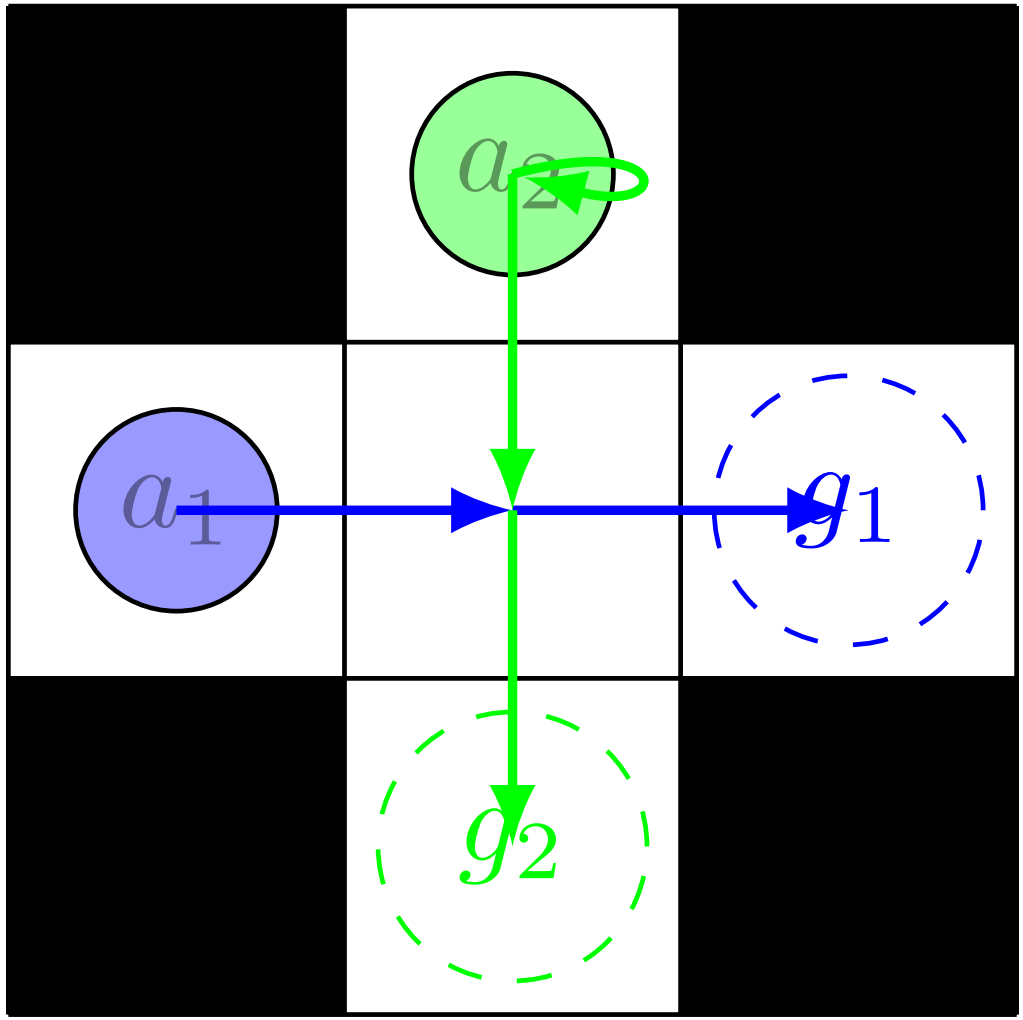
Lifelong Multi-Agent Path Finding for Online Pickup and Delivery Tasks

Hang Ma Jiaoyang Li T. K. Satish Kumar Sven Koenig
University of Southern California
Tsinghua University

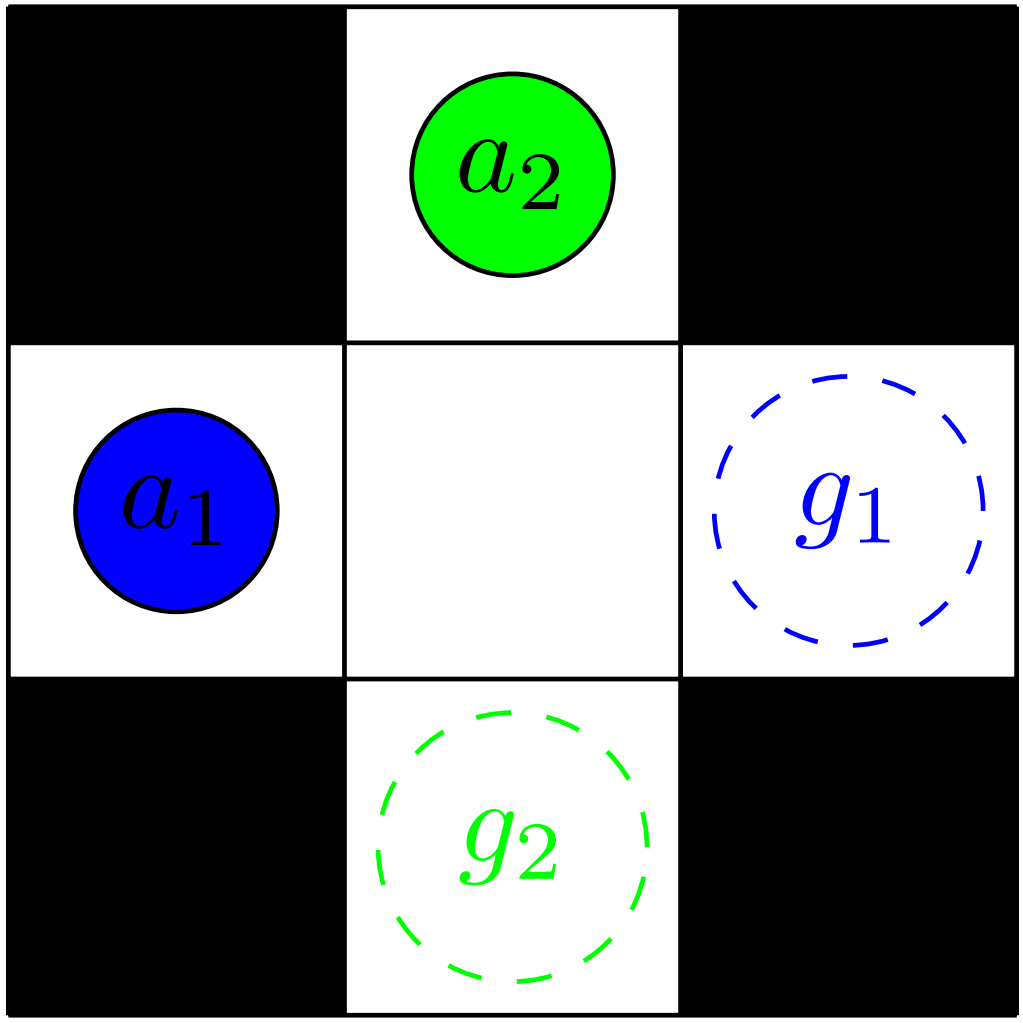
May 11, 2017
AAMAS

Multi-Agent Path Finding

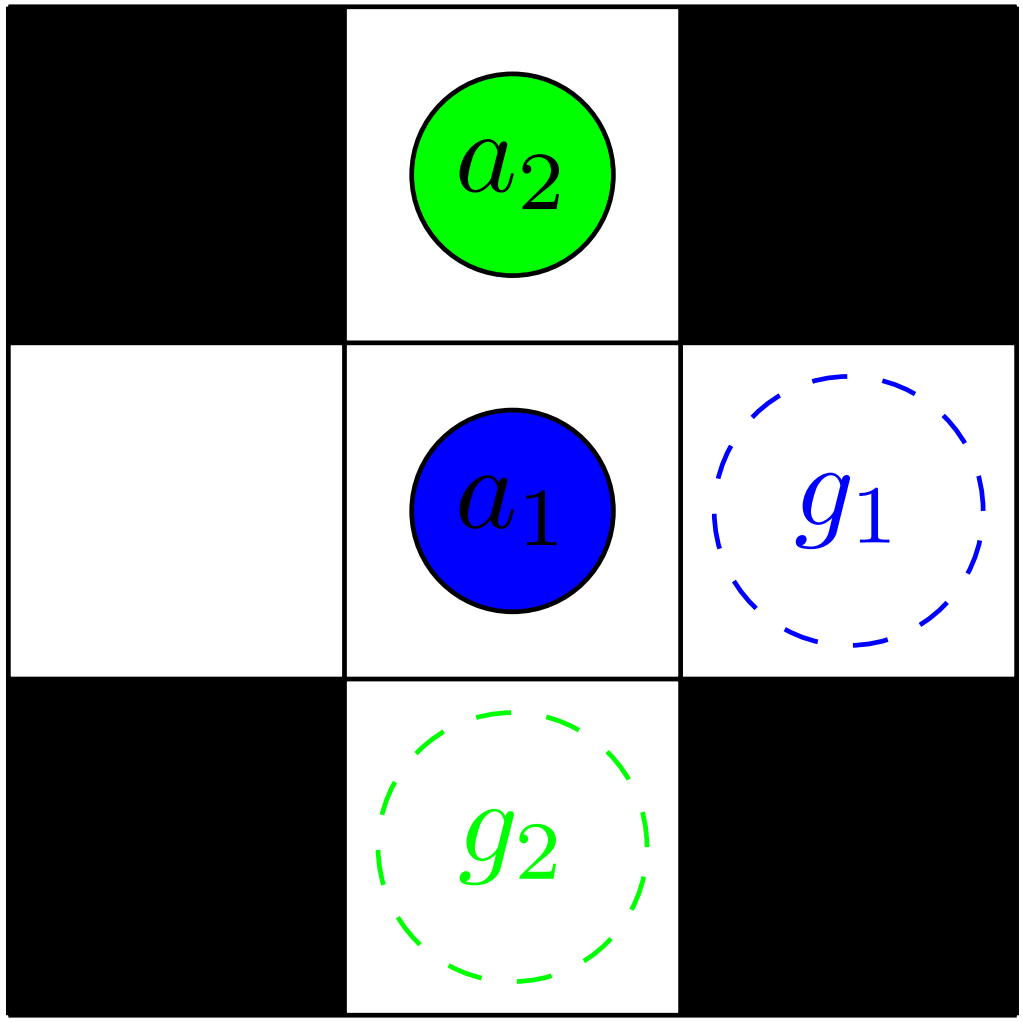
Find collision-free paths for all agents from their current locations to their predefined goal locations in a known environment.



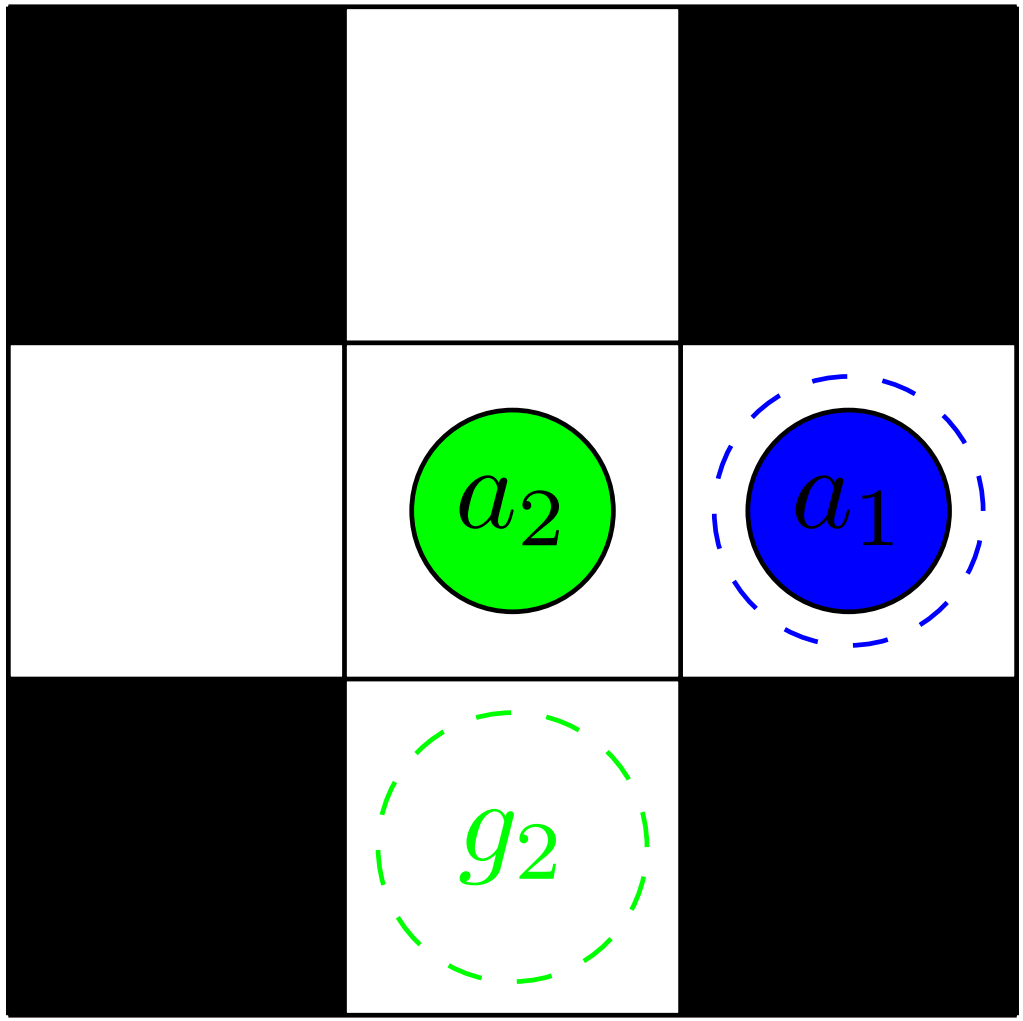
Time Step 0



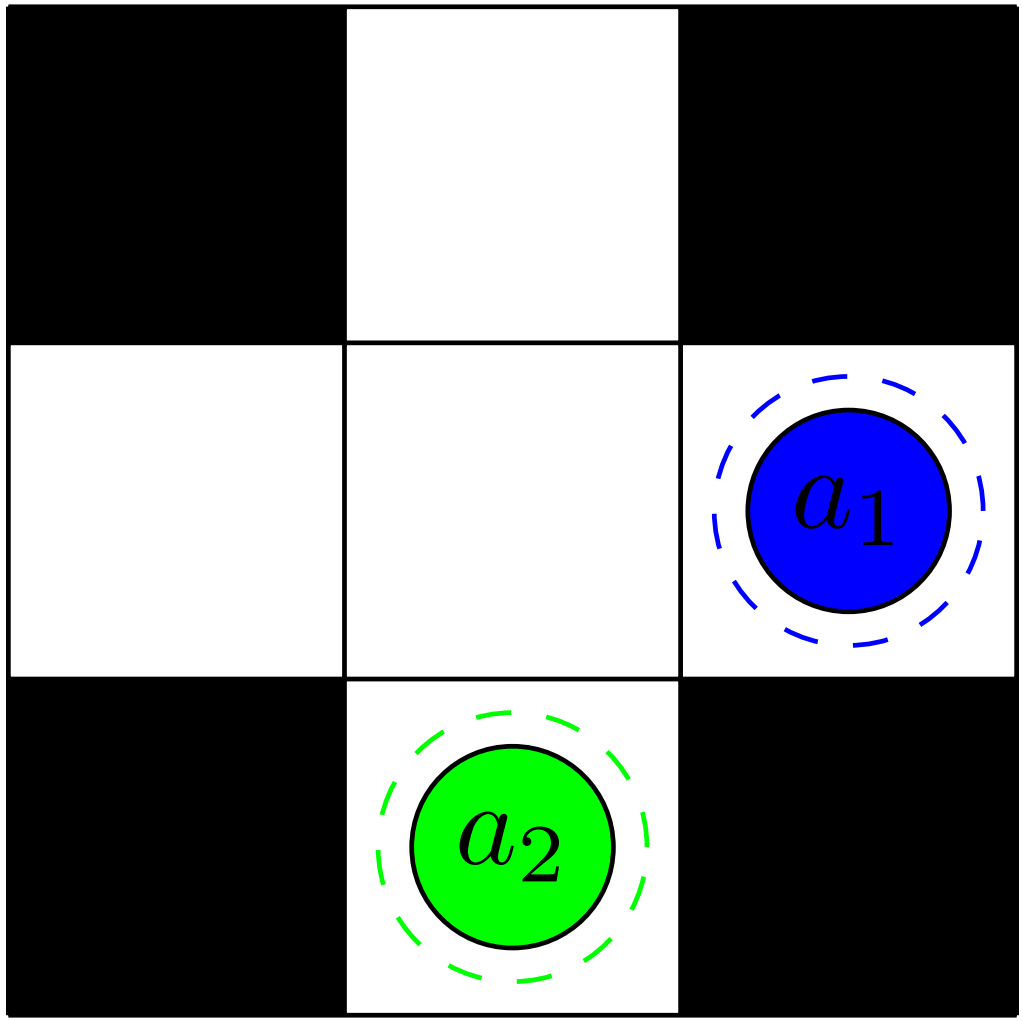
Time Step 1



Time Step 2



Time Step 3

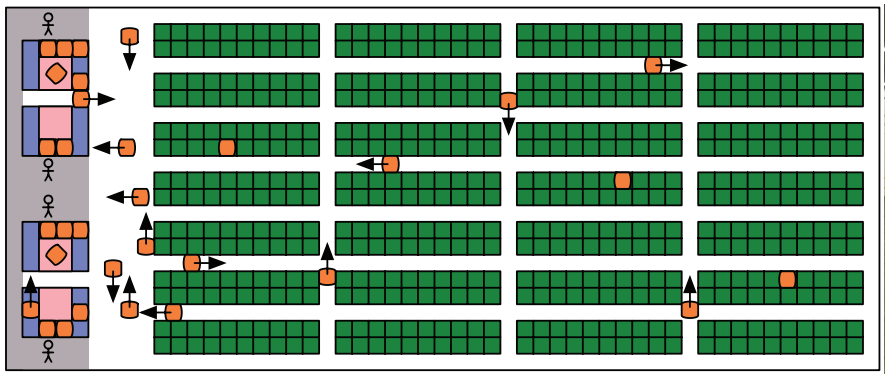


Motivated by Real-World Applications:

Automated aircraft-towing vehicles, warehouse robots, office robots, and game characters in video games.

Amazon Warehouse Robots[1]

Tasks: Move inventory shelves from storage locations to inventory stations or vice versa.



[1] P. R. Wurman, R. D'Andrea, and M. Mountz. "Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses". In: *AI Magazine* 29.1 (2008), pp. 9–20.

Multi-Agent Pickup and Delivery (MAPD) Problem

- Existing research on multi-agent path finding — a "one-shot" version:
  One pre-determined task for each agent — navigates to its goal location.
- MAPD — a "lifelong" version of multi-agent path finding:
  - A task can enter the system at any time.
  - Agents have to constantly attend to a stream of new tasks.

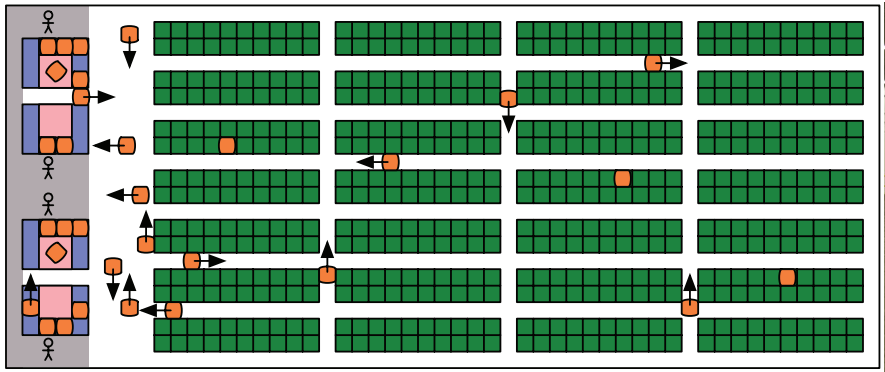
MAPD Algorithms

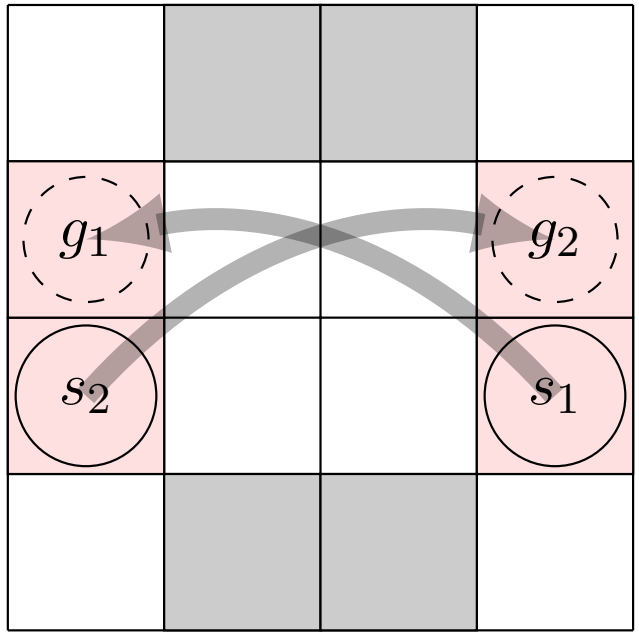
1. Decoupled Task Assignment and Path Finding
   - Token Passing (**TP**): Greedy task assignment and no task reassignment.
   - Token Passing with Task Swaps (**TPTS**): Local task reassignment between two agents.
2. Centralized Task Assignment and Path Finding
   CENTRAL

Roughly:

- Effectiveness: TP $<$ TPTS $<$ CENTRAL
- Efficiency: CENTRAL $<$ TPTS $<$ TP

Tasks

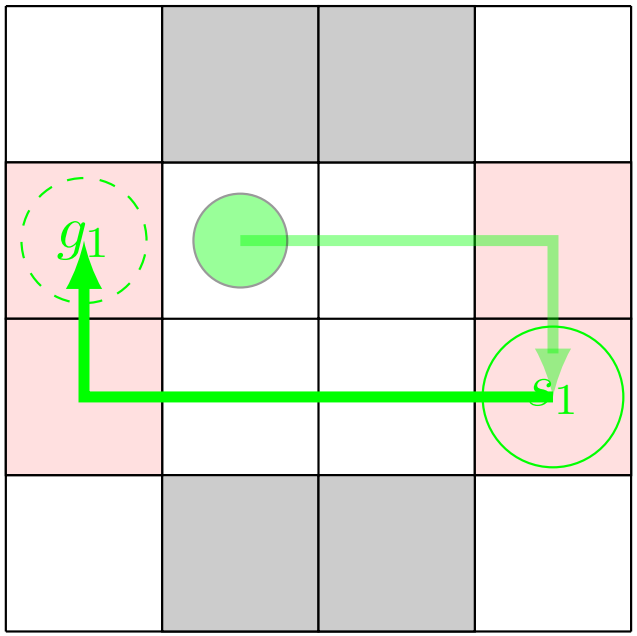




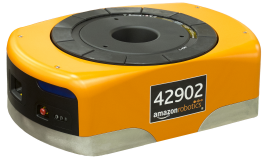
Executing Task

In order to execute a task, the agent has to move from its current location via the pickup location to the delivery location:

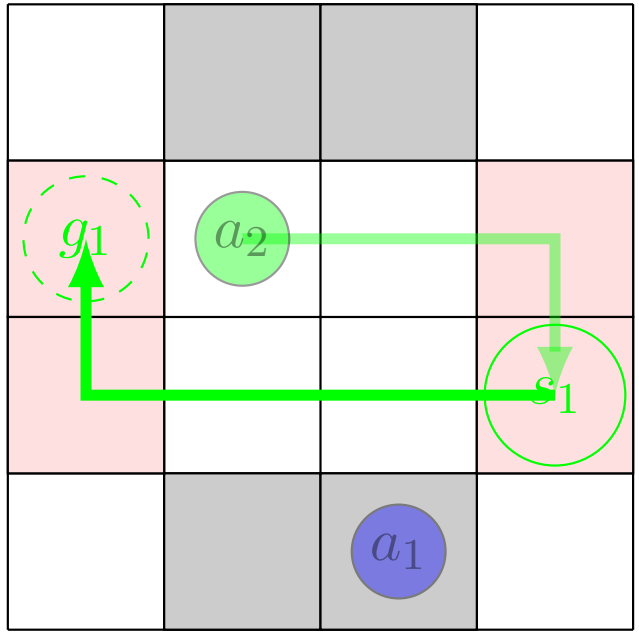
1. When the agent reaches the pickup location, it starts to execute the task.
2. When it reaches the delivery location, it finishes the task.



Free Agents

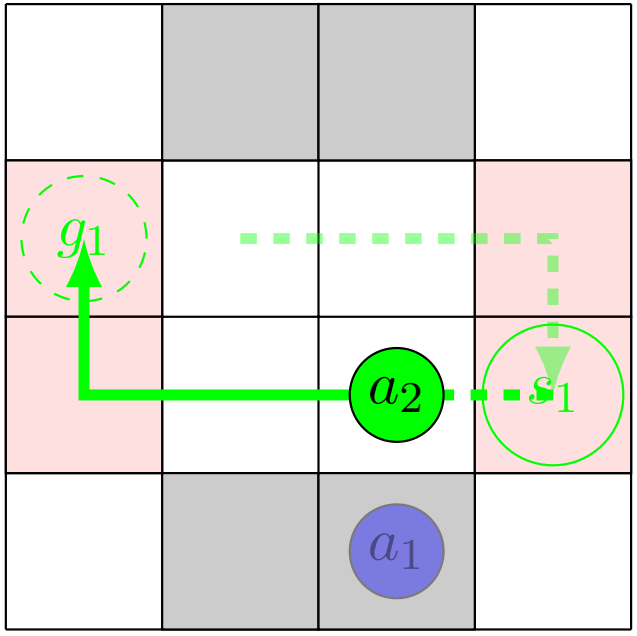


Free Agents:



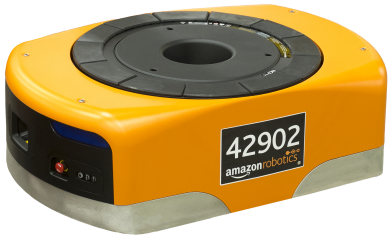
Occupied Agents

Occupied Agents:



Assignment of Agents to Tasks

- A free agent can be assigned to any unexecuted task.



- An occupied agent has to finish executing its current task.

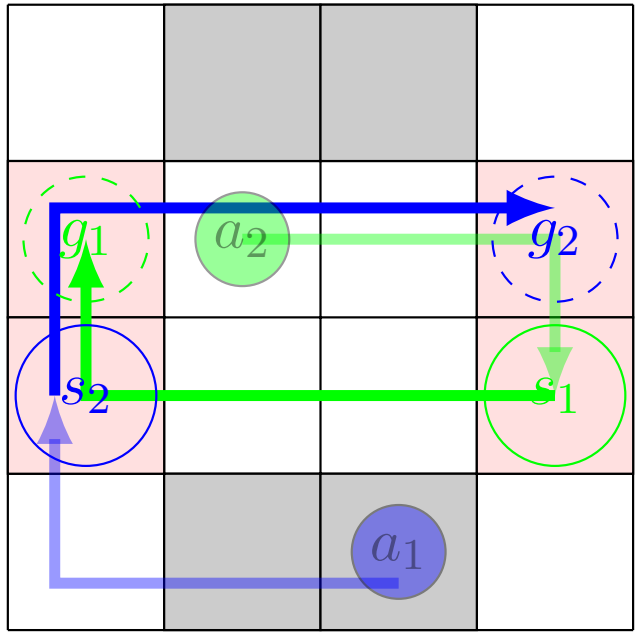
Objective of MAPD

Finish executing each task as quickly as possible.

Effectiveness of a MAPD algorithm

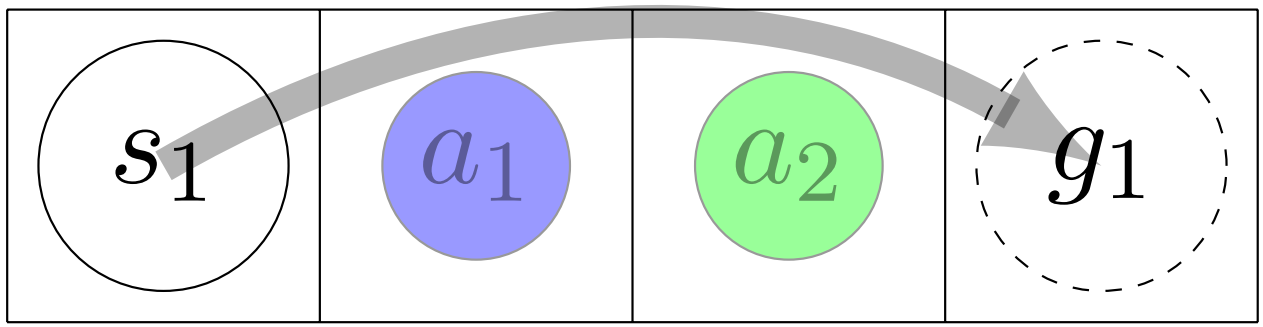
Service time: the average number of timesteps needed to finish executing each task after it enters the system.
An algorithm solves a MAPD instance $\iff$ Service time of all tasks is bounded.

Service time: $\frac{7+7}{2} = 7$



Solvability

Not every MAPD instance is solvable.



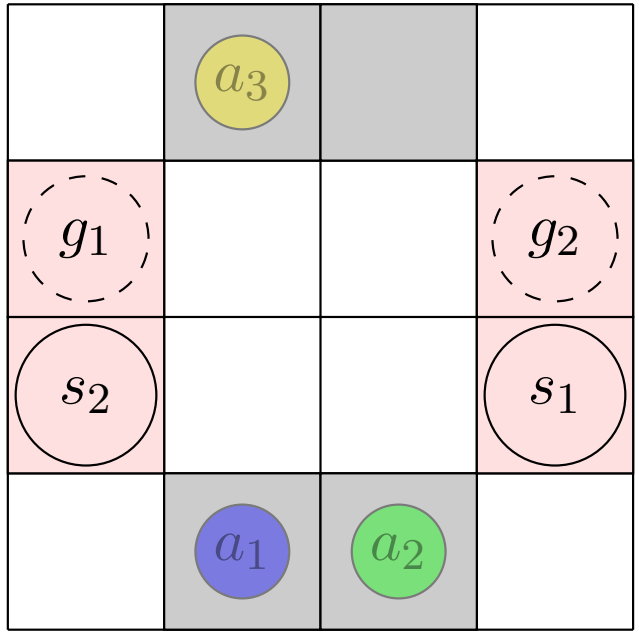
Well-Formed MAPD Instances

Being **well-formed** (based on [M. Cáp et al 2015][2]): a sufficient condition that makes MAPD instances solvable.
Intuition: agents should only be allowed to rest (that is, stay forever) in locations, called **parking locations**, where they cannot block other agents.

[2] M. Cáp, J. Vokrínek, and A. Kleiner. "Complete Decentralized Method for On-Line Multi-Robot Trajectory Planning in Well-formed Infrastructures". In: *International Conference on Automated Planning and Scheduling*. 2015, pp. 324–332.

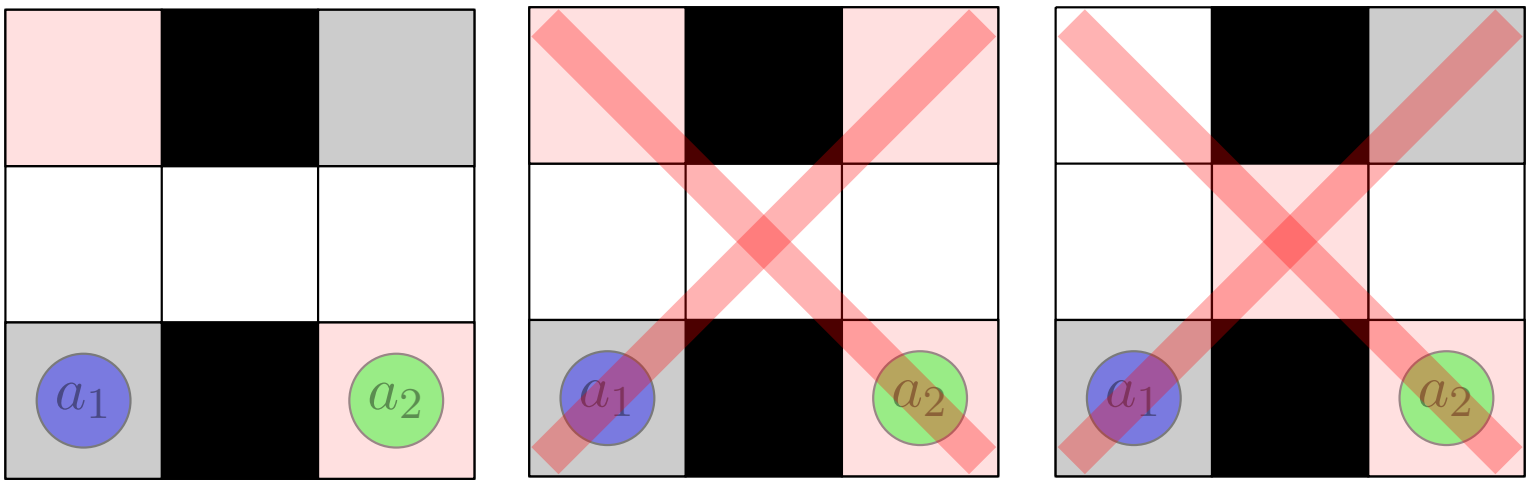
Parking Locations

- **Task Parking Locations**: all pickup and delivery locations of tasks
  (storage locations, inventory stations, etc.)
- **Non-task Parking Locations**:
  - All initial locations of agents
  - Additional designated parking locations



Well-Formed MAPD Instances

1. # tasks is finite;
2. # non-task parking locations $\geq$ # agents;
3. For any two parking locations, there exists a path between them that traverses no other parking locations.



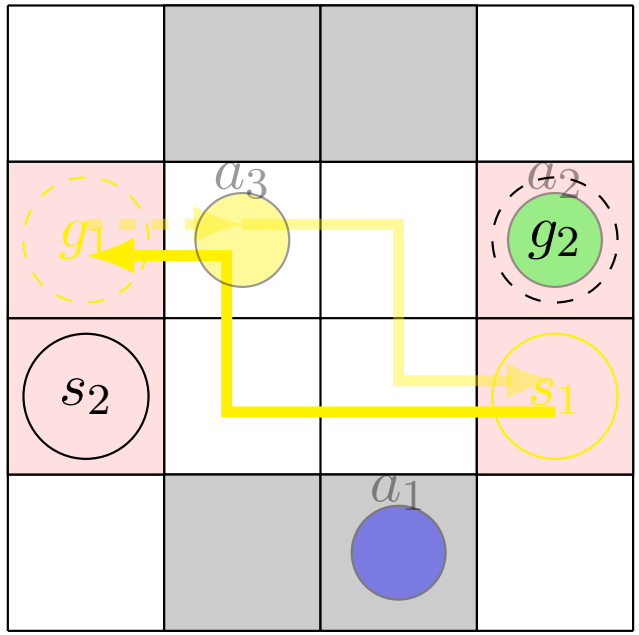
MAPD Algorithms

We present

1. Two Decoupled Algorithms:
   complete for well-formed MAPD instances (solve all well-formed instances)
   - Token Passing (**TP**)
   - Token Passing with Task Swaps (**TPTS**)
2. One Centralized Algorithm:
   CENTRAL

A Running Example

Unexecuted Tasks: $task_1$, $task_2$.
Agent a_1 and agent a_2 are resting.
Agent a_3 is assigned to $task_1$ and on the way to the pickup location s_1.



Token Passing (TP)

Based on an idea similar to Cooperative A*[3]:

- Token: a synchronized shared block of memory that contains the current paths of all agents, set of unexecuted task, and agent assignments.
- Only one agent has access to the token at each time.
- Each agent assigns itself a task, plan its path, and passes the token to the next agent.

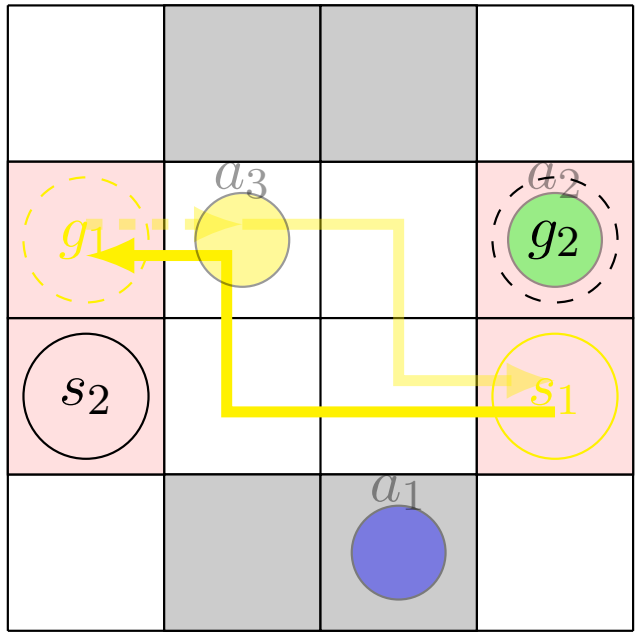
[3] D. Silver. "Cooperative Pathfinding". In: *Artificial Intelligence and Interactive Digital Entertainment*. 2005, pp. 117–122.

TP: Key Idea

- A task can only be assigned once.
- Once an agent is assigned to a task, it cannot be assigned to other tasks until it finishes the task.

TP: Running Example

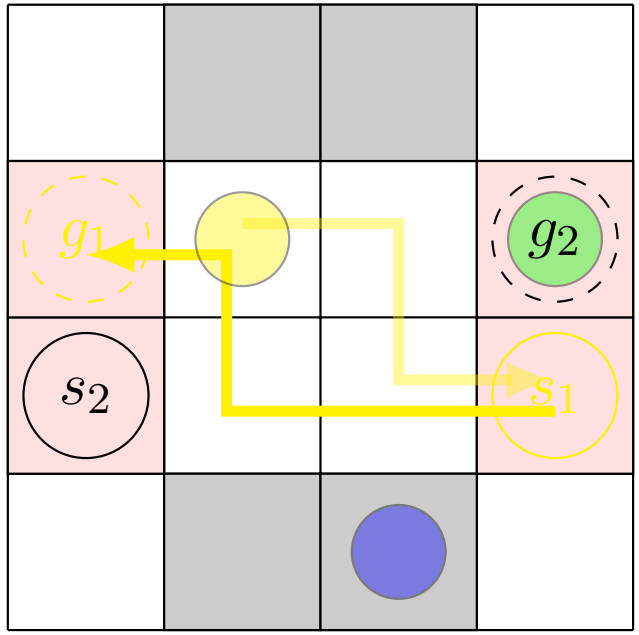
Task Available for Assignment: $task_2$.
Agent a_1 and agent a_2 request for token.



TP: Agent a_1's Turn

Agent a_1 Has Token

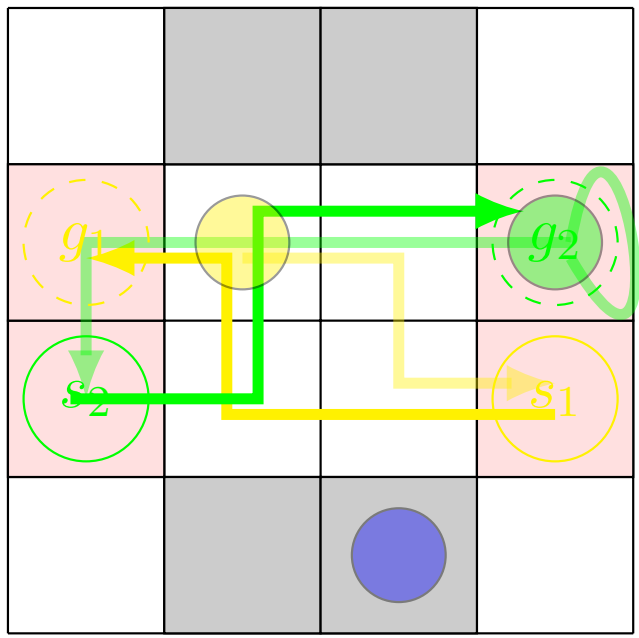
1. it cannot assign itself to any task because agent a_2 rests in g_2, the only task available to it;
2. it has to rest in a parking location that will not create any deadlock;
3. it can continue to rest in its current location.



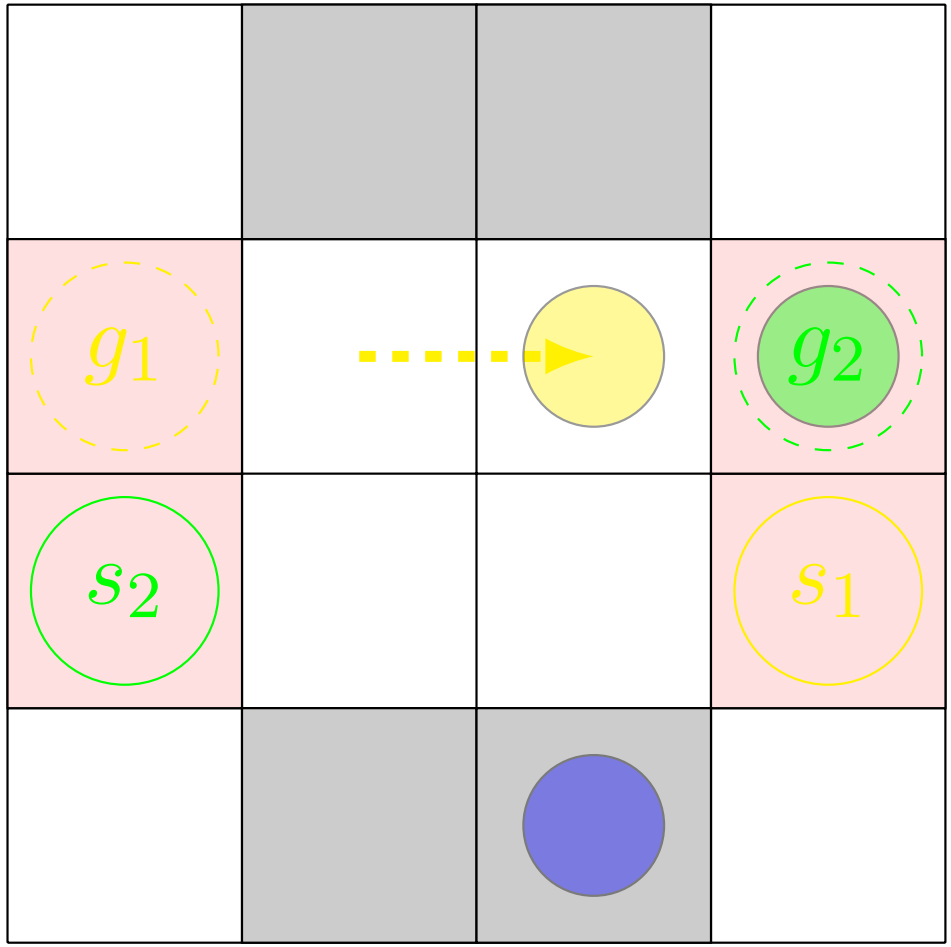
TP: Agent a_2's Turn

Agent a_2 Has Token

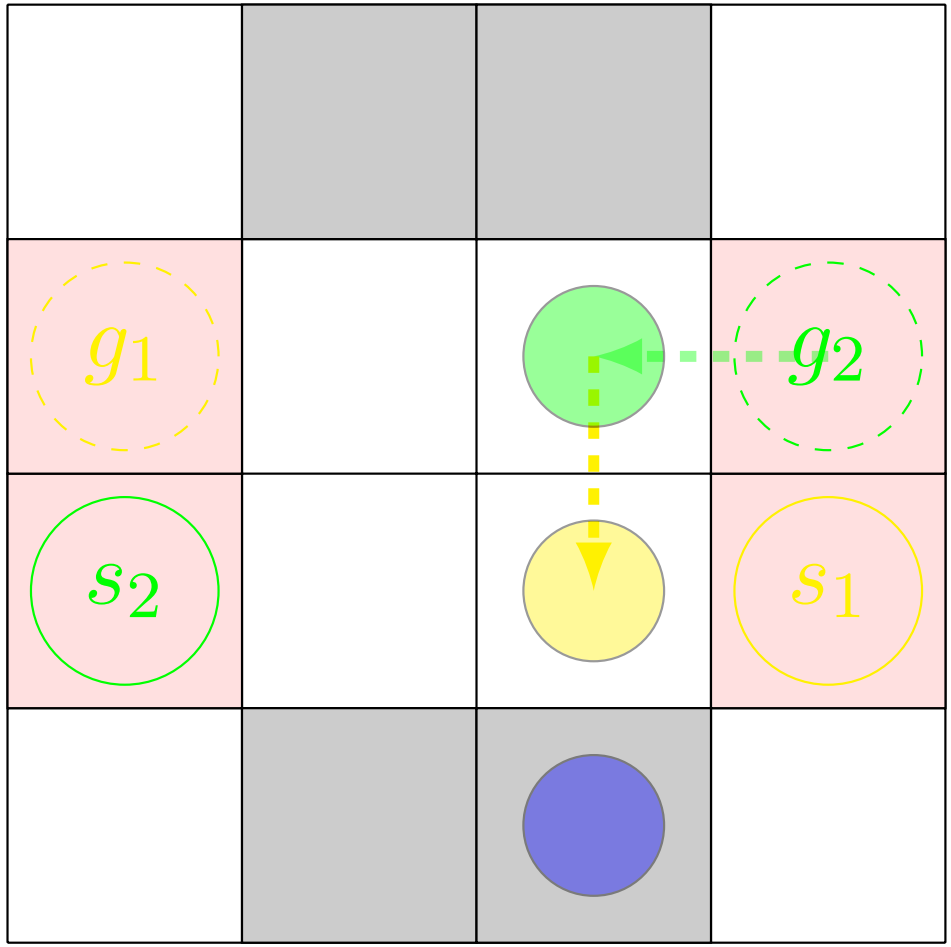
1. it assigns itself to $task_2$;
2. $task_2$ is no longer available to other agents;
3. it plans a cost-minimal collision-free path to execute $task_2$.



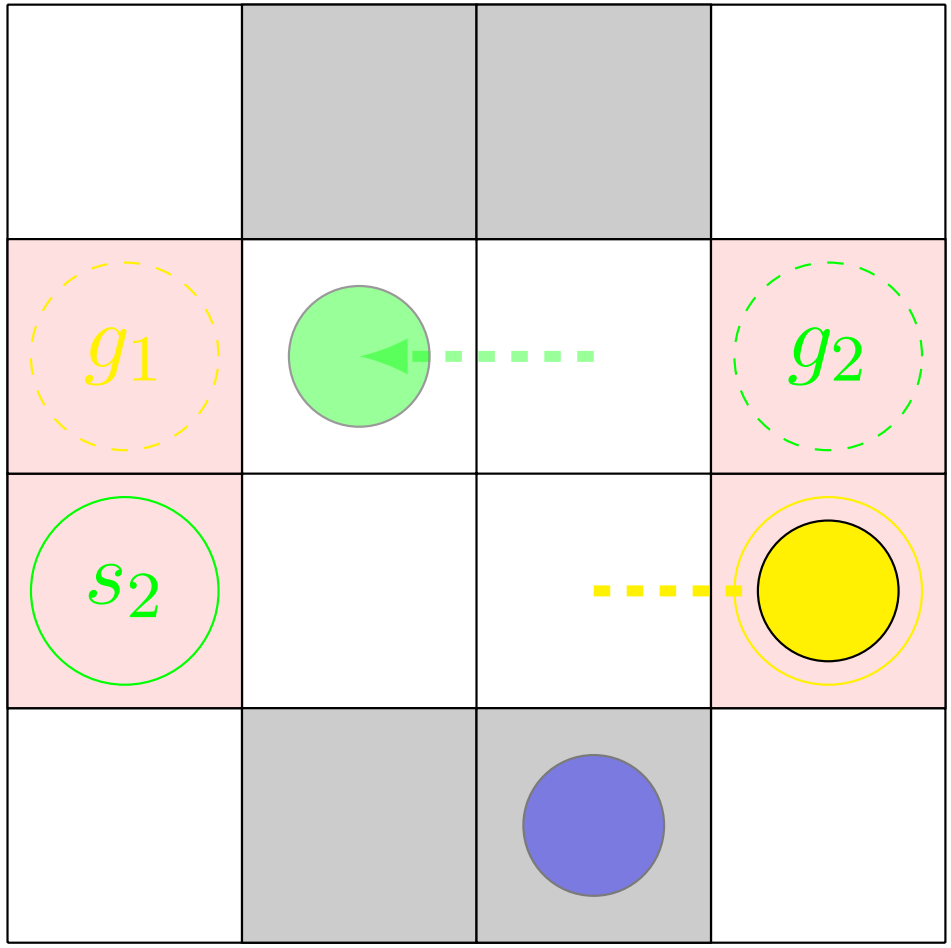
TP: Animation



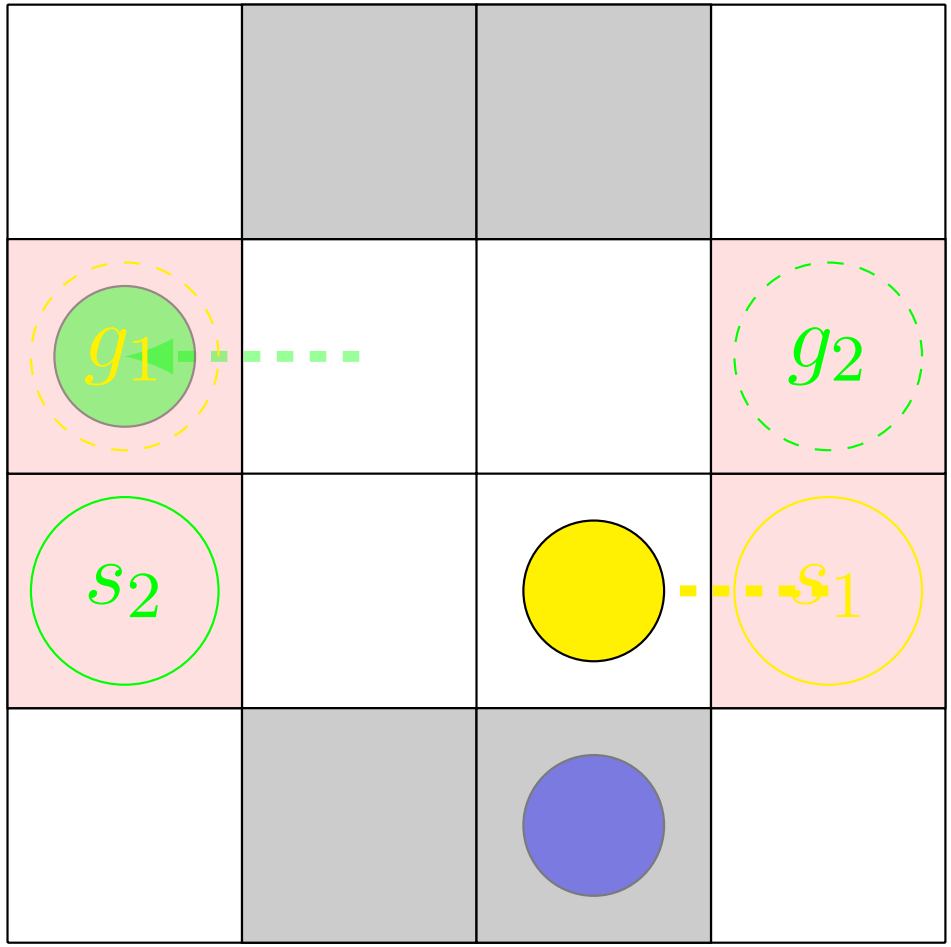
TP: Animation



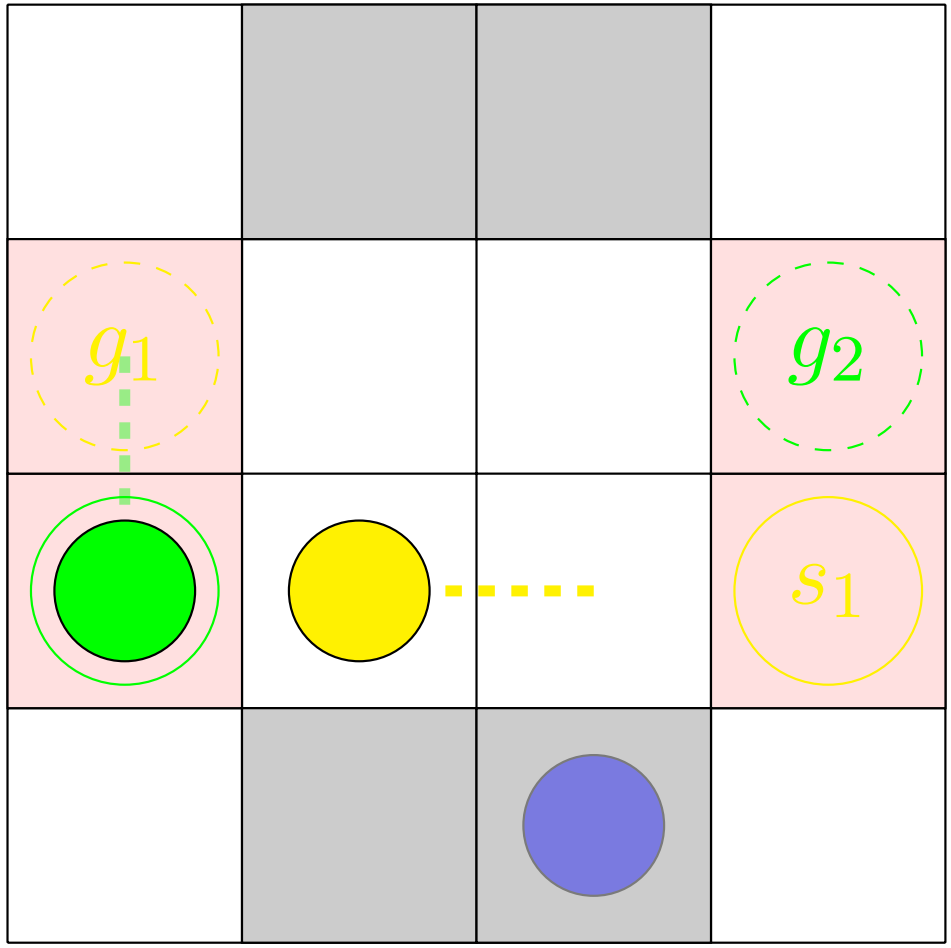
TP: Animation



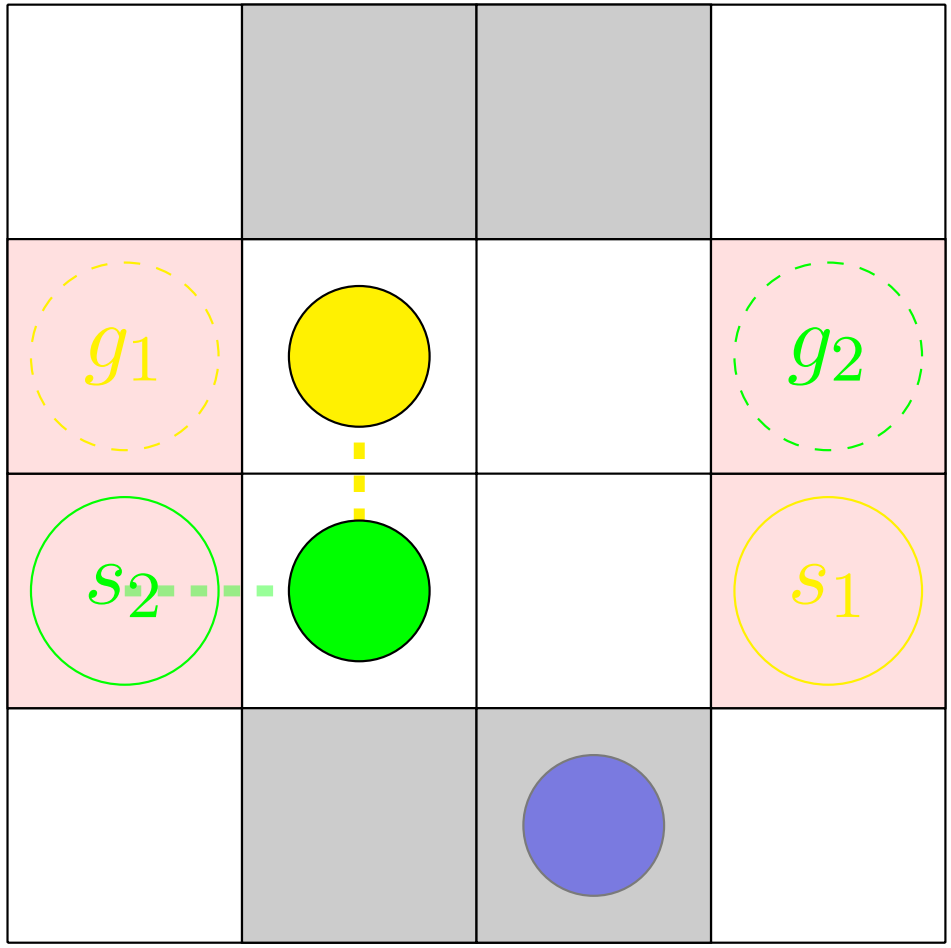
TP: Animation



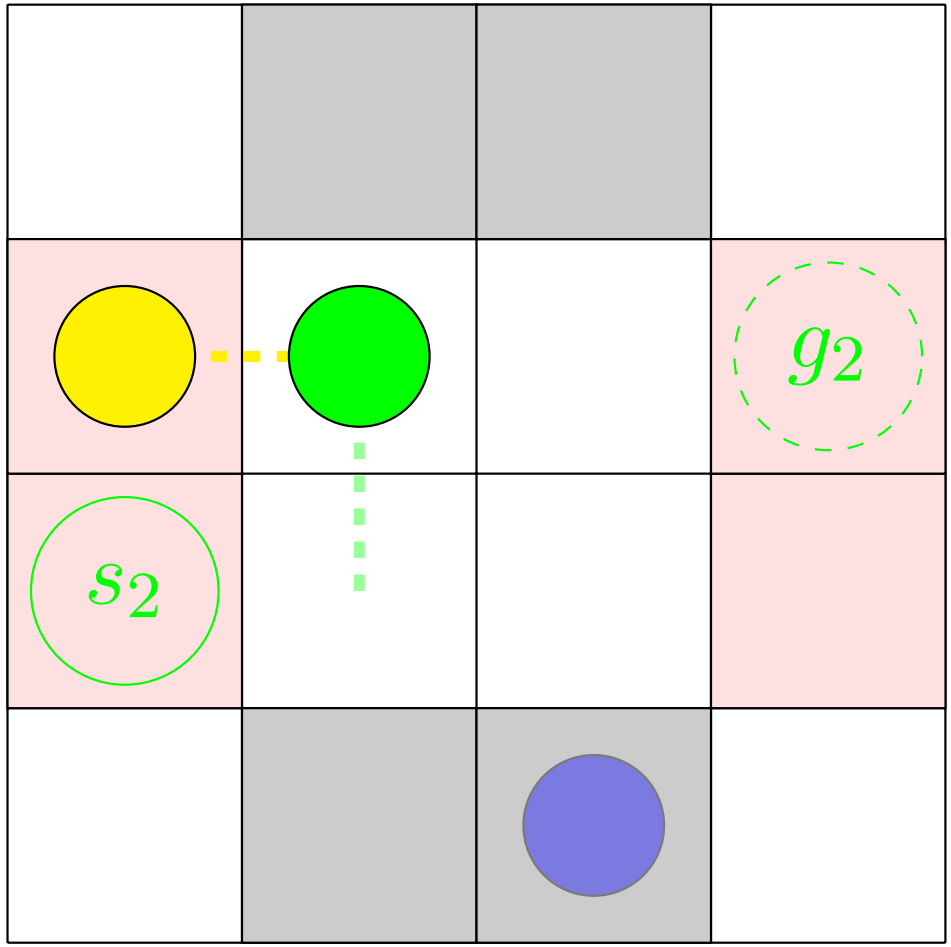
TP: Animation



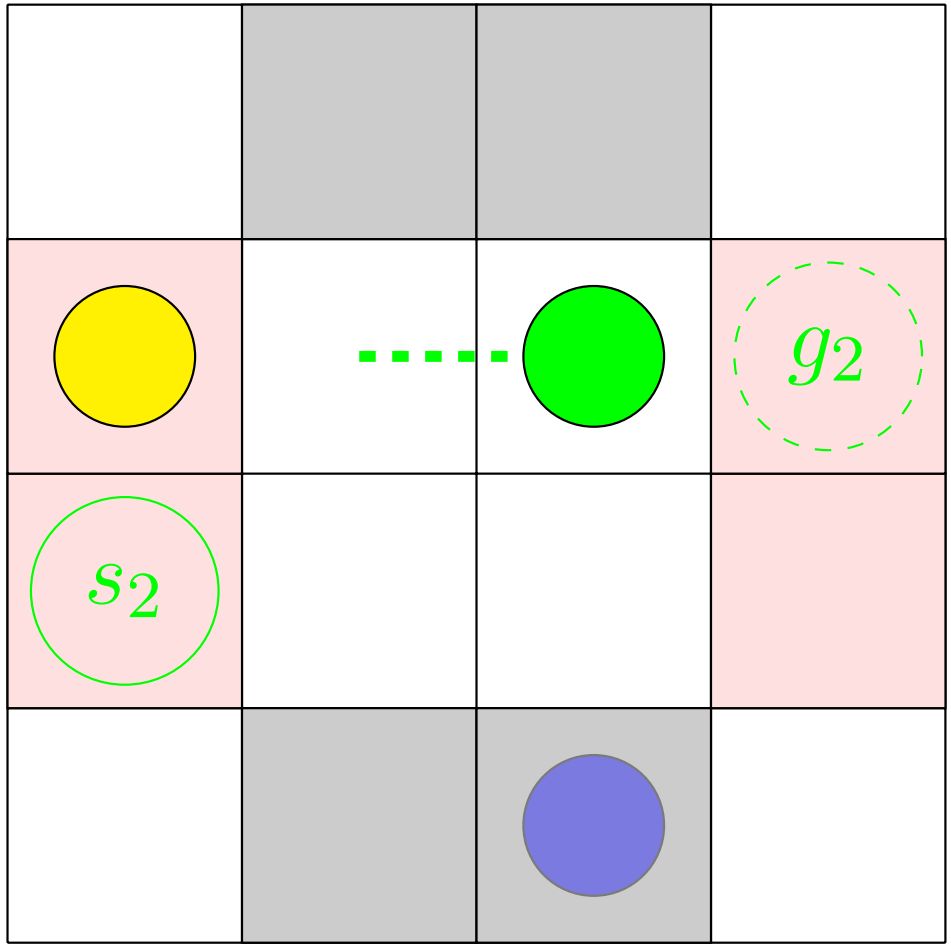
TP: Animation



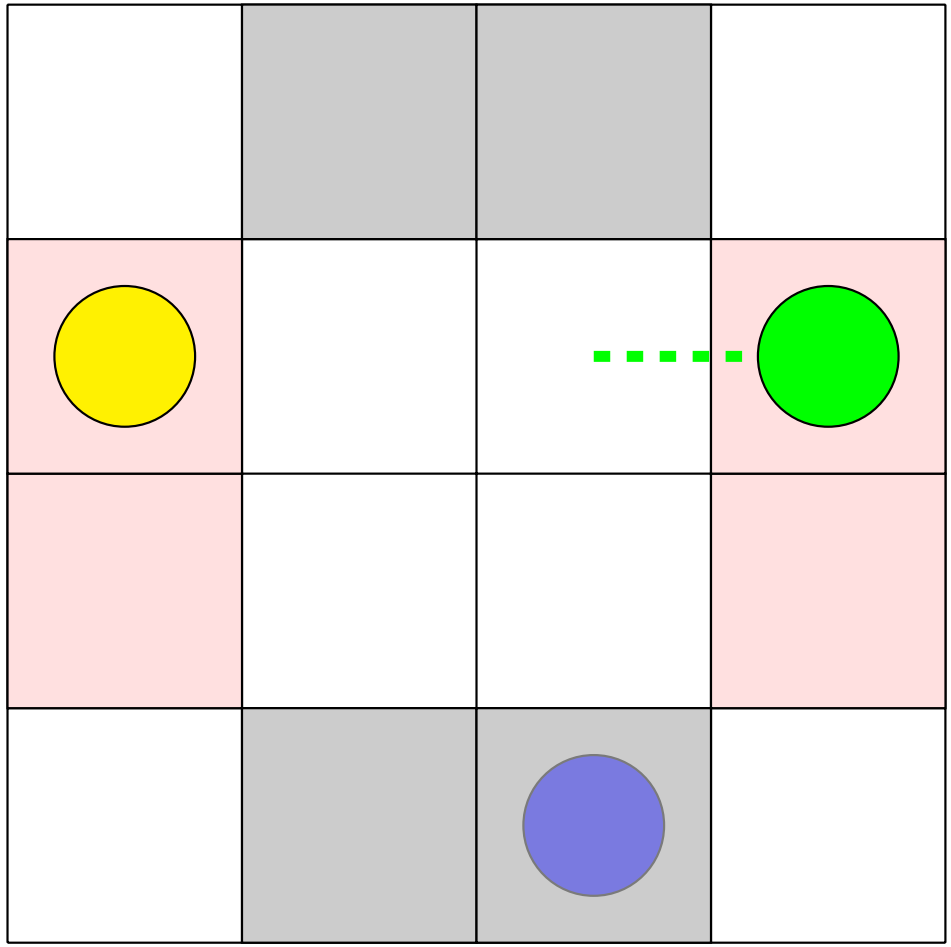
TP: Animation



TP: Animation



TP: Animation



TP: Completeness

Theorem

All well-formed MAPD instances are solvable, and TP solves them.

Improving the Effectiveness of TP

TP is simple but can be made more effective:

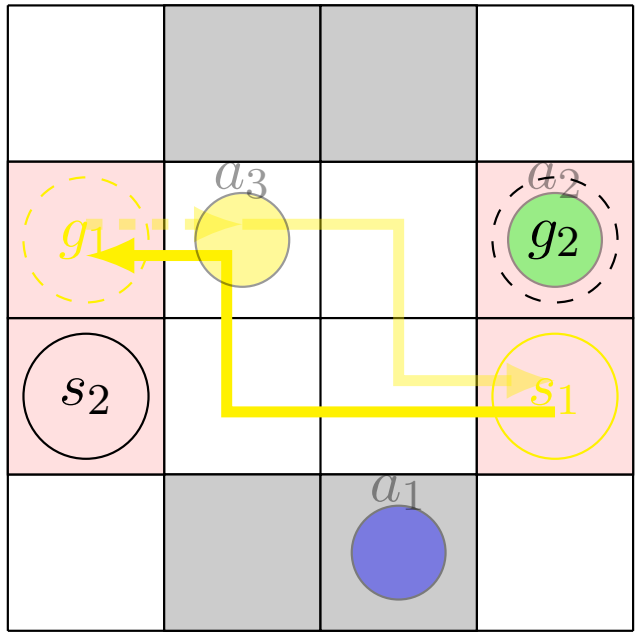
- A task with an assigned agent can be assigned a new agent (as long as the task has not been executed).

Token Passing with Task Swaps (TPTS)

An agent is allowed to grab a task from another agent if it can finish the task earlier.

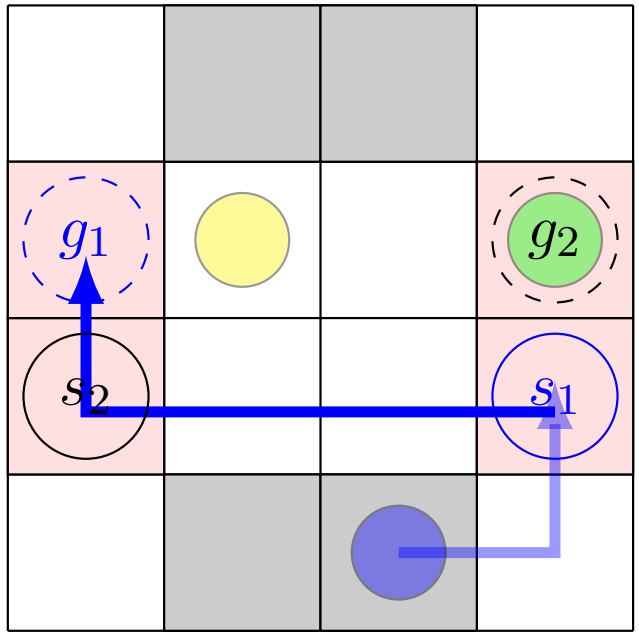
TPTS: Running Example

Tasks Available for Assignment: $task_1$, $task_2$.
Agent a_1 and agent a_2 request for token.



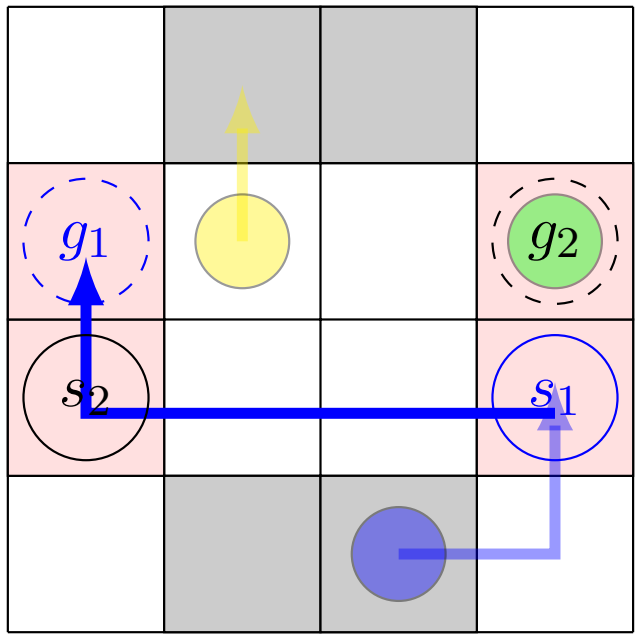
TPTS: Agent a_1's Turn

Agent a_1 has token.
Agent a_1 grabs $task_1$ from agent a_3.



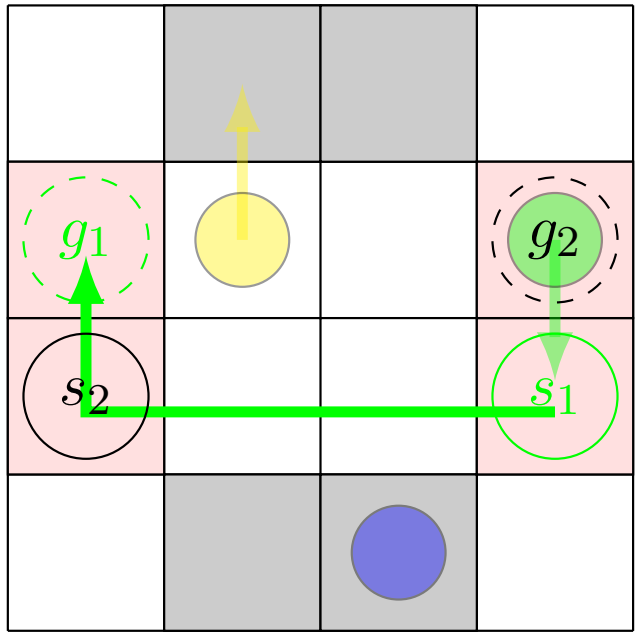
TPTS: Agent a_3 Making Decisions

Agent a_3 has token.
Agent a_3 moves to a parking location that will not create any deadlock in the future.



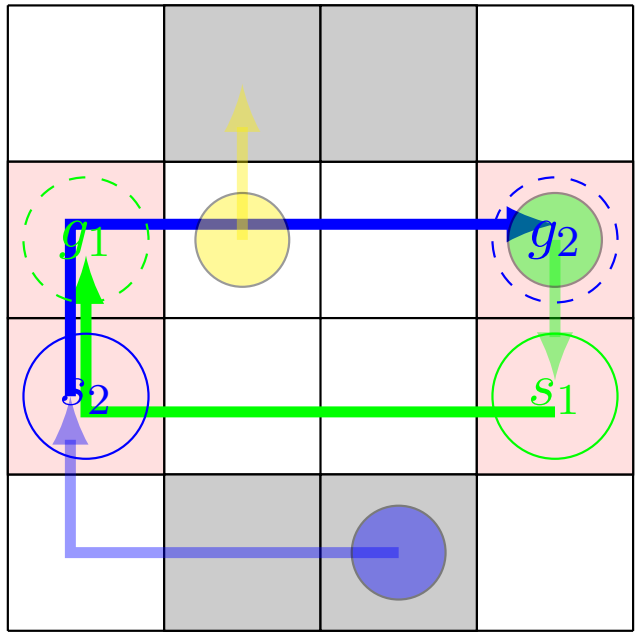
TPTS: Agent a_2's Turn

Agent a_2 has token.
Agent a_2 grabs $task_1$ from agent a_1.



TPTS: Agent a_1 Making Decisions

Agent a_1 has token.
Agent a_1 assigns itself to $task_2$.



TPTS: Completeness

Theorem

TPTS solves all well-formed MAPD instances.

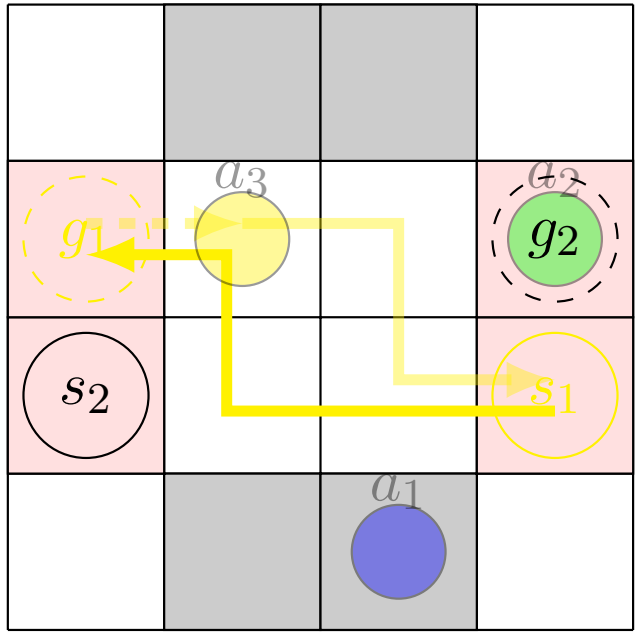
Centralized MAPD Algorithm: CENTRAL

CENTRAL assigns agents to tasks in a centralized way:

1. assigns parking locations to all free agents using Hungarian method;
2. plans paths for all of them from their current locations to their assigned parking locations by solving the resulting "one-shot" multi-agent path-finding problem.

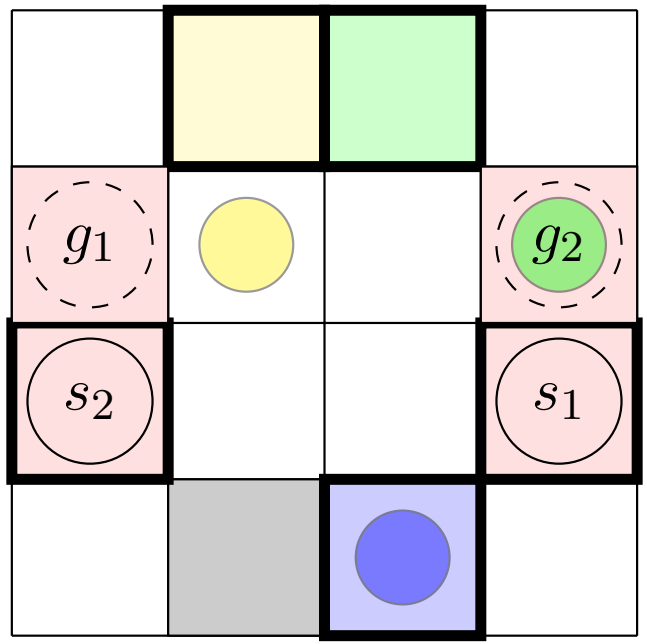
CENTRAL: Running Example

Tasks available for assignment: $task_1$, $task_2$



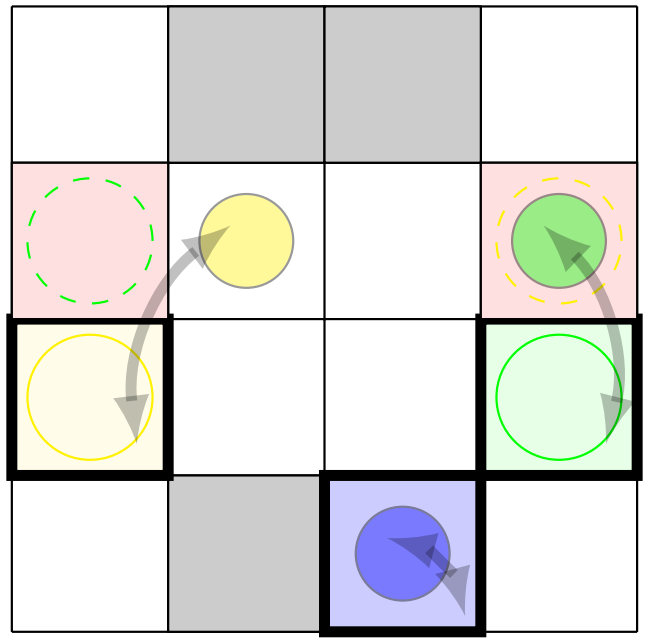
CENTRAL: Candidate Parking Locations

Pickup locations s_1 and s_2 + three additional "good" parking locations, one for each agent:



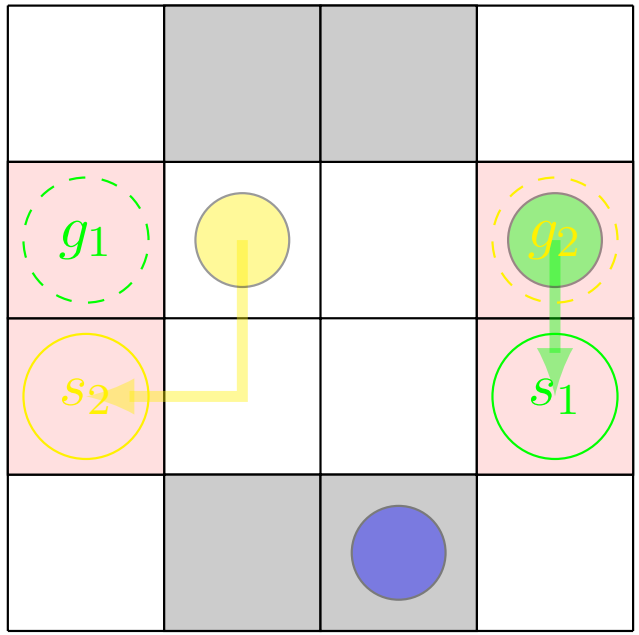
CENTRAL: Assignment of Parking Locations to Agents

CENTRAL uses Hungarian method to find a cost-minimal assignment from parking locations to agents (pickup locations have priority over other parking locations):

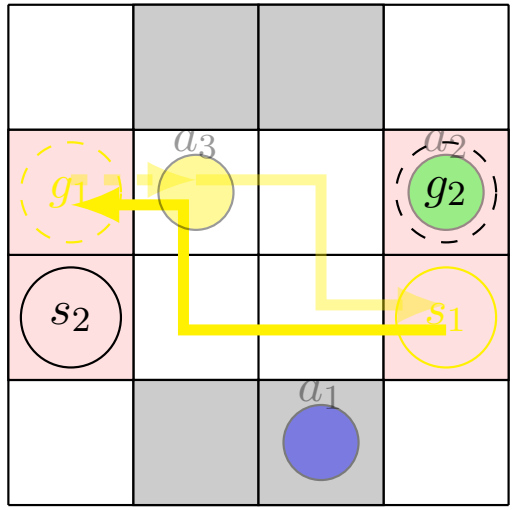


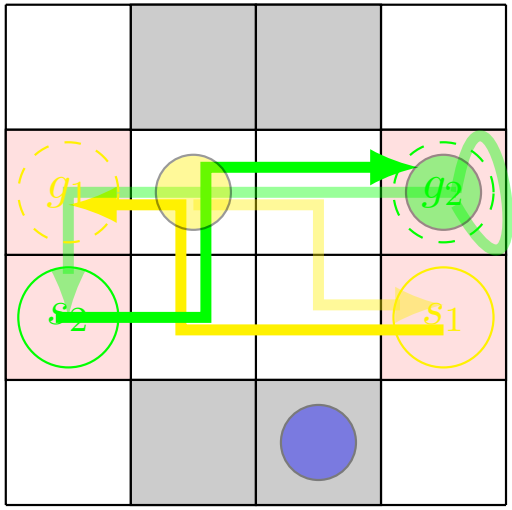
CENTRAL: Path Finding

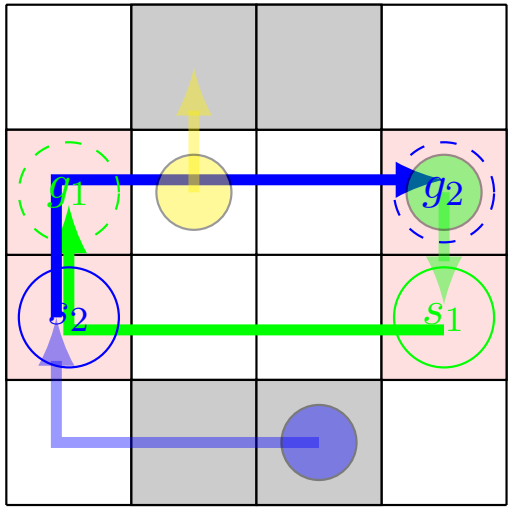
CENTRAL plans collision-free paths for all agents from their current locations to their assigned parking locations.
CENTRAL plans paths to delivery locations only when agents reach pickup locations.

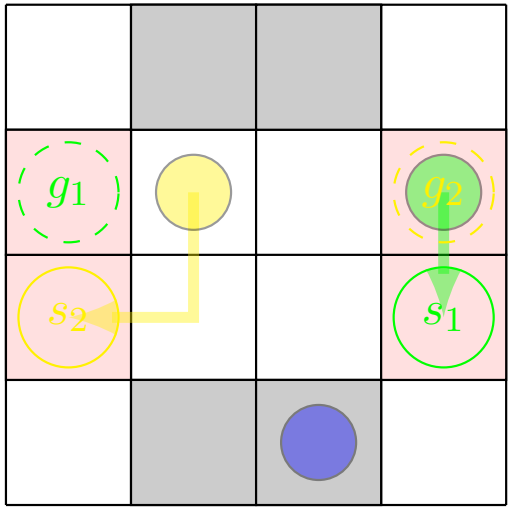


Comparisons of Three Algorithms









Small Simulated Warehouse Environment

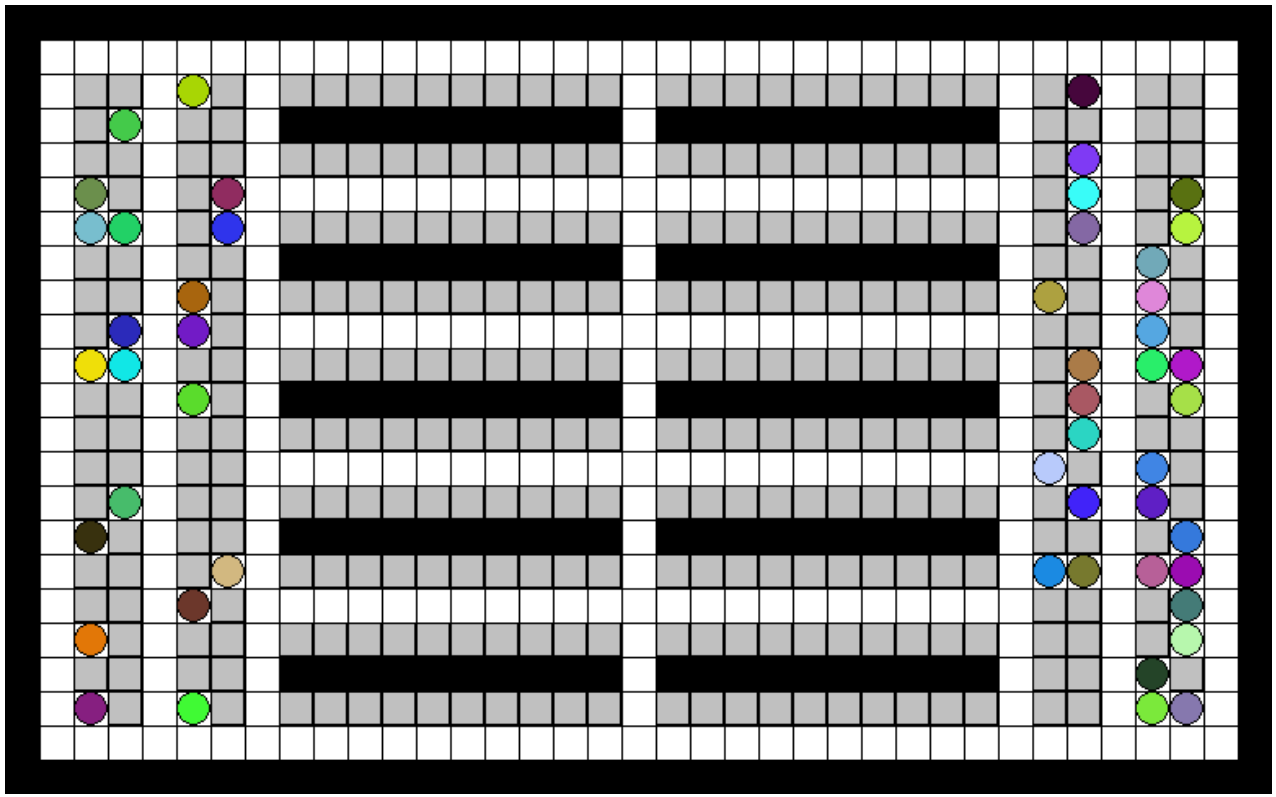


Figure: 21×35 4-neighbor grid with 50 agents. Gray cells are inventory stations and storage locations. Colored circles are the initial locations of agents.

Experimental Results: 500 Random Tasks, 10 to 50 Agents

- Effectiveness
  1. **Service Time**:
     CENTRAL $<$ TPTS $<$ TP
  2. **Throughput** – # tasks executed per 100 timesteps:
     TP $<$ TPTS $<$ CENTRAL
  3. **Makespan** – timestep when all tasks are finished:
     CENTRAL $<$ TPTS $<$ TP
- **Runtime per Timestep**:
  TP $<$ 10 milliseconds
  TPTS $<$ 200 milliseconds
  CENTRAL $<$ 4,000 milliseconds

Large Simulated Warehouse Environment

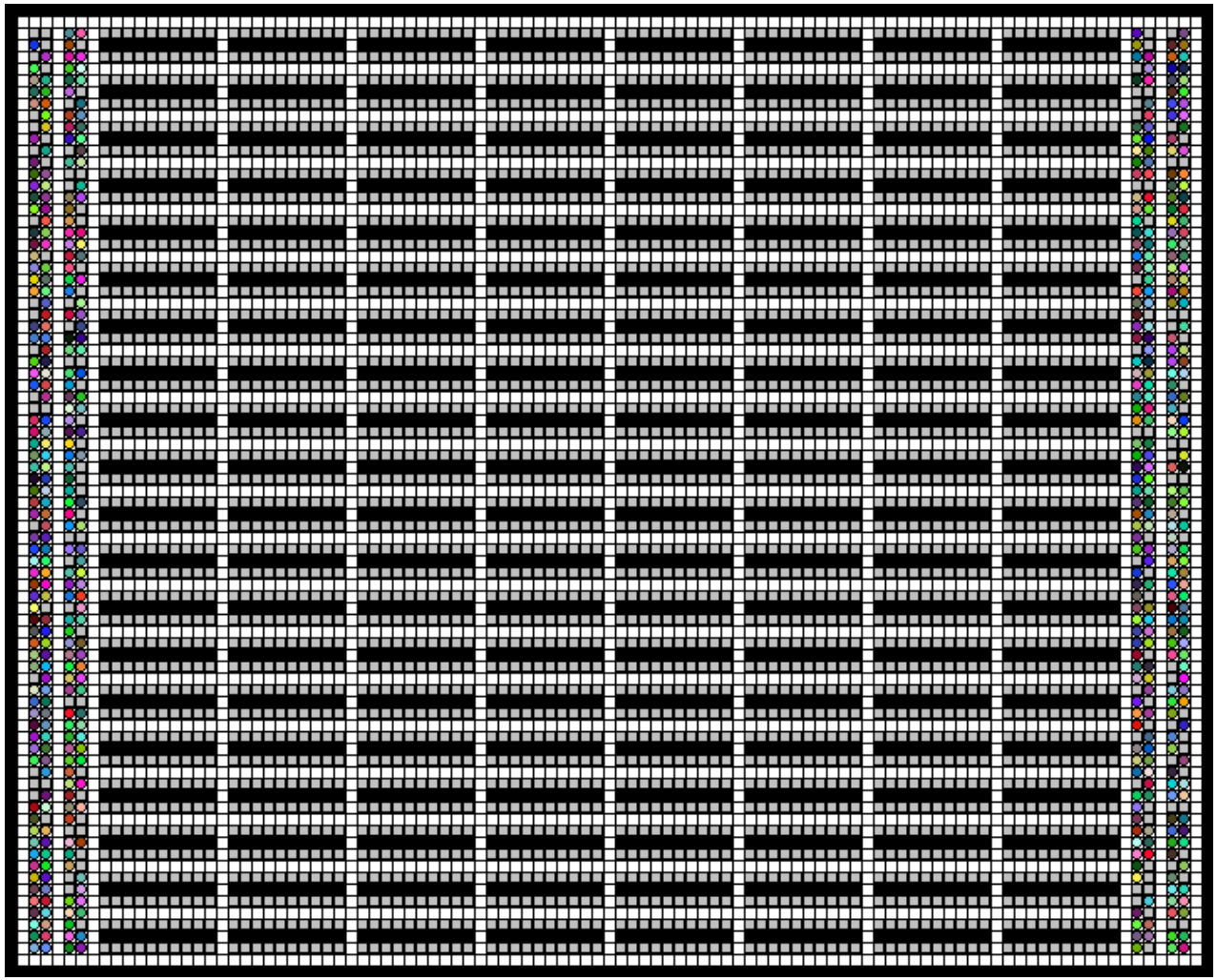


Figure: 81×81 4-neighbor grid with 500 agents.

Results for TP: 1000 Random Tasks, 100 to 500 Agents

100 agents: ~ 0.09 seconds per timestep
500 agents: ~ 6 seconds per timestep

agents	100	200	300	400	500
service time	463.25	330.19	301.97	289.08	284.24
runtime (milliseconds)	90.83	538.22	1,854.44	3,881.11	6,121.06

Takeaways

MAPD: A “lifelong” version of multi-agent path finding.
Three Algorithms:

- Decoupled and complete for well-formed MAPD instances: TP, TPTS.
- Centralized: CENTRAL.

Task Assignment Effort: TP $<$ TPTS $<$ CENTRAL
Effectiveness: TP $<$ TPTS $<$ CENTRAL
Efficiency: CENTRAL $<$ TPTS $<$ TP