# Iterative sample scenario planning for the dynamic dispatch waves problem

Leon Lan[1*] Jasper van Doorn[1]
Niels Wouda[2] Arpan Rijal[2] Sandjai Bhulai[1]

[1] Vrije Universiteit Amsterdam, The Netherlands
[2] Rijksuniversiteit Groningen, The Netherlands

* Presenting author

July 25, 2023

# Key points of this talk

- We formulate the *dynamic dispatch waves problem* introduced during the EURO meets NeurIPS 2022 vehicle routing competition.
- We present a sample scenario planning method to solve the problem.
- Our results show that the method is scalable and overcomes the large computational efforts often associated with sampling-based methods.

# Outline of this talk

# Same-day delivery

- Rapid growth of e-commerce with *same-day delivery services*.
- Offered by e-commerce companies to enhance customer satisfaction, e.g., Amazon, Instacart, Walmart, and Google.
- Same-day delivery problems are dynamic vehicle routing problems with *stochastic requests*.
- 90% of the studies on same-day delivery problems have been published in the last five years.[1]

1 J. Zhang and T. V. Woensel (2023). "Dynamic Vehicle Routing with Random Requests: A Literature Review". en. In: *International Journal of Production Economics* 256, p. 108751

# Dynamic dispatch waves problem

- We study a same-day delivery problem named the *dynamic dispatch waves problem*[2] (DDWP) introduced in the *EURO meets NeurIPS 2022* vehicle routing competition.[3]
- In the DDWP, vehicles are dispatched at fixed decision moments to deliver goods.
- All delivery requests must be served within their requested time window, and we assume that a sufficient number of vehicles is available to serve all requests on time.
- The goal is to minimize the total distance traveled.

2 M. A. Klapp, A. L. Erera, and A. Toriello (2018). "The Dynamic Dispatch Waves Problem for Same-Day Delivery". en. In: *European Journal of Operational Research* 271.2, pp. 519–534

3 W. Kool, D. Numeroso, R. Reijnen, R. R. Afshar, T. Catshoek, K. Tierney, E. Uchoa, and J. Gromicho (2022). *EURO Meets NeurIPS 2022 Vehicle Routing Competition*.

# Problem formulation

- Time horizon of $H$ units divided into $\mathcal{T} = \{1, \ldots, |\mathcal{T}|\}$ epochs.
- Each epoch $t \in \mathcal{T}$ starts at time $T_t \geq T_0 = 0$, with $T_t > T_{t-1}$.
- At the start of epoch $t \in \mathcal{T}$, a set of requests $\omega_t$ with known support $\Omega_t$ is revealed.
- A request $n$ has a location, a demand, a hard time window $[e_n, l_n]$, and a release time $r_n = T_t$.
- Unlimited fleet of vehicles available, each with capacity $Q$.

# Problem formulation

- In each decision epoch, decide which of the known requests to *dispatch*, and how to route them, and which ones to *postpone* to later epochs.
- Requests that cannot be postponed to the next epoch must be dispatched in the current epoch, e.g., a request that has a time window $[T_t, x]$ with $x < T_{t+1}$.
- For the dispatched requests: solve a VRPTW with release times (VRPTW-RT) with earliest departure time $T_t$.
- **Goal**: deliver all requests within their time windows at minimum traveling distance.

# Markov decision process

- Let $\mathcal{N}_t$ denote the set of known requests at epoch $t$. Define the *state* $s_t$ as

$$s_t = \{n \in \mathcal{N}_t \mid \text{request } n \text{ not yet dispatched}\}.$$

- Let $m_t \subseteq s_t$ denote the set of *must-dispatch* requests. Define the action space $\mathcal{A}(s_t)$ as

$$\mathcal{A}(s_t) = \{a_t \subseteq s_t \mid m_t \subseteq a_t\}.$$

- The direct cost $C(s_t, a_t)$ is given by the cost of routing all dispatched requests $a_t$, i.e., the optimal cost of the VRPTW-RT with departure time $T_t$.

# Markov decision process

- The transition to the next state $s_{t+1}$ is given by

$$s_{t+1} = (s_t \setminus a_t) \cup \omega_{t+1}.$$

- The objective of the DDWP is to select for each epoch $t \in \mathcal{T}$ a minimum cost action, such that the following (Bellman) optimality conditions are satisfied:

$$V(s_t) = \begin{cases} C(s_t, s_t) & \text{if } t = |\mathcal{T}|, \\ \min\limits_{a \in \mathcal{A}(s_t)} \left[ C(s_t, a) + \mathbb{E}_{\omega_{t+1}}[V(s_{t+1})] \right] & \text{otherwise.} \end{cases} \tag{1}$$

# Sample scenario planning

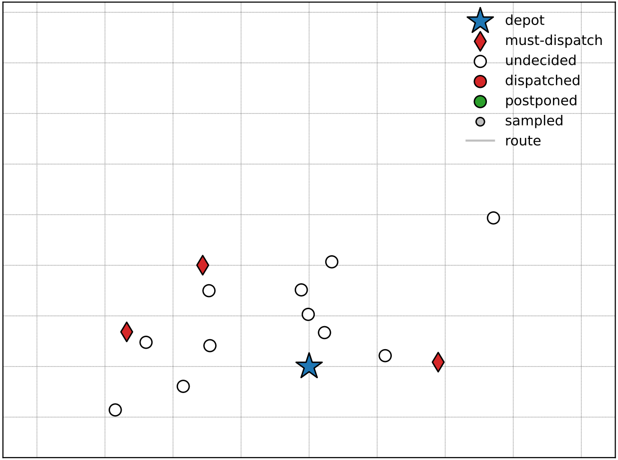
- Main idea: sample future scenarios, solve each scenario as a static problem, and combine solutions to derive an action $a_t$.
- Drawbacks: computationally expensive (need to solve many and large scenarios).
- What if we *incrementally* build an action to minimize computational challenges?
- Similar idea: dynamic stochastic hedging heuristic[4] for a dynamic pickup-and-delivery problem.

4 L. M. Hvattum, A. Løkketangen, and G. Laporte (2006). "Solving a Dynamic and Stochastic Vehicle Routing Problem with a Sample Scenario Hedging Heuristic". In: *Transportation Science* 40.4, pp. 421–438

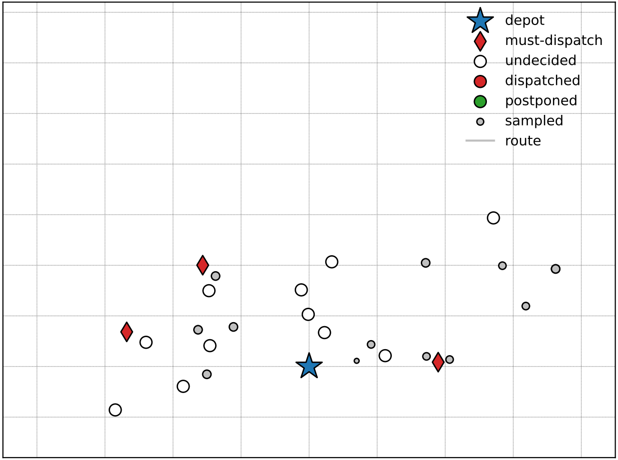
# Iterative conditional dispatch

- Let $d_t$ denote the set of dispatched requests and let $p_t$ denote the set of postponed requests.
- **Initialize** $d_t = m_t$ and $p_t = \varnothing$.
- **Repeat** until iterations exceeded:
  - **Step one**: Solve $|\mathcal{S}|$ sample scenarios conditioned on $d_t$ and $p_t$.
  - **Step two**: Classify undecided requests based on scenario solutions into either the set of dispatched requests $d_t$, the set of postponed requests $p_t$, or leave undecided.
- **Return** action $a_t = d_t$.

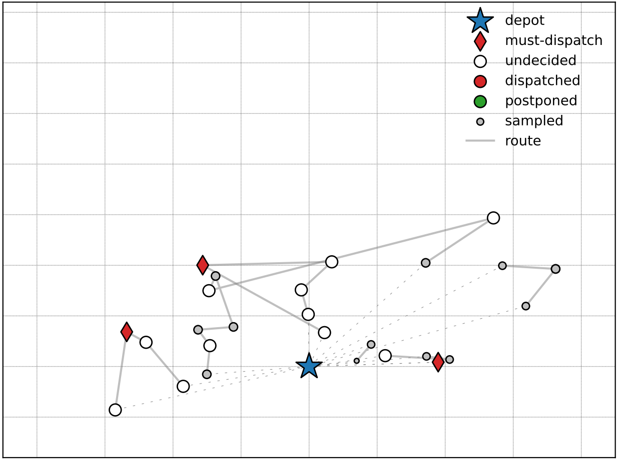
# Epoch instance



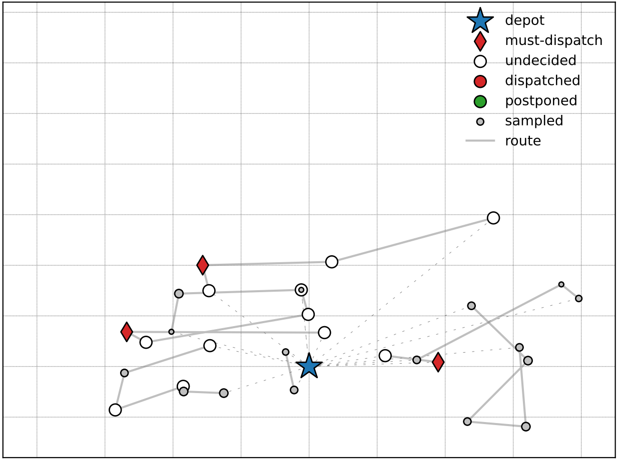
# Scenario instance #1



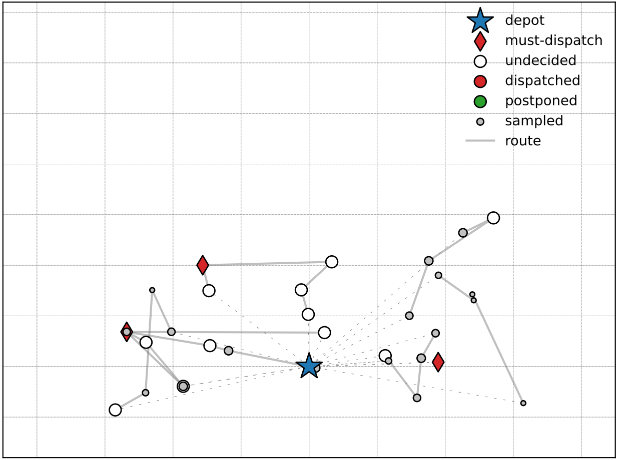
# Scenario solution #1



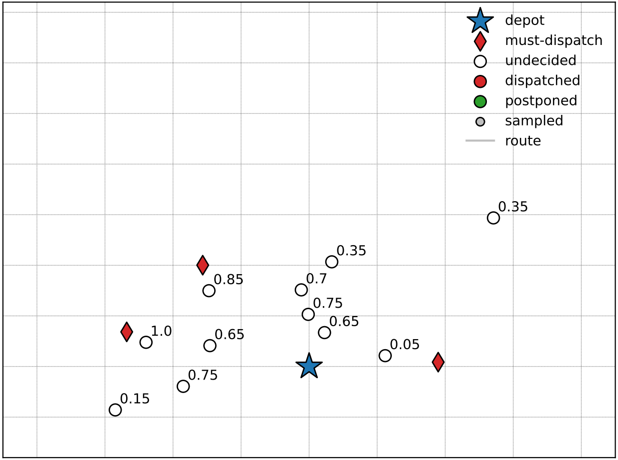
# Scenario solution #2



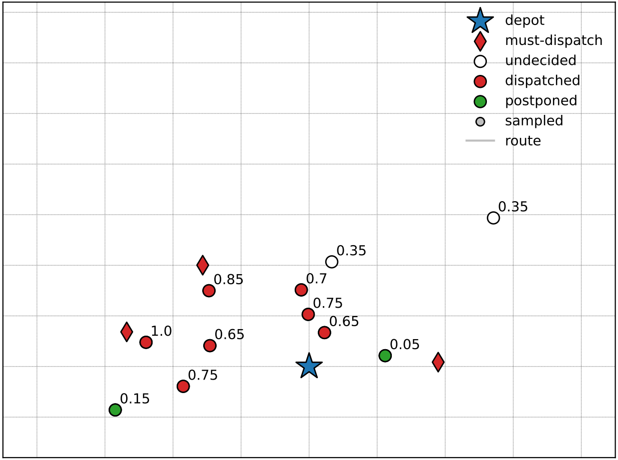
# Scenario solution #3



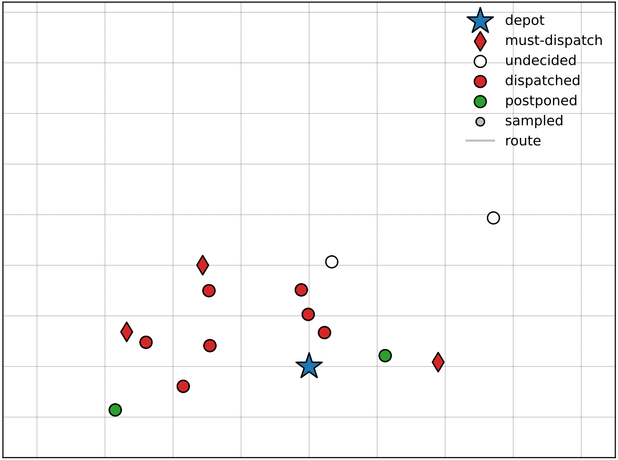
# Dispatch score



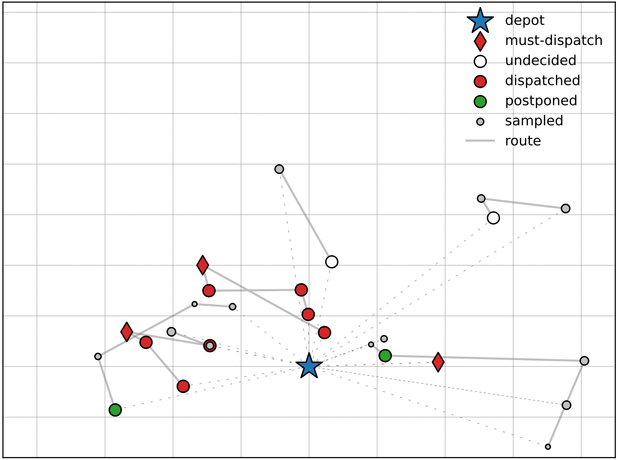
# Classify based on dispatch score



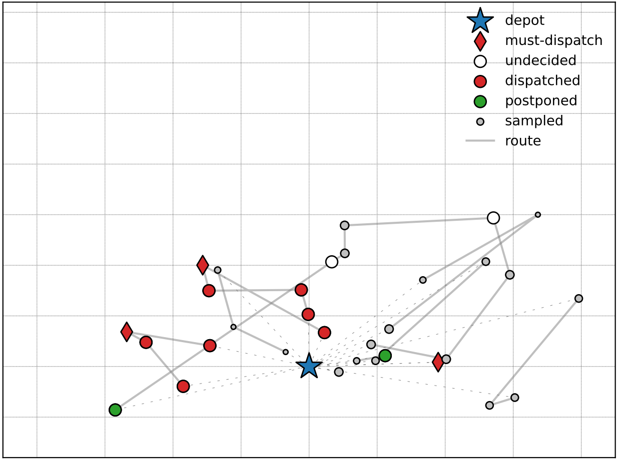
# Classify based on dispatch score



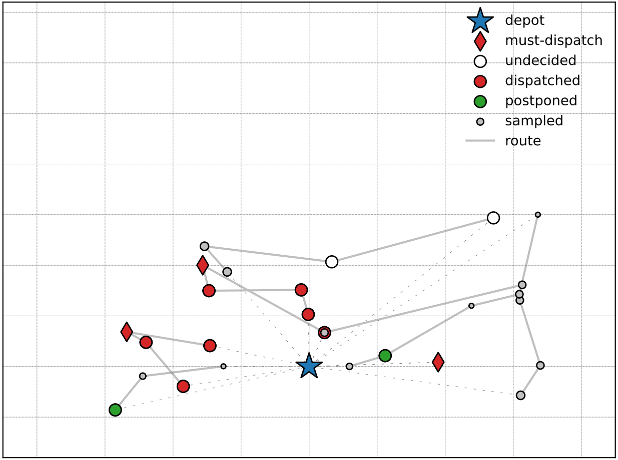
# Rinse and repeat!



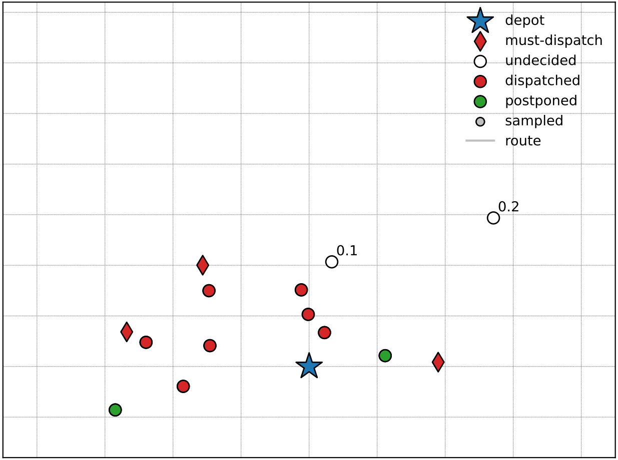
# Rinse and repeat!



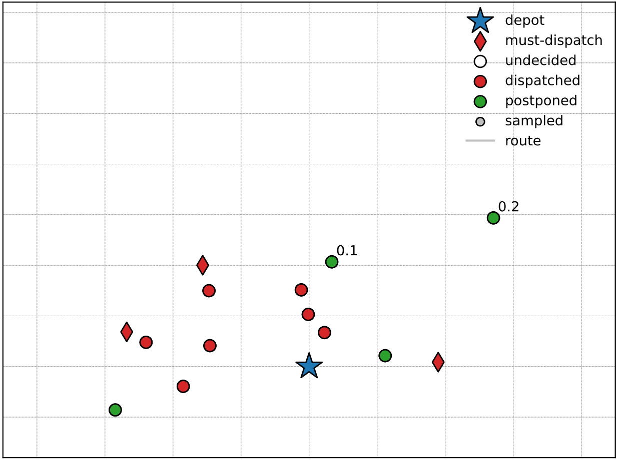
# Rinse and repeat!



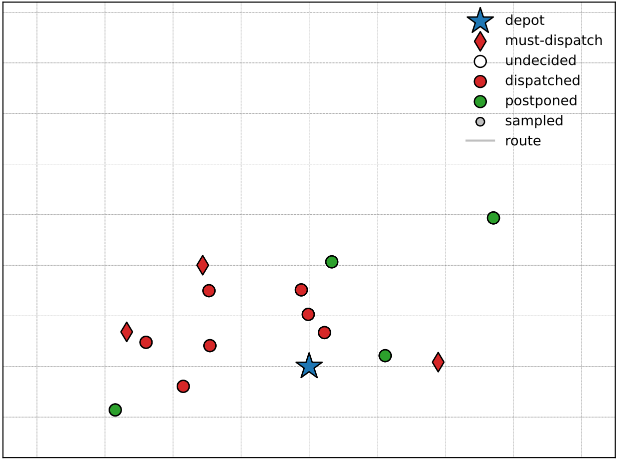
# Rinse and repeat!



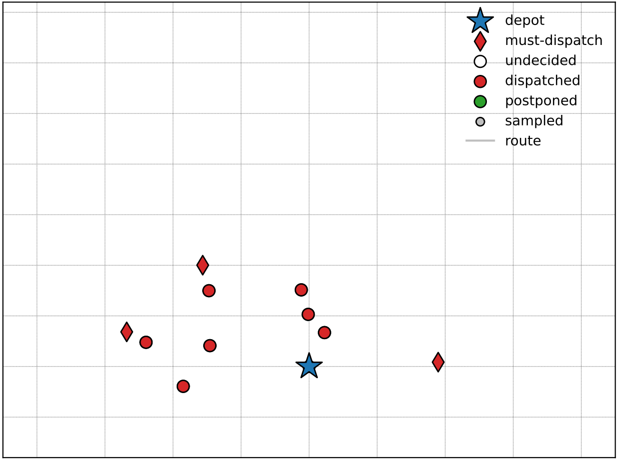
# Rinse and repeat!



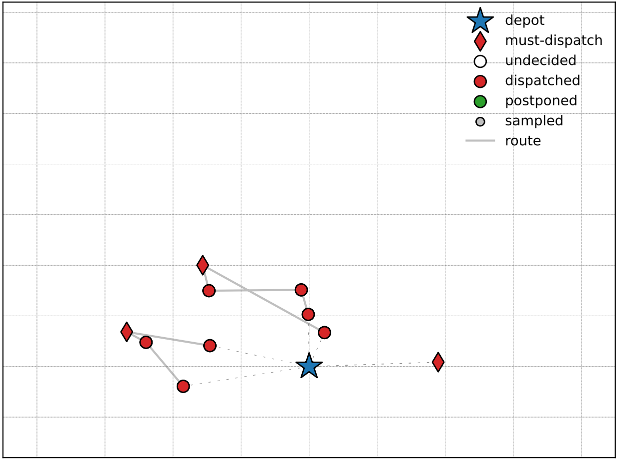
# Rinse and repeat!



# Dispatched requests



# Solve VRP for dispatched requests)



# Details of solving scenarios

- Let $L$ denote the number of lookahead epochs.
- A scenario $S \in \mathcal{S}$ is a set of requests formed by
  - the current state $s_t$, and
  - a sample path realization $(\tilde{\omega}_{t+1}, \tilde{\omega}_{t+2}, \ldots, \tilde{\omega}_{t+L})$, where $\tilde{\omega}_{t'}$ denotes the set of future requests that arrive in decision epoch $t'$.
- Note that requests in $\tilde{\omega}_{t'}$ have release times $T_{t'}$.
- But what about $d_t$ and $p_t$?

# Dispatch windows

- Instead of solving VRPTW-RT, we solve scenarios as VRPTW with dispatch windows (VRPTW-DW).
- Define a request *dispatch window* $[r_n^-, r_n^+]$, where $r_n^-$ denotes the *earliest* time that request $n$ can be dispatched, and $r_n^+$ denotes the *latest* time that it can be dispatched.

### Route dispatch window feasibility

Consider a route $R$ consisting of a sequence of requests and let $\theta_R$ denote its departure time. Then a route is dispatch window feasible if

$$\max_{n \in R}\{r_n^-\} \leq \theta_R \leq \min_{n \in R}\{r_n^+\}.$$

# Modifying request dispatch windows

- Consider a scenario $S$ with dispatched requests $d_t$ and postponed requests $p_t$. We modify the dispatch windows as follows:
  - For dispatched requests $n \in d_t$,

$$[r_n^-, r_n^+] = [T_t, T_t].$$

  - For postponed requests $n \in p_t$,

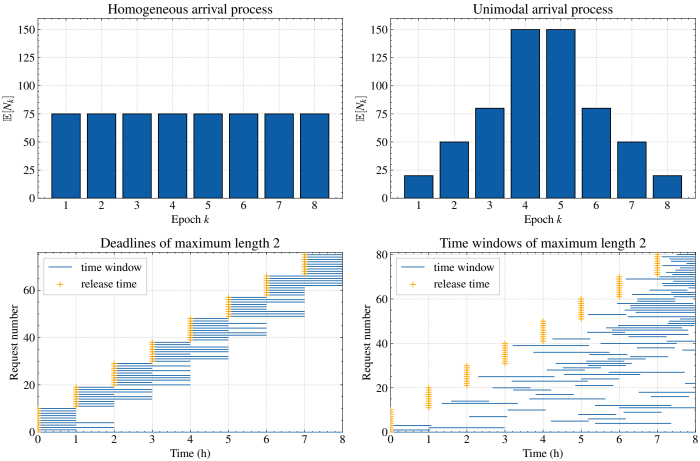
$$[r_n^-, r_n^+] = [T_{t+1}, H].$$

  - For all other requests $n \in S \setminus (d_t \cup p_t)$,

$$[r_n^-, r_n^+] = [r_n, H].$$

# Benchmark instances

- Planning horizon of 8 hours, divided into 1-hour epochs.
- Gehring and Homberger (R, RC, C) instance geographies.



# Parameter tuning

- We use PyVRP[5] to solve all VRP instances.
    - Optimized to solve scenarios well on short time limits ($< 1$ second).
- ICD parameters:
    - Number of iterations: 3
    - Number of scenarios per iteration: $|\mathcal{S}| = 30$
    - Number of lookahead epochs: $L = 1$
- Epoch time limit of 120 seconds
    - 90 seconds total for scenarios → 1 second per scenario
    - 30 seconds to compute the cost of dispatching action

5 N. A. Wouda, L. Lan, and W. Kool (2023). *PyVRP: a high-performance VRP solver package [Manuscript submitted for publication]*. URL: https://github.com/PyVRP/PyVRP/

# Algorithms

- Greedy baseline: dispatch all available requests
- Variants of ICD:
  - Postpone: only use a postponement threshold (and dispatch all not-postponed)
  - Dispatch: only use a dispatch threshold
  - Both: use both dispatch and postponement threshold
- Compare against solution assuming perfect information.

# Results: GH instances

| Method | greedy | postpone | dispatch | **both** |
|---|---|---|---|---|
| Avg. gap | 36.57 | 10.82 | 10.13 | **9.33** |

- 27.24% improvement over greedy baseline algorithm.
- Double threshold shows benefit over single threshold.

# Results: GH instances (subset)

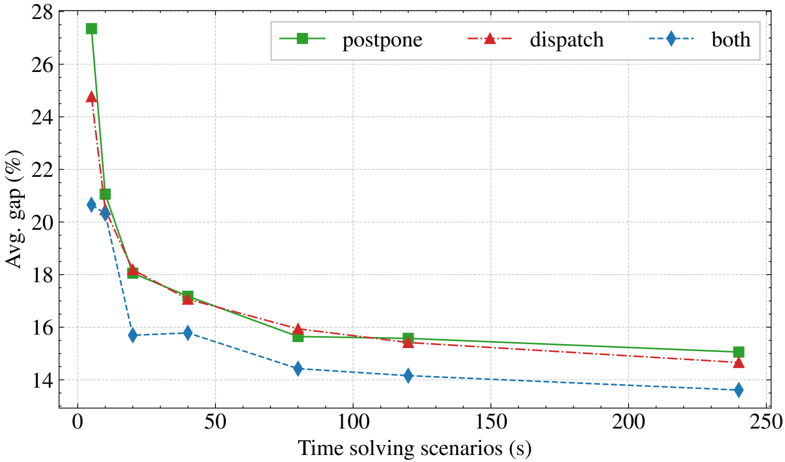



Figure: Average gaps vs. time solving scenarios.

# Results: GH instances (subset)

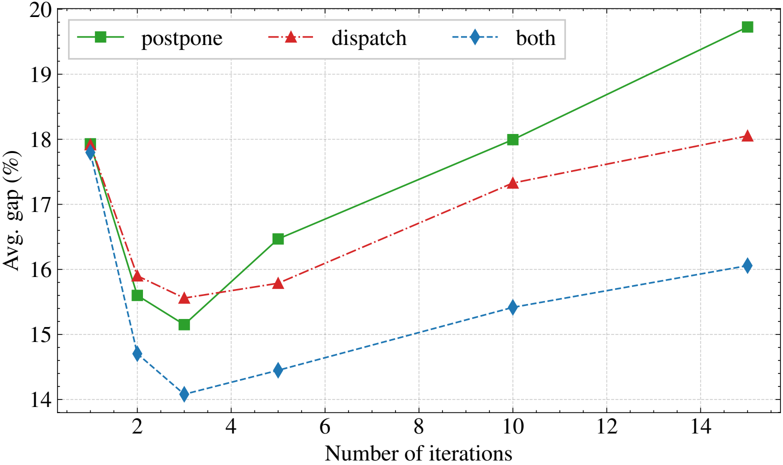



Figure: Average gaps vs. the number of iterations (with fixed 120s epoch time limits).

# EURO-NeurIPS 2022 Vehicle Routing Competition

- 100 instances to evaluate *final* rankings during competition
- Epoch time limits of 120 seconds
- Roughly 400-500 expected number of requests
- Expected number of maximized requests decreases over time

# EURO-NeurIPS 2022 Vehicle Routing Competition

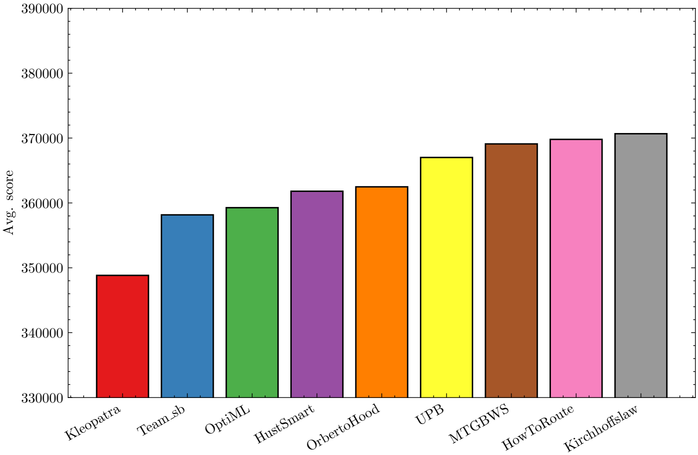



Figure: Results on EURO-NeurIPS final 100 instances

# EURO-NeurIPS 2022 Vehicle Routing Competition

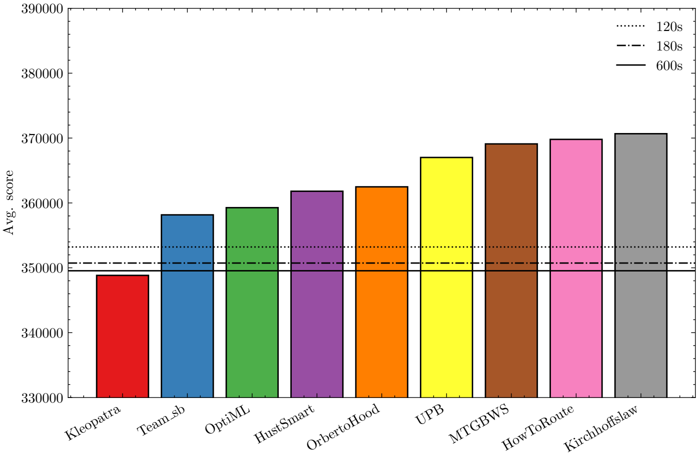



Figure: Results on EURO-NeurIPS final 100 instances

# Conclusion

- We formulate the *dynamic dispatch waves problem* introduced during the EURO meets NeurIPS 2022 vehicle routing competition.
- We present a sample scenario planning method to solve the problem.
- Our results show that the method is scalable and overcomes the large computational efforts often associated with sampling-based methods.

# Future work

Some directions for future work:

- **Variants of the DDWP**: limited fleet, limited inventory, limited route duration, maximizing the number of served requests, event-based triggered epochs
- **Solve DDWP (near-)optimally**: L-shaped method
- **Consensus functions**: hamming distance, prize-collecting

# Thanks for attending!

Slides: https://github.com/leonlan/slides/tsl2023.pdf

l.lan@vu.nl — leonlan — www.leonlan.com

# Bibliography I


Hvattum, L. M., A. Løkketangen, and G. Laporte (2006). "Solving a Dynamic and Stochastic Vehicle Routing Problem with a Sample Scenario Hedging Heuristic". In: *Transportation Science* 40.4, pp. 421–438.

Klapp, M. A., A. L. Erera, and A. Toriello (2018). "The Dynamic Dispatch Waves Problem for Same-Day Delivery". en. In: *European Journal of Operational Research* 271.2, pp. 519–534.

Kool, W., D. Numeroso, R. Reijnen, R. R. Afshar, T. Catshoek, K. Tierney, E. Uchoa, and J. Gromicho (2022). *EURO Meets NeurIPS 2022 Vehicle Routing Competition*.

Wouda, N. A., L. Lan, and W. Kool (2023). *PyVRP: a high-performance VRP solver package [Manuscript submitted for publication]*. URL: https://github.com/PyVRP/PyVRP/.

Zhang, J. and T. V. Woensel (2023). "Dynamic Vehicle Routing with Random Requests: A Literature Review". en. In: *International Journal of Production Economics* 256, p. 108751.