A Tale of Middle-Mile Logistics, Graph Neural Networks, and Reinforcement Learning

July 1st, 2024

Thibaut Cuvelier (tcuvelier@google.com)

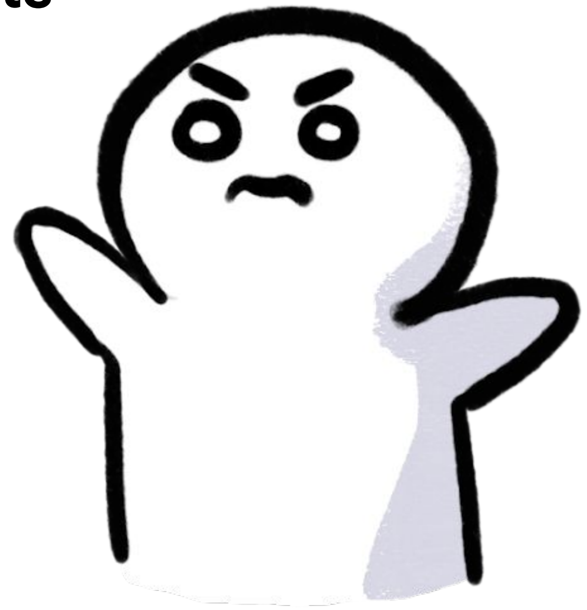
Joint work with Onno Eberhard (onnoeberhard@gmail.com) and Bruno De Backer (bdb@google.com)

Operations Research team, Google Research Paris

How to scale middle-mile logistics?

- Equivalent to large-scale multicommodity flows

- State-of-the-art matheuristic can scale **up to hundreds of shipments**
- The industry needs **millions of shipments**



What does the OR team do?

Routing (VRPs): last-mile logistics, StreetCar exploration

- Historical open-source solver
- New product: GMPRO

Solving practical problems:

- Workforce scheduling
- Shipping network design

Solving LPs and MIPs:

- Glop: robust simplex (LP only)
- CP-SAT: CP engine based on SAT, won 10+ gold medals at the MiniZinc competition
- PDLP: first-order LP solver
- MathOpt: modelling layer, also for cloud solves

Open-source product: OR-Tools

What is middle-mile logistics?

1

First mile

Tours within a metropolitan area

Short duration: 1 day

2

Middle mile

Within a region

Horizon of several days

3

Last mile

Tours within a metropolitan area

Short duration: 1 day

Source: UPS

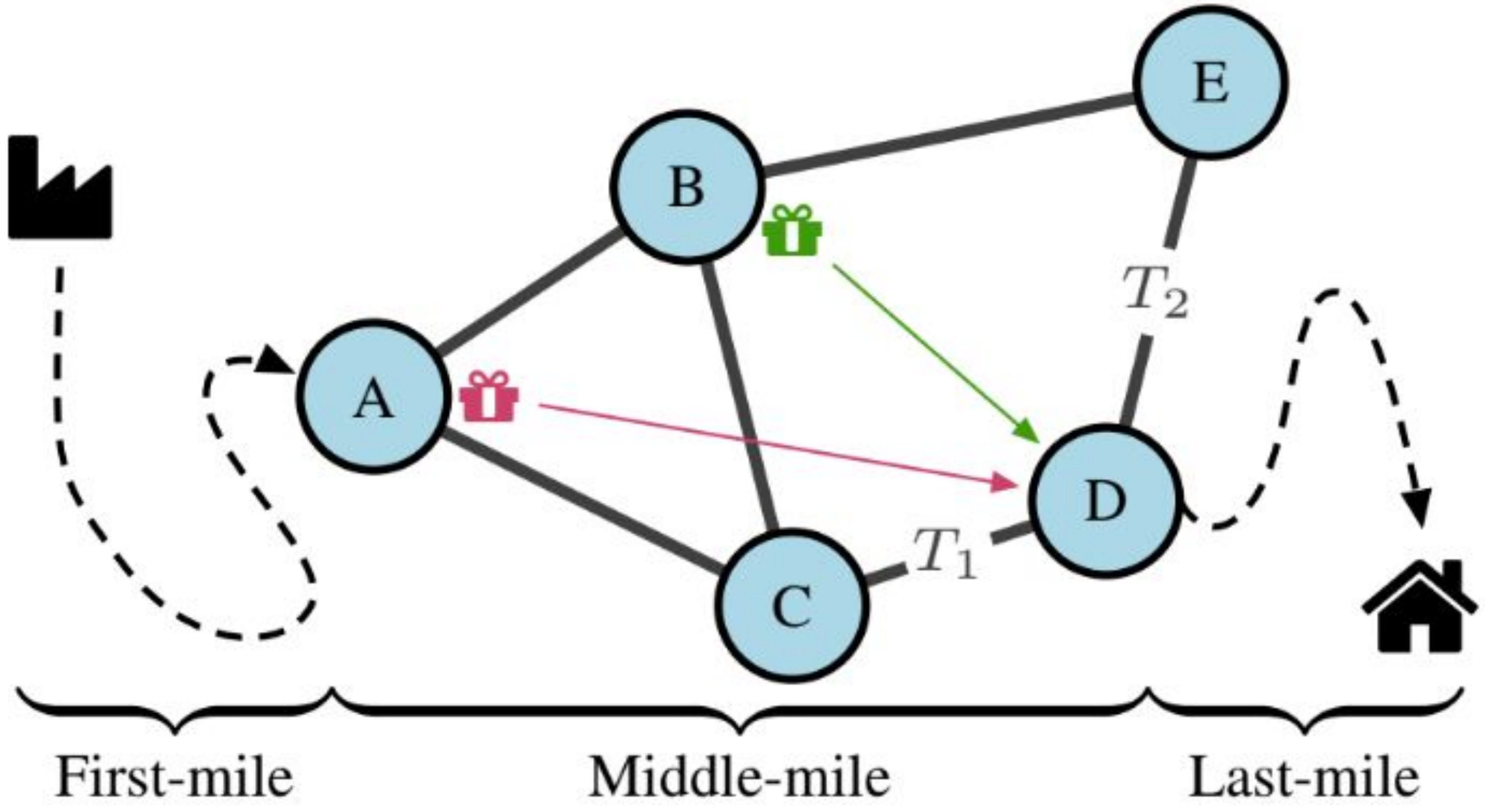
Source: UPS

Source: DHL

What is middle-mile logistics?

- **Shipments/containers**: size, time window
- **Hubs**: where shipments start/end
- **Vehicles**
- **Predefined lines**: list of hubs with visit times

- **Crossdocking**: a vehicle drops shipments at a hub and picks up others



What path should a shipment take?

Why is middle-mile logistics hard?

State-of-the-art matheuristic:

20
hubs

400
shipments

0
time steps

Customer needs:

100s
hubs

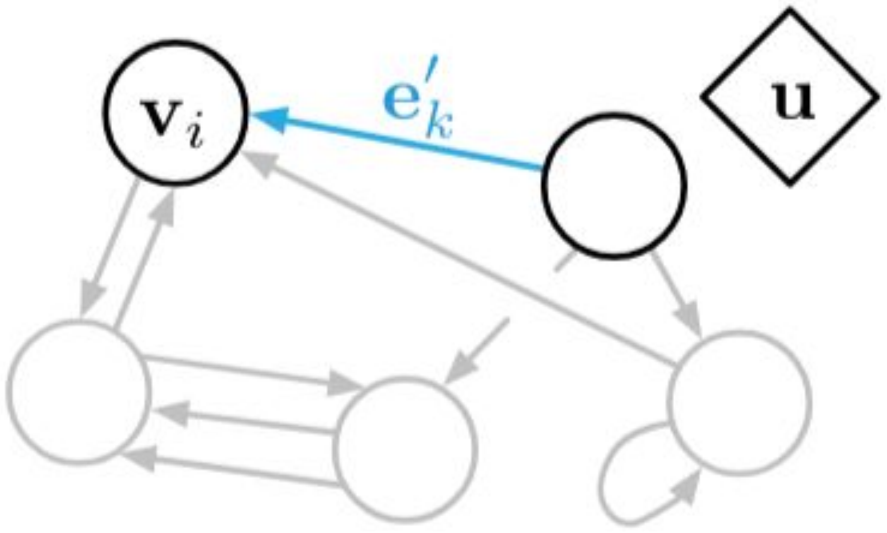
10^9s
shipments

1,000s
time steps

Graph neural networks and GraphNet

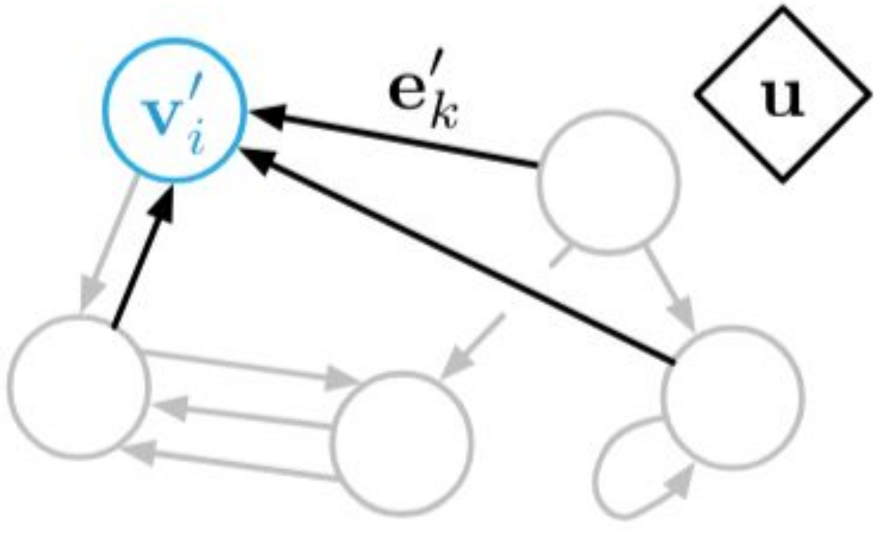
GraphNet is a specific GNN architecture with features for edges, nodes, and the whole graph

The network organised in blocks of “message passing”



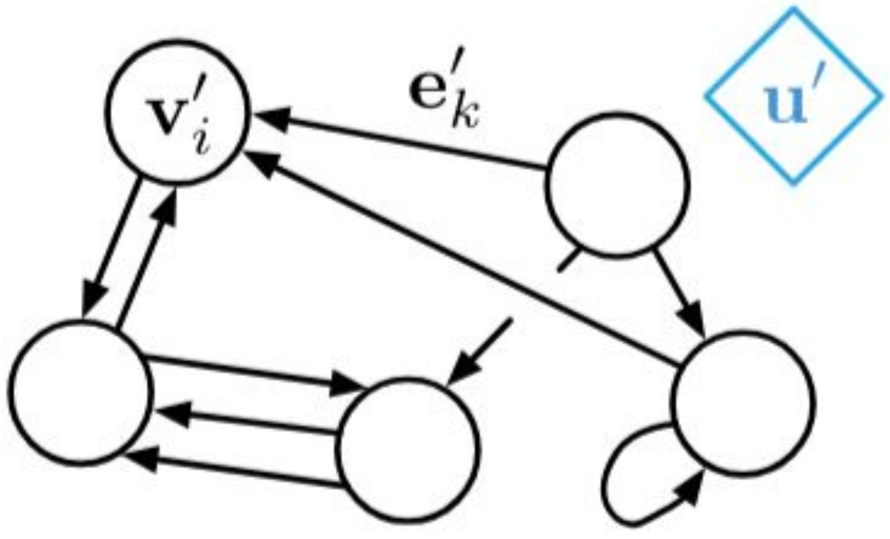
Edge-level message passing

Gather information from the end nodes



Node-level message passing

Gather information from adjacent edges, one by one, then aggregate



Graph-level message passing

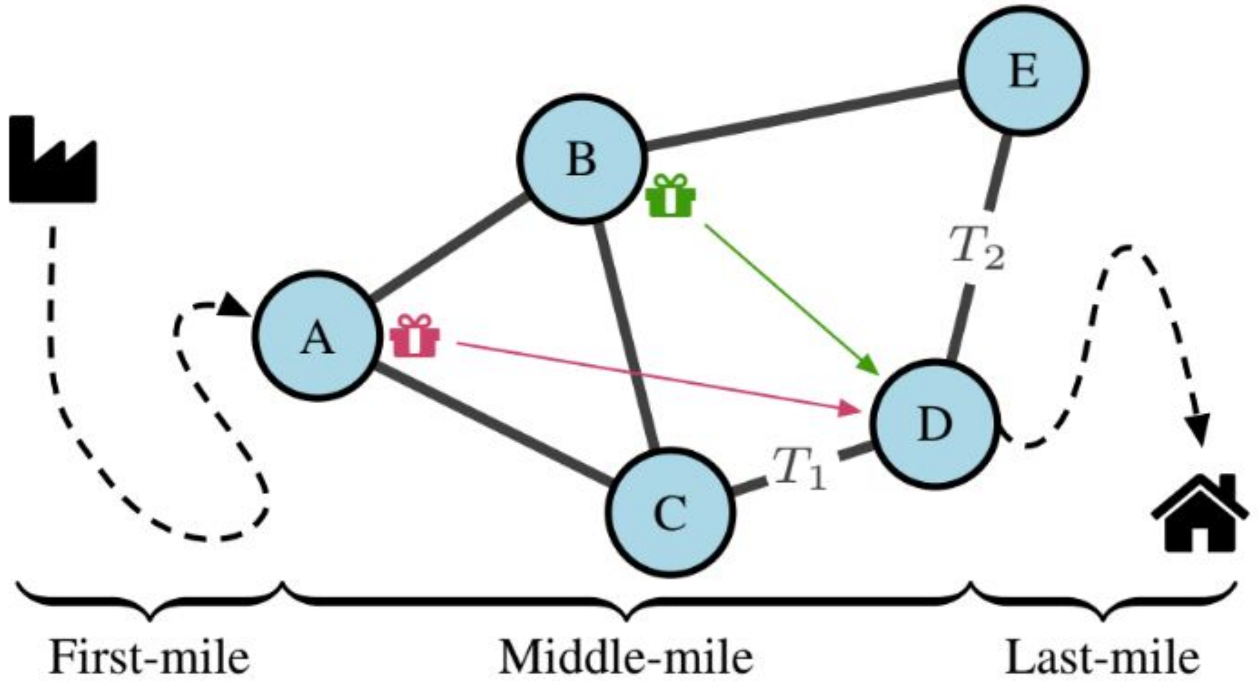
Gather information from all the edges and all the nodes

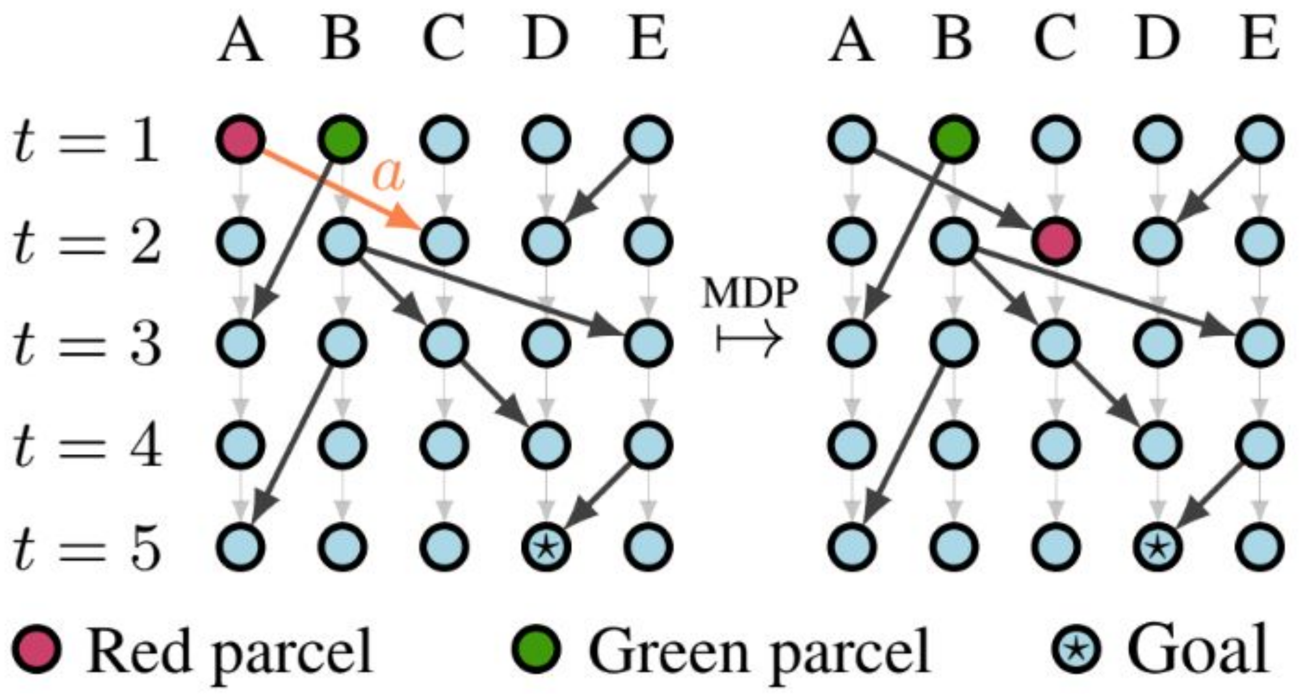
Image credit: Relational inductive biases, deep learning, and graph networks, P. Battaglia et al.

Middle-mile logistics as end-to-end reinforcement learning

Model it as an MDP:

- **State**: a time-expanded graph
  - Nodes are hubs with time
  - Edges are trucks with their schedule
  - Parcels are located at nodes
- **Transition**: one shipment moves from one hub to another (or stays at the current hub)
- **Reward**: whenever a shipment reaches its destination





Do you need to be smart?

Is pruning the graph useful?

- The time-expanded graph is large:
  - 100s hubs
  - 1,000s time steps
  - 100,000s of nodes
- Only a small part is useful for any given parcel
- GNNs have a high complexity
- **Pruning**: remove parts of the graph that will never be used

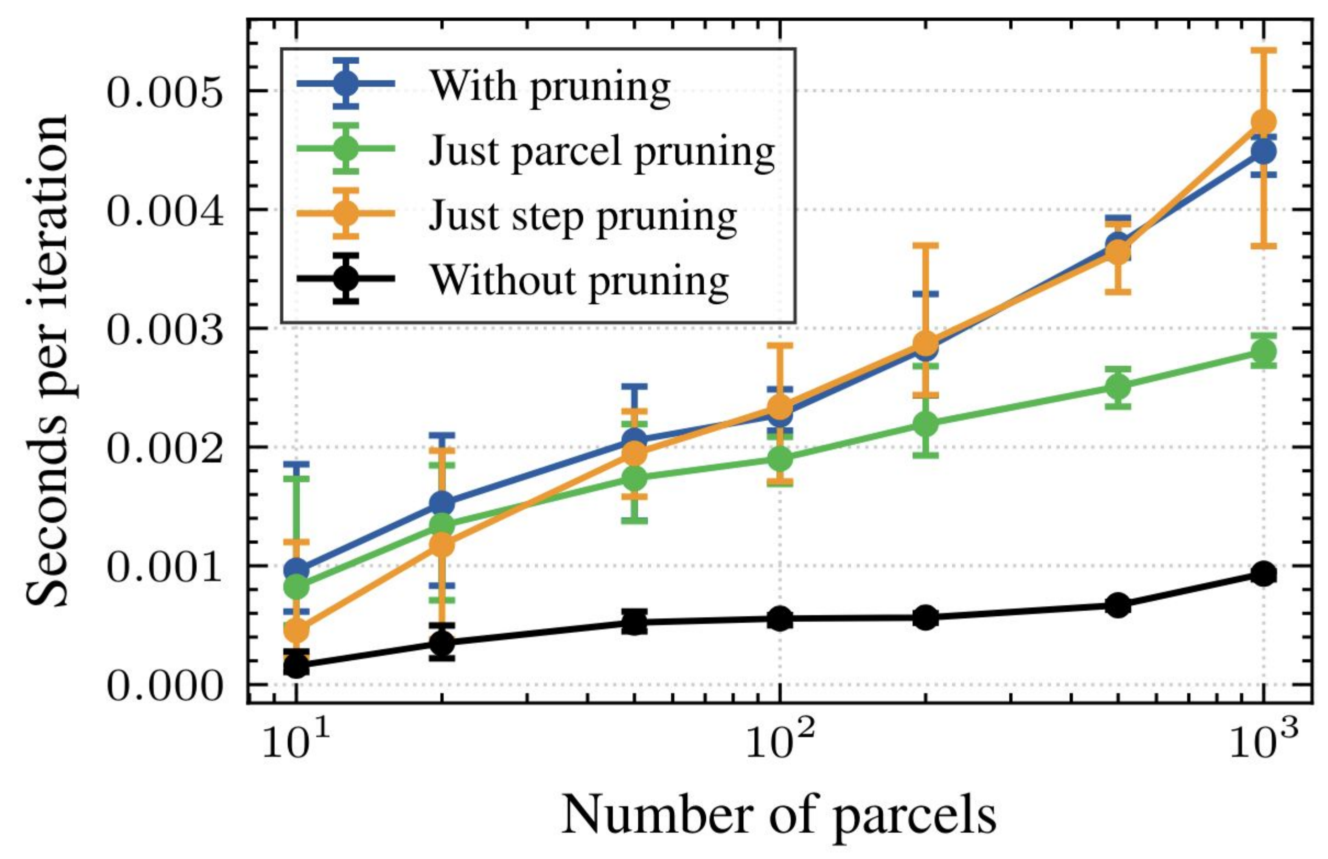
Do you need to be smart?

Is pruning the graph useful?

- The time-expanded graph is large:
  - 100s hubs
  - 1,000s time steps
  - 100,000s of nodes
- Only a small part is useful for any given parcel
- GNNs have a high complexity
- **Pruning**: remove parts of the graph that will never be used

Result?

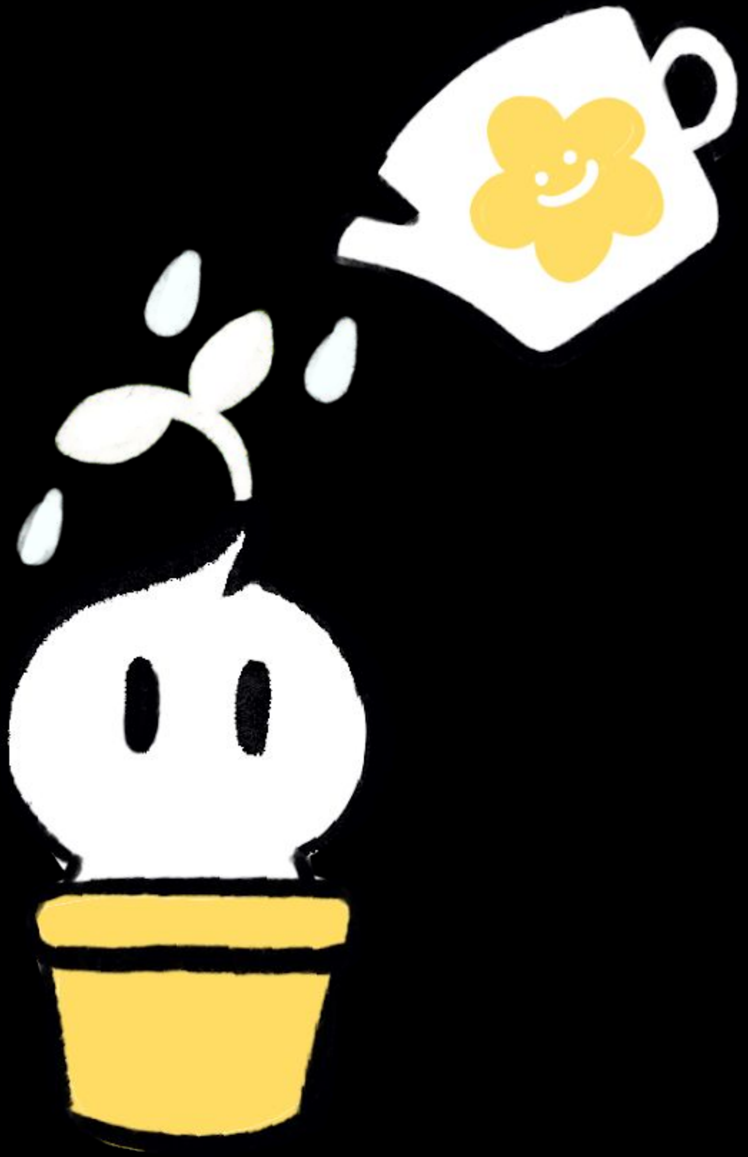
- Much, much, much slower (5x)

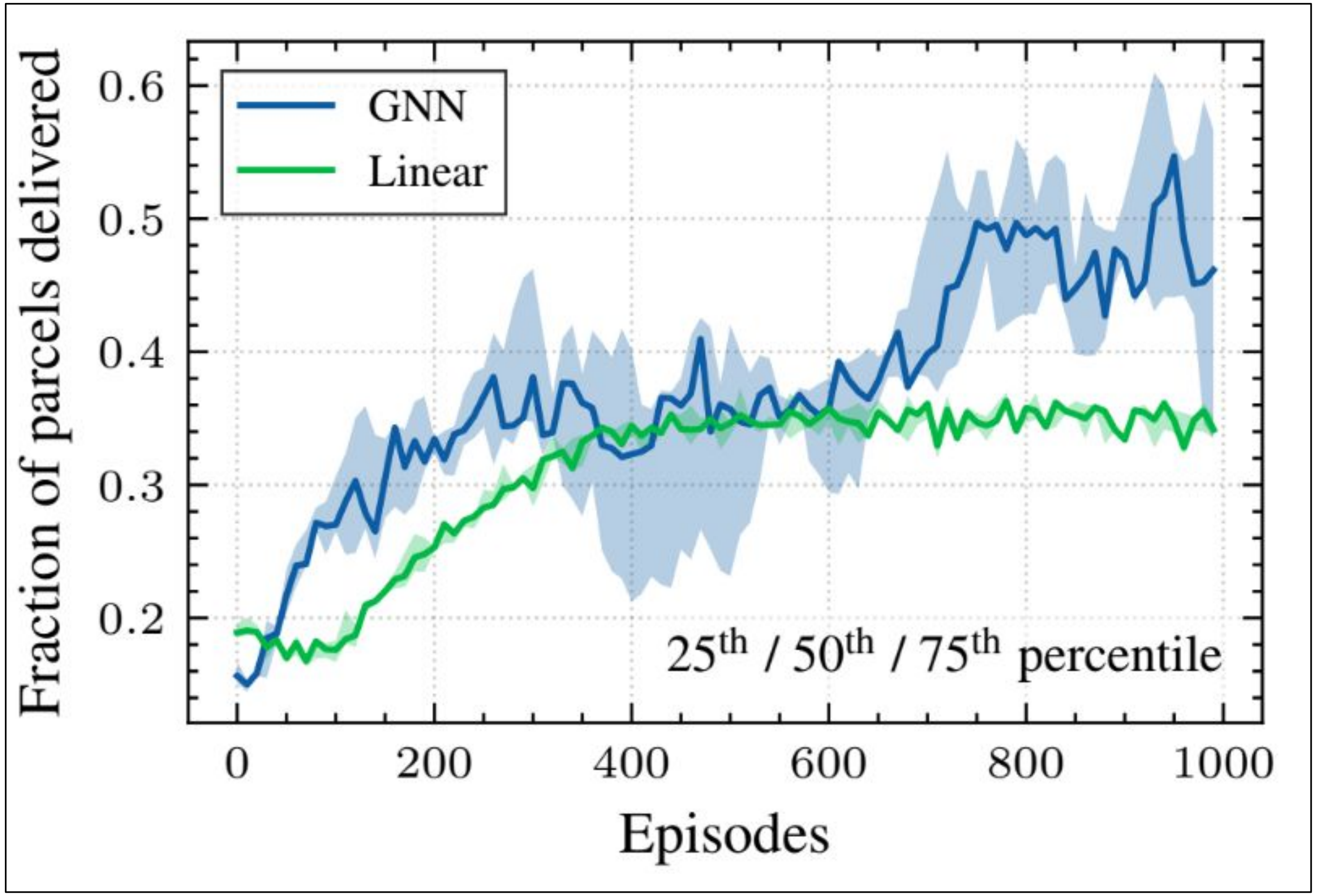


Feature graphs

- Instead of pruning, go the other way around
- Take all nodes within K edges of the origin and destination nodes
    - Easy to obtain!
    - Good results

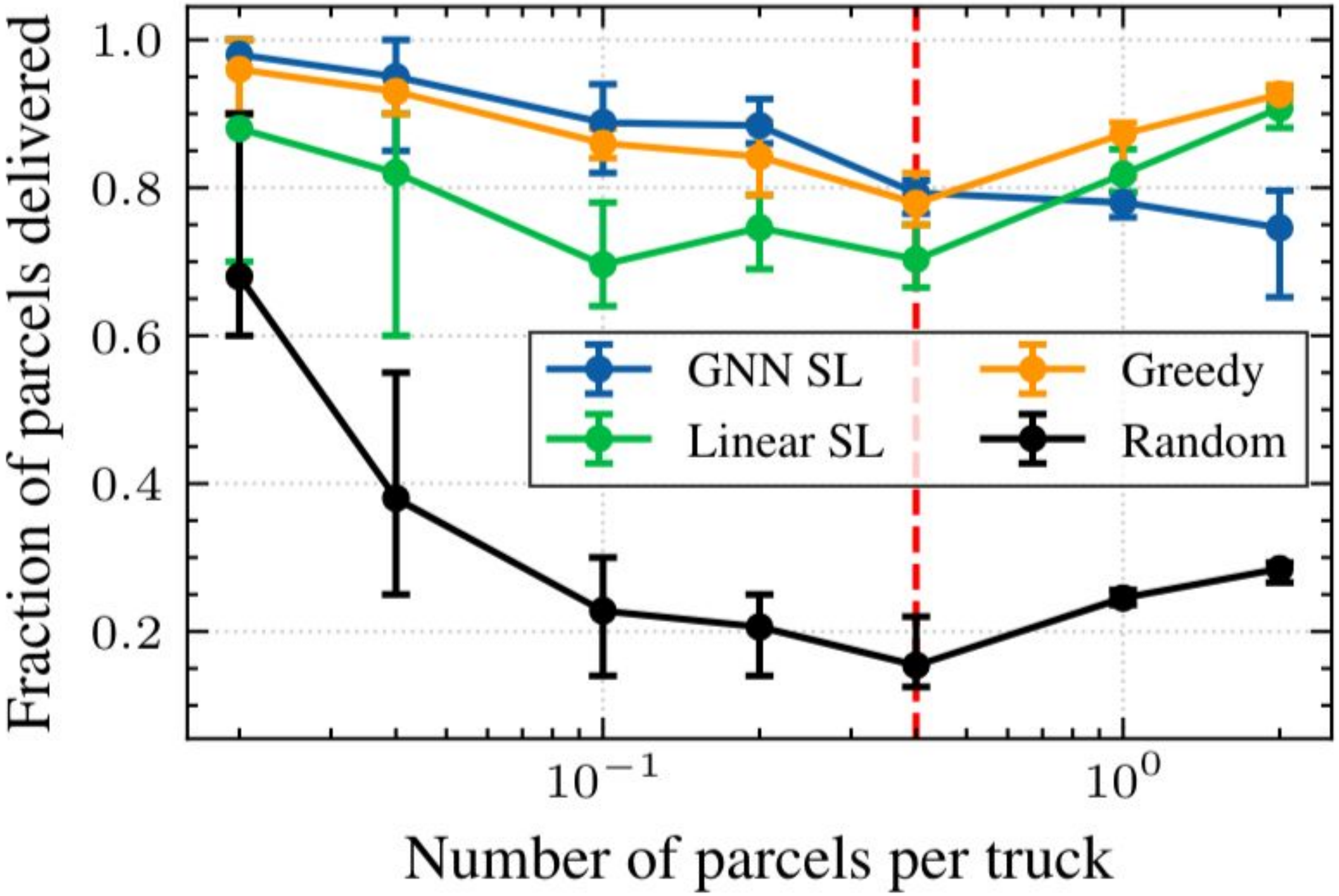
- But… no longer correct!
    - If there are more than 2 K edges between source and destination, no solution
    - Maybe the optimal path is outside the feature graph
- Hence:
    - Add features, such as distance to final node





Methodology and results

- RL algorithm:
  - PPO: model-free, policy gradient, actor-critic architecture
- Actor and critic: GNNs
  - Number of rounds: 10 hops, paths expected to be shorter in general
  - For feature graphs, K = 5
  - Compare to a linear approximator
- Evaluation: one new network per episode
  - Great generalisation!



Results

What about supervised learning?
What about instance hardness?

- Both RL and supervised learning **generalise well** to new instances
  - Even trained on a single instance hardness (red dashed line)

- Comparison to supervised learning:
  - **RL uses scarce resources** better

Next steps?

- Greedy decisions are not reconsidered
- Is RL the best tool? Monte Carlo tree search (like AlphaZero) would use more of the structure
- Is RL a good way to solve *other* combinatorial problems?

References

Middle-Mile Logistics Through the Lens of Goal-Conditioned Reinforcement Learning
Onno Eberhard, Thibaut Cuvelier, Michal Valko, Bruno De Backer
Goal-Conditioned Reinforcement Learning Workshop, NeurIPS 2023

Source code for the experiments: https://github.com/google-research/laurel

Relational inductive biases, deep learning, and graph networks
Peter W. Battaglia , Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, et al.
https://arxiv.org/abs/1806.01261

Jraph: A library for graph neural networks in Jax
Jonathan Godwin, Thomas Keck, Peter Battaglia, Victor Bapst, et al.
https://github.com/google-deepmind/jraph

Doodles drawn by Xinni Chng (xinni@google.com)

Thank You

Same topic, same team:
how do you generate **good test instances**?

Wednesday, VeRoLog stream,
WD-58

A Novel Instance Generator for Simulating
Middle-Mile Logistics Networks