Faster Latency Constrained Service Placement in Edge Computing with Deep Reinforcement Learning[1]

Séminaire Clourd Optimization DVRC/ESILV

Orso Forghieri [2]; *Emmanuel Hyon* [1]; Erwan Le Pennec [2] , Hind Castel Taleb [3] , Nancy Perrot, Yannick Carlinet [4].

[1] LIP6, Université Paris Nanterre

[2] CMAP, École Polytechnique

[3] Telecom SudParis

[4] Orange R&D

29/05/2025

[1] Issued From TX4Nets Workshop — IFIP Networking 2025 — Limassol, Cyprus

Table of Contents

Problem Statement

Smart Warehouse Usecase

From Orange

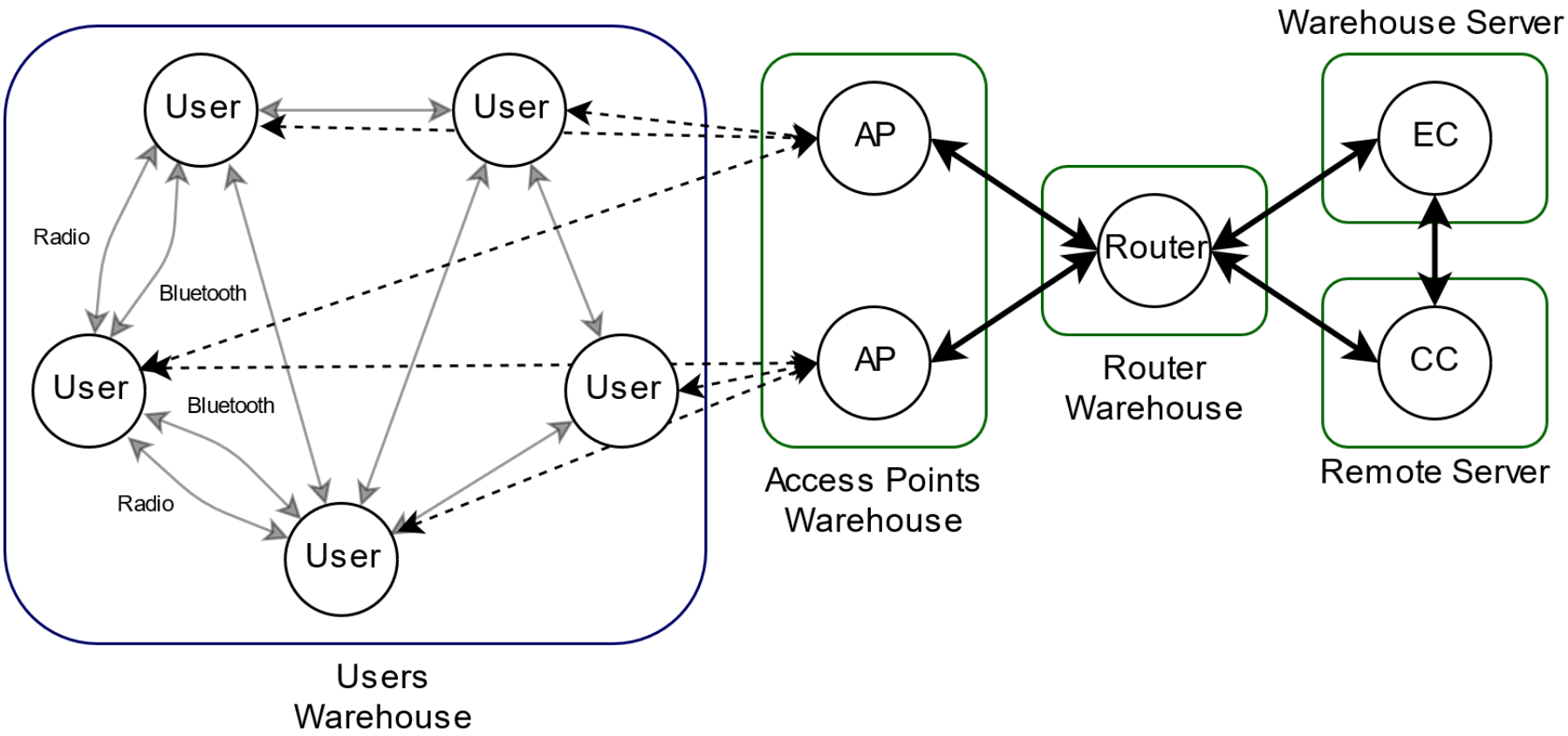


Figure 1: Smart Warehouse instance [11].

Problem Statement

Mobile Edge Computing (MEC)

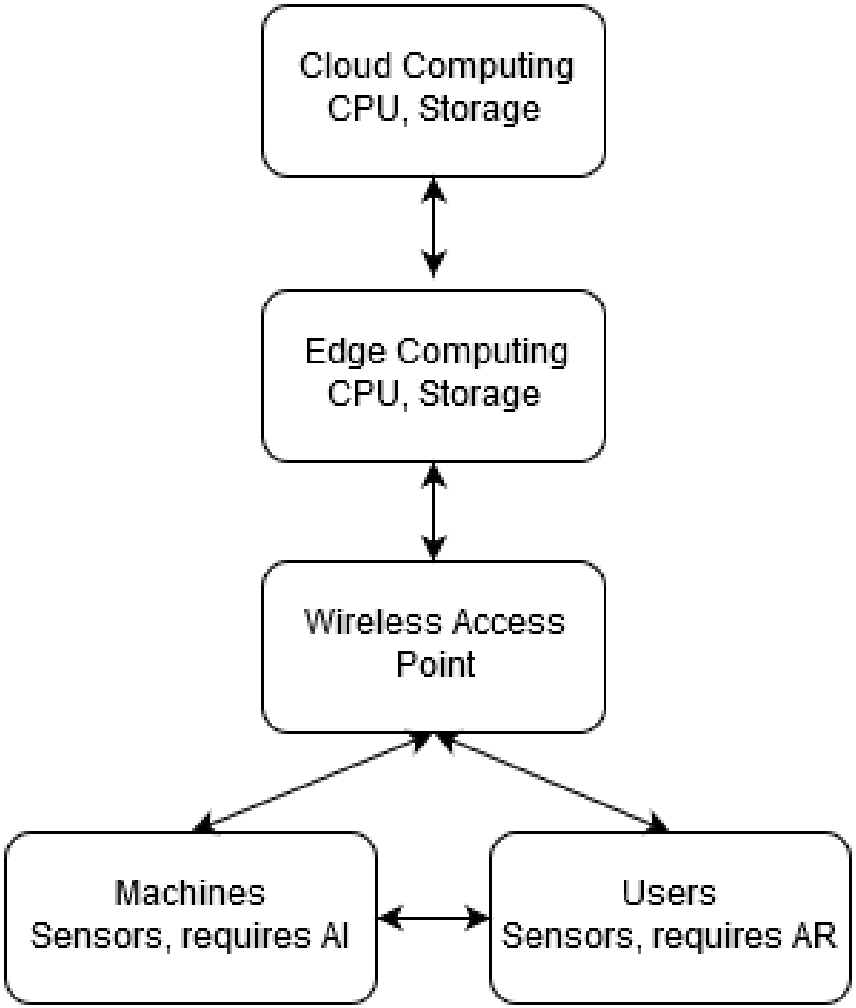


Figure 2: Smart warehouse in the global network.

Customer request:

- A service to deploy.

ISP's decision: Where to place the service?

1. Cloud (high power, high latency)
2. Edge (low latency, limited power)
3. Access Points (Tradeoff between latency and resources)

Optimize QoS for all services simultaneously and provide an **adapting and deployable** method.

Description of Services

A **service** S includes:

- Multiple **demands** with known origins and the same destination. (User data and server data meet at an intermediate point)

Each **demand** d of S is defined by:

- An (known) origin d_o and a variable destination;
- A maximum latency l_d (to decide);
- A QoS level $\mathbf{q}_d \in \{q_{d,\min}, \ldots, q_{d,\max}\}$;
- A path from origin to selected destination (to find).

Description of the Network

The network

An undirected graph $\mathcal{G} = (\mathcal{E}, \mathcal{A})$ with:

- $\text{res}^e \in \mathbb{N}$: available resources at node $e \in \mathcal{E}$ (e.g., CPU, storage);
- $\text{capa}^a \in \mathbb{N}^+$: capacity of edge $e \in \mathcal{E}$;
- lat^a: latency associated with edge e.

Latency model (linear approximation)

Using edge $e \in \mathcal{E}$ with bandwidth q_a induces a latency:

$$\text{lat}^a = \alpha^a \sum_{d \in \mathcal{D}} \mathbf{q}_a + \beta^a$$

- $\alpha^a \in \mathbb{R}^+$: latency increase per unit of traffic;
- $\beta^a \in \mathbb{R}^+$: base latency of edge e.

Orchestration Problem Description

Key subproblems:

- **Service placement:** Assign services to nodes with available resources (e.g., GPU, memory);
- **Routing:** Define paths from sources to destinations;
- **QoS management:** Allocate bandwidth to satisfy demand requirements.

This turn to be an Assignment problem coupled with multicommodity flow (NP Hard).

Core question: Can deep RL provide a scalable and deployable solution?

Literature Review

- **Use Cases:** [9, 11]
- **Service Assignment:** surveys [3, 10], RL [6]
- **Network Flows:** linear [4], nonlinear [2], multicommodity [5]
- **RL for Combinatorial Optimization:** original work [1], survey [8]
- **Deep RL for Multicommodity Flow:** CDN [13, 12]

Table of Contents

Decision Variables

Decision variables:

- **Service placement**:
  $y_S^e \in \{0, 1\}$ (1 if S is on node e, 0 otherwise)
- **Demands routing**:
  x_d^a (1 if request d uses edge e, 0 otherwise)
- **Quality of Service/Bandwidth**
  $q_d \in \{q_{d,min}, ..., q_{d,max}\}$ the quality of demand d.

Objective function :

$$\max \sum_d q_d$$

Integer Program

$$\max \sum_{S, d \in S} q_d \qquad \text{(Maximize quality)}$$

$$\text{s. t.} \sum_{f \in \delta(e)} \left(\mathbf{x}_d^{(e,f)} - \mathbf{x}_d^{(f,e)} \right) = \mathbf{1}_{e=d_o} - \mathbf{y}_S^e \,, \ \forall e, \forall d \in S, \forall S \quad \text{(routing)}$$

$$\sum_{S \in \mathcal{S}} \mathbf{y}_S^e r_S \leq \text{res}^e \,, \quad \forall e \in \mathcal{E} \qquad \text{(Resources on nodes)}$$

$$\sum_{S \in \mathcal{S}} \sum_{d \in S} \mathbf{q}_d \mathbf{x}_d^a \leq \text{capa}^a \,, \quad \forall a \in \mathcal{A} \qquad \text{(Edges capacities)}$$

$$\sum_{a \in \mathcal{A}} \mathbf{x}_d^a \left(\alpha^a \sum_{S' \in \mathcal{S}} \sum_{d' \in S'} \mathbf{q}_{d'} \mathbf{x}_{d'}^a + \beta^a \right) \leq l_d \,, \quad \forall d \in S, \forall S \in \mathcal{S}$$

(Demands latencies)

$$\sum_{e \in \mathcal{E}} \mathbf{y}_S^e \geq 1 \,, \quad \forall S \in \mathcal{S} \qquad \text{(Services placement)}$$

$$\mathbf{q}_d \in \{q_{d,\min}, \ldots, q_{d,\max}\}, \ \mathbf{y}_S^e \in \{0,1\}, \ \mathbf{x}_d^a \in \{0,1\}$$

Resolution of the Mathematical Program

This Integer Program:

- Can be linearized by introducing new variables;
- Can be finally solved using ILP exact solvers (e.g., CPLEX Optimizer).

Studied improvements:

- Path-Formulation,
- Continuous relaxation and rounding,
- Decomposition: Service placement (solved by local search) and Multi commodity flow (with Linear Progamming).

Possible Improvements: Dantzig Wolfe decomposition

Anyway ILP solvers are slow in practice. $\rightarrow$ **RL approach**.

Local search heuristic

Solution description

A solution is a vector (e_S, q_d, SP_d) where

- e_S describes the *service placement* for each service ($\in [1, |\mathcal{E}|]^S$).
- q_d describes the *quantity* for each demand ($\in [1, Q]^D$).
- SP_d describes the *shortest path* chosen (index of the path) by demand ($\in [1, MAXPCC]^D$).

Greedy Algorithm

We assume that there is a feasible solution.

- Find the feasible solution (minimal quantity and all services on cloud)
- Repeat while improvement exists
  - Increases the flows greedily
  - If possible decrases load of the most loaded arc (by changing path)
  - If possible change the placement of services

Local search heuristic

Solution description

A solution is a vector (e_S, q_d, SP_d) where

- e_S describes the *service placement* for each service ($\in [1, |\mathcal{E}|]^S$).
- q_d describes the *quantity* for each demand ($\in [1, Q]^D$).
- SP_d describes the *shortest path* chosen (index of the path) by demand ($\in [1, MAXPCC]^D$).

Metaheurisics

Here we consider Evolutionary approaches **Using CMAES**

Anyway Local Search and metaheuristics have poor performances. $\rightarrow$ **RL approach**.

Reinforcement Learning for Combinatorial Optimization

Markov Decision Process (MDP) Framework

- **State space:** the set of feasible solutions;
- **Action space:** modifications applicable to a solution;
- **Transition:** deterministic evolution to a new solution based on the selected action;
- **Reward:** a measure of improvement from the action.

Neural network: enables generalization of the policy to unseen solutions.

RL Model

Observation Space

Input to the neural network: $o = (\mathbf{x}, \mathbf{y}, \mathbf{q}, \mathcal{G}_{\text{des}}, \mathcal{S}_{\text{des}})$

- $\mathbf{x}, \mathbf{y}, \mathbf{q}$: Current solution represented as vectors;
- $\mathcal{G}_{\text{des}}$: Complete description of the graph (capacities, resources, edge usage, etc.);
- $\mathcal{S}_{\text{des}}$: Description of the instance requirements.

$\rightarrow$ **Integration of any relevant technical feature — the more, the better.**

RL Model

Action Space

For each service, **the action operates**:

- An explicit selection of the placement node;
- An explicit selection of the shortest path index;
- An adjustment of the QoS q_d by ± 1.

$\rightarrow$ Fine-grained action specification $\implies$ complex exploration.

RL Model

Reward Shaping

Reward function:

- Penalty for **infeasible actions**:

$$\begin{cases} -10 \text{ for service displacement} \\ -2 \text{ for demand routing} \\ -1 \text{ for QoS change} \end{cases}$$

- $\max(0, q_{\text{now}} - q_{\text{best}})$ otherwise.

where q_{best} denotes the maximum quantity achieved so far during the trajectory.

$\rightarrow$ Reward is related to an **exploration/exploitation trade-off.**

RL algorithm

Actor-critic method:

1. Policy parameterized by θ.
2. Optimization of θ by SGD following:

$$\nabla_\theta J(\pi_\theta) = \mathbf{E}_{\pi_\theta}\left(\sum_t \nabla_\theta \log \pi_\theta(a_t|s_t) G(s_t, a_t)\right)$$

with $G(s,a) = \sum_{t=t'}^{H} \gamma^{t-t'} r(s_t, a_t)$.

Two algorithms considered:

- **A2C**: Less stable, require tuning.
- **PPO**: Explore if reward is well shaped, stable, sample efficient;

$\rightarrow$ **PPO more efficient** in practice

Table of Contents

Instances description

Parameterization

Parameter	**Typical Value**
Maximum flow	1-100 Mbps
Resources	10 TB-10 PB
Base latency	1 ms
Induced latency	1 ms/Mbps
Required letency	10 ms

Table 1: Main graph parameters and variables [7].

Instances design

Tree-shaped graph

- Cloud and edge nodes;
- Close users are connected.

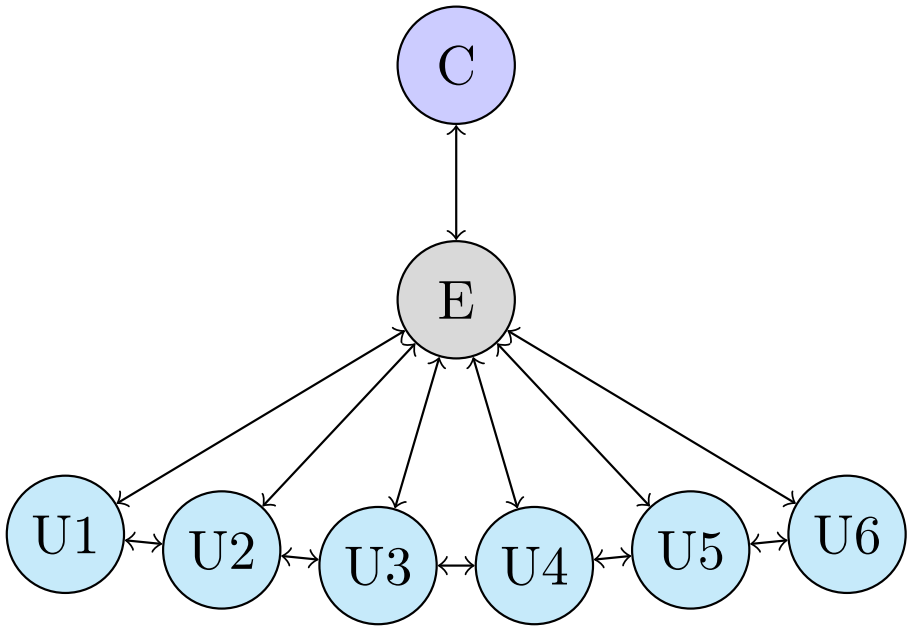


Figure 3: Tree-shaped graph

Solving Approaches

We selected:

- An **exact ILP-solving** approach using the CPLEX Optimizer;
- The previously proposed **RL method**;
- The previously proposed **heuristic placement and routing** approach based on traditional Multi-commodity Flow algorithms.

Runtime comparison for static tree instance

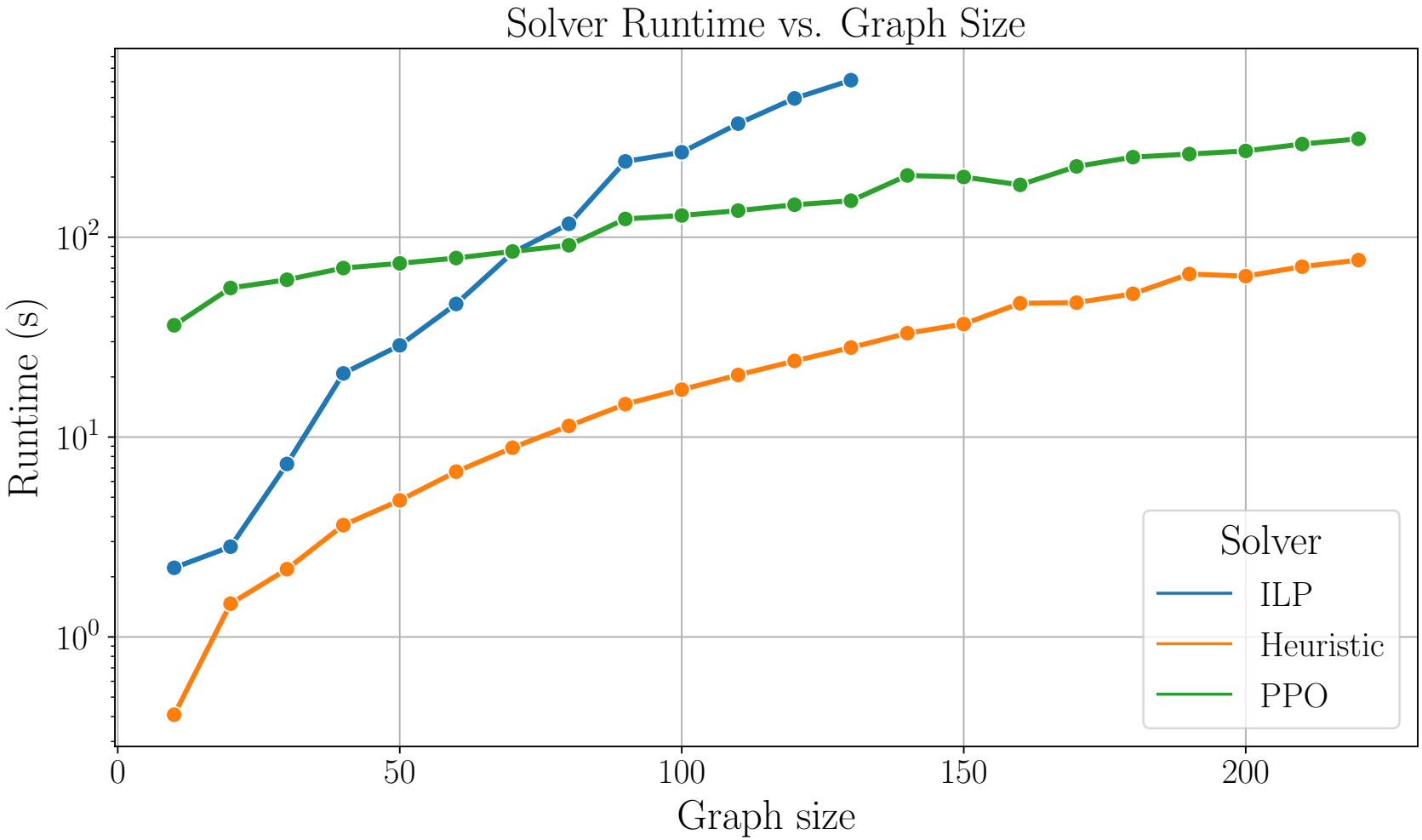


Figure 4: RL training is scalable considering both memory and runtime.

Performance comparison for static tree instance

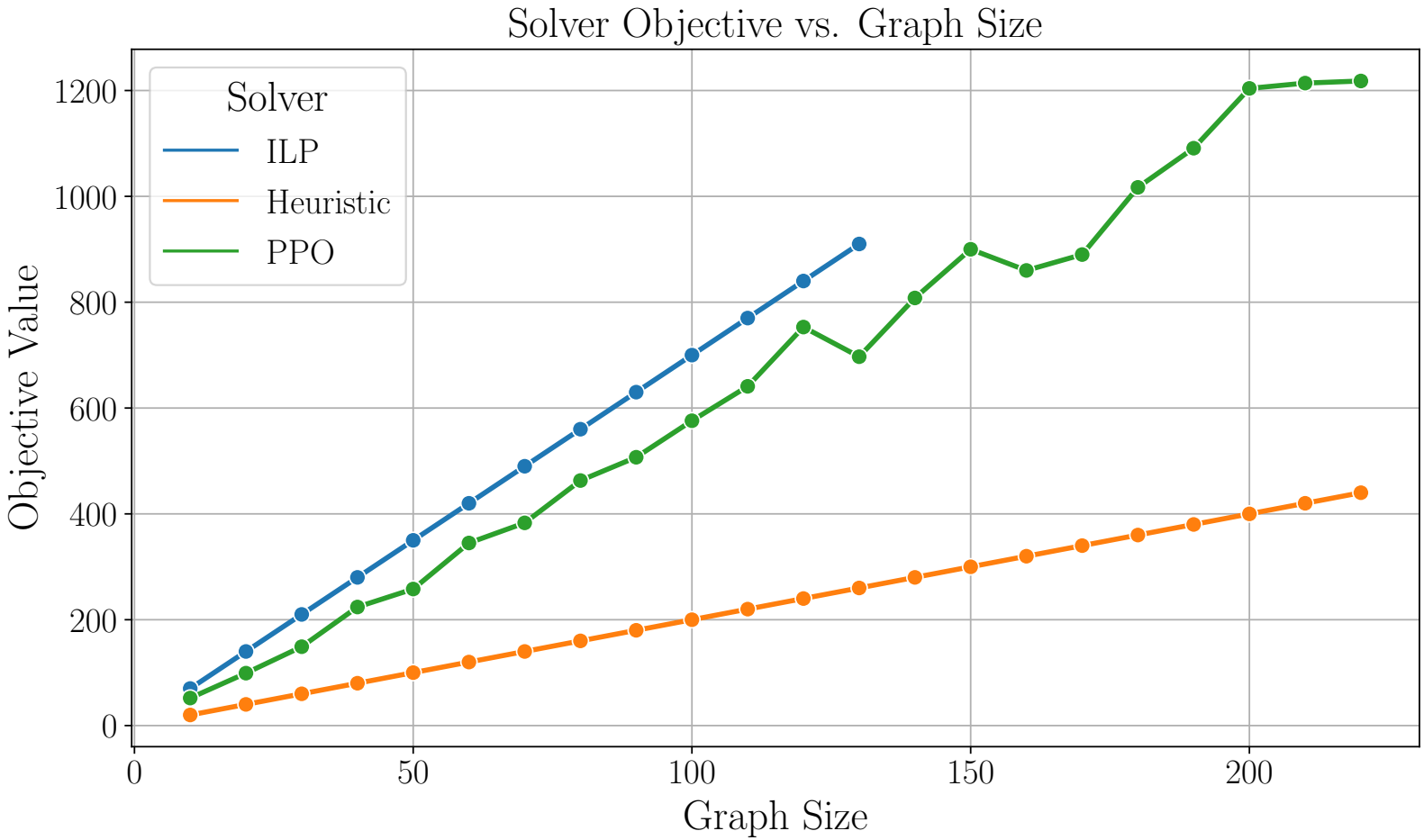
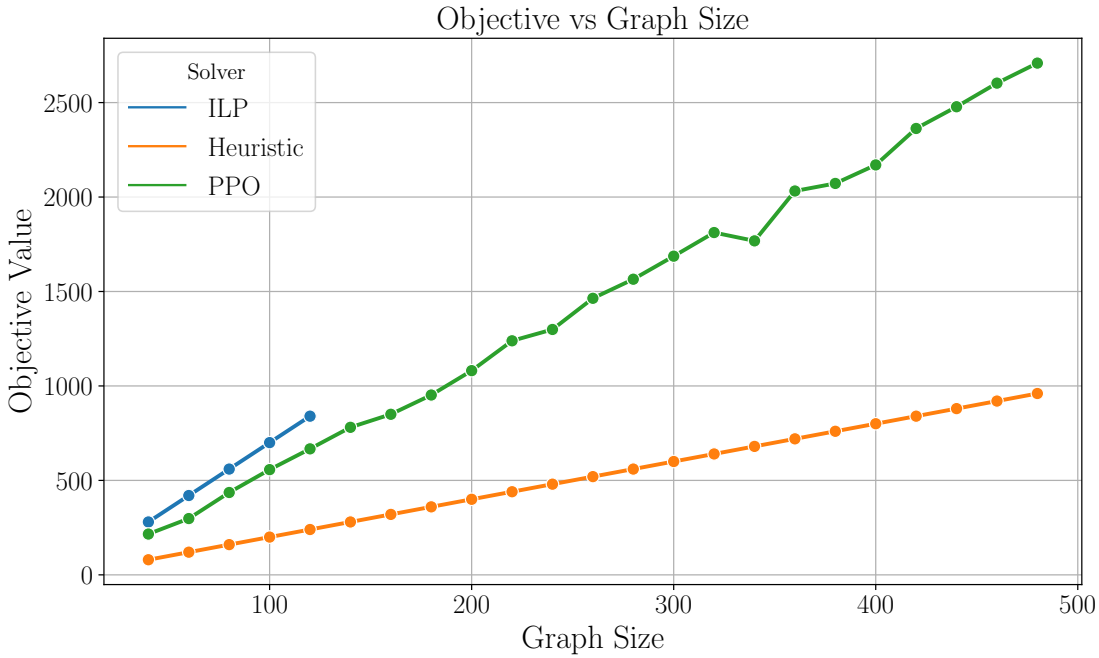
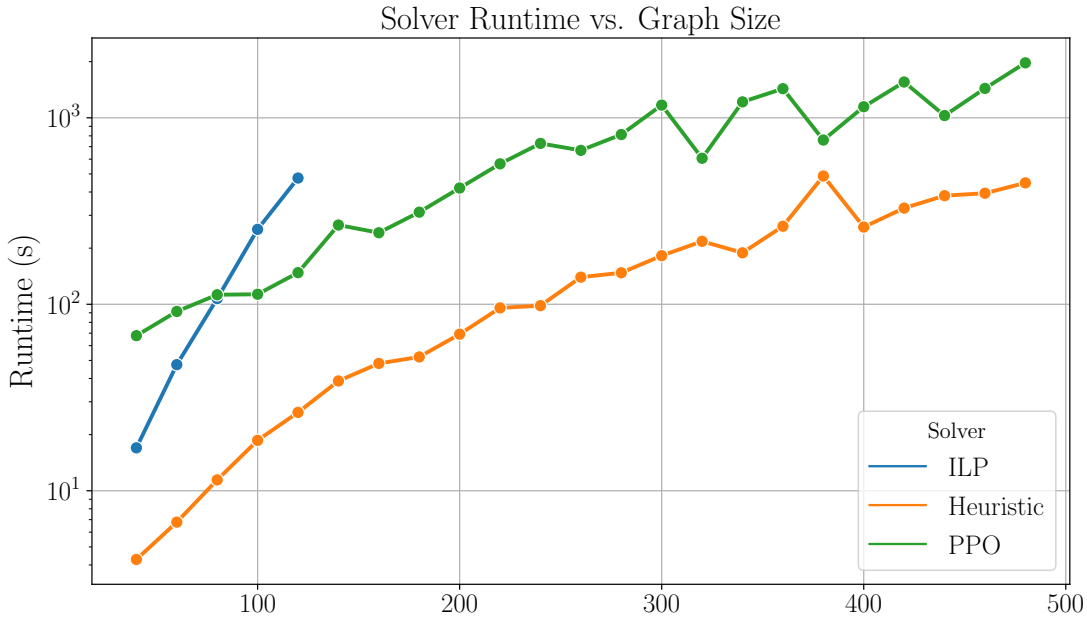


Figure 5: Tree instance: RL usually reaches a 20% optimality gap.

Performance comparison for static star instance





Dynamic Instances

RL takes advantage of learning ?

In industrial settings decision placement should be taken very often either **on the same graph** or **on a different graphs**.

We consider dynamics instances.

- Differents fixed instances close to each other on the same graphs.
- **NOT** the dynamic stochastic control problems where instances at each time step are interdependent through a stochastic evolution.

A new instance ask a new resolution of ILP (or a warm start).

Dynamic Instances – Performance and runtime

	Instance	Tree-50	Tree-100
Runtime (s)	**ILP**	80.2	266.0
	RL	**1.7**	**4.6**
Objective	**ILP**	370.0	700.0
	RL	278.5	582.7
Opt. Gap (%)	**RL**	24.7	16.8

Table 2: Average objective values and runtimes on Tree topologies.

Table of Contents

Conclusion

We presented:

- A Smart Warehouse model;
- Linear and RL solving methods with explicit features.

The RL approach:

- Scales to large instances;
- Achieves a consistent optimality gap between 15% and 25% in practice;
- Makes decisions in seconds, enabling transfer learning across instances.

Perspectives

- Improved implementations could enable deployable orchestration;
- Incorporating more technical features can reduce the optimality gap.

Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio.
Neural combinatorial optimization with reinforcement learning.
In *ICLR*, 2017.

Walid Ben-Ameur and Adam Ouorou.
Mathematical models of the delay constrained routing problem.
Algorithmic Operations Research, 1(2), 2006.

Amal Benhamiche, Wesley da Silva Coelho, and Nancy Perrot.
Routing and resource assignment problems in future 5g radio access networks.
In *International Network Optimization Conference*, 2019.

Pierre Bonami, Dorian Mazauric, and Yann Vaxès.
Maximum flow under proportional delay constraint.
Theoretical Computer Science, 689:58–66, 2017.

Moses Charikar, Yonatan Naamad, Jenifer Rexford, and X Kelvin Zou.
Multi-Commodity flow with in-network processing.

In *ALGOCLOUD 2018*, pages 73–101. Springer, 2019.


Piotr Frohlich, Erol Gelenbe, Jerzy Fiolka, Jacek Checinski, Mateusz Nowak, and Zdzislaw Filus.
Smart SDN management of fog services to optimize QoS and energy.
Sensors, 21(9):3105, 2021.


Fang Liu, Guoming Tang, Youhuizi Li, Zhiping Cai, Xingzhou Zhang, and Tongqing Zhou.
A survey on edge computing systems and tools.
Proceedings of the IEEE, 107(8):1537–1562, 2019.


Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev.
Reinforcement learning for combinatorial optimization: A survey.
Computers & Operations Research, 134:105400, 2021.


Gopika Premsankar, Mario Di Francesco, and Tarik Taleb.
Edge computing for the internet of things: A case study.
IEEE Internet of Things Journal, 5(2):1275–1284, 2018.

Farah Ait Salaht, Frédéric Desprez, and Adrien Lebre.
An overview of service placement problem in Fog and Edge Computing.
ACM Computing Surveys (CSUR), 53(3):1–35, 2020.

Vera Stavroulaki, E Calvanese Strinati, François Carrez, Yannick Carlinet, Mickael Maman, Drasko Draskovic, Drazen Ribar, Arthur Lallet, Klaus Mößner, Milenko Tosic, et al.
DEDICAT 6G-Dynamic coverage extension and distributed intelligence for human centric applications with assured security, privacy and trust: From 5G to 6G.
In *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pages 556–561. IEEE, 2021.

Kai-Chu Tsai, Lei Fan, Li-Chun Wang, Ricardo Lent, and Zhu Han.

Multi-commodity flow routing for large-scale leo satellite networks using Deep Reinforcement Learning.

In *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 626–631. IEEE, 2022.

Yang Wang, Yutong Li, Ting Wang, and Gang Liu.
Towards an energy-efficient data center network based on Deep Reinforcement Learning.
Computer Networks, 210, 2022.