

Scheduling for Real-Time Online Multi-Agent Pickup and Delivery

2024/9/1 wtcs2024

Tokyo Institute of Technology, Défago Lab

Keita Nakajima

A solid teal horizontal bar at the bottom of the slide.

Abstract

- Multi-Agent Path Finding (MAPF) problem is an extended class of pathfinding problems where multiple paths are assigned to multiple agents without collision. MAPF problem can be applied to various practical domains including path-planning of robot carriers in warehouses, automated taxiing of airplanes, and video games. Multi-Agent Pickup and Delivery (MAPD) problem that is a continuous MAPF problem has been studied for more practical cases such as automated warehouses in which robot carriers are assigned to tasks that appear over time. A major goal of MAPD problems is to reduce time to be completed or the total path length for the tasks.
- In this study, we focus on agent placement and action policies that allow agents to immediately pickup tasks wherever they occur in the environment in which they operate.

Context



robot carriers in warehouses

[Wurman+ 08]



autonomous vehicles

[Dresner+ 08]



formation with drones

[Murray 07]

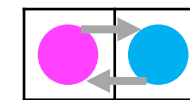
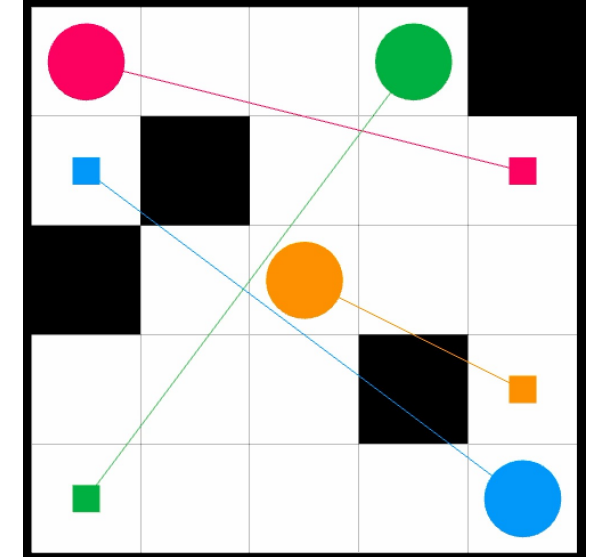
one shot
synchronized

huge gap

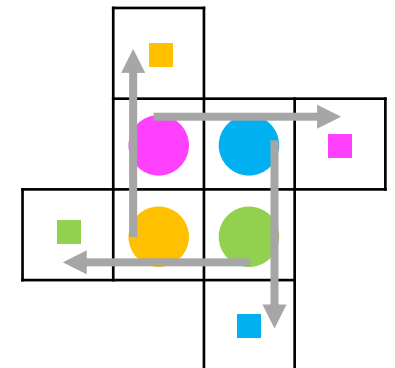
lifelong
task
assignment
asynchronized

Multi-Agent Pathfinding

[Stern+ SoCS-19]



collision



deadlock

MAPF: Multi-Agent Pathfinding

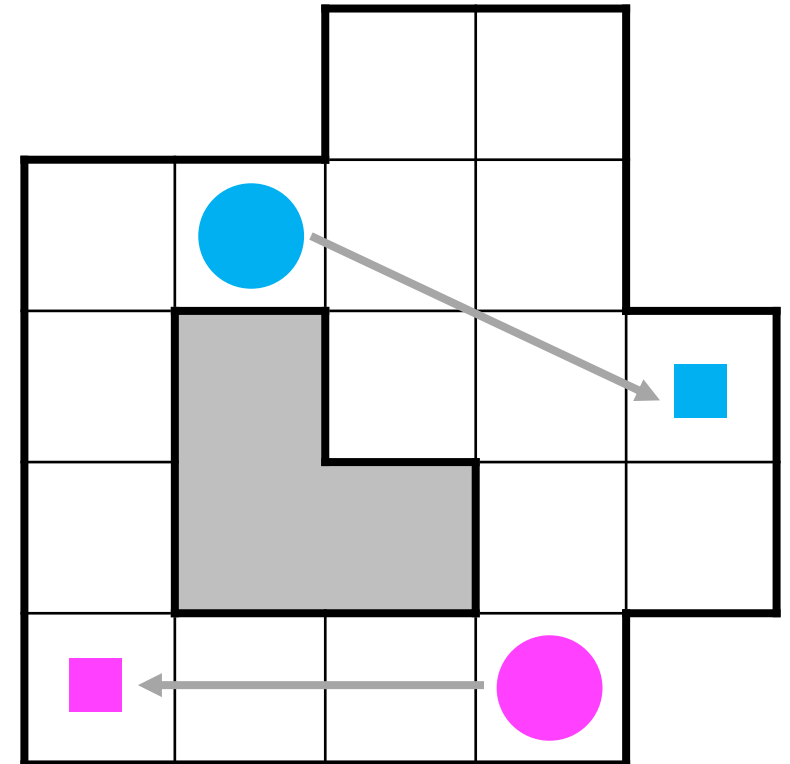
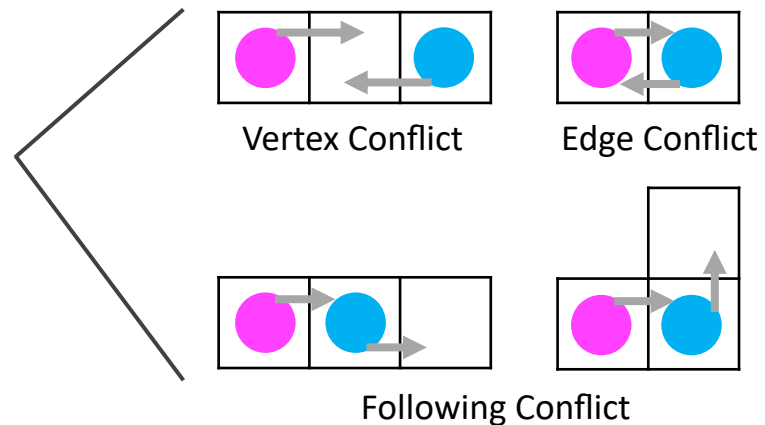
3

Input

- graph $G = (V, E)$
- agents' initial location ● , goal location ■

Output

- conflict-free paths



MAPD: Multi-Agent Pickup and Delivery

4

[Ma+ AAMAS-17]

Input

- graph $G = (V, E)$
- agents' initial location ●
- tasks (pickup ▲ , delivery ■ location)

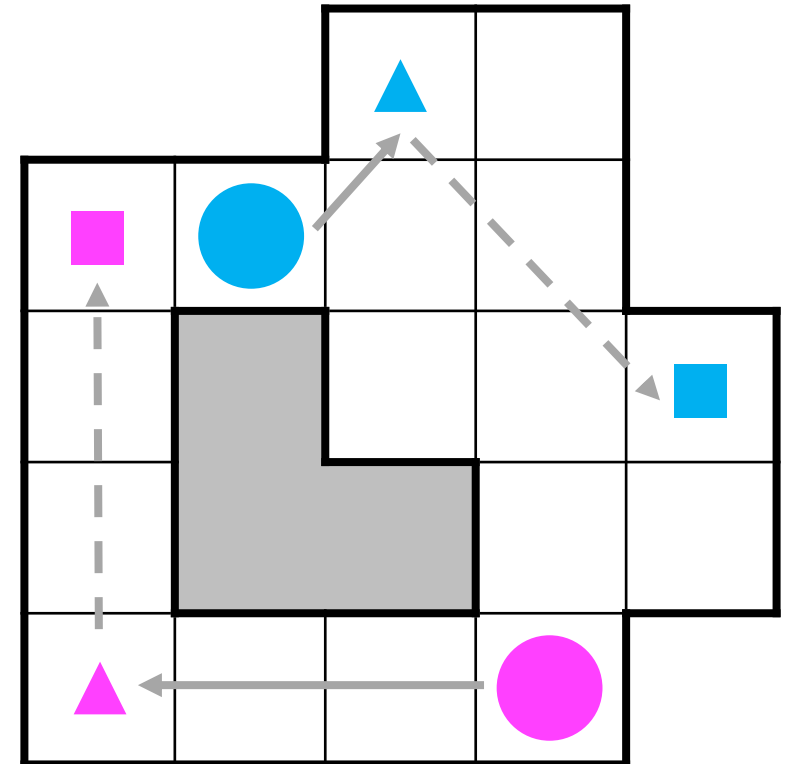
Output & Goal

- conflict-free paths
- minimize service time

└─ Time from task occurrence to completion

Constraint

- online: tasks are priori unknown

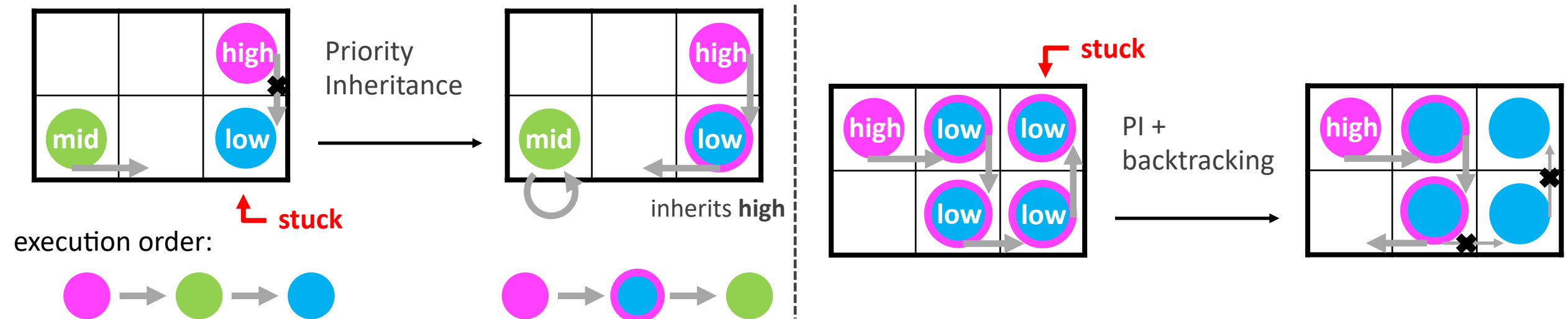


Related work

State-of-the-art algorithm (short-time horizon, decentralized)

• Priority Inheritance with backtracking (PIBT)* [Okumura+ AIJ-22]

- extremely scalable (suboptimal), solving iterative MAPF
- ensure reachability in biconnected-like graph



*Not considering following conflict

Research Questions

Problem: Multi-Agent Pickup and Delivery (task assignment + path planning)

Goal: Minimize the **service time** of arbitrary tasks

└─→ PIBT solves sub-optimally

RQ1 : How to improve task assignment?

RQ2 : How free agents handling impacts system performance?

Issue : PIBT at MAPD

7

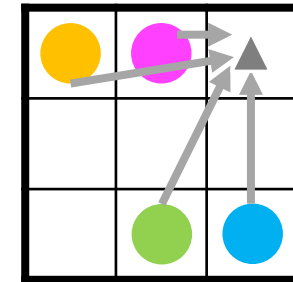
Naïve solution (First arrival)

Policy:

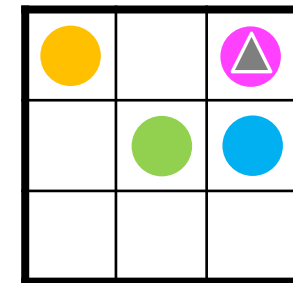
let all free agents move to the **nearest pickup** location

Consequences:

- **only one agent** can be in charge of the task
 - └ other agents' actions are wasteful
- prone to **localized traffic congestion**
 - └ high frequency of following conflict



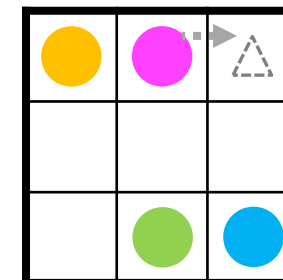
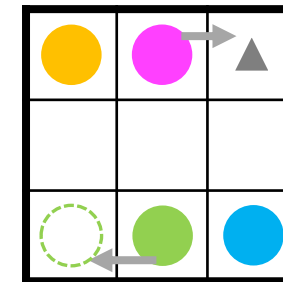
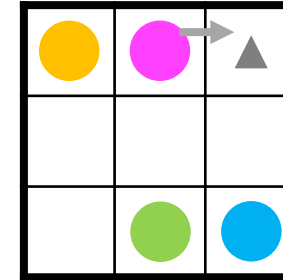
↓ next step



Proposal from issue

Policies:

- **Matching** with agents and tasks
 - + reduce wasteful movement
 - + avoid localized traffic congestion
- **Balancing**: equally distribute agents
 - + avoid **further** localized traffic congestion
- **Predicting** the distribution of task occurrences
 - + fundamentally speed up task pickup time



Method of handling free agents

Greedy matching

1. for each task, find the most appropriate agent among all free agents

2. calculate **matching cost** of agent a and task τ at time t :

$$matching_cost(a, \tau) = dist(a_{loc}, \tau_{pickup_loc}) + w(t - \tau_{release_time})$$

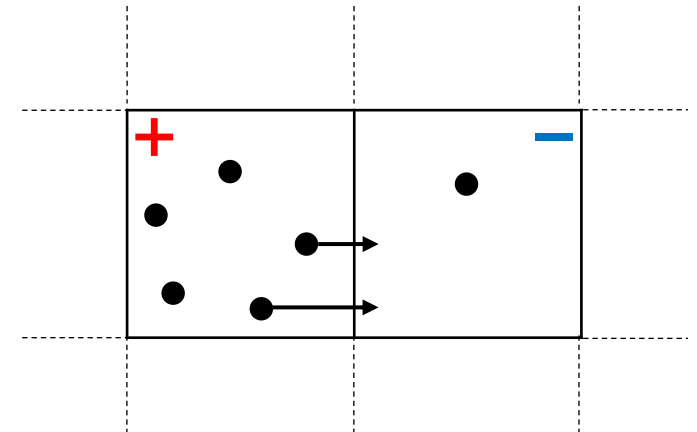
3. sort matching costs in descending order and set target from the top

Repeat while
of free agents > 0 and
of untargeted tasks > 0

Balancing

1. divides the region and calculates the **density of free agents**

2. selected and moved to have the same density in adjacent regions



Experiment

Main parameters

- free agents heuristics { first arrival, greedy, greedy + balancing }
- number of agents [500, 1000, 1500, 2000, 2500]
- average task arrival rate [0.2, 1, 5]

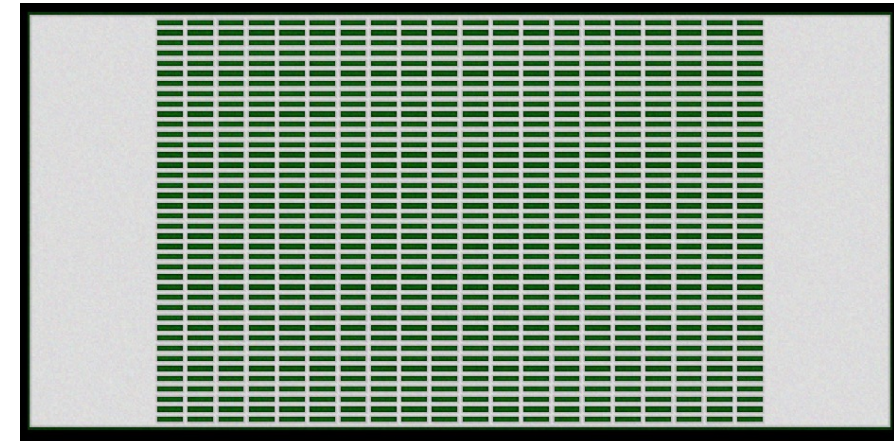
Metrics

- **pickup timestep**: time from task occurrence to pickup
- **throughput**: completed tasks rate at given timestep
- **computation time***

Environment

- C++, M2 macbook air 24GB

Simulation map image:



warehouse-20-40-10-2-2.map

H = 340, W = 164

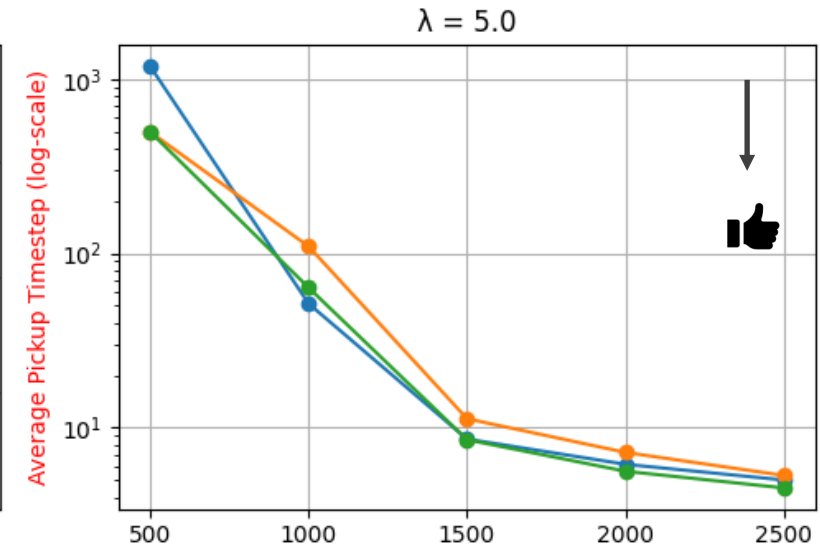
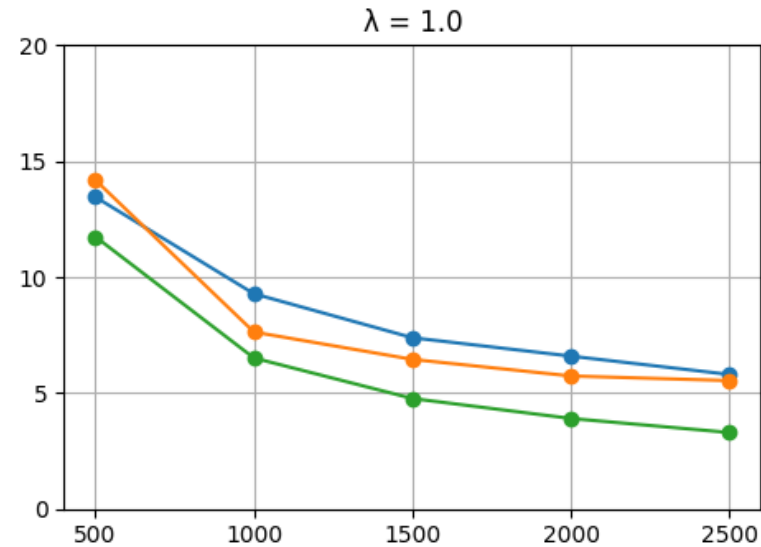
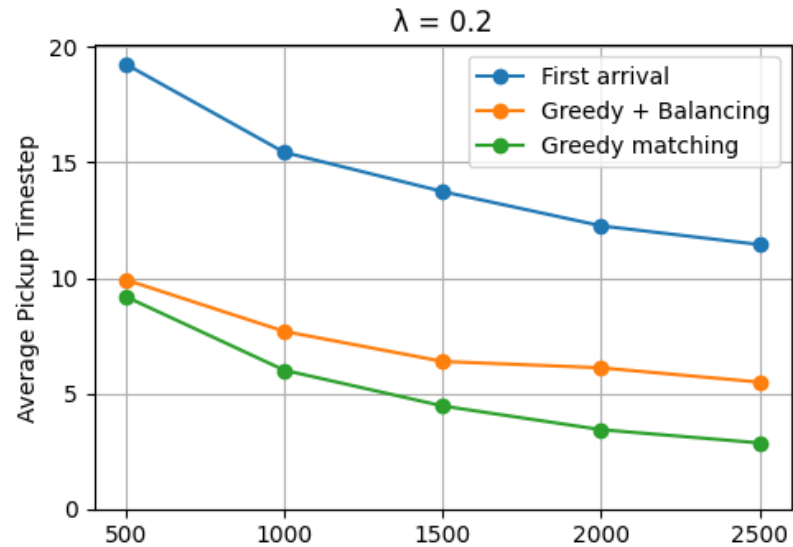
38,756 vertices

[Stern+ SoCS-19]

*Results are in appendix

Result : Pickup timestep

Average task arrival rate λ [task / timestep]

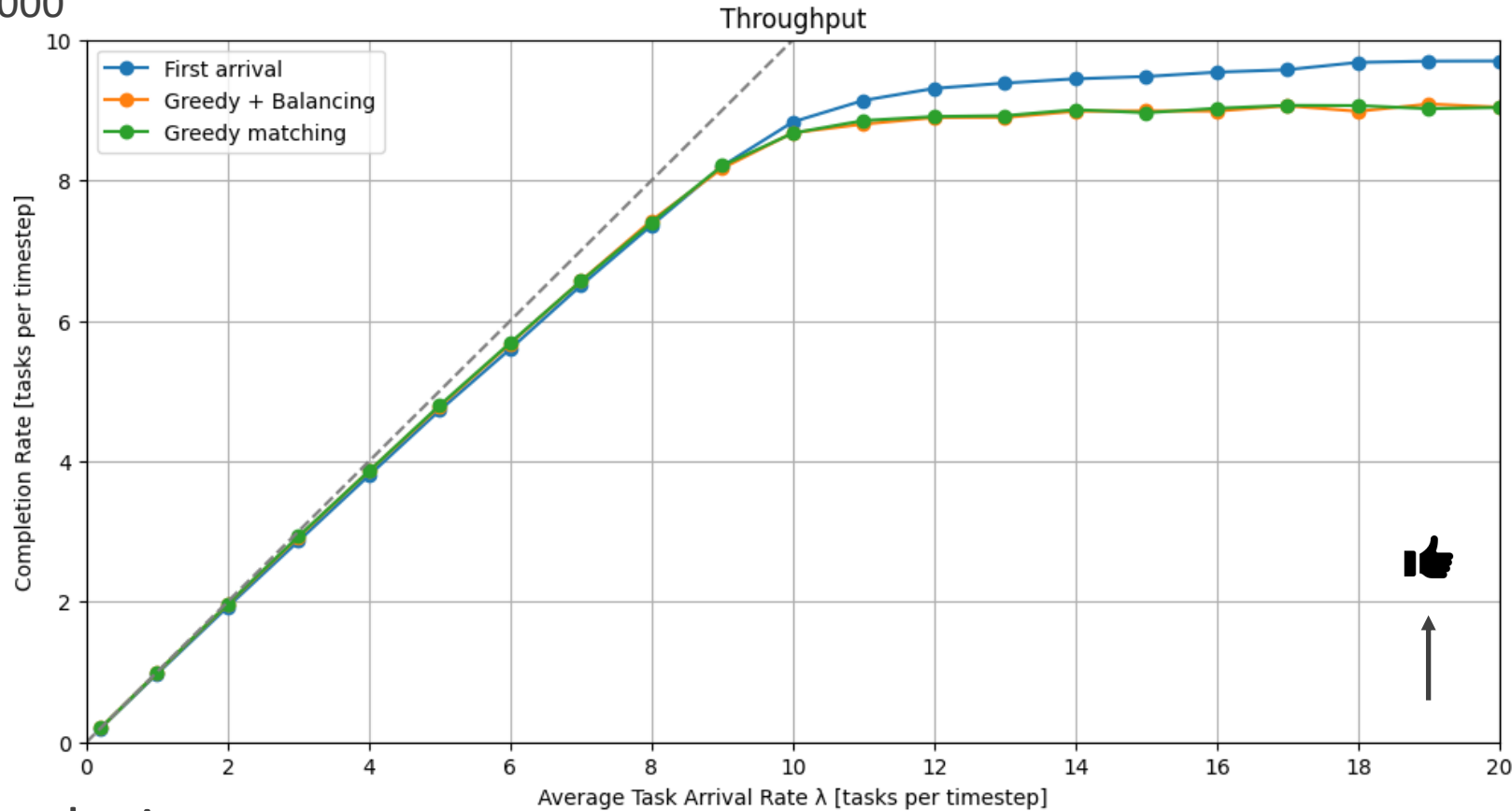


Quick conclusion

- Free agent handling has potential to improve solution quality
- Time to pickup is highly dependent on **traffic conditions** and **agent busyness**

Result : Throughput

Number of agents : 2000



Quick conclusion

- First arrival method maintains slightly better throughput at higher rates
- All methods eventually reach a saturation point at $\lambda = 10$

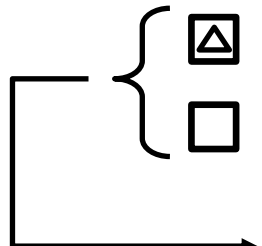
Summary

Problem: Multi-Agent Pickup and Delivery (task assignment + path planning)

Goal: Minimize the **service time** of arbitrary tasks

- ☐ RQ1 : How to improve task assignment?
- ☒ RQ2 : How free agents handling impacts system performance?
 - Trade-off: faster pickup vs. more costly and time-consuming

Roadmap

- ☐ Propose and evaluate the method to handle free agents
 - ☒ greedy matching
 - ☒ balancing
 - ☐ predicting
 - ☐ Build a theory-based rationale
 - Improved balancing and prediction accuracy based on theoretical evidence
- 

Reference

- [Dresner+ 08] Kurt Dresner and Peter Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research (JAIR)*, 2008.
- [Ma+ 17] Hang Ma, Jiaoyang Li, T. K. Satish Kumar, and Sven Koenig. Lifelong multi-agent path finding for online pickup and delivery tasks. In *Proceedings of International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, 2017.
- [Murray 07] Richard M. Murray. Recent Research in Cooperative Control of Multivehicle Systems. ASME. *J. Dyn. Sys., Meas., Control.*, 2007
- [Okumura+ 22] Keisuke Okumura, Manao Machida, Xavier Défago, and Yasumasa Tamura. Priority inheritance with backtracking for iterative multi-agent path finding. *Artificial Intelligence (AIJ)*, 2022.
- [Stern+ 19] Roni Stern, Nathan Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, TK Kumar, et al. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of Annual Symposium on Combinatorial Search (SOCS)*, 2019.
- [Wurman+ 08] Peter R Wurman, Raffaello D’Andrea, and Mick Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine*, 2008.

Appendix. Result : Computation time [ms]

Average task arrival rate λ [task / timestep]

