



# **Multi-Agent Path Finding using Deep RL with Communication**

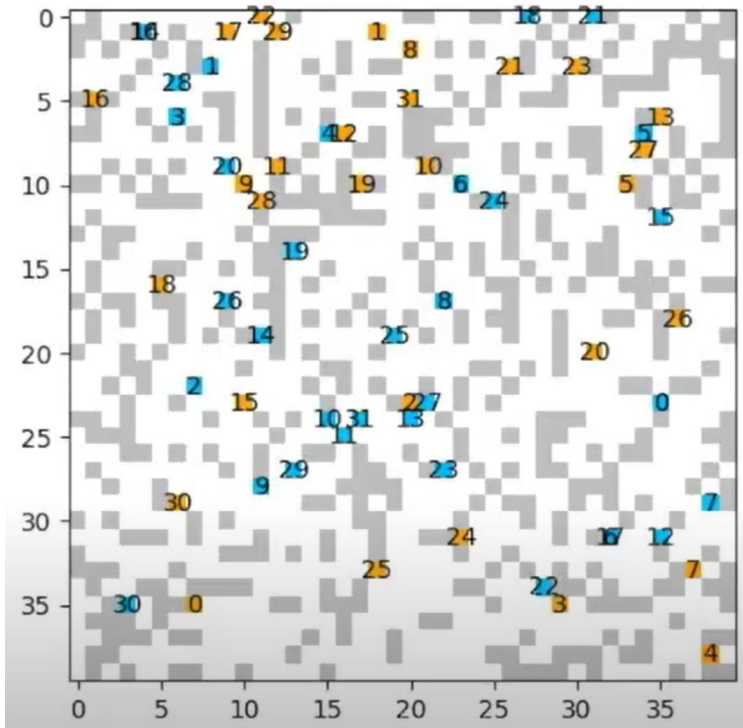
ICRA 2021 and ICRA 2022

Yudong Luo email: [yudong.luo@uwaterloo.ca](mailto:yudong.luo@uwaterloo.ca)

# Problem

---

- Problem: N agents in an environment, each agent has a starting point and goal location. Agents all move to their goals without congestion.
- One of the commonly used environments is the grid world



32 agents in 40 by 40 grids

Blue color is the starting location, number is the ID

Orange color with the same ID is the goal

# Existing methods

---

- Existing methods:
- 1. Searching algorithms: search path for each agent and adjust if conflicts.
  - Conflict-based Search
- 2. RL: partial observability, usually guided by imitation learning
  - PRIMAL<sub>[1]</sub>: Use A3C as backend (each thread is an agent), switch to (supervised) imitation periodically
  - Others<sub>[2,3,4]</sub>: reward shaping: give a penalty reward if agent does not follow the imitation path (multi-agent path / single-agent shortest path)

[1] PRIML: Pathfinding via Reinforcement and Imitation Multi-Agent Learning, RA-L'19

[2] Glas: Global-to-local safe autonomy synthesis for multi-robot motion planning with end-to-end learning, RA-L'20

[3] Mobile Robot Path Planning in Dynamic Environments Through Globally Guided Reinforcement Learning, RA-L'20

[4] MAPPER: Multi-Agent Path Planning with Evolutionary Reinforcement Learning in Mixed Dynamic Environments, IROS'20

# Existing methods

---

- Searching: Can find optimal path. The computation cost is high, should solve the problem instance every time.
- RL with imitation: Imitation is costly.
  - Global optimal multi-agent path: generated by searching algorithm.
  - Single-agent shortest path: usually not unique and global optimal. Force agent to follow is not reasonable

# Existing methods

---

- Searching: Can find optimal path. The computation cost is high, should solve the problem instance every time.
- RL with imitation: Imitation is costly.
  - Global optimal multi-agent path: generated by searching algorithm.
  - Single-agent shortest path: usually not unique and global optimal. Force agent to follow is not reasonable
- Goal: Can we do RL without imitation from global optimal path? Even if we need some guidance for RL, can we design some heuristic instead of forcing agent to follow sub optimal path (single-agent shortest path)

# Environment design

---

- Grid World

- Obstacle density: triangular distribution (0, 0.5) peak at 0.33 [same as PRIMAL]
- Observation: FOV 9x9 (10x10 in PRIMAL), one channel is obstacle, one channel is other agents
- Action space: 5 , four direction + stay still
- Reward:

Actions	Reward
Move (Up/Down/Left/Right)	-0.075
Stay (on goal, away goal)	0, -0.075
Collision (obstacle/agents)	-0.5
Finish	3

# Environment design

---

- Grid World

- Obstacle density: triangular distribution (0, 0.5) peak at 0.33 [same as PRIMAL]
- Observation: FOV 9x9 (10x10 in PRIMAL), one channel is obstacle, one channel is other agents
- Action space: 5 , four direction + stay still
- Reward:

Actions	Reward
Move (Up/Down/Left/Right)	-0.075
Stay (on goal, away goal)	0, -0.075
Collision (obstacle/agents)	-0.5
Finish	3

- Question: Should we do Multi-Agent RL or Single-Agent RL ?

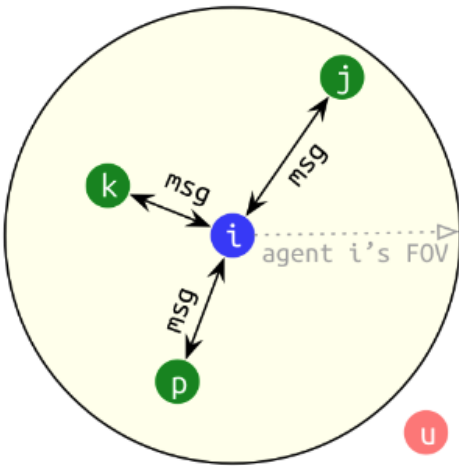
- Multi-Agent RL: cooperative setting, agents share the same team reward, critic (Q value function) is trained on joint actions
- Single-Agent RL: train each agent separately.

- Agents have their individual goals. Each agent plays the same role.

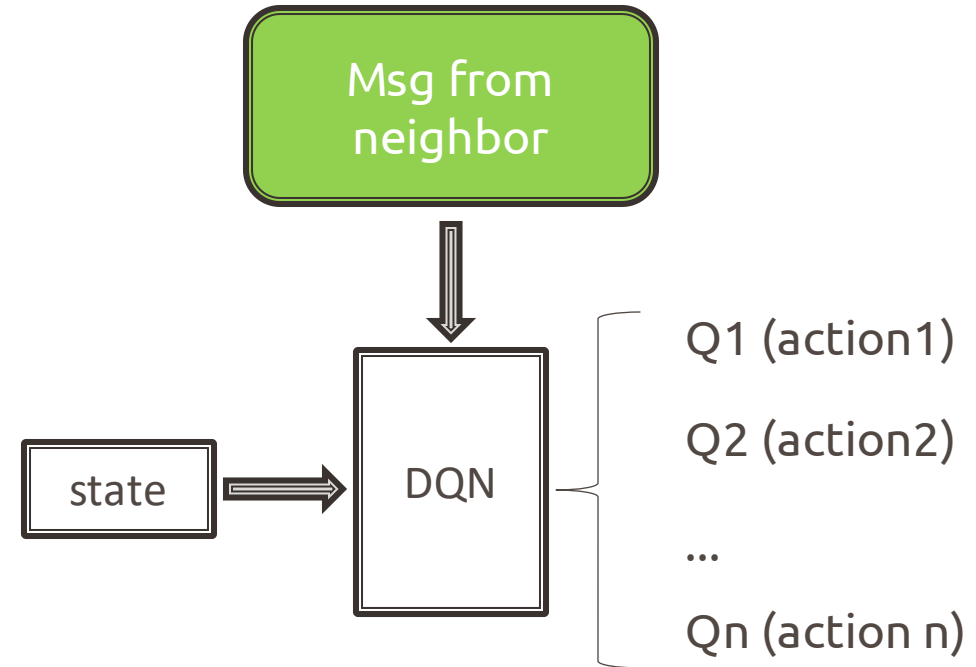
# RL design : Communication

---

- If simply apply single-agent RL to MAPF, the interaction between agents cannot be learned.
- One promising direction to achieve cooperation is communication.



- How to embed msg from neighbor

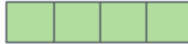




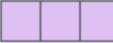
# RL design : Communication

- Graph convolution (Transformer / multi head attention)
- Each agent first embed its local obs with NN (green vector)

Embedding  $i$  

$j$  

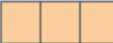
Queries  $q_1$  

$q_2$  



$W^Q$

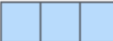
Keys  $k_1$  

$k_2$  



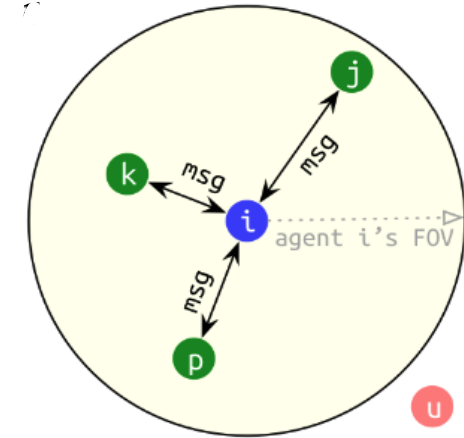
$W^K$

Values  $v_1$  

$v_2$  



$W^V$



Output for agent  $i$

$$w_1, w_2 = \text{softmax}(\boxed{q_1} \cdot k_1, \boxed{q_1} \cdot k_2)$$

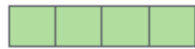
$$o_1 = w_1 \cdot v_1 + w_2 \cdot v_2$$

$W^Q, W^K, W^V$  are learning parameters

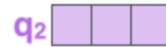
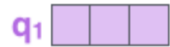
# RL design : Communication

- Graph convolution (Transformer / multi head attention)
- Each agent first embed its local obs with NN (green vector)

Embedding  $i$

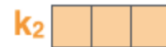
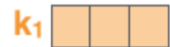


Queries



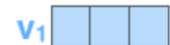
$W^Q$

Keys

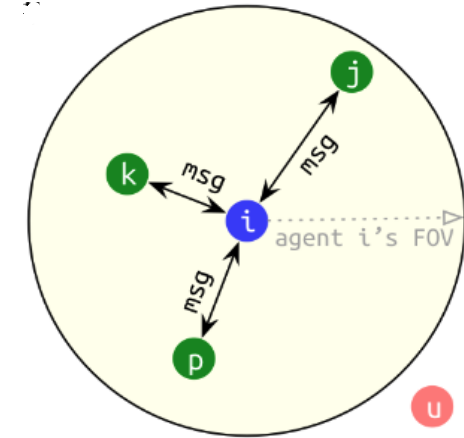


$W^K$

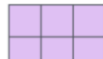

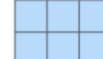
Values

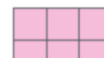


$W^V$



$$\text{softmax} \left( \frac{Q \times K^T}{\sqrt{d_k}} \right) V$$

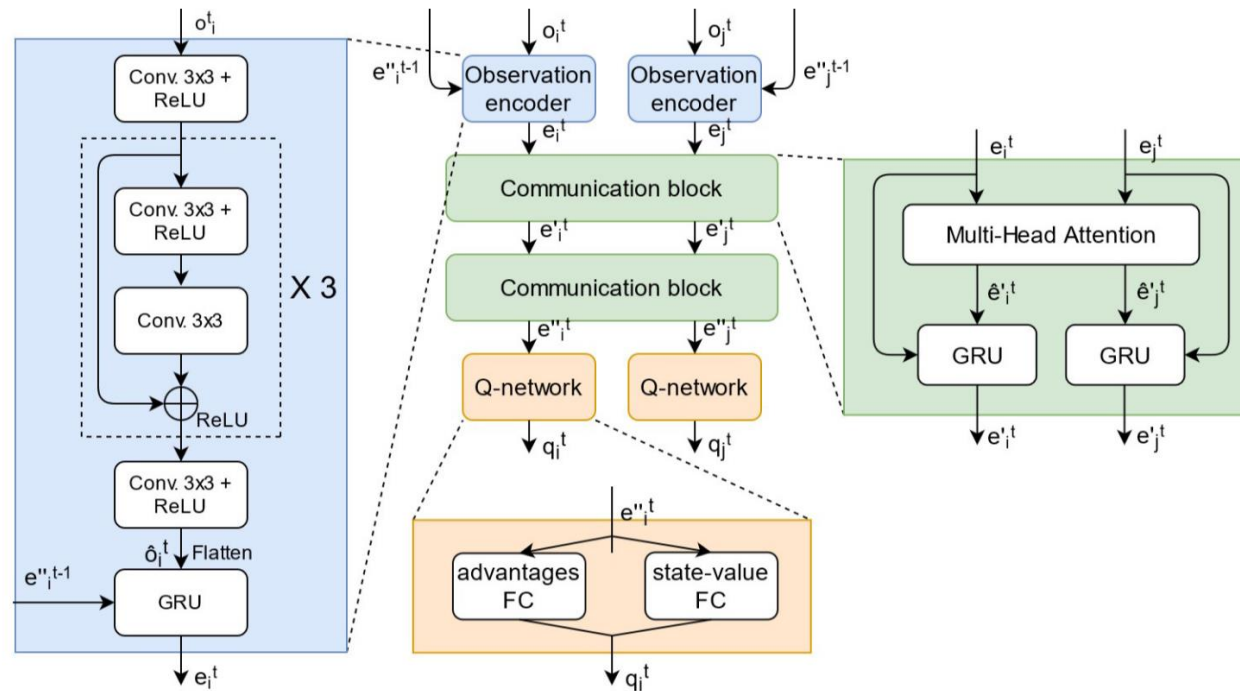






$W^Q, W^K, W^V$  are learning parameters

# RL design : Communication

- Full architecture

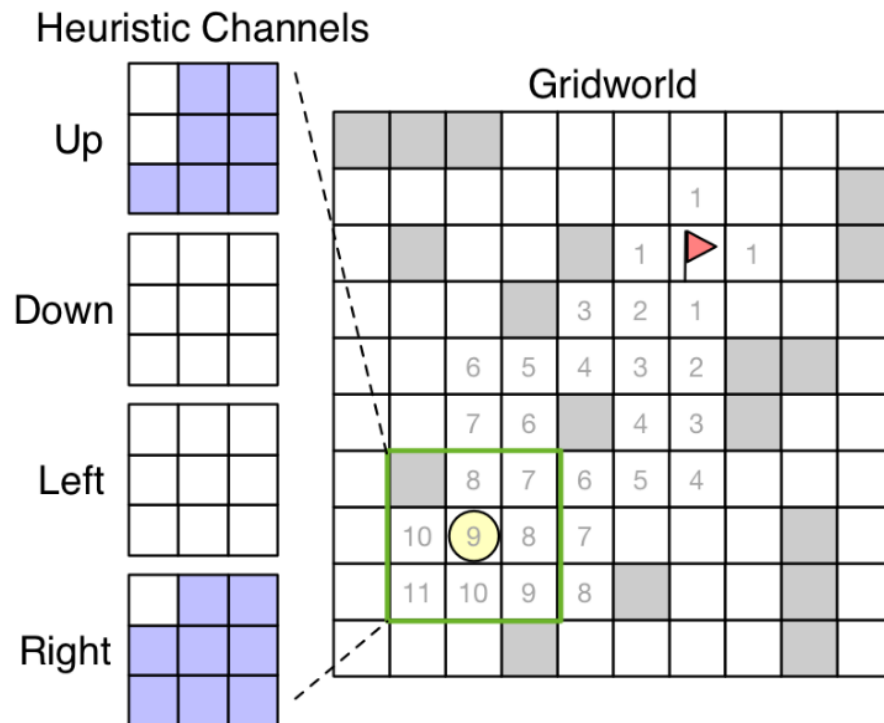


- Note: communication requires the information of neighbors. This information should be recorded in the replay buffer when doing offline training.

# RL design : Heuristic guidance

---

- Using multi-agent optimal path as guidance is expensive. Using single-agent shortest path is biased.
- Embed all the potential single-agent shortest path to the model input

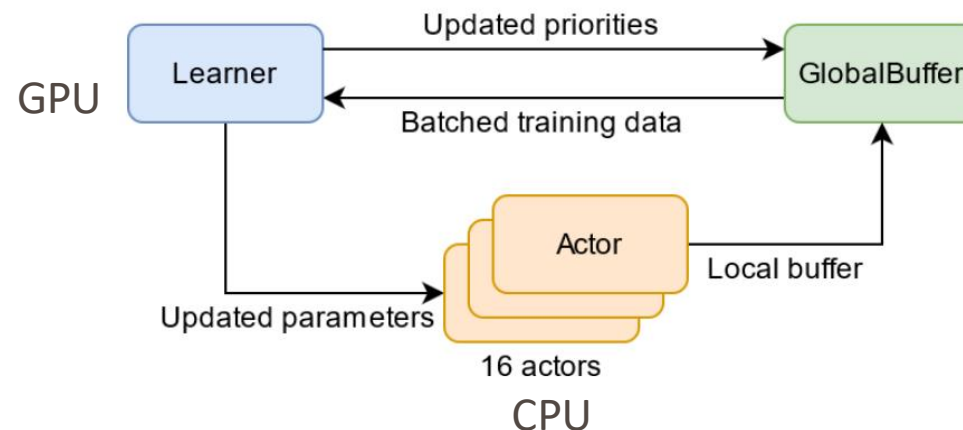


These four channels along with the original two observation channels serve as the model input

# RL design : Distributed training

---

- Distributed training can accelerate learning, e.g. used by PRIMAL (A3C)
- Two directions:
  - 1. parallelizing the gradient computation, e.g. A3C
  - 2. parallelizing experience data generation and selection with a shared replay memory
- Benefit of the second: requires single GPU



# Training

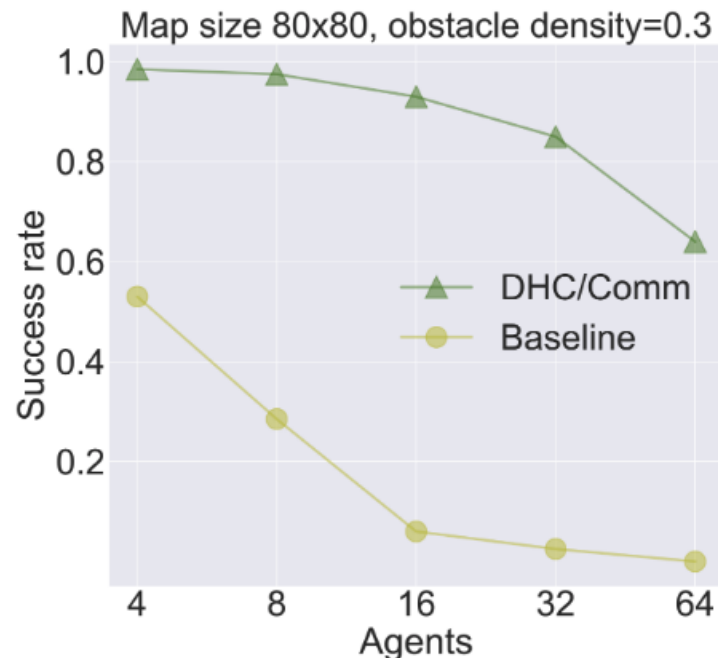
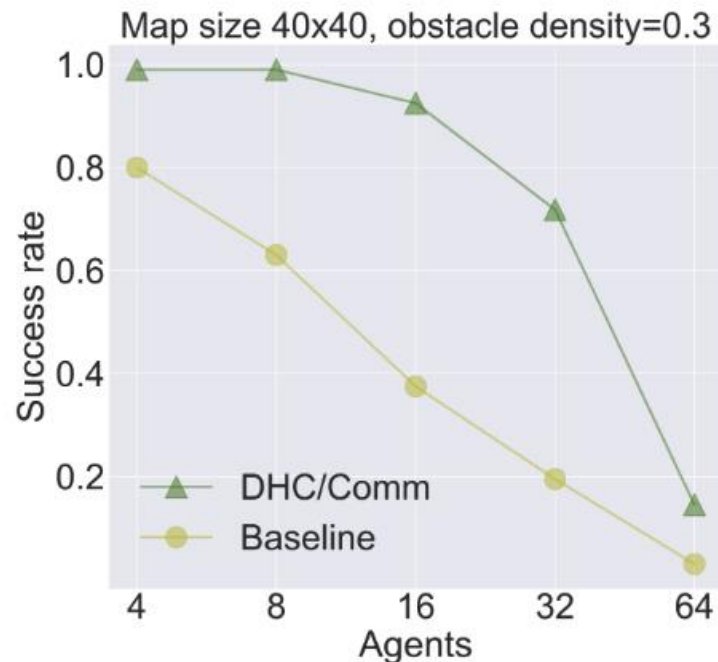
---

- Communication scope: nearest two neighbors
- Curriculum learning:
  - From easy to hard. Start from 1 agent in a 10x10 env
  - Move to next level if success rate exceeds 0.9
  - Next level is adding 1 agent, or increase the env size by 5
  - Final task: 12 agents in a 40x40 env

# Experiments

---

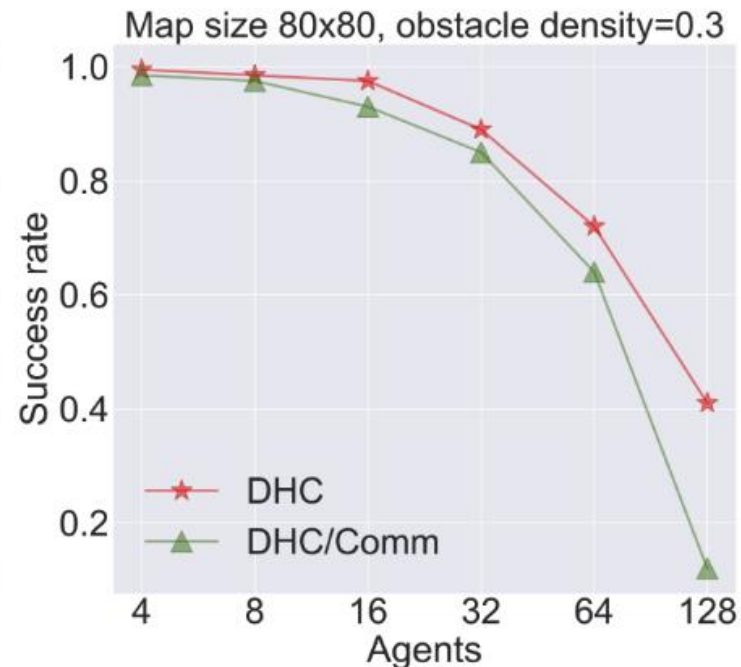
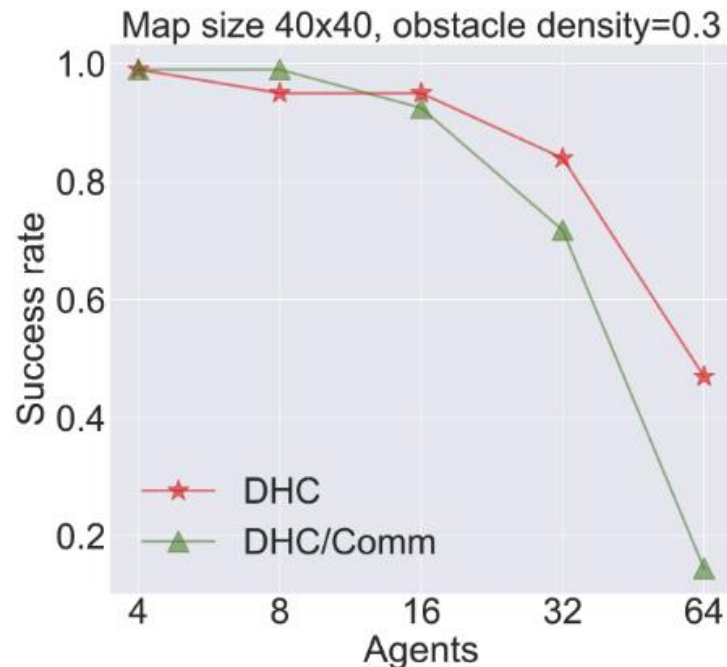
- Can heuristic guidance and communication really help MAPF
- Heuristic guidance
- [Baseline]: (no heuristic guidance, no communication) use more informative input as PRIMAL V.S. [DHC/Comm]: Our model with communication removed .



# Experiments

---

- Communication
- [DHC/Comm]: Our model with communication removed.
- [DHC]: Our full model





# Further direction

---

- Communication Efficiency:

- In this work, the agent communicates with its two nearest neighbors.
- Which neighbor to choose: are the nearest neighbors the most relevant agents to communicate with?
- How many neighbors should be chosen? In practice, the less the better.

# Learning selective communication

---

- Communication Scheme
  - Before communication, agent can compute a local policy
  - No communication: if the presence of other agents in FOV does not change its local policy
  - Intuition: to reduce communication frequency, agents follow heuristic guidance as much as possible, unless communication is needed
- Significantly reduced the communication overhead
  - Better than communicating with nearest two neighbors



Thank you!