Multi-Agent Reinforcement Learning

Unleashing Collective Intelligence

Gianluca Aguzzi gianluca.aguzzi@unibo.it

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

27/07/2023

What is Multi-Agent Reinforcement Learning?

What is Multi-Agent Reinforcement Learning?

Multiple agents **learn** to take the *right* actions (**policy**) to maximise a *reward* signal.

Multi-Agent Reinforcement Learning (MARL)

Applications

a) Videogames
b) Trafic Control
c) Robotics (*Swarm robotics*)
d) Trading
e) Energy Management
f) Environmental Monitoring

Today Outline

- MARL differences with respect to Single Agent RL
- MARL classification
  - Cooperative vs. Competitive agents
  - Centralised vs Decentralised learning and acting
  - Homogeneous vs Heterogeneous agents
- Reference to MARL algorithms
- Scale to **infinity**: MARL large scale applications

Motivating Examples

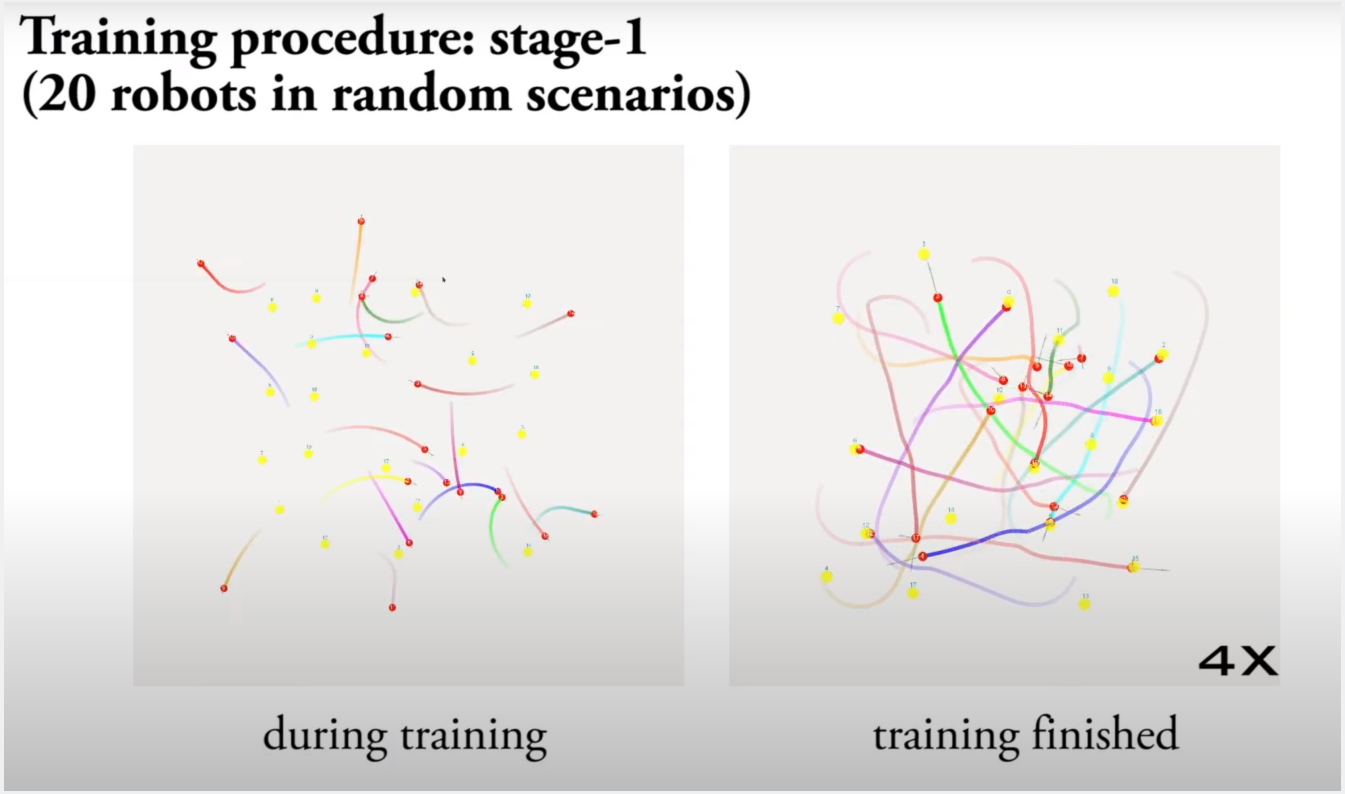
OpenAI Hide And Seek: https://openai.com/blog/emergent-tool-use/

Motivating Examples

Capture The Flag:
https://deepmind.com/blog/article/capture-the-flag-science

Motivating Examples

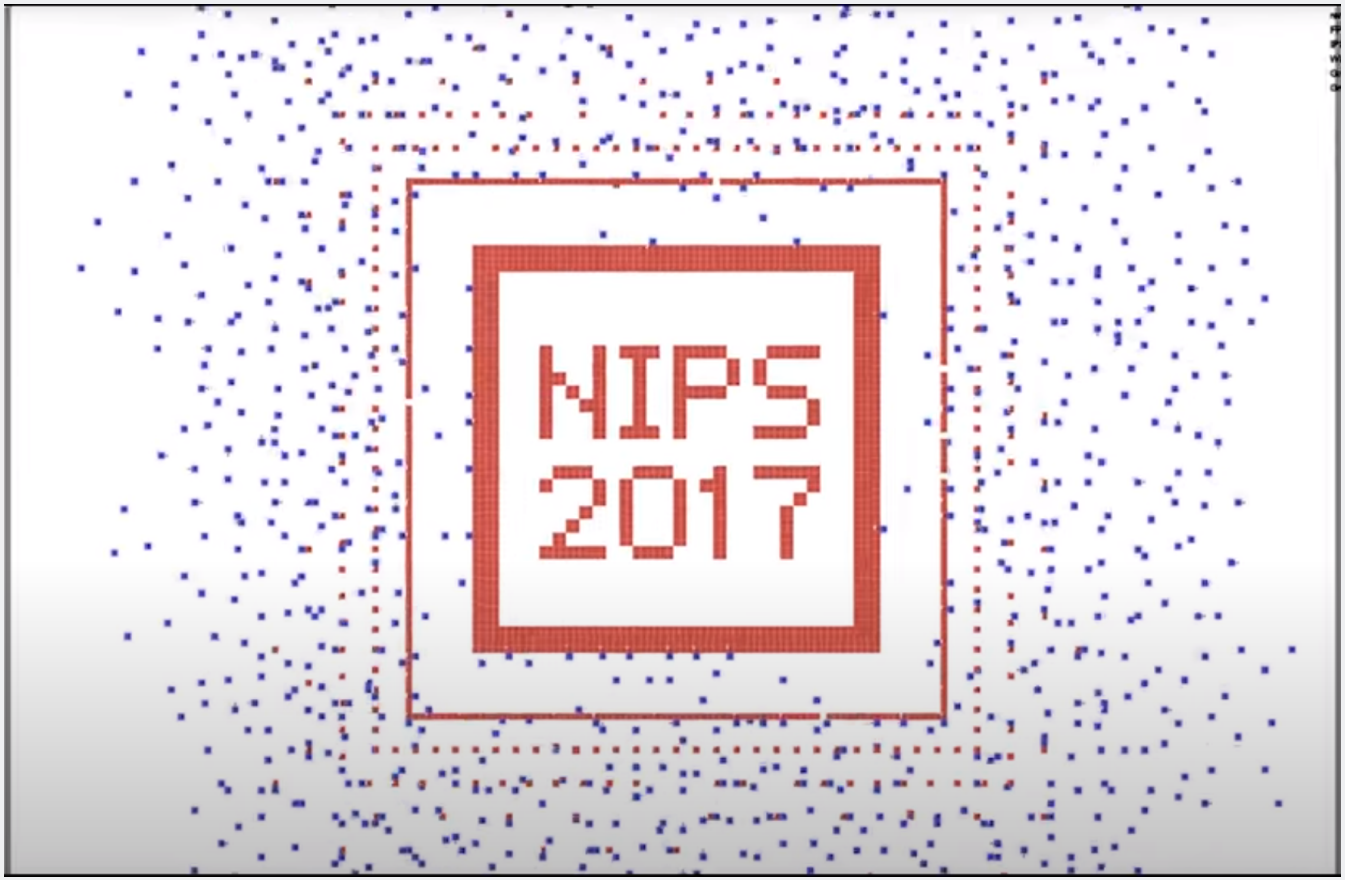
Robot Collision Avoidance [1] https://sites.google.com/view/drlmaca



[1] Pinxin Long et al. "Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning". In: IEEE, 2018, pp. 6252–6259. url: https://doi.org/10.1109/ICRA.2018.8461113

Motivating Examples

MAgent [2] https://github.com/geek-ai/MAgent

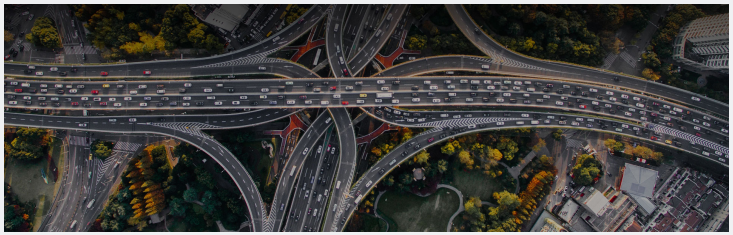


[2] Lianmin Zheng et al. "MAgent: A many-agent reinforcement learning platform for artificial collective intelligence". In: 2018

Real-case applications I

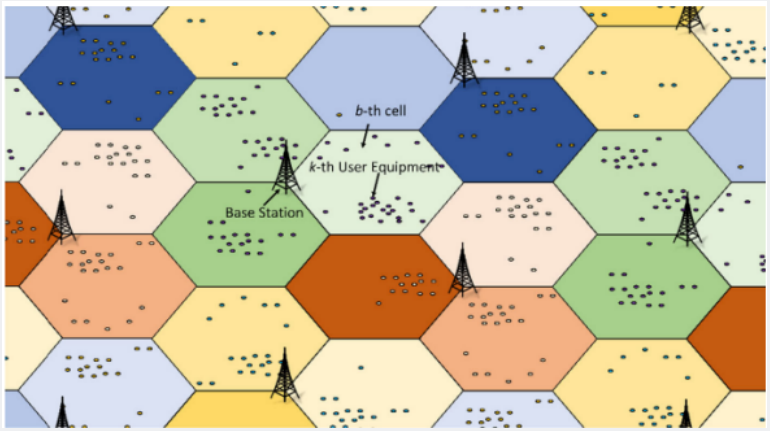
Trafic control reduction

- Trafic control is an important issue in modern cities (long waiting times, accidents)
- Coordinating the actions of multiple agents (cars) can lead to a better overall performance



Antenna tilt control

- The joint configuration base stations can be optimized according to the distribution



Benefits of Multi-Agent Learning

- **Sharing experience**
  - via communication, teaching, imitation
- **Parallel computation**
  - due to decentralized task structure
- **Robustness**
  - redundancy, having multiple agents to accomplish a task

Single Agent Reinforcement Learning

What have you seen so far . . .

- We have seen how to model the agent-environment interactions
- Find a learning process that eventually leads to an *optimal* policy π^*
- Q-Learning (in general *value-based approaches*) as a prominent algorithm to reach converge

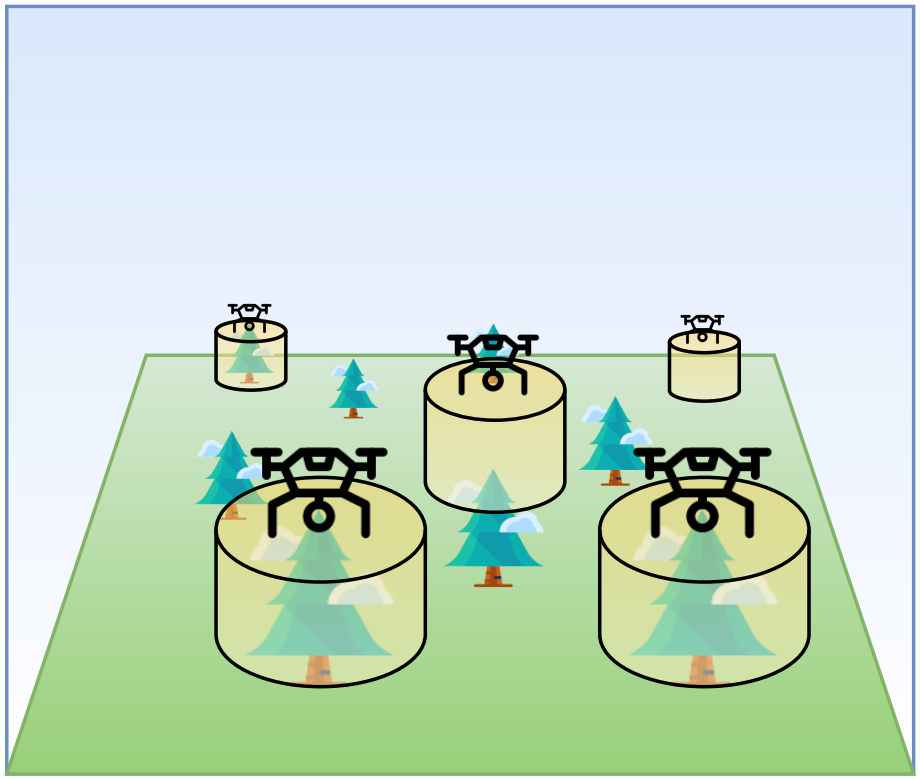
. . . But plain RL works only under some conditions

- Full environment observability and Markovian property
- Stationary environment
- State/action space should be small enough to be stored in memory (otherwise, we should leverage function approximators)

Partial Observable environments

Definition

An agent does not have a *perfect* and *complete* knowledge of the state of the environment

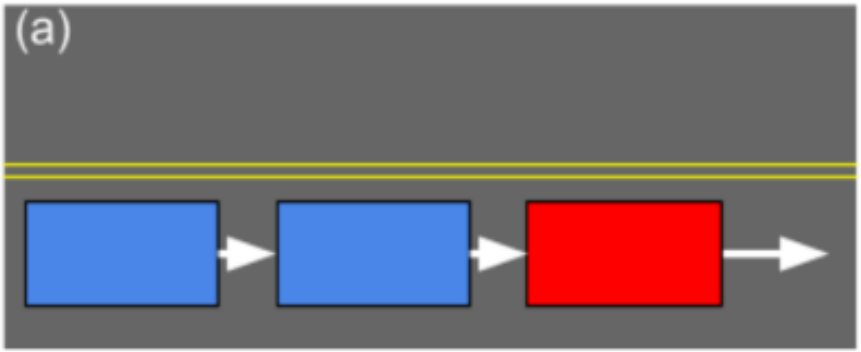


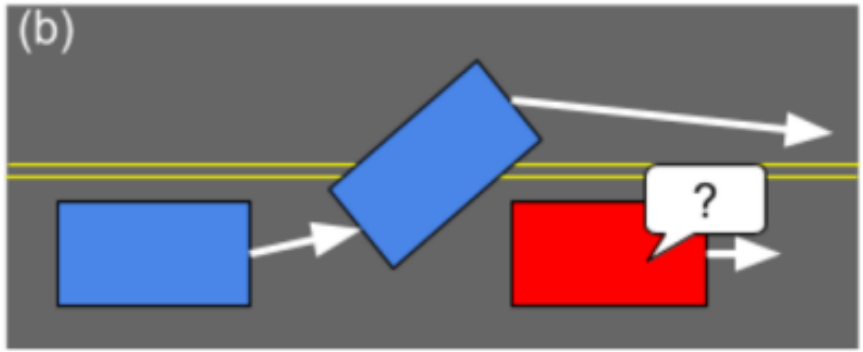
Non-stationary environments

Definition

The environment model (e.g. the random variable associated with it) changes over time.

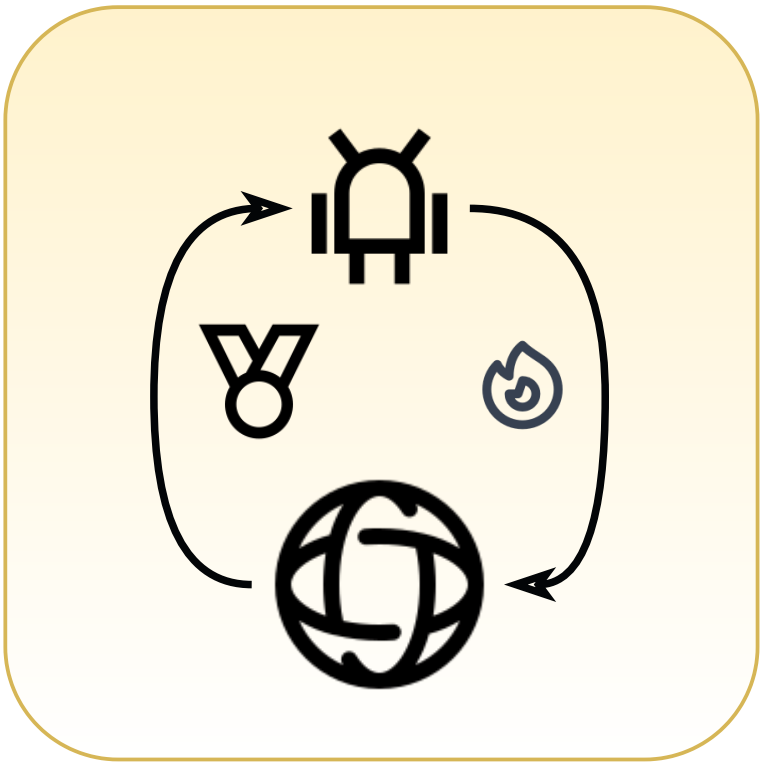
- Real-case environment dynamics could change over time (e.g. markets, city traffic, routing networks)
- Practically, it seems that RL (in particular Temporal Difference (TD) methods) works well even in this case
- *But, we lose convergence guarantees.*

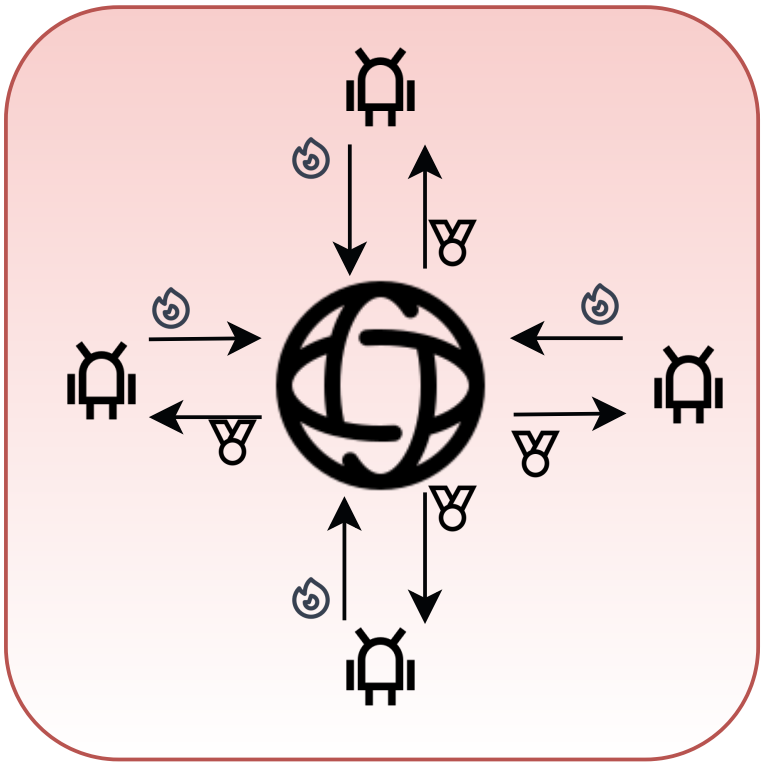




Agents-environment interaction

- Simple modelation:
  - At each time step t
    - Each agent i observes the environment state s_t (or a part of it, o_t^i)
    - Each agent i selects an action a_t^i (possibly different from the others)
    - The collective action is $a_t = (a_t^1, \ldots, a_t^N)$ is applied to the environment
    - The environment evolves to a new state s_{t+1} and provides a reward r_{t+1} to each agent





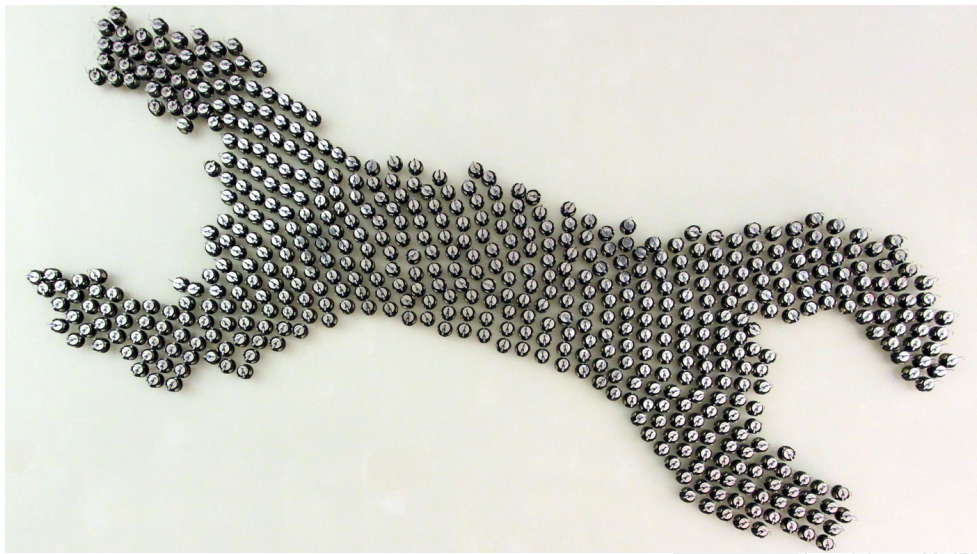
Multi but how many?

- Well-studied applications ➔ 2 agents
  - mainly board games
- Finite number of agents ➔ N agents
  - e.g. traffic control, antenna tilt control
  - Two-agents approaches can be extended to N agents
  - Finite but small
- "Infinite"
  - Very large-scale systems
  - We cannot know the number of agents in advance
  - Typically referred as *many* agent systems

MARL Systems: Task type

Cooperative

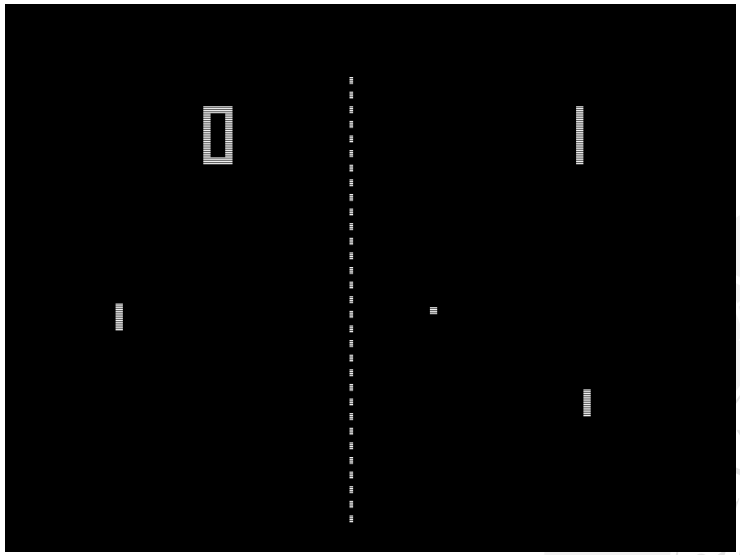
- Agents share the same reward function ($R^1 = \cdots = R^N$) in order to accomplish a collective goal



MARL Systems: Task type

Competitive

- Agents compete with each other to maximise a long term return
- Board Games, Video games, . . .



MARL Systems: Task type

Mixed

- Agents can both compete and cooperate in order to maximise a global reward function
- Also called *General Sum games*

On Cooperative Task

Homogeneous

- Each agent has same capabilities ($A^1 = \cdots = A^N$)
- The overall goal is to find the best policy that is the same for each agent ($\pi^* = \pi^*{}_1 = \pi^*_2 = \cdots = \pi^*{}_N$)

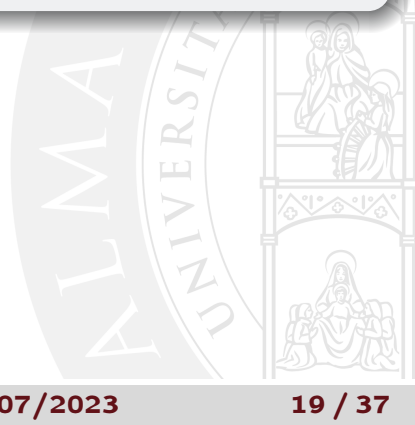
Heterogeneous

- Each agent could have different capabilities (in the worst case, $A^1 \neq \ldots \neq A^N$)
- Each agent has its local policy that should be maximised following the global collective goal

MARL Systems: Learning Scheme

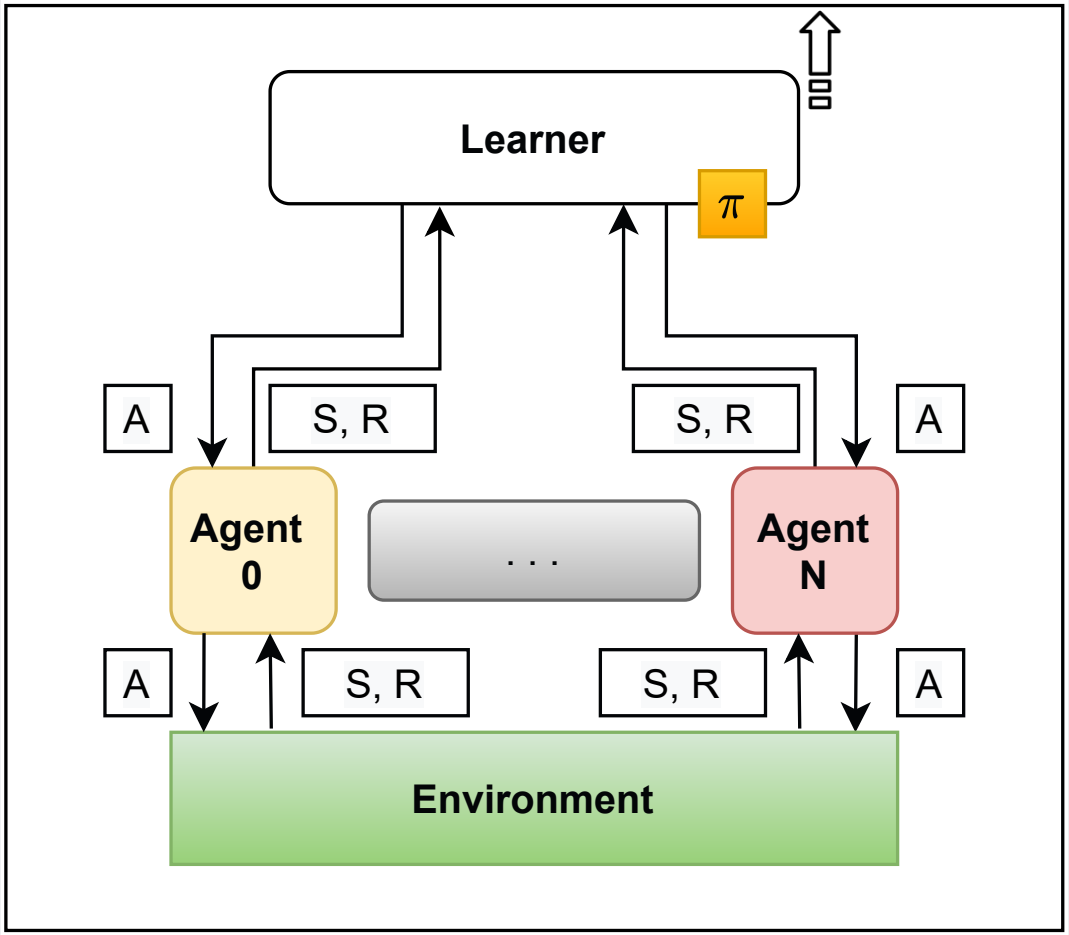
Centralised Training and Centralised Execution (CTCE)

- *One* agent with a global view of the system (in the cloud? or in a server?)
- Nodes send their perception to that central agent
- With these perceptions, the central node creates a global state of the system
- With the current policy, it chooses the actions that nodes should perform (*Control*)
- In the next time step, it evaluates the reward function and updates the policy accordingly (*Learn*)
- *Are the nodes agents?*
- Used mainly in offline (i.e. at simulation time) setting



MARL Systems: Learning Scheme

Centralised Training and Centralised Execution (CTCE)



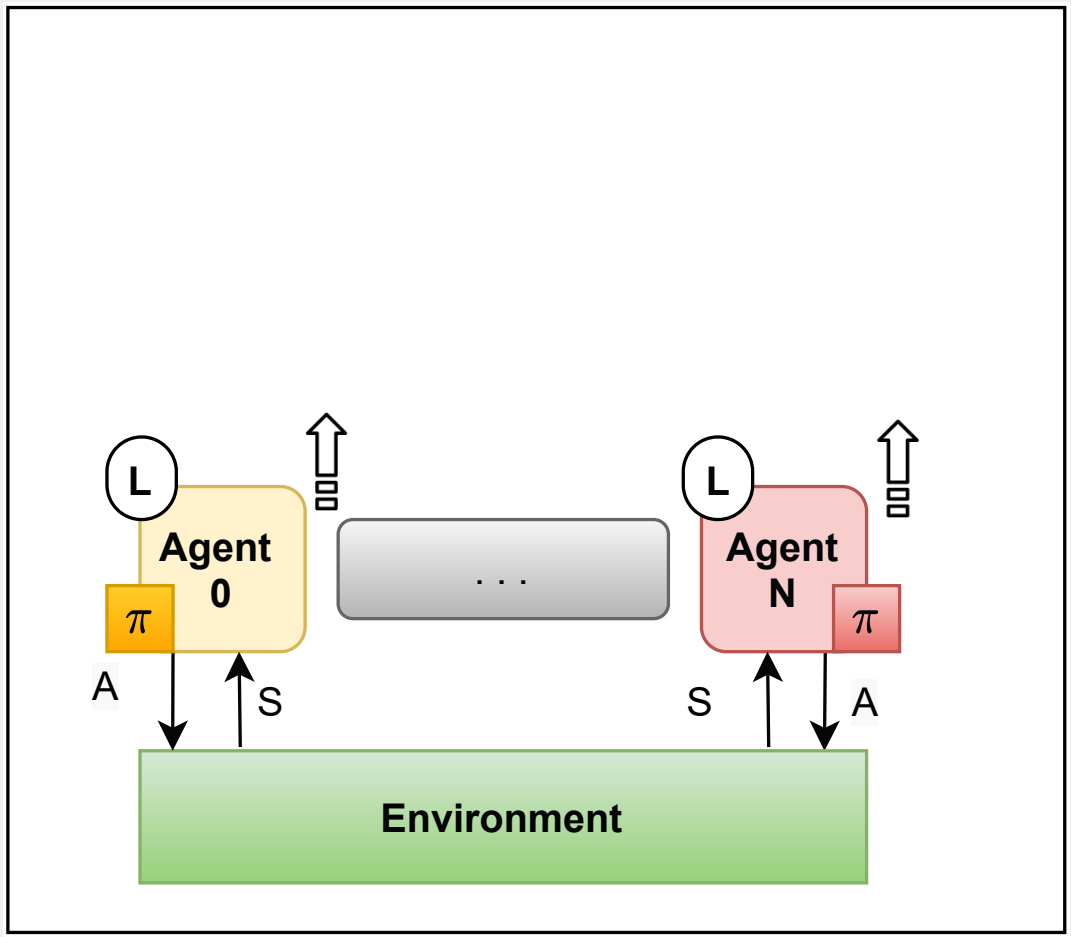
MARL Systems: Learning Scheme

Decentralised Training and Decentralised Execution (DTDE)

- Each node has its local policy/value table
- They can perceive the environment state (or they can observe a part of it)
- With the current state, they perform an action following their local policy (*Control*)
- In the next time step, they update their policy following a local reward function (*Learn*)
- Both used in offline and online (i.e. deployment time) setting

MARL Systems: Learning Scheme

Decentralised Training and Decentralised Execution (DTDE)



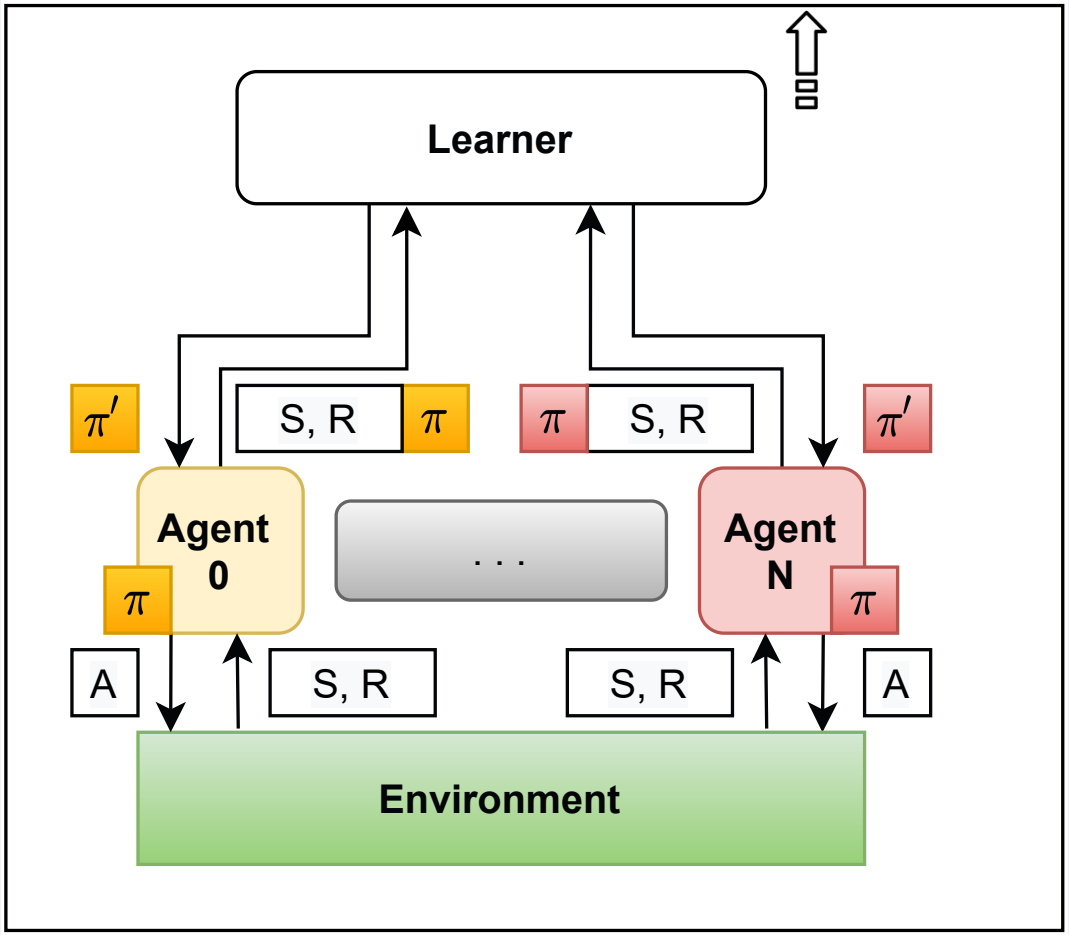
MARL Systems: Learning Scheme

Centralised Training and Decentralised Execution (CTDE)

- A simulation time learning online execution pattern
- *Simulation time*
  - Each node follows the typical $o_t, a_t, o_{t+1}, a_{t+1}, \ldots$ trajectory using a local policy
  - After an episode, this trajectory (or something derived from it) will be sent to a central learner
  - It, using a global view of the system, improves the policies of the agents
  - An then the policies will be shared with the agents
- *Execution time*
  - Each agent has the local policy distilled during the simulation time
  - With it, they act in the environment

MARL Systems: Learning Scheme

Decentralised Training and Decentralised Execution (DTDE)



Single Agent ➔ Multi Agent

Big Question: Can we use Single Agent RL in MAS?

- It depends on the *task type*, *policy type*, *communication* constraints, *execution* constraints, and *learning* constraints
- Single agent RL could be easily adapted in some cases:
  - Task type: cooperative
  - Policy type: homogeneous
  - Communication constraints: independent
  - Execution constraints: centralised/decentralised
  - Learning constraints: centralised/decentralised
- What about the reward function?
  - A reward function for each agent?
  - A collective reward function?
  - Combination?

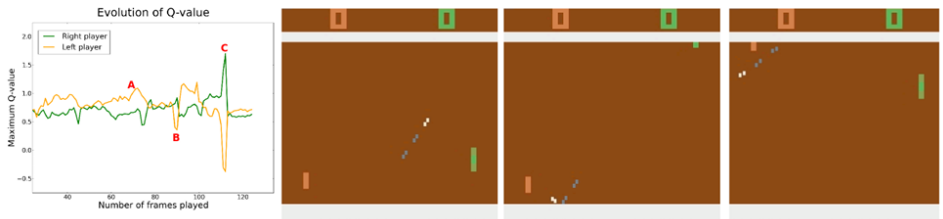
Independent Q Learning [3]

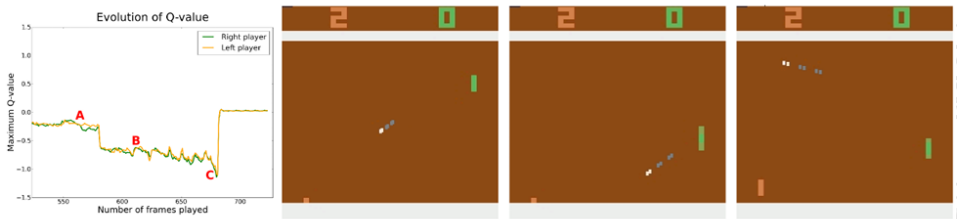
Idea

- Each agent has its own Q table
- Each agent improves its Q table independently following a local policy

Problems

- The agents could learn a policy that is not optimal for the collective goal
- The agents could learn a policy that is not optimal for the other agents
- The agents could learn a policy that is not optimal for the environment

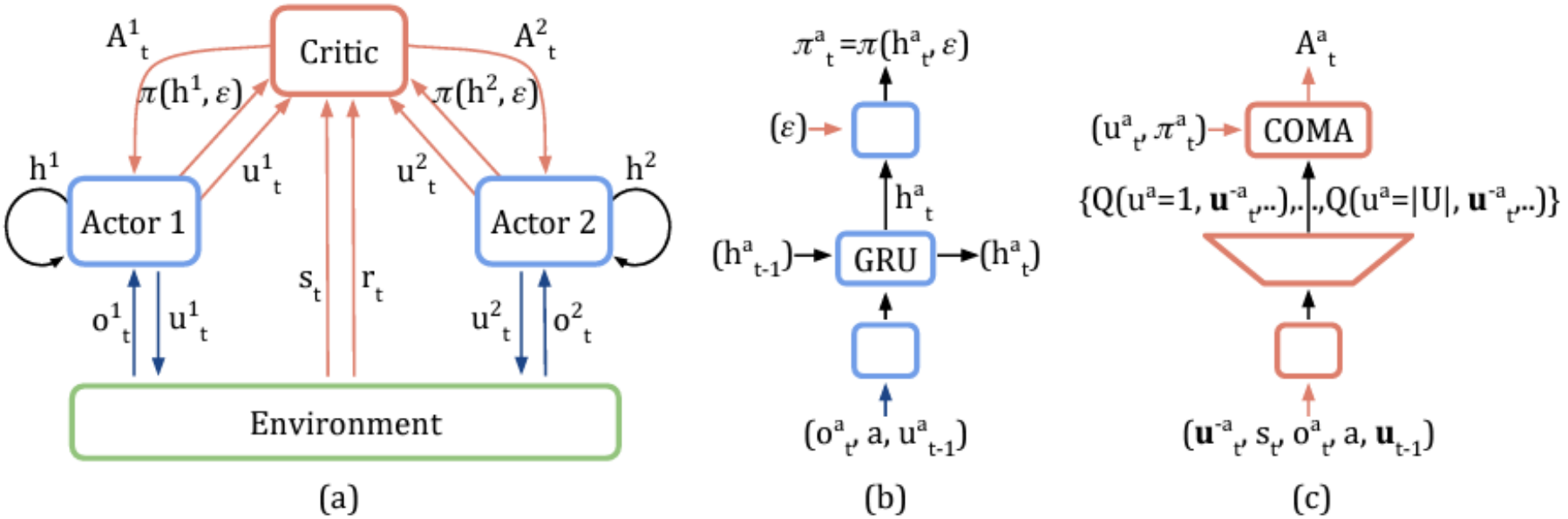




[3] Ming Tan. "Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents". In: ed. by Paul E. Utgoff. Morgan Kaufmann, 1993, pp. 330–337. doi: 10.1016/b978-1-55860-307-3.50049-6. url: https://doi.org/10.1016/b978-1-55860-307-3.50049-6

COMA [4]

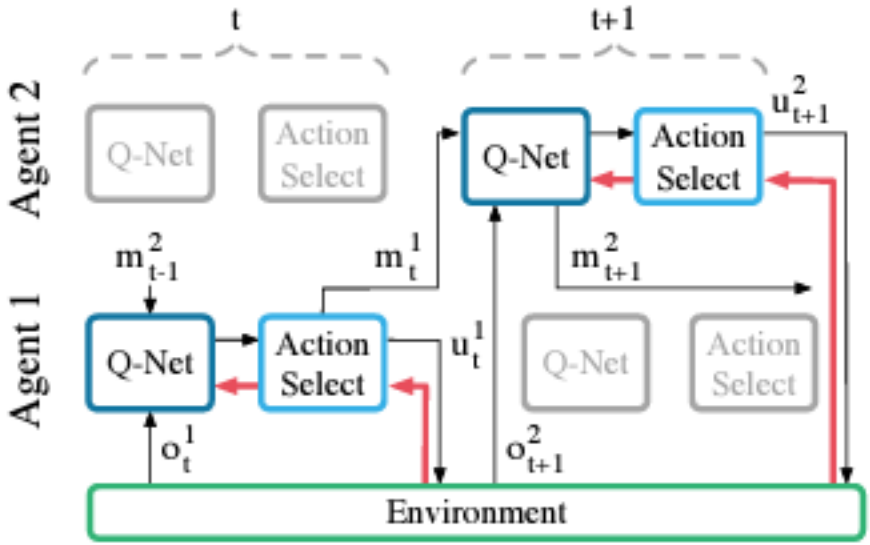
- COMA (Counterfactual Multi-Agent) is a technique that uses a *centralised* training and a *decentralised* execution
- Idea: ease the learning processes by giving a *global* view of the environment
- Counterfactual baseline: a baseline that is independent of the actions taken by the agent
  - Used to understand the impact of the agent's actions



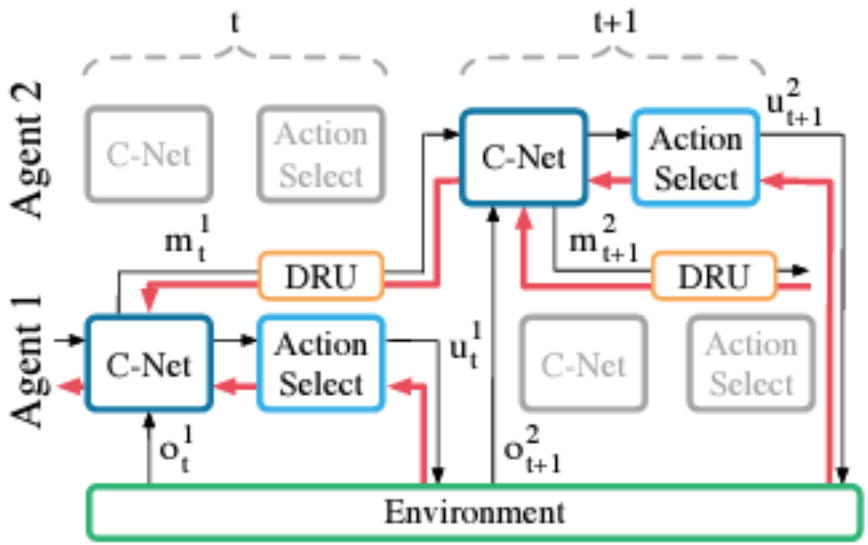
[4] Ming Tan. "Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents". In: ed. by Paul E. Utgoff. Morgan Kaufmann, 1993, pp. 330–337. doi: 10.1016/b978-1-55860-307-3.50049-6. url: https://doi.org/10.1016/b978-1-55860-307-3.50049-6

Learning to Communicate [5]

- The agent coordination can be ruled via *communication*
- This communication can be learned through *reinforcement learning*



(a) RIAL - RL based communication



(b) DIAL - Differentiable communication

[5] Jakob Foerster et al. "Learning to communicate with deep multi-agent reinforcement learning". In: *Advances in neural information processing systems* 29 (2016)

MAPPO [6]

- MAPPO (Multi-Agent Proximal Policy Optimisation) is the extension of PPO to MARL
  - Promising results in cooperative and competitive tasks with few agents



[6] Chao Yu et al. "The surprising effectiveness of ppo in cooperative multi-agent games". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24611–24624

Decentralised Hysteretic Deep-Recurrent-Q Learning [7]

- The previous approach suppose to have a centralised controller
- cons:
  - offline learning
  - hard to scale (without weight sharing)
- Decentralised Hysteretic Deep-Q Learning (DH-DQL) is a technique that uses a *decentralised* training and a *decentralised* execution
- The agents maximise a local reward function
  - Collective goal reached by emergent behaviour
- Strategies:
  - *Hysteretic*: heuristic that prevents the agents to change their policy too often bringing in miscoordination
  - *Recurrent Neural Network*: to manage the partial observability

[7] Shayegan Omidshafiei et al. "Deep decentralized multi-task multi-agent reinforcement learning under partial observability". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2681–2690

Learning in Many Agent Systems

Problems

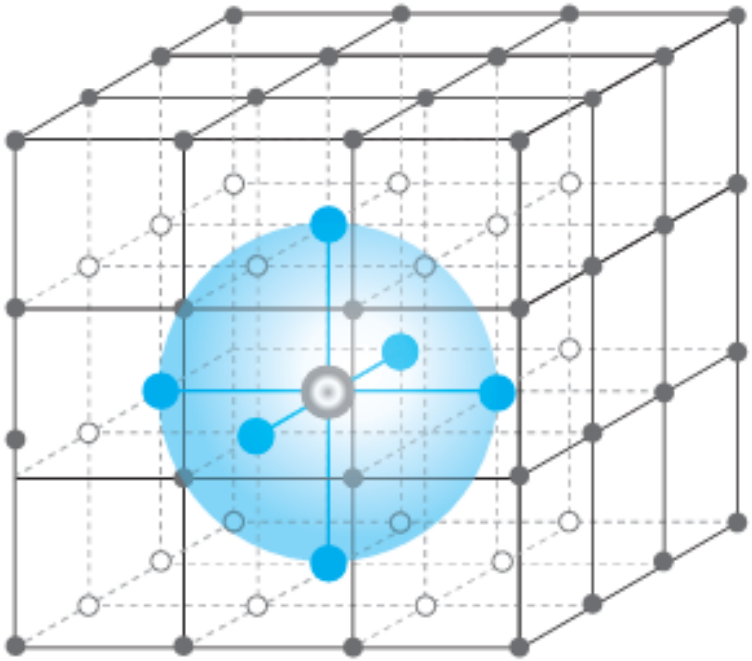
- Many Agent Systems are challenging environments for learning
  - The agents involved could be hundreds or thousands
  - The state space could be very large
  - The agents only have partial information about the state
  - No central controller (at runtime)

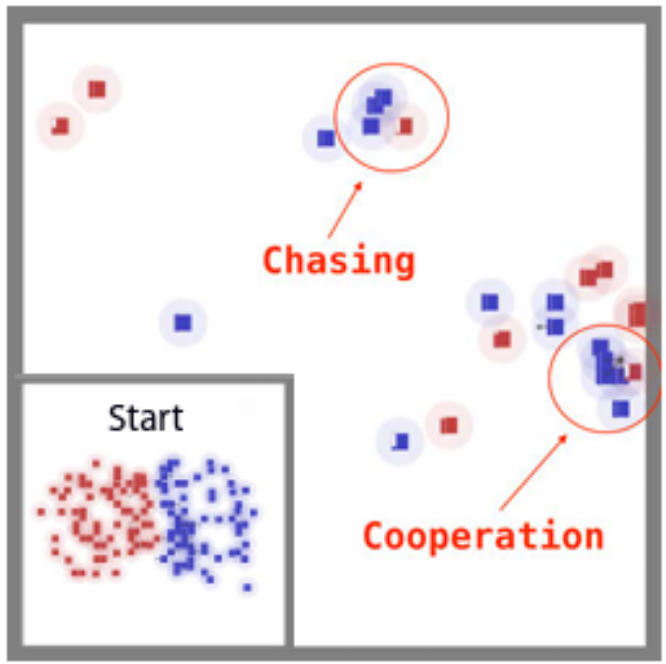
How?

- Constraints on policy: *homogeneous*
  - The **complexity** emerges through interactions
- Partial observability handled through *deep learning* (e.g., Deep Q-Learning)
- Centralised training and decentralised execution strategy
  - During the simulation the agents are trained using a centralised training algorithm
  - The Q table is local to each agent
  - After the simulation, it will be copied into each agent
  - No need for a central controller at runtime!
- Communication handled through *message passing* and encoded as a local state
- *Dense* reward to simplify the learning process and to enable cooperation

Mean-field Reinforcement Learning [8]

- **Mean-field** Reinforcement Learning (MRL) is a technique to reduce the complexity of a multi-agent system
- The idea is to approximate the behaviour of the agents as a single agent (i.e., the **mean-field** agent)

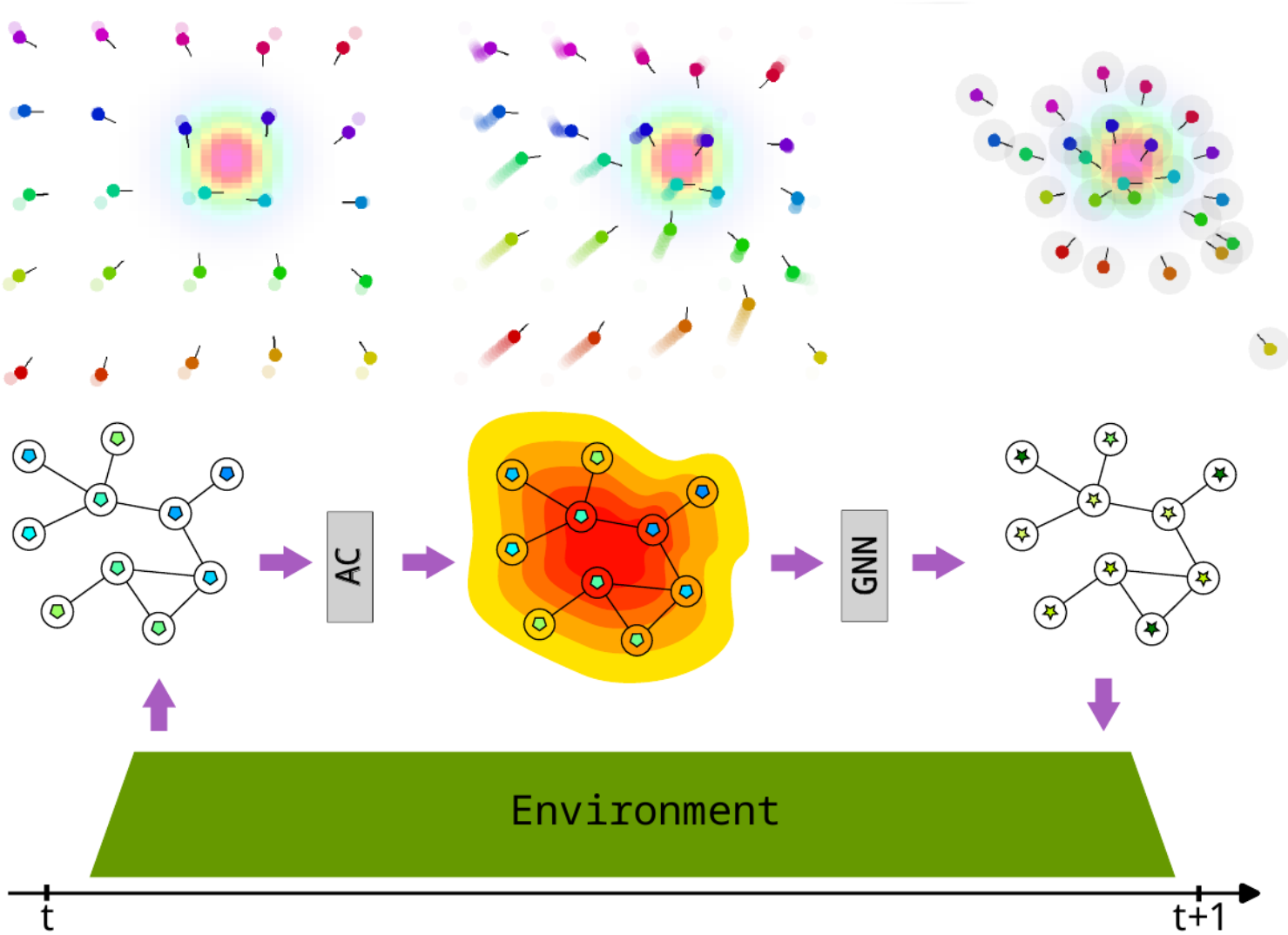




[8] Yaodong Yang et al. "Mean field multi-agent reinforcement learning". In: *International conference on machine learning*. PMLR. 2018, pp. 5571–5580

Field-Informed Multi-Agent Reinforcement Learning

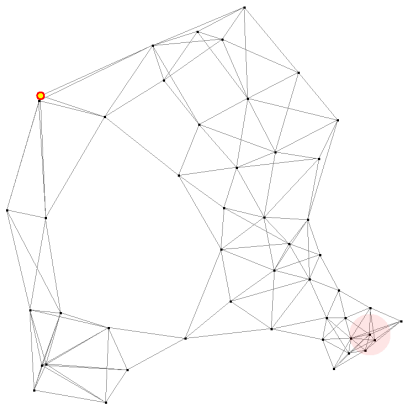
- Field-Informed MARL is a technique that guides the learning process of the agents using a *computational* field
- Idea: ease the learning process by providing a *prior* knowledge about the environment

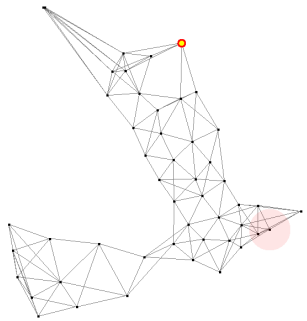


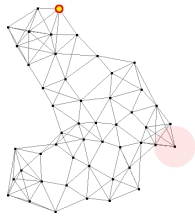
Example: Follow the leader

Scenario

- Configuration: N agents, M neighbours, 8 directions + standstill
- But aggregate computing is used to share the leader towards a gradient
- Reward function is composed of two terms
  - collision: the nearest agent distance should be as large as possible
  - leader: the distance to the leader should be as small as possible







Open Challenges

- **Curse of dimensionality**
  - Exponential growth in computational complexity from an increase in state and action dimensions. Also a challenge for single-agent problems.
    - Partially handled by homogeneous policies
- **Specifying a good (learning) objective**
  - Agent returns are correlated and cannot be maximized independently.
    - Multi-agent credit assignment problem (how can I know if my action was good?)
- **Need for coordination**
  - Agent actions affect other agents and could confuse other agents (or herself) if not careful. Also called destabilizing training
- **No formal guarantees**
  - No formal guarantees on convergence, stability, or optimality

Conclusion

- Multi-Agent Reinforcement Learning is a very active research field
- From Single to Multi-Agent RL, different axes to consider:
  - Training strategy: centralised or decentralised
  - Policy constraints: homogeneous or heterogeneous
  - Agent population: few or many
  - ...
- Several techniques have been proposed to handle the complexity of the problem
  - MAPPO, Mean-Field RL, COMA, ...
- No one size fits all solution
  - Some handle only cooperative problems
  - Other works only with a few agents
  - ...
- My hope: a general framework for MARL (like the one for RL)

Multi-Agent Reinforcement Learning

Unleashing Collective Intelligence

Gianluca Aguzzi gianluca.aguzzi@unibo.it

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

27/07/2023