Announcements

- Homework 5 Deadline **extended to Friday (March 28, 11:59pm).**
- No TA hours tomorrow, normal TA hours resume next week.
- Your Full Proposal + Lit Review are due April 4th!

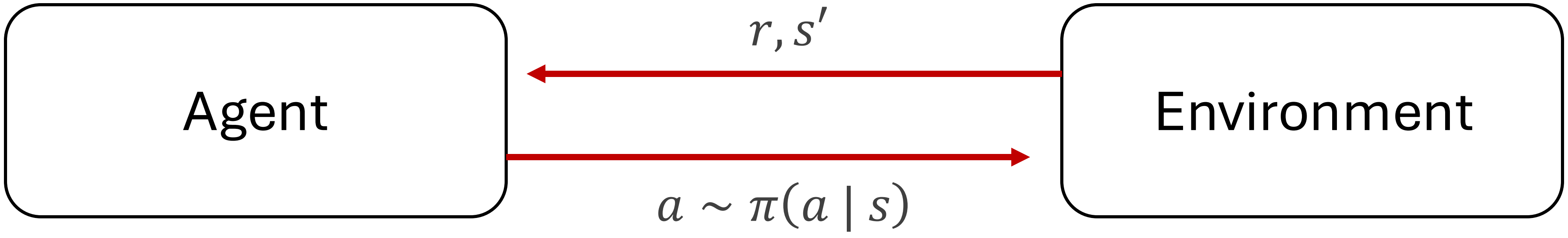
Multi-Agent Reinforcement Learning

Guest Lecturer: Connor Mattson

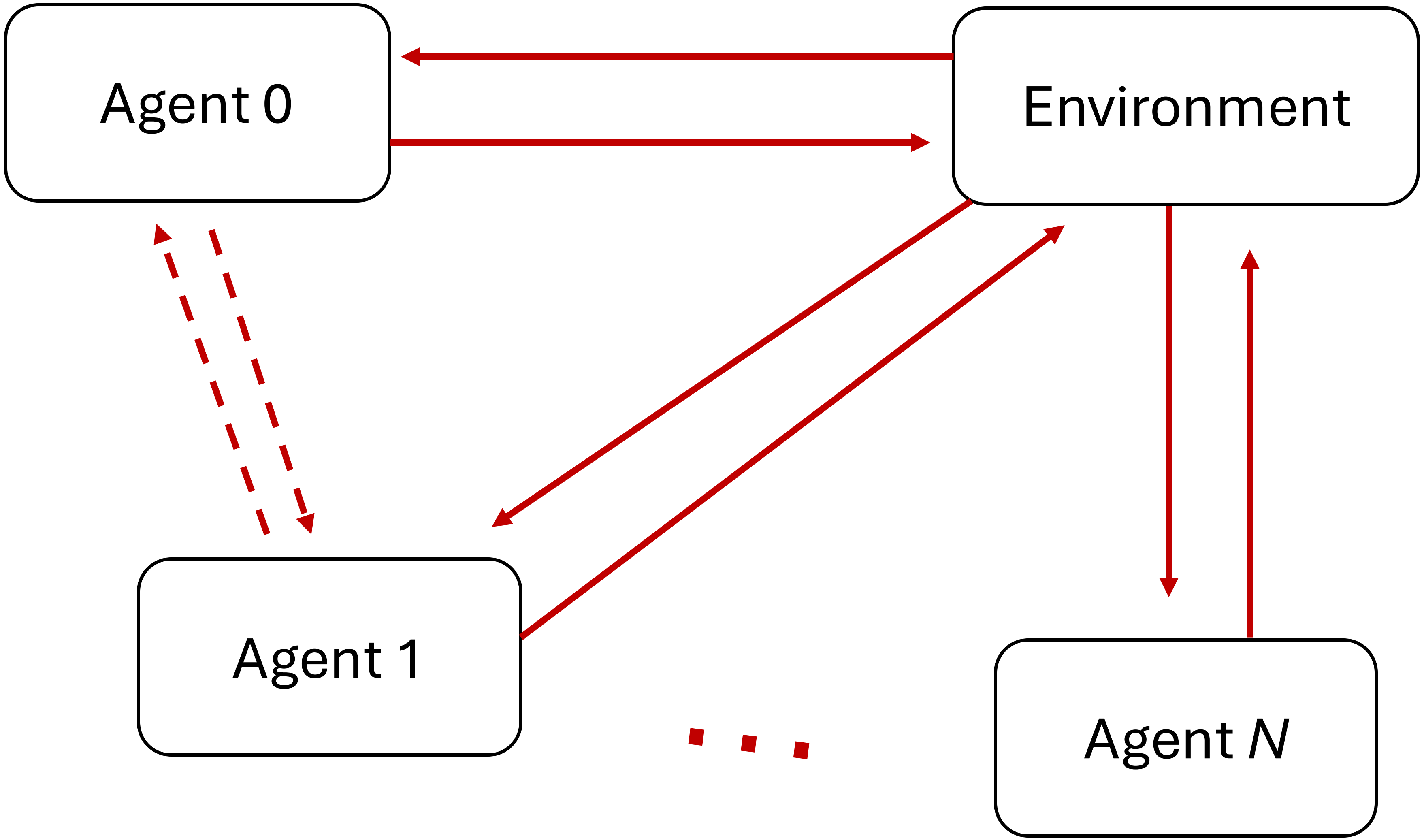
CS5955/CS6955: Advanced Artificial Intelligence, Spring 2025

What are Multi-Agent Systems?

Recall our simple single-agent interaction loop!

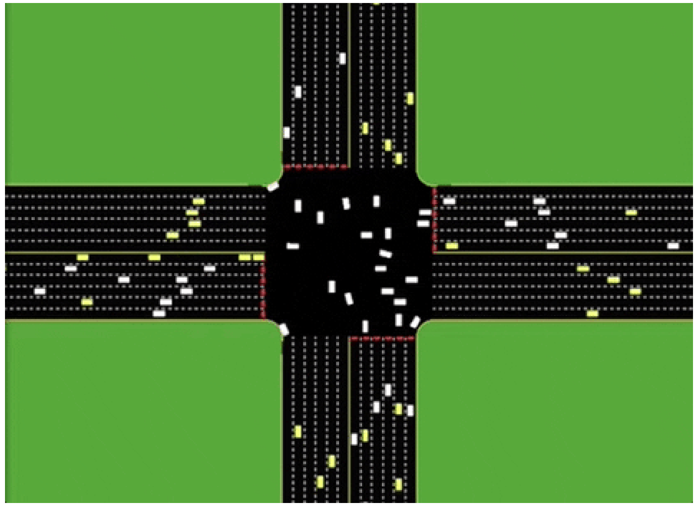


What are Multi-Agent Systems?

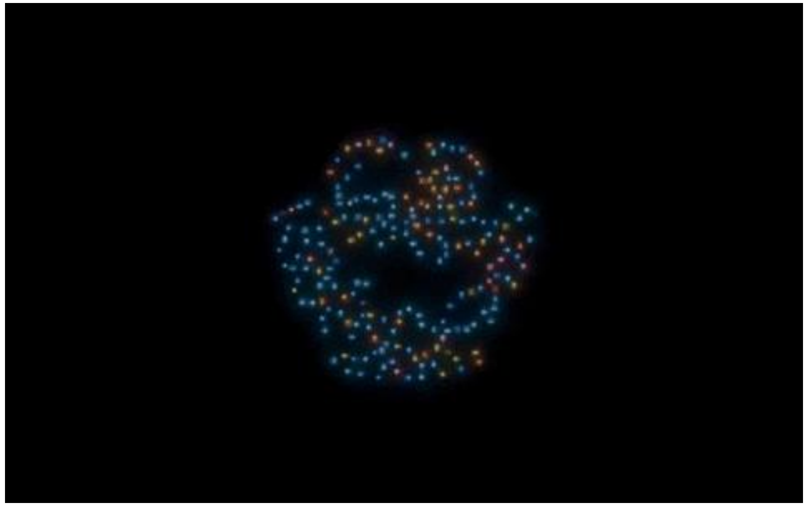


<u>What information is being passed along the red arrows?</u>

Examples of Multi-Agent Applications



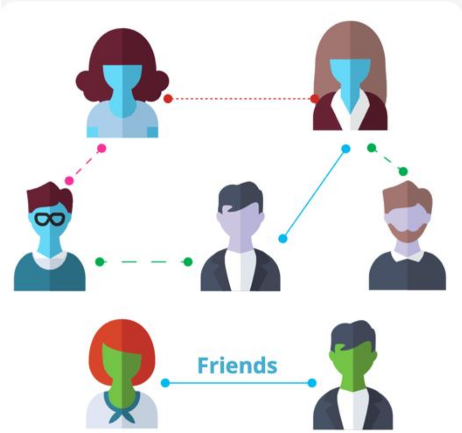
Autonomous Driving
+ Traffic Control



UAVs + Drones

Coordinated Robot Motion

n-Player Games



Psychology + Sociology

Human Multi-Robot Interaction

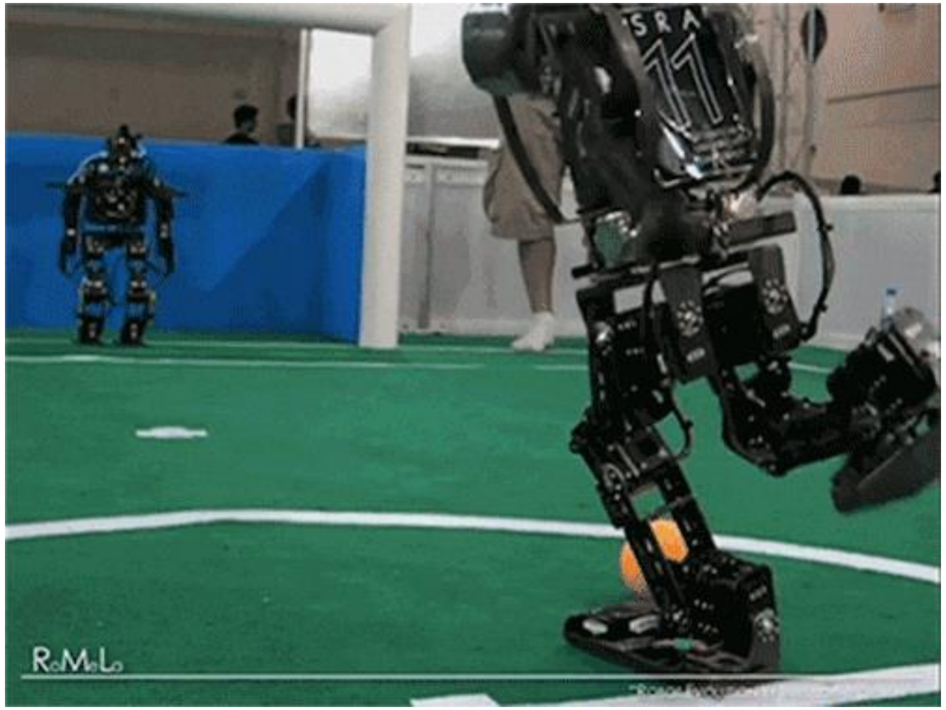
Amazon Robotics

Why Study Multi-Agent Systems?

Some of humanities greatest feats are multi-agent problems!

Examples:
Team Sports, Government, International Relations, Academic Research, Game Theory, Resource Allocation, etc.

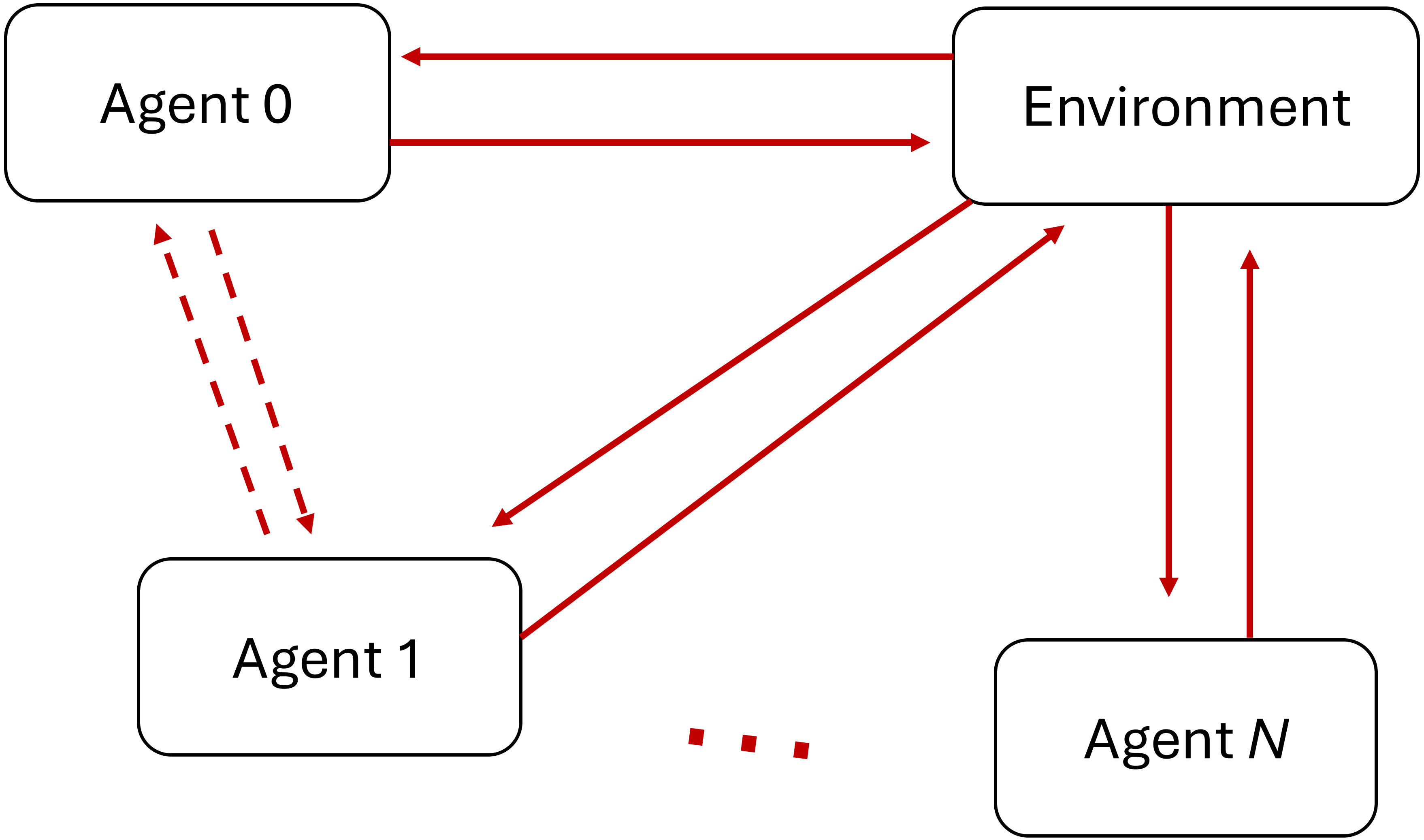
Robots/Agents should also possess these capabilities!



(we have a long way to go!)

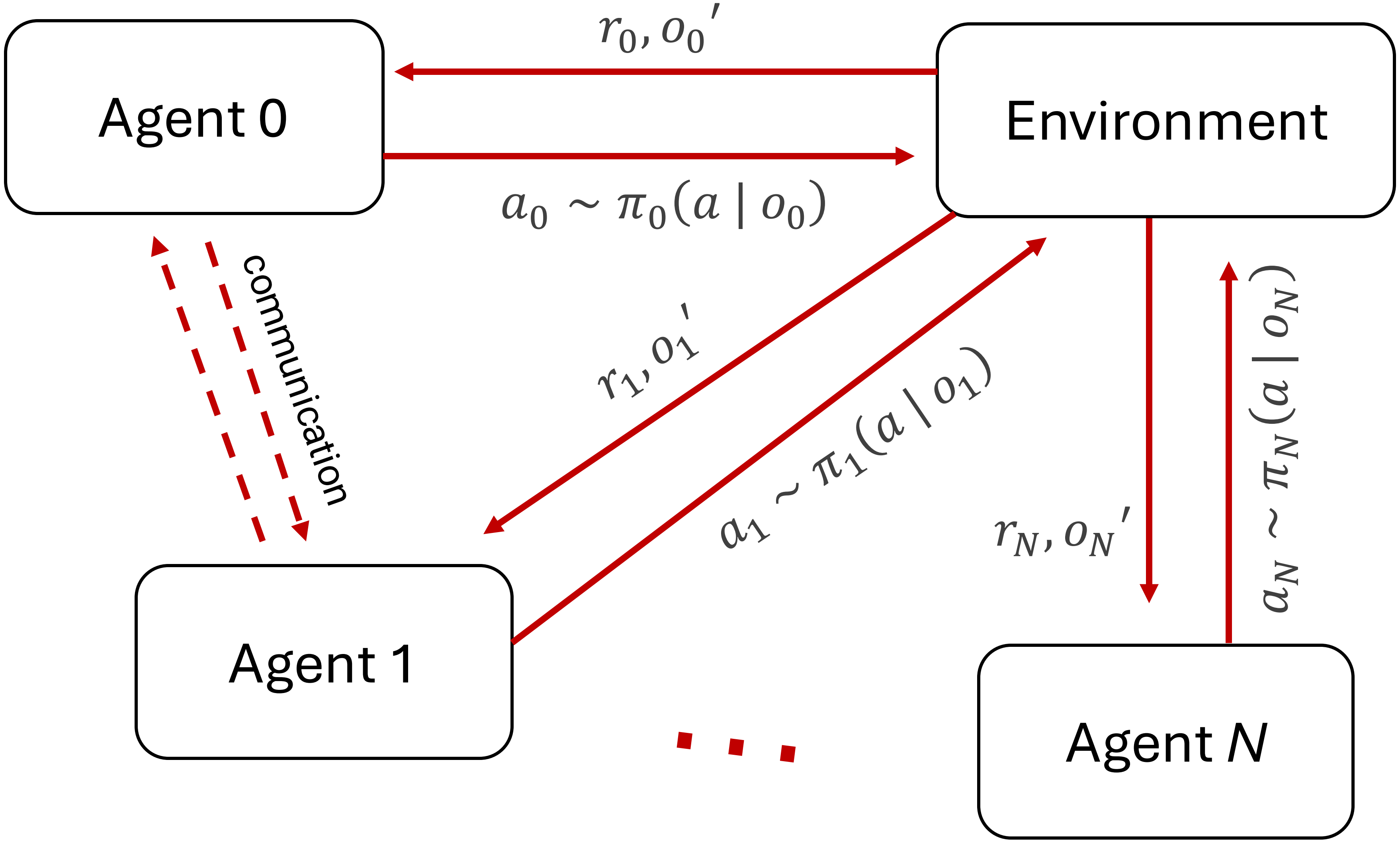
Videos: Fox News, RoboCup

What are Multi-Agent Systems?



<u>What information is being passed along the red arrows?</u>

What are Multi-Agent Systems?



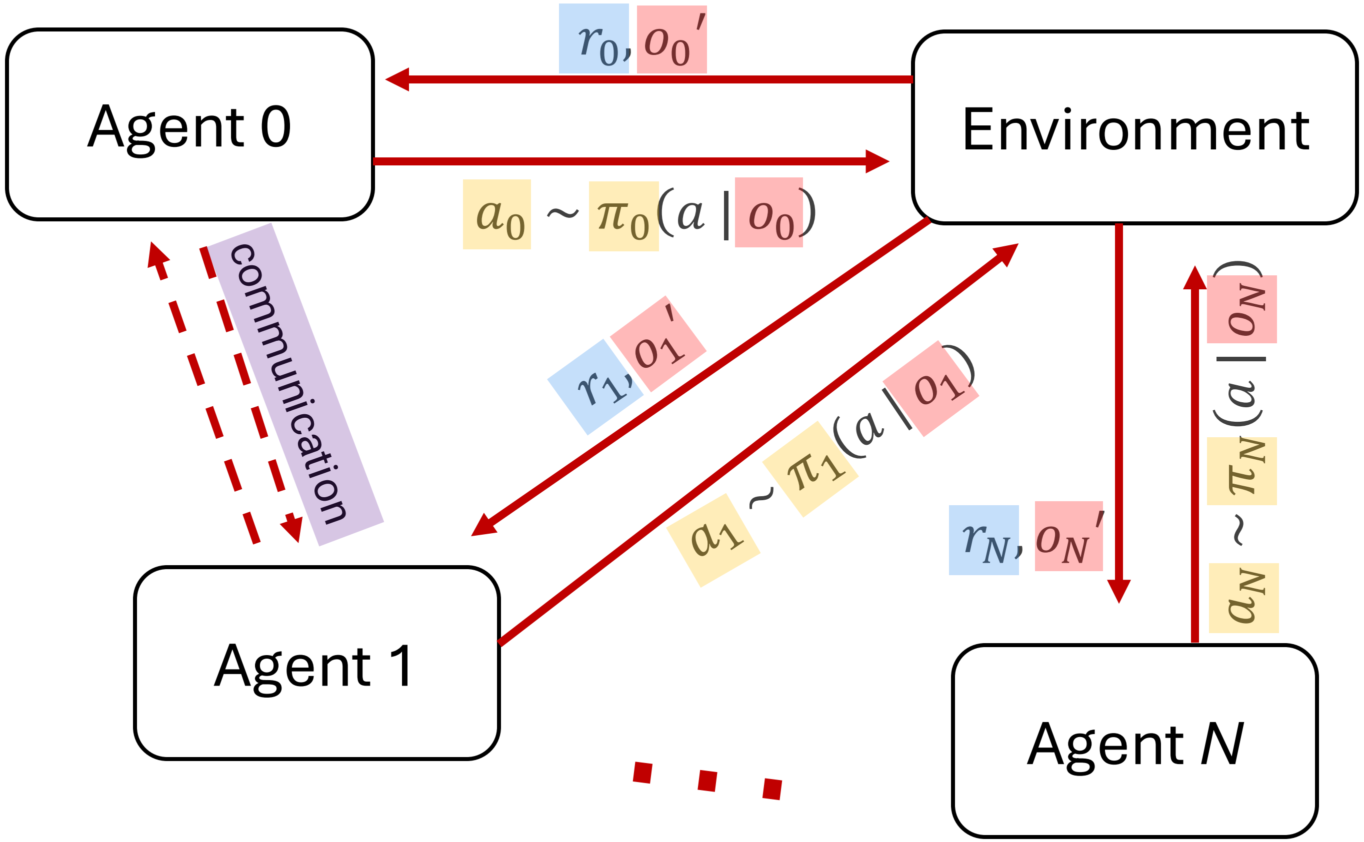
<u>What do you notice?</u>

Rewards?

Actions/Policies?

State/Observations?

How does MARL differ from single-agent RL?



Separate State Signals: Agent 0 and Agent 1 receive o_0' and o_1', respectively.

Separate Reward Signals: Agent 0 and Agent 1 receive r_0 and r_1, respectively.

Separate Policies: π_i can be different for each agent (so a_i are different!)

Communication: Agents can communicate with each other (if allowed)

Parallels to Single-Agent RL

Same Desired Outcome: Policies that maximize an expected return.

Assumptions: We don’t know transition dynamics but have reward signals.

Same Definitions of Value: We can model/approximate V(s) and Q(s, a).

MARL is often the expansion of Single-Agent RL to trickier problems!

Single-Agent Learning	Multi-Agent Learning
Behavioral Cloning	Imitation-MA (2007) Coord-MAIL (2017)
Advanced Behavioral Cloning	GMA-BC (2018)
Multi-Armed Bandits	Game Theory (Normal-Form Games)
Markov Decision Processes	Game Theory (Markov Games)
Exact Solutions for MDPs	MARL-Textbook
Value-Based RL	MARL-Textbook
TD Learning	MARL-Textbook

Single-Agent Learning	Multi-Agent Learning
Q-Learning	IQL (1993)
DQN	MA-DQL (2015) VDN (2017) QMIX (2018)
Policy-Gradient (REINFORCE)	Multi-Agent PG (2020)
A2C/A3C	SEAC (2020)
PPO	MAPPO (2022)
SAC	Dec-SAC (2021)
DDPG	MADDPG (2020)

Single-Agent Learning	Multi-Agent Learning
TD3	MA-TD3 (2020)
Model-Based RL	MB-MARL (2022)
Offline RL	MARL-IGL (2023)
RLHF	MARLHF (2024)
Inverse RL	MAIRL (2021)
LLM Agents	Generative Agents (2023)
…	…

We could spend an **entire semester** talking about MARL! See NYU's TR GY 8103.

Single Agent Reinforcement Learning

Multi-Agent Reinforcement Learning

Game Theory

Normal-Form Games (Bandits++)

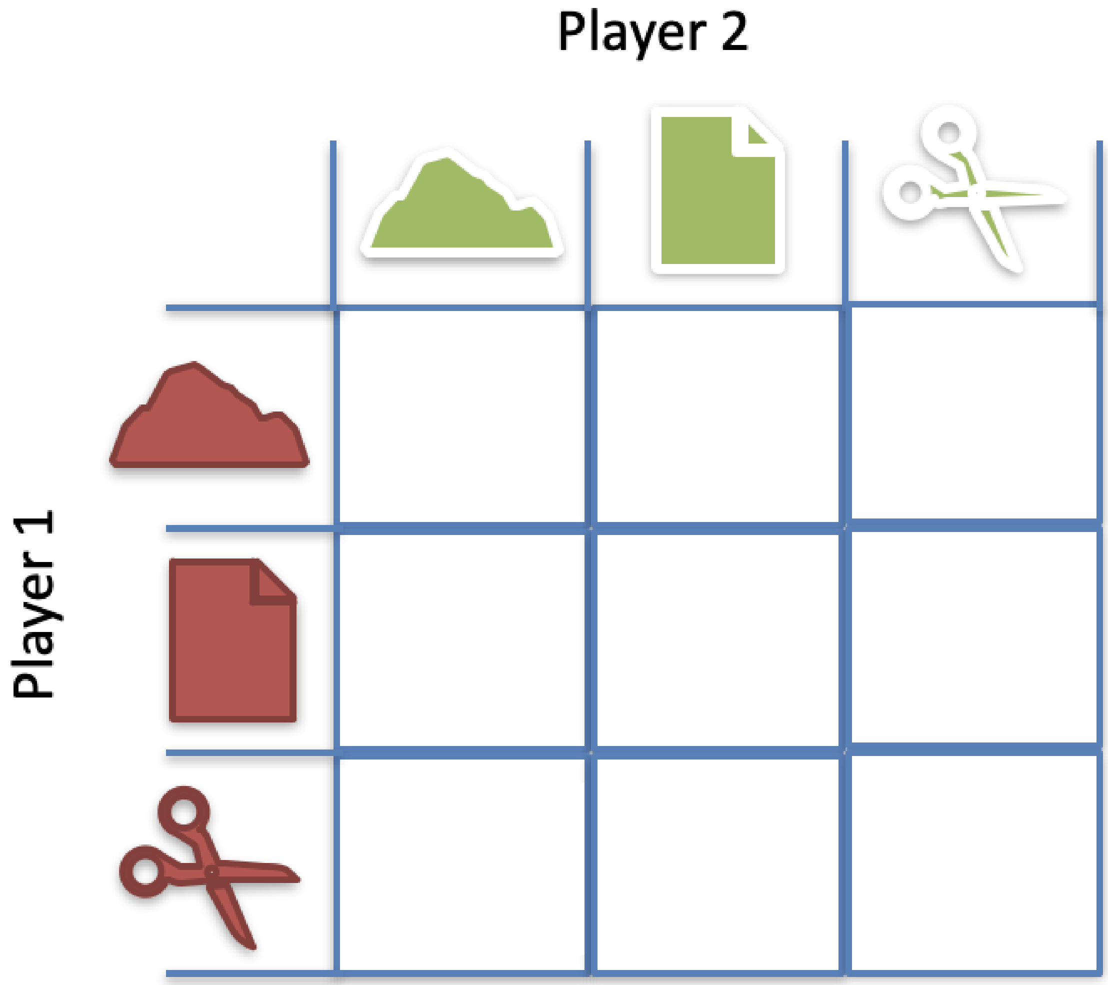
Definition 2 (Normal-form game) *A normal-form game consists of:*

- *Finite set of agents* $I = \{1, ..., n\}$
- *For each agent* $i \in I$*:*
  - *Finite set of actions* A_i
  - *Reward function* $\mathcal{R}_i : A \to \mathbb{R}$*, where* $A = A_1 \times ... \times A_n$

Albrecht (2024)

Most Popular Normal-Form Game: Rock, Paper, Scissors!

- A single-shot, simultaneous, full-information game
- **Single shot**: Played once
- **Simultaneous**: Both actions are revealed at once
- **Full-information**: Everything about the game is known (in this case, the costs / rewards)

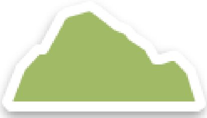
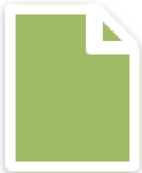
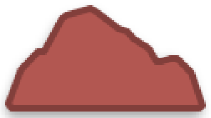
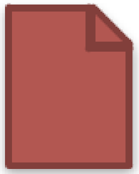


Most Popular Normal-Form Game: Rock, Paper, Scissors!

- A single-shot, simultaneous, full-information game
- **Single shot**: Played once
- **Simultaneous**: Both actions are revealed at once
- **Full-information**: Everything about the game is known (in this case, the costs / rewards)

Player 2

Player 1

			
	(0,0)	(-1,1)	(1,-1)
	(1,-1)	(0,0)	(-1, 1)
	(-1,1)	(1,-1)	(0,0)

Normal-Form Games (Bandits++)

	R	P	S
R	0,0	-1,1	1,-1
P	1,-1	0,0	-1,1
S	-1,1	1,-1	0,0

(a) Rock-Paper-Scissors

	A	B
A	10	0
B	0	10

(b) Coordination Game

	C	D
C	-1,-1	-5,0
D	0,-5	-3,-3

(c) Prisoner's Dilemma

Zero-Sum Game

Common Reward

Mixed Reward

We won't go over the various ways to solve NFGs from evaluative feedback, but this is a well studied area of game theory!

Vinitsky (2024)

Markov Games (MDPs++)

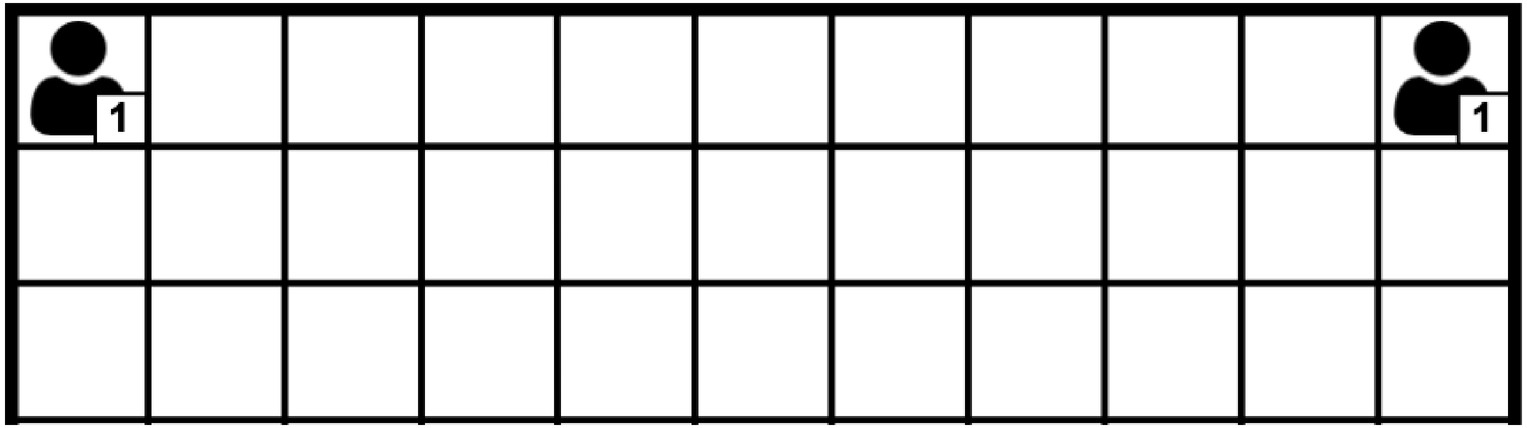
Definition 3 (Stochastic game) *A stochastic game consists of:*

- *Finite set of agents* $I = \{1, ..., n\}$
- *Finite set of states* S*. with subset of terminal states* $\bar{S} \subset S$
- *For each agent* $i \in I$*:*
  - *Finite set of actions* A_i
  - *Reward function* $\mathcal{R}_i : S \times A \times S \rightarrow \mathbb{R}$*, where* $A = A_1 \times ... \times A_n$
- *State transition probability function* $\mathcal{T} : S \times A \times S \rightarrow [0, 1]$ *such that*

$$\forall s \in S, a \in A : \sum_{s' \in S} \mathcal{T}(s, a, s') = 1 \tag{3.1}$$

Albrecht (2024)

Markov Games (MDPs++)



Each agent has four actions:

{UP, LEFT, DOWN, RIGHT}

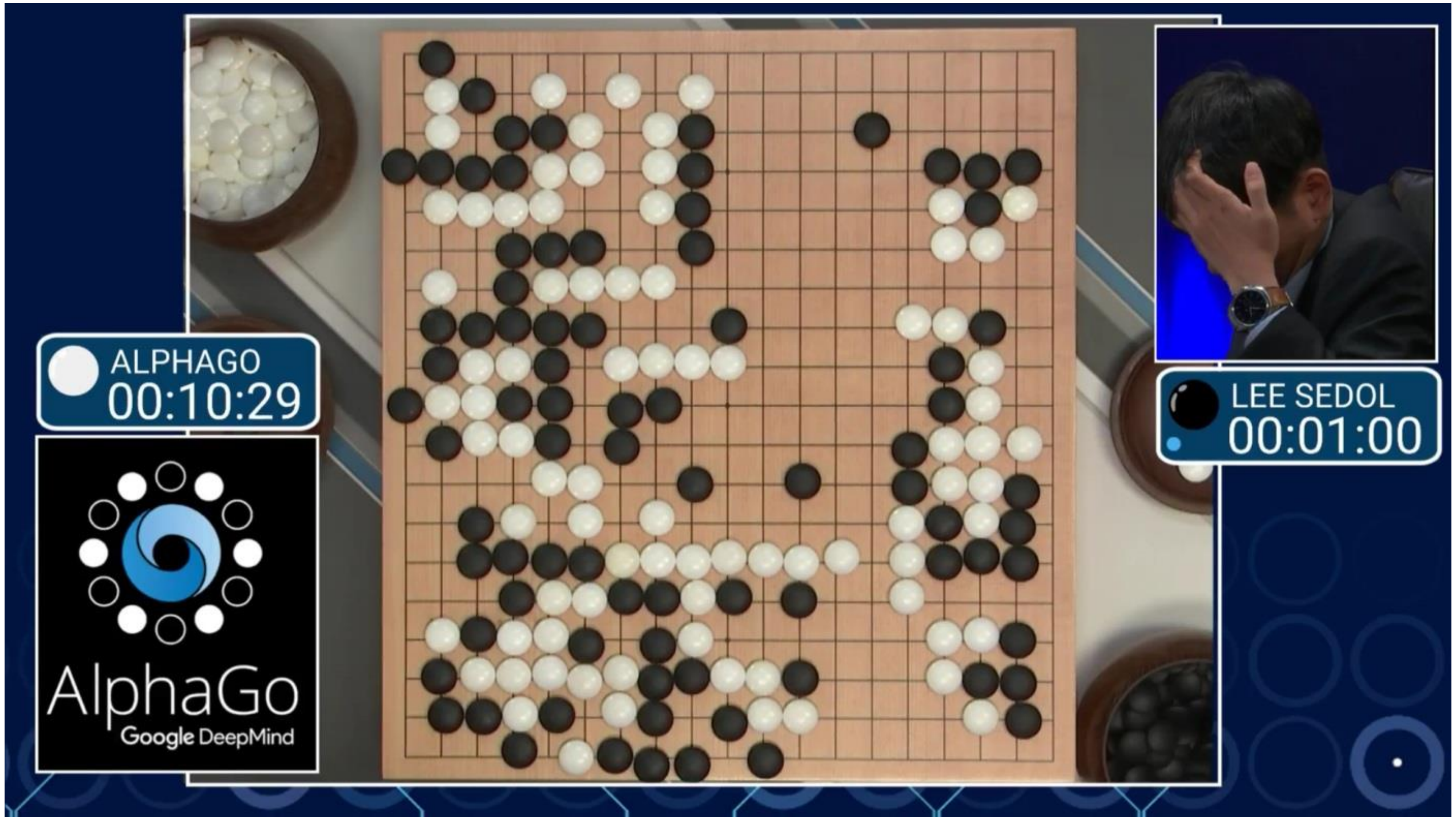
A step forward in the game is a **joint action**

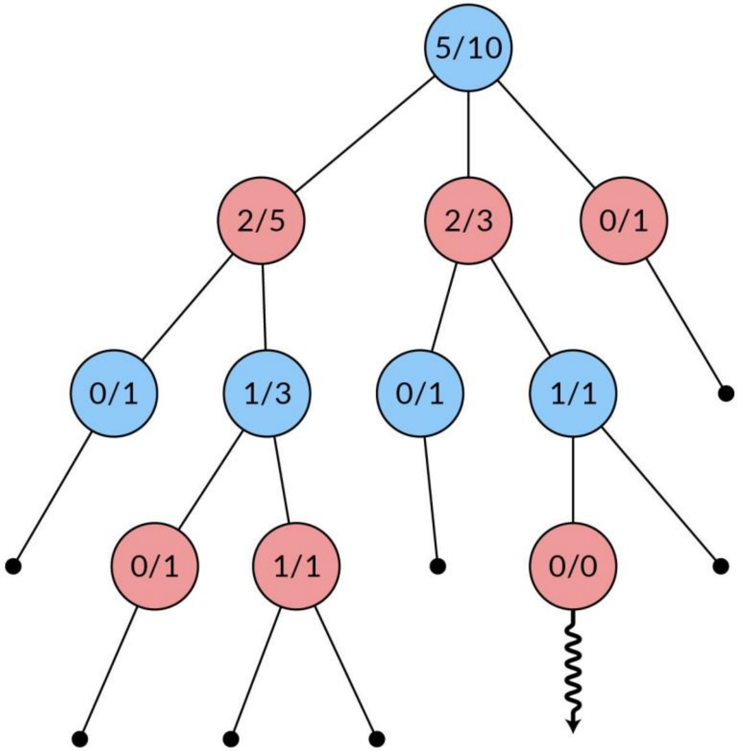
$$\boldsymbol{a} \in \boldsymbol{A} = \boldsymbol{A_0} \times \boldsymbol{A_1} \times \cdots \times \boldsymbol{A_N}$$

- **Sequential Decision Making**: state changes over time (could be finite or infinite horizon).
- **Simultaneous**: Both agents take actions at the same time.
- **System Dynamics:** The actions of both agents (joint action) determine the subsequent state.

Albrecht (2024)

Competition (2-player games)





We already talked about strategies for learning turn-based 2-player games when we covered AlphaGo and AlphaZero!

Cooperation!

For today, let's focus on the *collaborative* case**.**

All agents still have independent reward signals

$$r_0, r_1, \dots, r_N$$

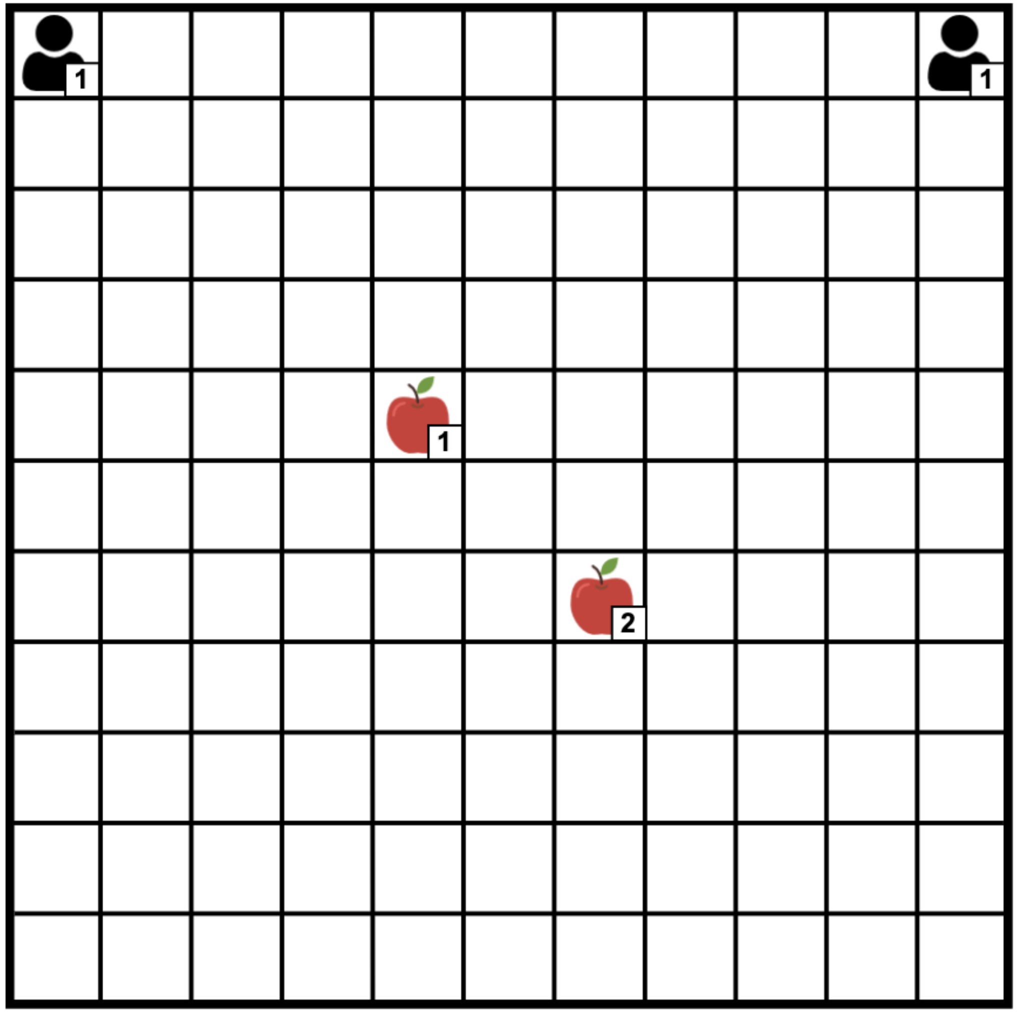
Our objective will be to **maximize the collective welfare** of the entire system.

$$\max_{(\pi_0,\pi_1,\dots,\pi_N)} E_{(\pi_0,\pi_1,\dots,\pi_N)}\left[\sum_{t=0}^{\infty}\sum_{i=0}^{N}\gamma^t\, R_i(s_t,\pi_i(s_t),s_{t+1})\right]$$

Albrecht (2024)

Apple Picking Task

- N agents must collect all apples, but some apples cannot be picked alone and require teamwork!
- Each agent has a "level" that indicates how skilled they are at picking apples.
- Each apple has a "level" that indicates how difficult it is to pick that apple.
- An apple can only be collected if the sum of agent levels simultaneously picking it is greater than or equal to the level of the apple.
- Discrete action space with 6 actions.
  - A = {UP, DOWN, LEFT, RIGHT, COLLECT, NO_OP}
- Reward is +1 for all agents every time an apple is picked.



Albrecht (2024)

Simplest Solution: Convert MARL to RL

The simplest way to apply RL algorithms to multi-agent settings is to **reduce the problem to a single-agent problem** and apply traditional RL techniques.

Central Learning

- Apply one single-agent RL algorithm to **control all agents centrally**.
- A central policy is learned over the joint action space $A = (A_1 \times A_2 \times \cdots \times A_N)$.

Independent Learning

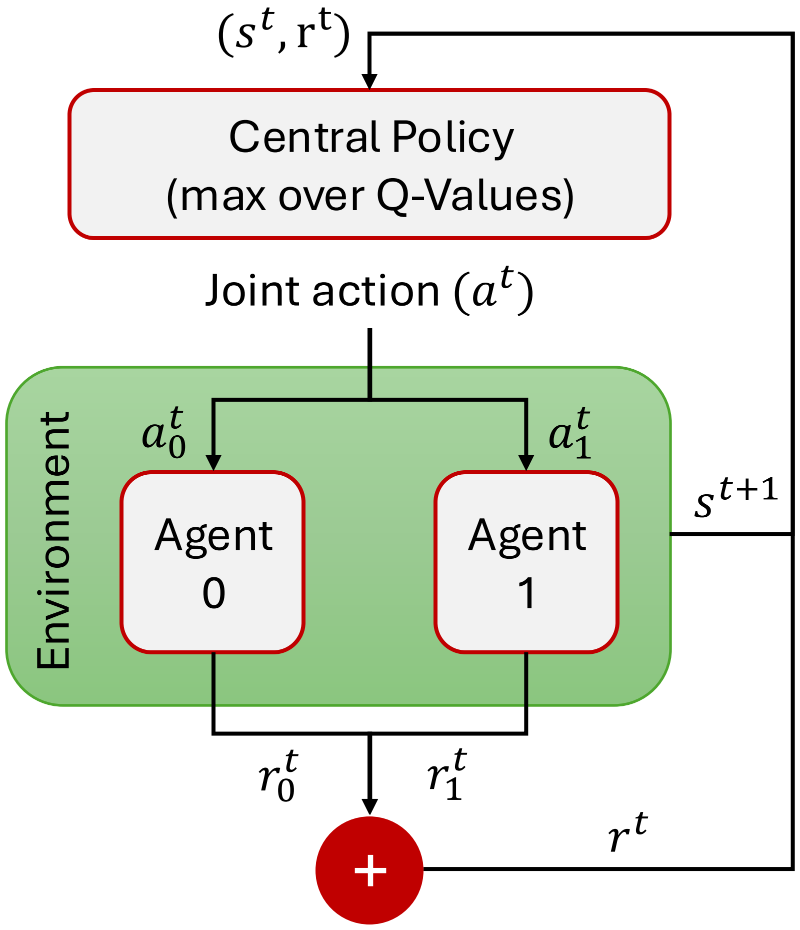
- Apply single-agent RL algorithms to each agent **independently**.
- Agents do not explicitly consider or represent each other!

Albrecht (2024)

Centralized Learning: Algorithm

Algorithm 4 Central Q-learning (CQL) for stochastic games

1: Initialize: $Q(s, a) = 0$ for all $s \in S$ and $a \in A = A_1 \times \ldots \times A_n$
2: Repeat for every episode:
3: **for** $t = 0, 1, 2, \ldots$ **do**
4: Observe current state s^t
5: With probability ϵ: choose random joint action $a^t \in A$
6: Otherwise: choose joint action $a^t \in \arg\max_a Q(s^t, a)$
7: Apply joint action a^t, observe rewards $r_1^t, \ldots, r_n^t$ and next state s^{t+1}
8: Transform $r_1^t, \ldots, r_n^t$ into scalar reward r^t
9: $Q(s^t, a^t) \leftarrow Q(s^t, a^t) + \alpha \left[r^t + \gamma \max_{a'} Q(s^{t+1}, a') - Q(s^t, a^t) \right]$



We learn q-values over the space of joint-actions, so the algorithm doesn't know that we are training multiple agents.

Single-agent RL doesn't know how to process multiple reward signals, so we must transform the joint reward into a scalar reward.

Independent Learning: Algorithm

Algorithm 5 Independent Q-learning (IQL) for stochastic games

// Algorithm controls agent i

Initialize: $Q_i(s, a_i) = 0$ for all $s \in S$, $a_i \in A_i$

Repeat for every episode:

for $t = 0, 1, 2, \ldots$ **do**

  Observe current state s^t

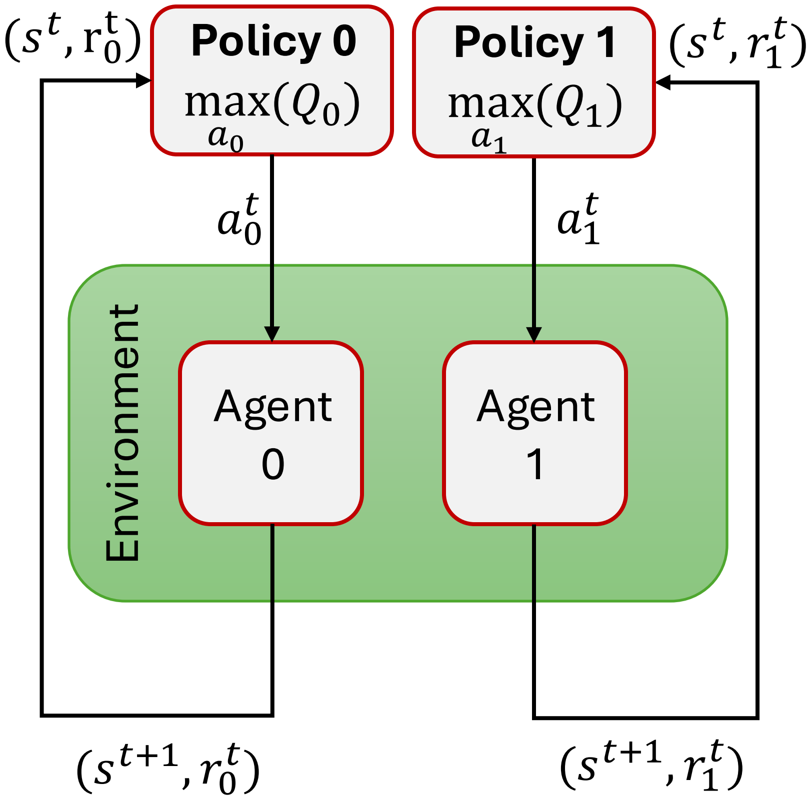
  With probability ϵ: choose random action $a_i^t \in A_i$

  Otherwise: choose action $a_i^t \in \arg\max_{a_i} Q_i(s^t, a_i)$

  (meanwhile, other agents $j \neq i$ choose their actions a_j^t)

  Observe own reward r_i^t and next state s^{t+1}

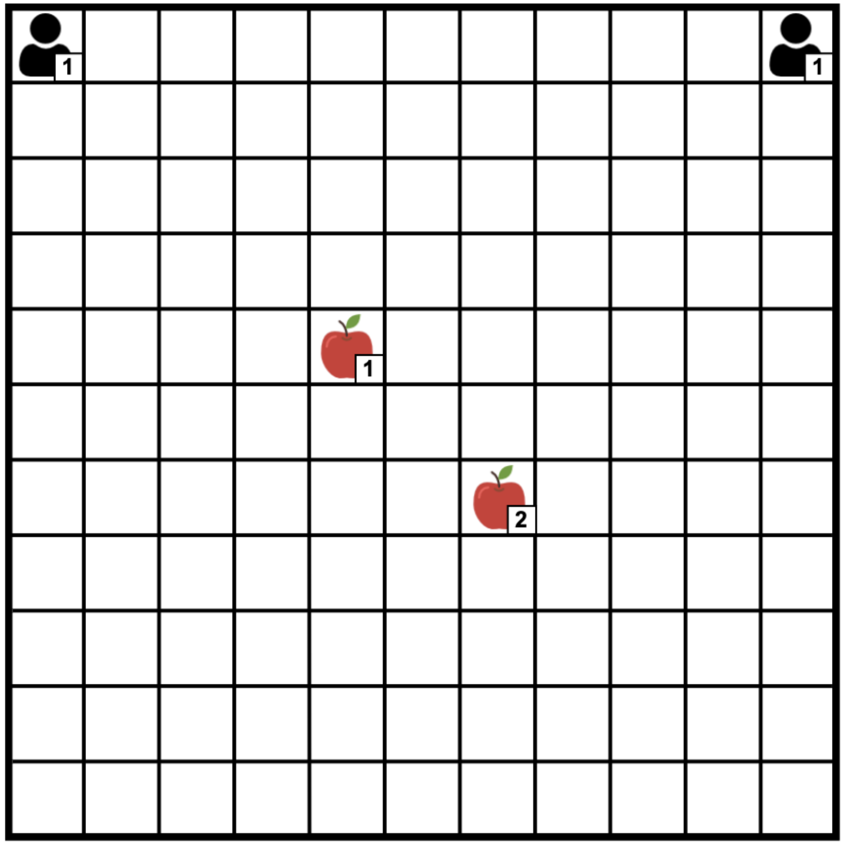
  $Q_i(s^t, a_i^t) \leftarrow Q_i(s^t, a_i^t) + \alpha \left[r_i^t + \gamma \max_{a_i'} Q_i(s^{t+1}, a_i') - Q_i(s^t, a_i^t) \right]$

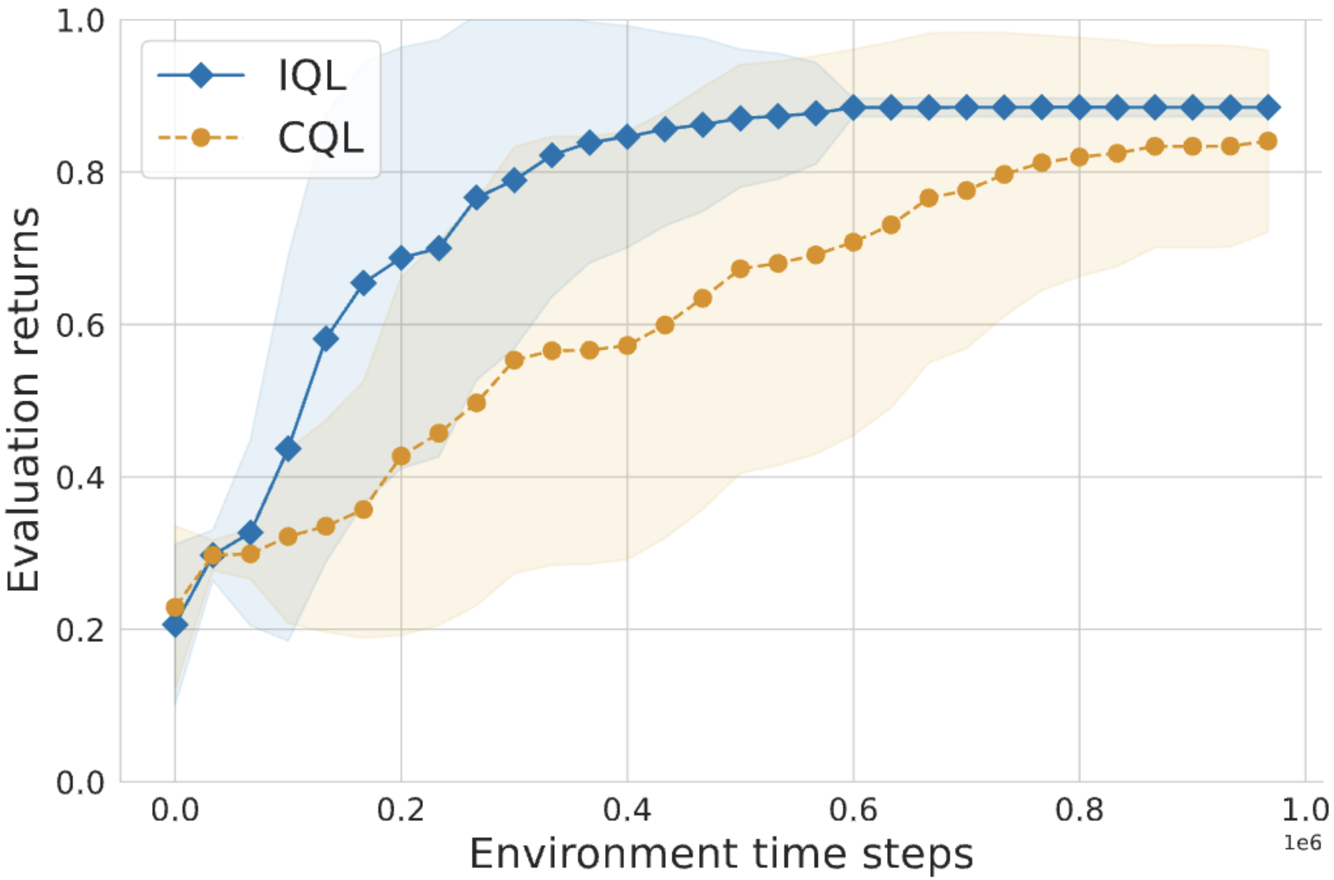


IQL doesn't have any information about the other agents.

It solely tries to model its own reward interactions in it's Q table.

CQL + IQL: Results





Albrecht (2024)

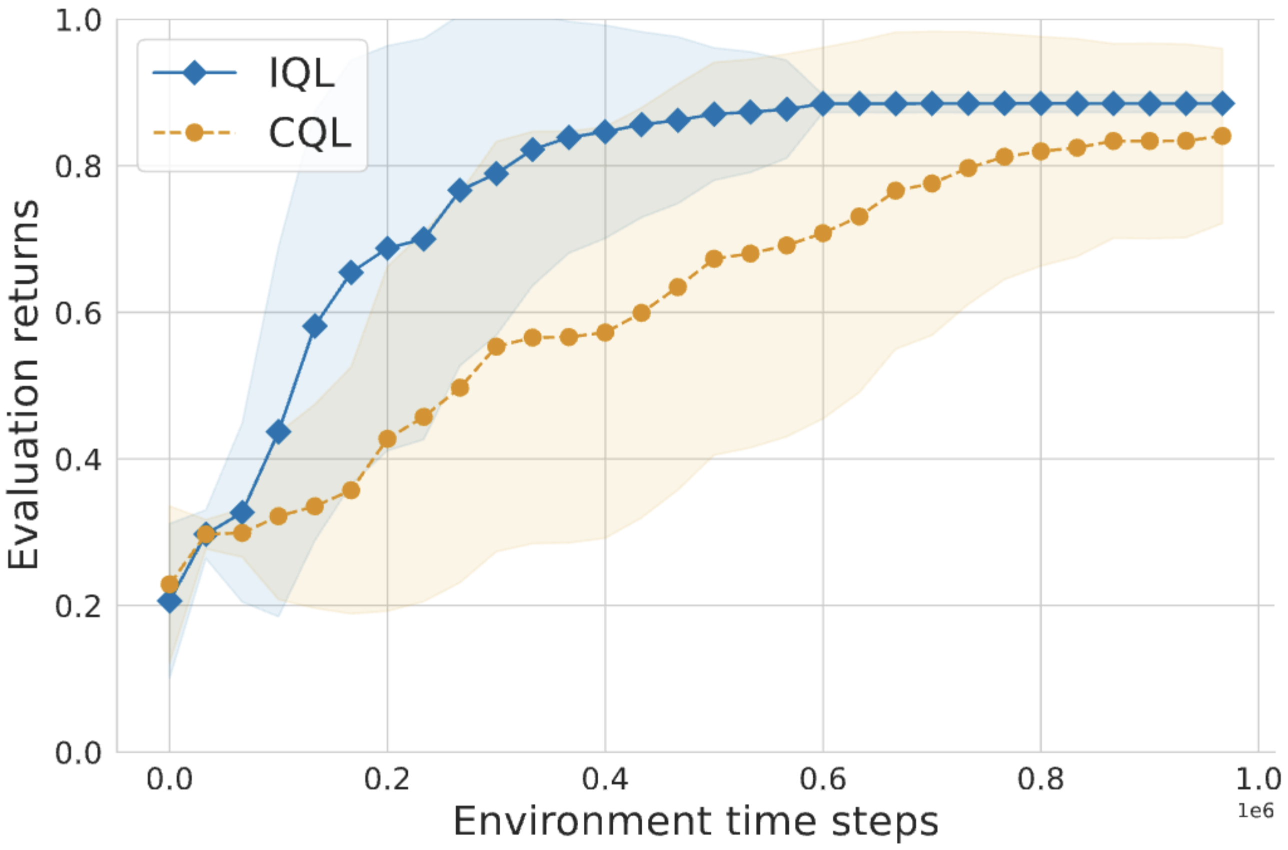
Quiz: Why does IQL learn faster than CQL?

Algorithm 4 Central Q-learning (CQL) for stochastic games

Initialize: $Q(s,a)=0$ for all $s \in S$ and $a \in A = A_1 \times \ldots \times A_n$
Repeat for every episode:
for $t = 0, 1, 2, \ldots$ **do**
  Observe current state s^t
  With probability ϵ: choose random joint action $a^t \in A$
  Otherwise: choose joint action $a^t \in \arg\max_a Q(s^t, a)$
  Apply joint action a^t, observe rewards $r_1^t, \ldots, r_n^t$ and next state s^{t+1}
  Transform $r_1^t, \ldots, r_n^t$ into scalar reward r^t
  $Q(s^t, a^t) \leftarrow Q(s^t, a^t) + \alpha \left[r^t + \gamma \max_{a'} Q(s^{t+1}, a') - Q(s^t, a^t) \right]$

Algorithm 5 Independent Q-learning (IQL) for stochastic games

// Algorithm controls agent i
Initialize: $Q_i(s, a_i) = 0$ for all $s \in S, a_i \in A_i$
Repeat for every episode:
for $t = 0, 1, 2, \ldots$ **do**
  Observe current state s^t
  With probability ϵ: choose random action $a_i^t \in A_i$
  Otherwise: choose action $a_i^t \in \arg\max_{a_i} Q_i(s^t, a_i)$
  (meanwhile, other agents $j \neq i$ choose their actions a_j^t)
  Observe own reward r_i^t and next state s^{t+1}
  $Q_i(s^t, a_i^t) \leftarrow Q_i(s^t, a_i^t) + \alpha \left[r_i^t + \gamma \max_{a_i'} Q_i(s^{t+1}, a_i') - Q_i(s^t, a_i^t) \right]$



Centralized Learning: Challenges

- Works great for low N! But exploration suffers exponentially as N grows.

Number of Agents in Apple Picking (N)	Size of Joint Action Space (|A|)
2	36
3	216
4	1296
5	7776

$$\prod_{i=0}^{N} |A_i| = 6^N$$

- Physical Limitations. In robotics/real world tasks we cannot assume that we have centralized control over all agents (or even if we do there are major communication/latency issues!)
- Requires the scalarization of reward signals, which is sometimes not possible.

Albrecht (2024)

Independent Learning

- It performs fairly compared to a lot of SOTA MARL algorithms!
- All single-agent algorithms have been independently applied to MARL.

Algorithm 17 Independent deep Q-networks

Initialize n value networks with random parameters $\theta_1, \ldots, \theta_n$
Initialize n target networks with parameters $\bar{\theta}_1 = \theta_1, \ldots, \bar{\theta}_n = \theta_n$
Initialize a replay buffer for each agent $D_1, D_2, \ldots, D_n$
for time step $t = 0, 1, 2, \ldots$ **do**
  Collect current observations $o_1^t, \ldots, o_n^t$
  for agent $i = 1, \ldots, n$ **do**
    With probability ϵ: choose random action a_i^t
    Otherwise: choose $a_i^t \in \arg\max_{a_i} Q(h_i^t, a_i; \theta_i)$
  Apply actions $(a_1^t, \ldots, a_n^t)$; collect rewards $r_1^t, \ldots, r_n^t$ and next observations $o_1^{t+1}, \ldots, o_n^{t+1}$
  for agent $i = 1, \ldots, n$ **do**
    Store transition $(h_i^t, a_i^t, r_i^t, h_i^{t+1})$ in replay buffers D_i
    Sample random mini-batch of B transitions $(h_i^k, a_i^k, r_i^k, h_i^{k+1})$ from D_i
    if s^{k+1} is terminal[2] **then**
      Targets $y_i^k \leftarrow r_i^k$
    else
      Targets $y_i^k \leftarrow r_i^k + \gamma \max_{a_i' \in A_i} Q(h_i^{k+1}, a_i'; \bar{\theta}_i)$
    Loss $\mathcal{L}(\theta_i) \leftarrow \frac{1}{B} \sum_{k=1}^{B} \left(y_i^k - Q(h_i^k, a_i^k; \theta_i) \right)^2$
    Update parameters θ_i by minimizing the loss $\mathcal{L}(\theta_i)$
    In a set interval, update target network parameters $\bar{\theta}_i$

Algorithm 18 Independent REINFORCE

Initialize n policy networks with random parameters $\phi_1, \ldots, \phi_n$
Repeat for every episode:
for time step $t = 0, 1, 2, \ldots, T-1$ **do**
  Collect current observations $o_1^t, \ldots, o_n^t$
  for agent $i = 1, \ldots, n$ **do**
    Sample actions a_i^t from $\pi(\cdot \mid h_i^t; \phi_i)$
  Apply actions $(a_1^t, \ldots, a_n^t)$; collect rewards $r_1^t, \ldots, r_n^t$ and next observations $o_1^{t+1}, \ldots, o_n^{t+1}$
for agent $i = 1, \ldots, n$ **do**
  Loss $\mathcal{L}(\phi_i) \leftarrow -\frac{1}{T} \sum_{t=0}^{T-1} \left(\sum_{\tau=t}^{T-1} \gamma^{\tau-t} r_i^\tau \right) \log \pi(a_i^t \mid h_i^t; \phi_i)$
  Update parameters ϕ_i by minimizing the loss $\mathcal{L}(\phi_i)$

Algorithm 19 Independent A2C with synchronous environments

Initialize n actor networks with random parameters $\phi_1, \ldots, \phi_n$
Initialize n critic networks with random parameters $\theta_1, \ldots, \theta_n$
Initialize K parallel environments
for time step $t = 0 \ldots$ **do**
  Batch of observations for each agent and environment: $\begin{bmatrix} o_1^{t,1} \ldots o_1^{t,K} \\ \ddots \\ o_n^{t,1} \ldots o_n^{t,K} \end{bmatrix}$
  Sample actions $\begin{bmatrix} a_1^{t,1} \ldots a_1^{t,K} \\ \ddots \\ a_n^{t,1} \ldots a_n^{t,K} \end{bmatrix} \sim \pi(\cdot \mid h_1^t; \phi_1), \ldots, \pi(\cdot \mid h_n^t; \phi_n)$
  Apply actions; collect rewards $\begin{bmatrix} r_1^{t,1} \ldots r_1^{t,K} \\ \ddots \\ r_n^{t,1} \ldots r_n^{t,K} \end{bmatrix}$ and observations $\begin{bmatrix} o_1^{t+1,1} \ldots o_1^{t+1,K} \\ \ddots \\ o_n^{t+1,1} \ldots o_n^{t+1,K} \end{bmatrix}$
  for agent $i = 1, \ldots, n$ **do**
    if $s^{t+1,k}$ is terminal **then**
      Advantage $Adv(h_i^{t,k}, a_i^{t,k}) \leftarrow r_i^{t,k} - V(h_i^{t,k}; \theta_i)$
      Critic target $y_i^{t,k} \leftarrow r_i^{t,k}$
    else
      Advantage $Adv(h_i^{t,k}, a_i^{t,k}) \leftarrow r_i^{t,k} + \gamma V(h_i^{t+1,k}; \theta_i) - V(h_i^{t,k}; \theta_i)$
      Critic target $y_i^{t,k} \leftarrow r_i^{t,k} + \gamma V(h_i^{t+1,k}; \theta_i)$
    Actor loss $\mathcal{L}(\phi_i) \leftarrow \frac{1}{K} \sum_{k=1}^{K} Adv(h_i^{t,k}, a_i^{t,k}) \log \pi(a_i^{t,k} \mid h_i^{t,k}; \phi_i)$
    Critic loss $\mathcal{L}(\theta_i) \leftarrow \frac{1}{K} \sum_{k=1}^{K} \left(y_i^{t,k} - V(h_i^{t,k}; \theta_i) \right)^2$
    Update parameters ϕ_i by minimizing the actor loss $\mathcal{L}(\phi_i)$
    Update parameters θ_i by minimizing the critic loss $\mathcal{L}(\theta_i)$

Albrecht (2024)

Independent Learning: Challenges

- Much better scaling! Can be deployed easily on physical systems!

- Suffers from *non-stationary* environment dynamics.

Single Agent Dynamics

$$P(s' \mid s, a) = T(s' \mid s, a)$$

Probability of subsequent states stays constant over time.

Multi Agent Dynamics

$$P_i(s' \mid s, a_i) = \sum_{a_{-i} \in A_{-i}} T(s' \mid s, \langle a_i, a_{-i} \rangle) \left[\prod_{j \neq i} \pi_j(a_j \mid s^t) \right]$$

Changes to environment state are not guaranteed to be the result of my actions

- Depending on reward formulation, it suffers from poor *credit assignment*.

Albrecht (2024)

The Credit Assignment Problem

Temporal Credit Assignment — *Which of my **past actions** were responsible for my earned reward?*

Multi-Agent Credit Assignment — *Which **agent** was responsible for my earned reward?*

Recall that each agent its own reward signal

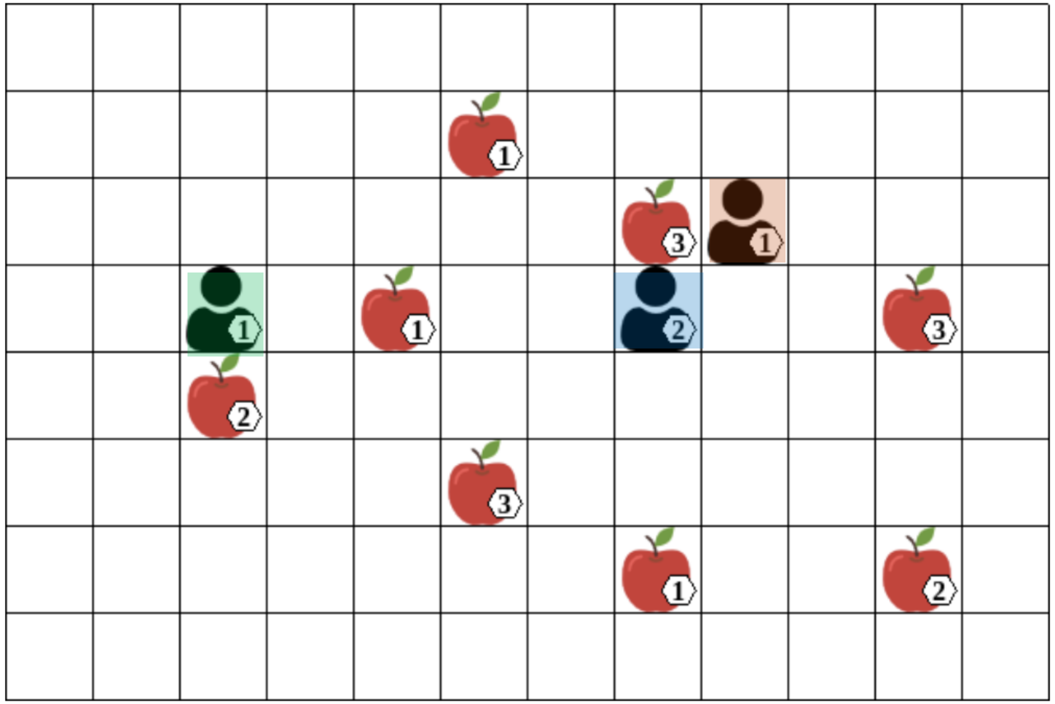
$$r_0, r_1, \dots, r_N$$

What happens when we shift to a common reward?
(i.e. all agents receive the same reward signal at time t)

$$r_0 = r_1 = \cdots = r_N$$

Now the green agent gets some reward for the apple that the orange and blue agents did!

Tasks are often formulated as common reward!



Middle Ground? CTDE Methods!

Can we create MARL algorithms that can still be deployed on a decentralized system while overcoming obstacles in scaling, stationarity, and credit assignment?

Independent Learning: Decentralized Training, Decentralized Execution

Central Learning: Centralized Training, Centralized Execution

Centralized Training, Decentralized Execution.

Utilize *privileged centralized information* at training time to determine how agent's actions affect the rewards of the system and share experiences between agents.

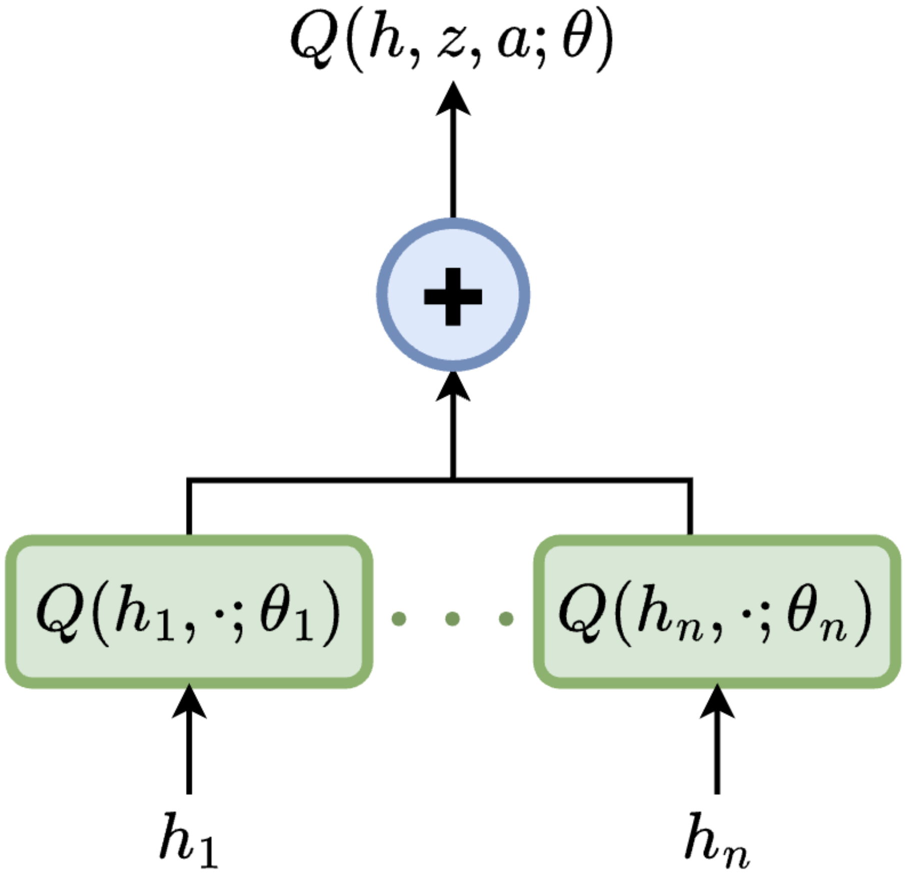
At execution time, ensure that all each policy can independently execute, with zero dependencies on centralized information.

Albrecht (2024)

Value Decomposition Networks

Algorithm 21 Value decomposition networks (VDN)

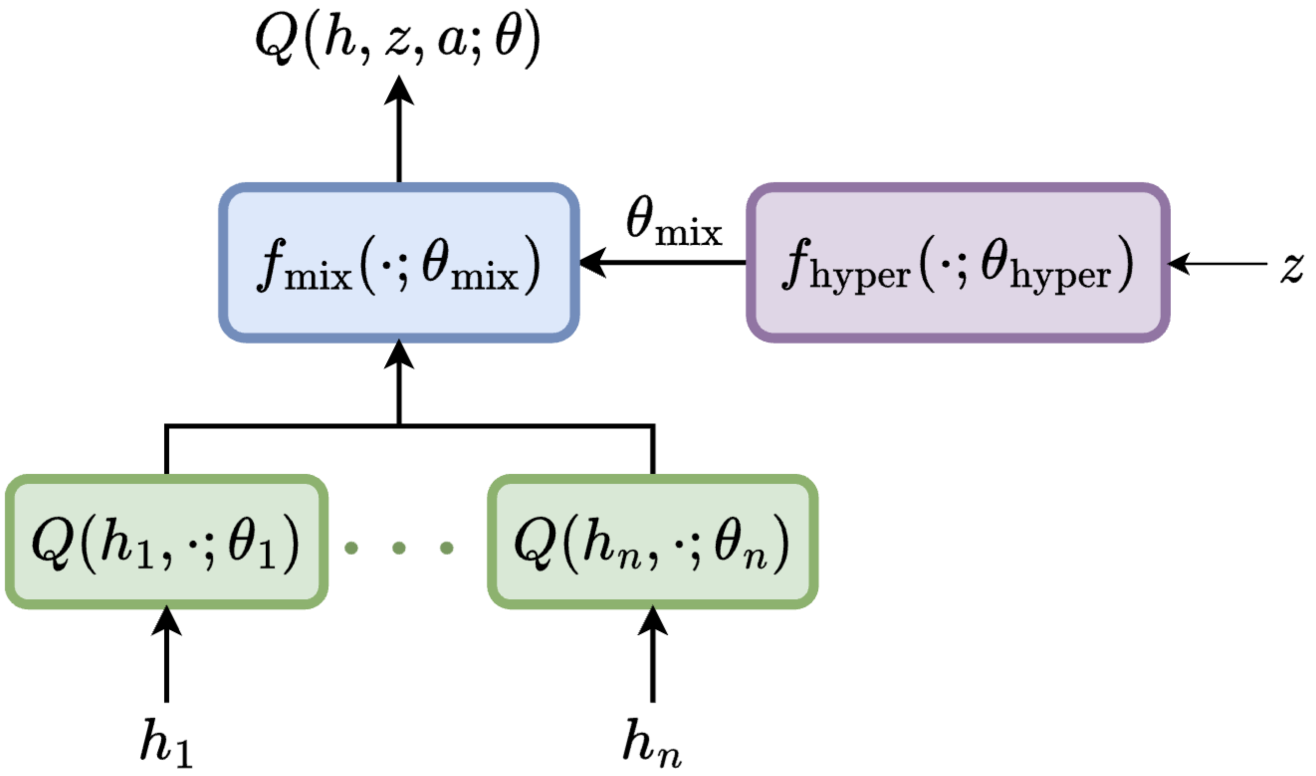
Initialize n utility networks with random parameters $\theta_1, \ldots, \theta_n$
Initialize n target networks with parameters $\bar{\theta}_1 = \theta_1, \ldots, \bar{\theta}_n = \theta_n$
Initialize a shared replay buffer D
for time step $t = 0, 1, 2, \ldots$ **do**
  Collect current observations $o_1^t, \ldots, o_n^t$
  for agent $i = 1, \ldots, n$ **do**
    With probability ϵ: choose random action a_i^t
    Otherwise: choose $a_i^t \in \arg\max_{a_i} Q(h_i^t, a_i; \theta_i)$
  Apply actions; collect shared reward r^t and next observations $o_1^{t+1}, \ldots, o_n^{t+1}$
  Store transition (h^t, a^t, r^t, h^{t+1}) in shared replay buffer D
  Sample mini-batch of B transitions (h^k, a^k, r^k, h^{k+1}) from D
  if s^{k+1} is terminal **then**
    Targets $y^k \leftarrow r^k$
  else
    Targets $y^k \leftarrow r^k + \gamma \sum_{i \in I} \max_{a_i' \in A_i} Q(h_i^{k+1}, a_i'; \bar{\theta}_i)$
  Loss $\mathcal{L}(\theta) \leftarrow \frac{1}{B} \sum_{k=1}^{B} \left(y^k - \sum_{i \in I} Q(h_i^k, a_i^k; \theta_i) \right)^2$
  Update parameters θ by minimizing the loss $\mathcal{L}(\theta)$
  In a set interval, update target network parameters $\bar{\theta}_i$ for each agent i



QMIX: Non-Linear VDN

Algorithm 22 QMIX

Initialize n utility networks with random parameters $\theta_1, \ldots, \theta_n$
Initialize n target networks with parameters $\bar{\theta}_1 = \theta_1, \ldots, \bar{\theta}_n = \theta_n$
Initialize hypernetwork with random parameters θ_{hyper}
Initialize a shared replay buffer D
for time step $t = 0, 1, 2, \ldots$ **do**
  Collect current centralized information z^t and observations $o_1^t, \ldots, o_n^t$
  for agent $i = 1, \ldots, n$ **do**
    With probability ϵ: choose random action a_i^t
    Otherwise: choose $a_i^t \in \arg\max_{a_i} Q(h_i^t, a_i; \theta_i)$
  Apply actions; collect shared reward r^t, next centralized information z^{t+1} and observations $o_1^{t+1}, \ldots, o_n^{t+1}$
  Store transition $(h^t, z^t, a^t, r^t, h^{t+1}, z^{t+1})$ in shared replay buffer D
  Sample mini-batch of B transitions $(h^k, z^k, a^k, r^k, h^{k+1}, z^{k+1})$ from D
  if s^{k+1} is terminal **then**
    Targets $y^k \leftarrow r^k$
  else
    Mixing parameters $\theta_{\text{mix}}^{k+1} \leftarrow f_{\text{hyper}}(z^{k+1}; \theta_{\text{hyper}})$
    Targets $$y^k \leftarrow r^k + \gamma f_{\text{mix}} \left(\begin{matrix} \max_{a_1'} Q(h_1^{k+1}, a_1'; \bar{\theta}_1), \\ \ddots \\ \max_{a_n'} Q(h_n^{k+1}, a_n'; \bar{\theta}_n) \end{matrix} ; \theta_{\text{mix}}^{k+1} \right)$$
  Mixing parameters $\theta_{\text{mix}}^k \leftarrow f_{\text{hyper}}(z^k; \theta_{\text{hyper}})$
  Value estimates $Q(h^k, z^k, a^k; \theta) \leftarrow f_{\text{mix}} \left(Q(h_1^k, a_1^k; \theta_1), \ldots, Q(h_n^k, a_n^k; \theta_n); \theta_{\text{mix}}^k \right)$
  Loss $\mathcal{L}(\theta) \leftarrow \frac{1}{B} \sum_{k=1}^{B} \left(y^k - Q(h^k, z^k, a^k; \theta) \right)^2$
  Update parameters θ by minimizing the loss $\mathcal{L}(\theta)$
  In a set interval, update target network parameters $\bar{\theta}_i$ for each agent i



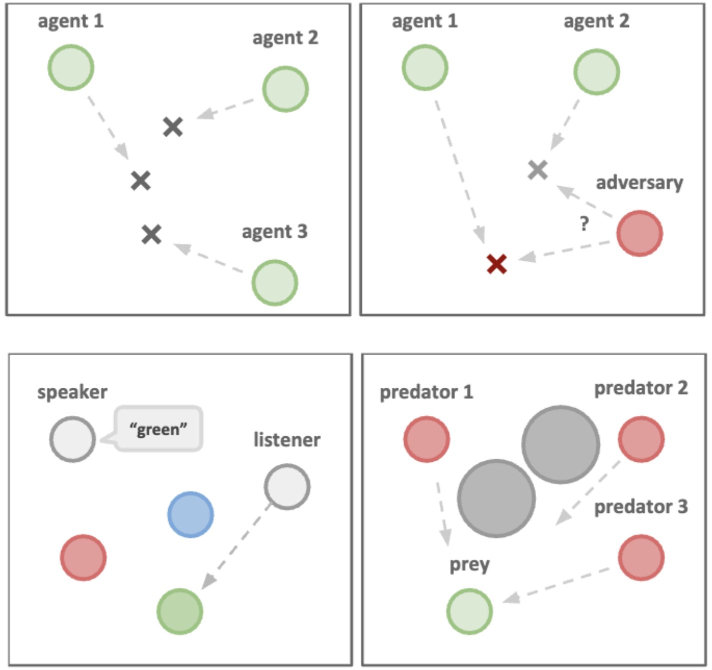
Other CTDE Extensions

Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments

Ryan Lowe* McGill University OpenAI

Yi Wu* UC Berkeley

Aviv Tamar UC Berkeley



The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games

Chao Yu[1♯*], Akash Velu[2♮*], Eugene Vinitsky[2♭], Jiaxuan Gao[1], Yu Wang[1♭], Alexandre Bayen[2], Yi Wu[13♭]

[1] Tsinghua University [2] University of California, Berkeley [3] Shanghai Qi Zhi Institute

♯zoeyuchao@gmail.com, ♮akashvelu@berkeley.edu

Current Benchmarks and Challenges

Where to Deploy?

What Challenges require Multi-Agent Learning right now?

How does deployment difficulty scale?

Current Benchmarks and Challenges

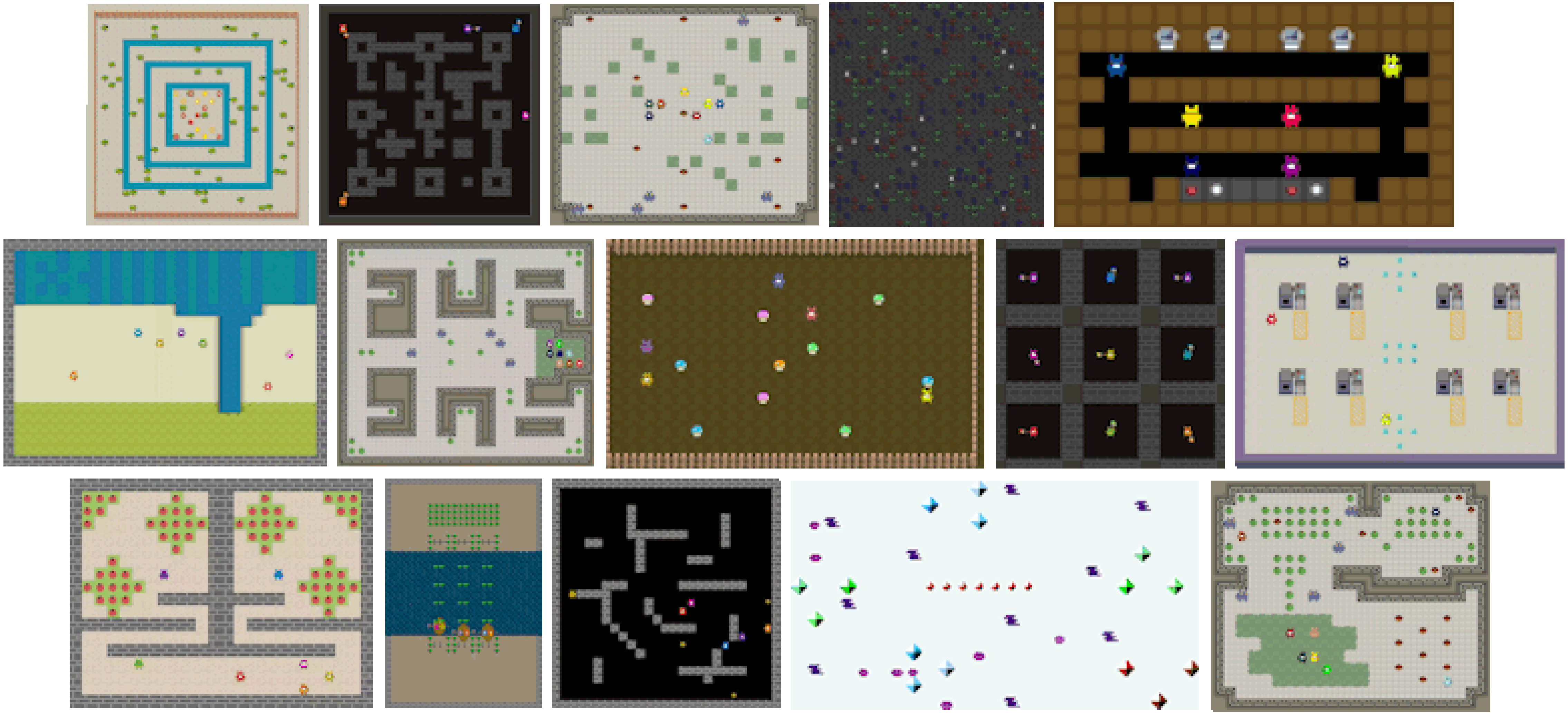
Article | Published: 30 October 2019

Grandmaster level in StarCraft II using multi-agent reinforcement learning

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, ... David Silver

AlphaStar (2019)

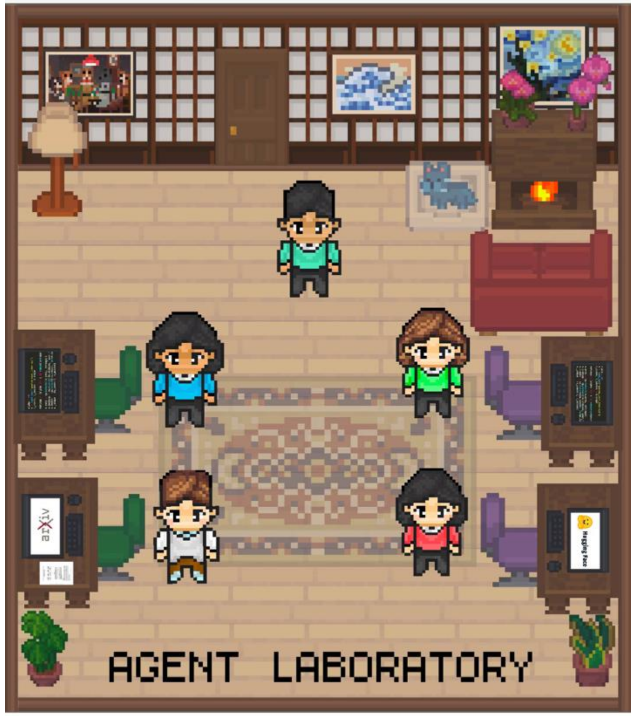
Current Benchmarks and Challenges

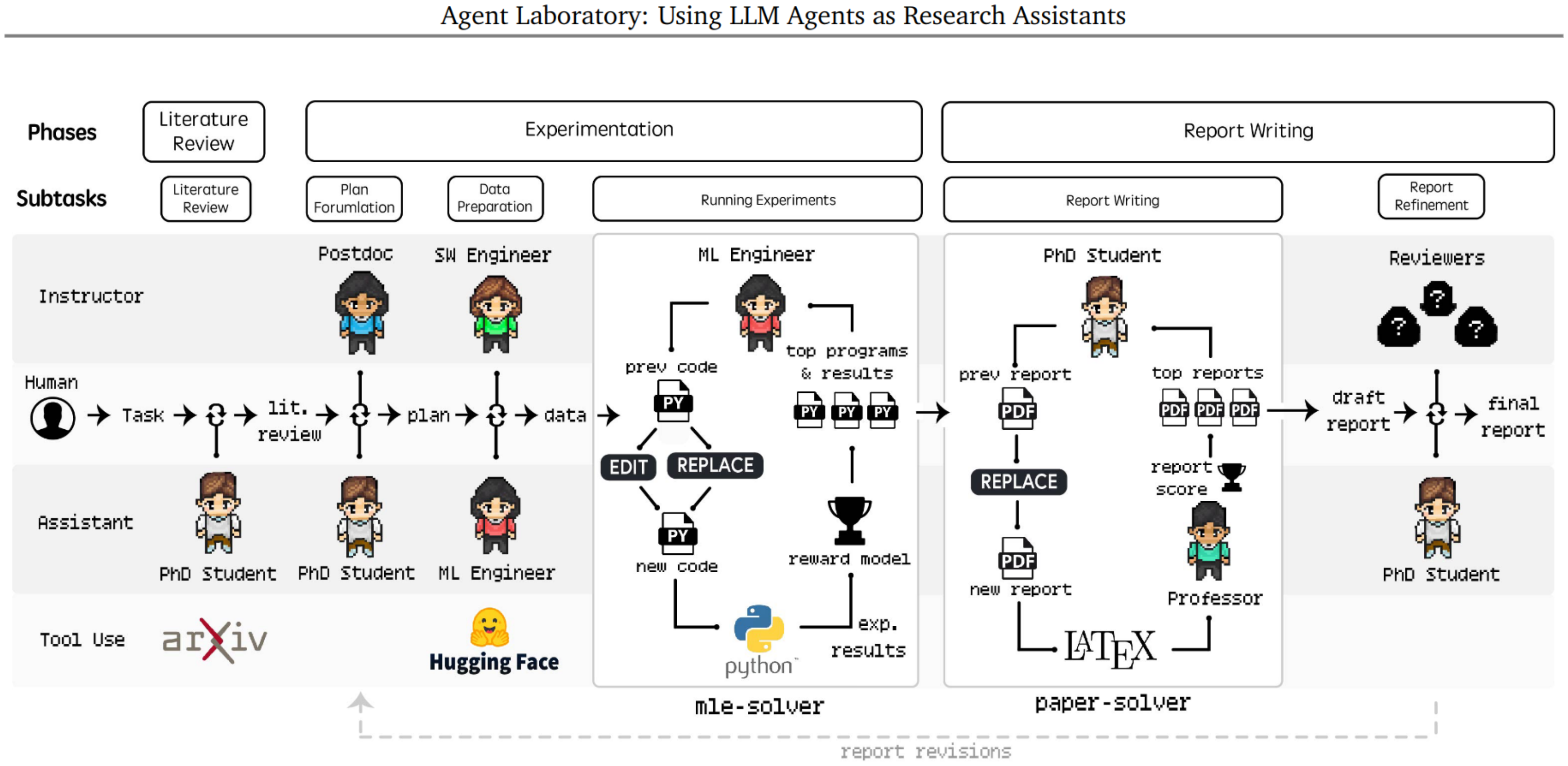


Melting Pot (2024)

Current Explorations: LLMs

Agent Laboratory: Using LLM Agents as Research Assistants



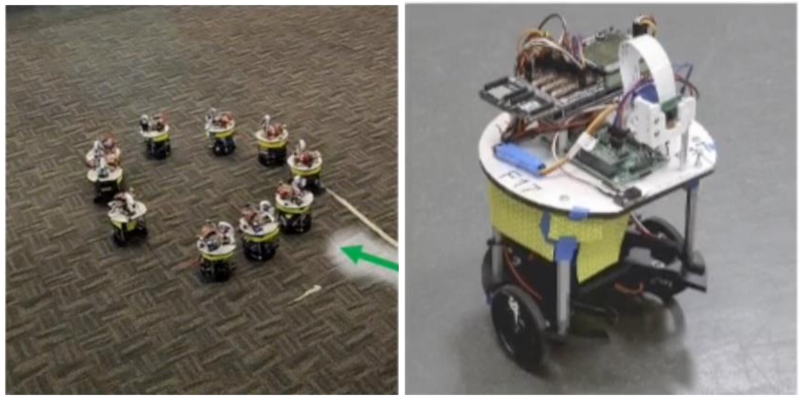


Schmidgall (2025)

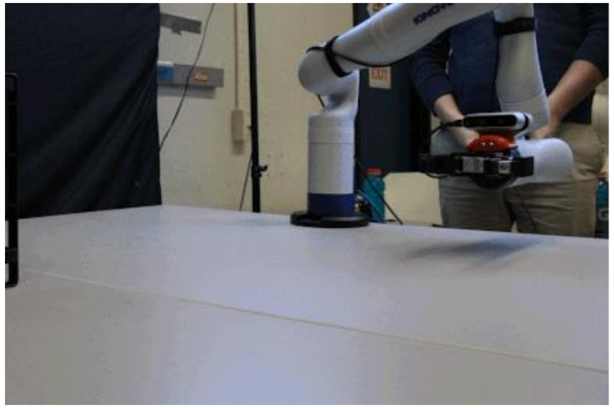
Current Explorations: LLMs

Park (2023)

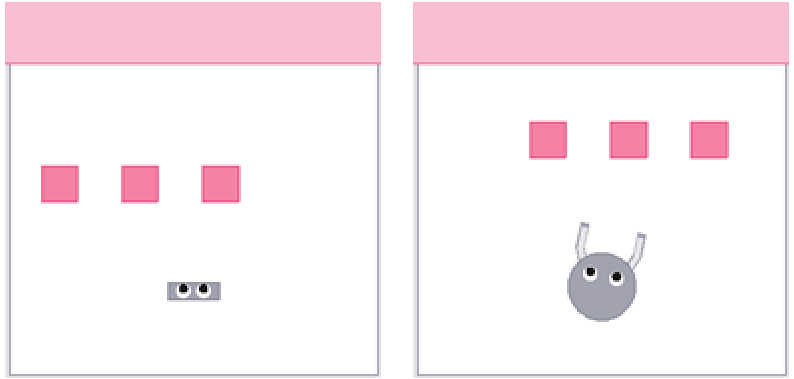
What does my PhD Research Look Like?



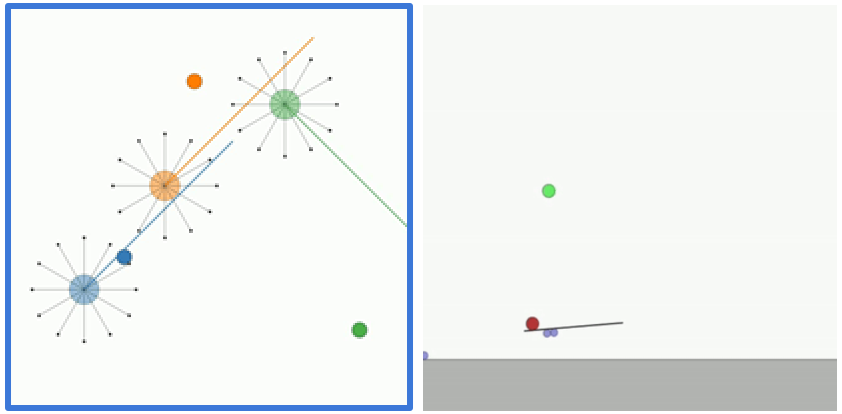
Robot Swarm Learning + Sim2Real
[ICONS'24], [CoRR'24], [AAMAS'25]



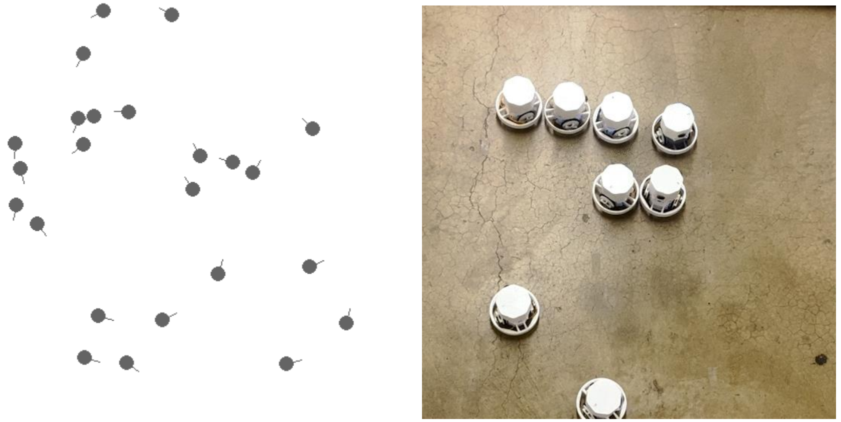
Assistive Robotics
[HRI'25], [SciData'24]



X-Embodiment Learning
[RLJ'24]



Multi-Agent Reinforcement Learning
[in preparation]



Repr. Learning + Behavior Discovery
[GECCO'23], [MRS'23] [AAMAS'25]

Multi-Agent Learning is still a young field and there's lots of exciting work to do!

Let's chat!

c.mattson@utah.edu