Rolling Horizon based Temporal Decomposition for the Offline Pickup and Delivery Problem with Time Windows

Youngseo Kim[1], Danushka Edirimanna[1], Michael Wilbur[2], Philip Pugliese[3], Aron Laszka[4], Abhishek Dubey[2], Samitha Samaranayake[1]

[1]Cornell University, [2]Vanderbilt University, [3]Chattanooga Area Regional Transportation Authority, [4]Pennsylvania State University

December 29, 2022

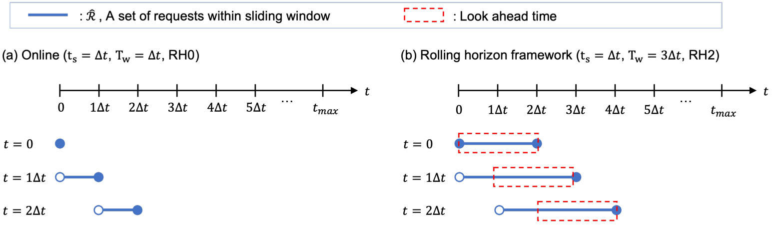
Background

Pickup and delivery problem with time windows (PDPTW)

- PDPTW problem is proven to be very challenging computationally, i.e., NP-hard.
- Due to the complexity of the problem, practical problem instances can be solved only via heuristics.
- One common strategy is problem decomposition, i.e., the reduction of a large-scale problem into a collection of smaller sub-problems.

Rolling Horizon Framework

- We utilize a state-of-the-art online solver along with the rolling horizon optimization framework.
- A sliding window moves forward in time after each iteration, and keeps some overlap with the previous window.
- The overlapped window allows the parts of the route to be rescheduled

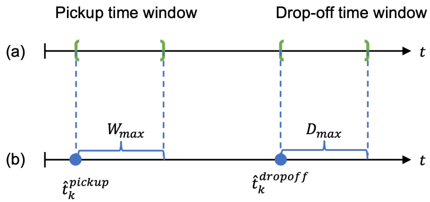


Problem Definition

Offline PDPTW

Given:

- Network, Requests $r_k \in \mathcal{R}$, Vehicles $v_j \in \mathcal{V}$
- Maximum waiting time W_{max} and maximum delay time D_{max}



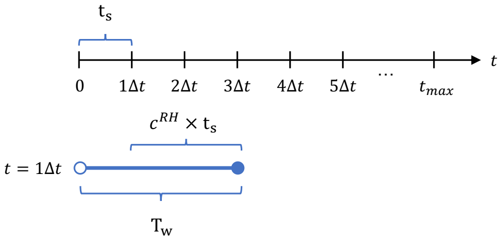
Find:

- $(v_j, [r_1, \ldots, r_k]) \in \mathit{Assignments}$, a set of tuples of a vehicle and a list of scheduled visits by order

Problem Definition

Hyperparameters

- Window size T_w and step size t_s.
- Rolling horizon factor c^{RH} is a factor for look ahead time.



- Adjusting window size and step size provides a better trade-off between solution quality and compute time.

Methodology

State-of-the-art Online Solver

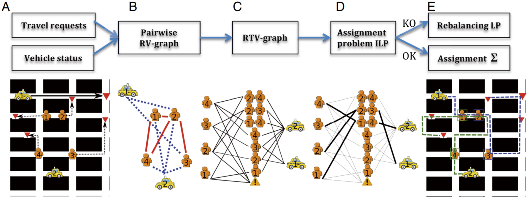
Challenges

- It is important to introduce a high performance PDPTW solver for subproblems.
- The subproblem is still NP-hard.

Techniques to solve the problem in polynomial time

- **Decomposition**: decoupling the NP-hard problem into routing and matching problems using RTV graph
- **Pruning**: feasibility constraints significantly reducing eligible matching pairs
- **Heuristics**: exhaustive search up to four passengers and insertion algorithm if more than four

Request Trip Vehicle (RTV) graph [AMSW+17]



Integer Linear Programming (ILP)

$$\underset{\epsilon_{ij},\chi_k}{\text{argmin}} \quad \sum_{\{ij:\epsilon_{ij}\in\mathcal{E}_{TV}\}} c_{ij}\epsilon_{ij} + \sum_{k\in\mathcal{R}} c_k\chi_k \tag{1}$$

$$\text{s.t.} \quad \sum_{\{i:T_i\in\mathcal{T}\}} \epsilon_{ij} \leq 1 \qquad ,\forall v_j \in \mathcal{V} \tag{2}$$

$$\sum_{\{i:T_i\in\mathcal{T}\}} \sum_{\{j:V_j\in\mathcal{V}\}} \epsilon_{ij} + \chi_k = 1 \qquad ,\forall r_k \in \mathcal{R} \tag{3}$$

Experimental design

Benchmark solvers

Google OR-Tools [PF22]

- A well-established, modern, and publicly available VRP solver developed by Google.
- Guided local search (GLS) which is known as the best performing setting for the OR-Tools PDPTW solver.

A modified Lin-Kernighan-Helsgaun heuristic (LKH3) [Hel17]

- LKH3 is the state-of-the-art solver to solve TSP and its variants.
- Among 322 PDPTW benchmark instances, LKH3 finds equal to or better than the best-known solutions in 319 instances.

Experimental design

Parameter settings

Table 1: Parameter settings for real-world dataset

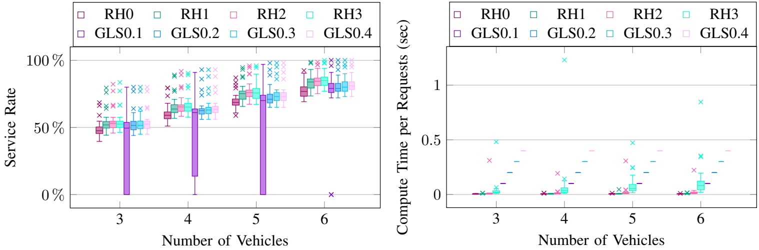
Parameter	**Values**
Vehicle capacity	8
Maximum waiting time	30 (min)
Maximum delay time	30 (min)
Dwell time	5 (min)

Table 2: Parameter for Chattanooga and New York City dataset

Parameter	**Chattanooga**	**NYC (small)**	**NYC (large)**
Data	Full data	1% sampled	20% sampled
Fleet size (M)	3, 4, 5, 6	3, 4, 5, 6	40
Step size	15 (min)	5 (min)	5 (min)
RH factor	0,1,2,3	0,2,4,6	0,1,2

Results

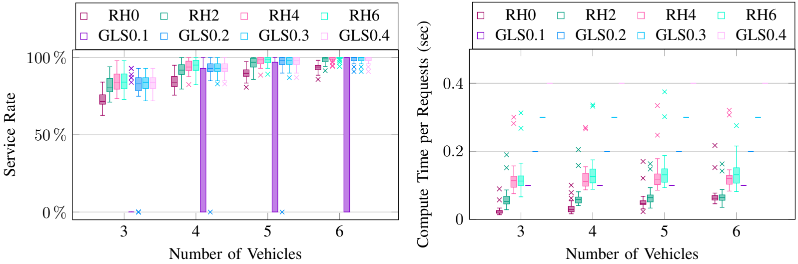
Chattanooga



RH* denotes the solutions with the * indicating the rolling horizon factor.
GLS* denotes the guided local search solutions with the * indicating time limit.
Number of requests: an average of 172 with standard deviation of 33

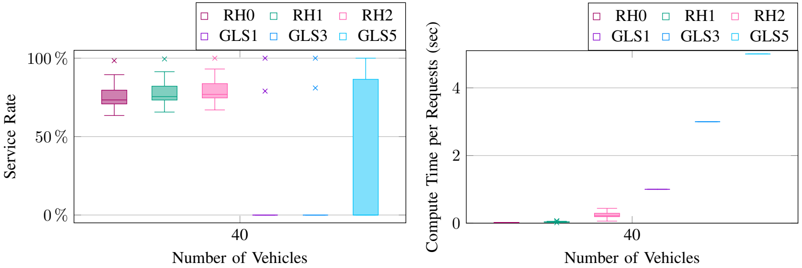
Results

NYC (small)



Number of requests: an average of 129 with standard deviation of 29

- Performance of the RH is as good as that of GLS.



Number of requests: an average of 2587 with standard deviation of 570

RH2 achieves a service rate of 79.0% in average and 100% maximum service rate, within 1 second per request.

GLS1 and 3 cannot get any feasible solution in 29 among 31 instances.

- The performance of GLS is insufficient in practice because operators need to obtain a schedule for the next day within a couple of hours.

Results

Benchmark instance

Instance	LKH3		Rolling horizon Framework						Gap (%)
	VMT[a]	Compute time[b]	Compute time[b]		Service rate (%)		VMT[a]		
			$T_w = 5$	$T_w = 10$	$T_w = 5$	$T_w = 10$	$T_w = 5$	$T_w = 10$	
lc101	**997**	**12.05**	0.42	**0.44**	100	**100**	1095	**1127**	13.00
lc105	**1011**	**15.92**	0.50	**0.50**	98	**100**	1116	**1140**	12.73
lc106	**1032**	**22.81**	0.49	**0.50**	100	**100**	1172	**1163**	12.69
lc107	**1021**	**18.51**	0.49	**0.60**	98.03	**100**	1165	**1085**	6.23
lc108	**1030**	**18.81**	0.50	**0.52**	96.15	**100**	1209	**1120**	8.74
lc201	**1779**	**50.85**	0.39	**0.40**	100	**100**	1981	**2021**	13.57

Table 3: Comparison of LKH3 and Rolling Horizon Framework

[a] Abbreviation for vehicle miles traveled. The unit of the compute time is second.

Conclusion

1. In this paper, we introduce a new temporal decomposition scheme to solve the PDPTW.
2. Rolling horizon framework provides better trade-off between solution quality and compute time.
3. We showcase the performance and scalability of the rolling horizon framework in different networks with different demand profiles.

References I

Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus.
On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment.
Proceedings of the National Academy of Sciences, 114(3):462–467, 2017.

Keld Helsgaun.
Lkh3.
`http://webhotel4.ruc.dk/~keld/research/LKH-3/`, 2017.
[Online; accessed 15-Dec-2022].

Laurent Perron and Vincent Furnon.
Or-tools v9.4.
`https://developers.google.com/optimization/routing/vrp`, 2022.
[Online; accessed 26-Oct-2022].