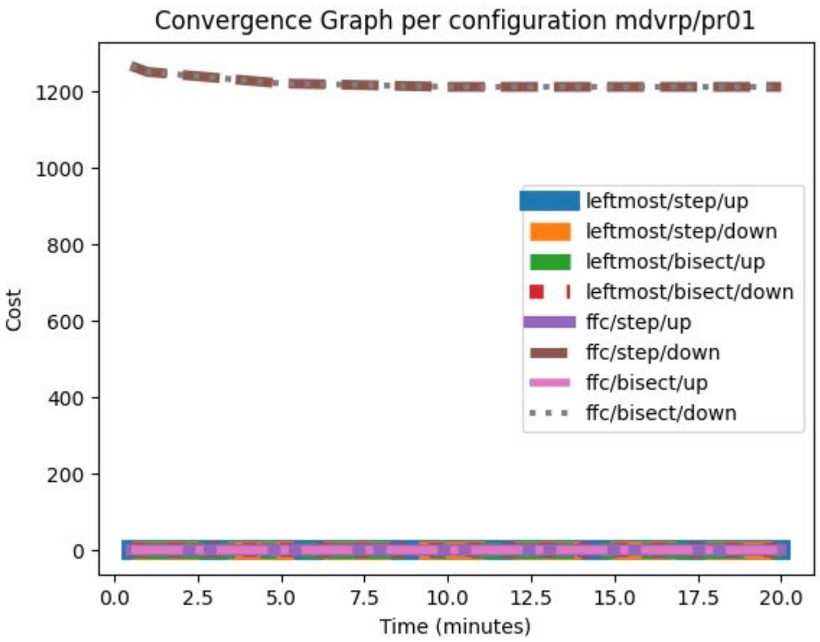
03 Time Limits

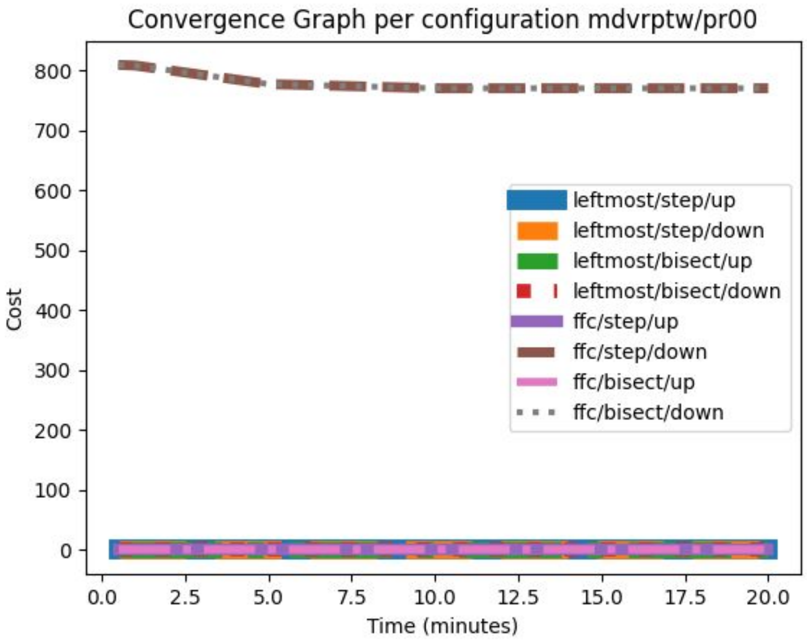
30 seconds
1, 5, 10, 20 minutes

04 Obtain results

Run the solver for each problem with all 8 variations with time limits.

Results - Prolog

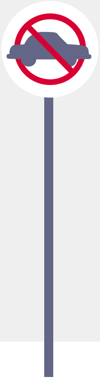




Results - Comparison

Solution*

Or-Tools solution is on average 2.64 times better than Prolog



Time*

Or-tools achieves the best solution 23703.70 times faster than Prolog

*For problems where Prolog found a solution (6%)

Conclusion

MDVRP

VRPTW

MDVRPTW

+

Prolog

OR Tools

Bibliography

[1] - G. B. Dantzig, J. H. Ramser, (1959) The Truck Dispatching Problem. Management Science 6(1):80-91.
[2] - Gupta, Ashima & Saini, Sanjay. (2017). An Enhanced Ant Colony Optimization Algorithm for Vehicle Routing Problem with Time Windows. 267-274. 10.1109/ICoAC.2017.8441175.
[3] - Silva Junior, Orivalde & Lopes, Luiz & Bergmann, Ulf. (2011). A Free Geographic Information System as a Tool for Multi-Depot Vehicle Routing. Brazilian Journal of Operations & Production Management. 8. 103-120. 10.4322/bjopm.2011.006.
[4] - Feng, B., Wei, L. An improved multi-directional local search algorithm for vehicle routing problem with time windows and route balance. Appl Intell (2022).
[5] - http://neumann.hec.ca/chairedistributique/data