Multi-Agent Interactions and Social Dilemmas

Advantage Alignment Algorithms by Juan Duque, Milad Aghajohari, Tim Cooijmans, Razvan Ciuca, Tianyu Zhang, Gauthier Gidel and Aaron Courville

Aaron Courville

Mila,
University of Montreal

April 17th, 2025
Safety-Guaranteed LLMs

What's next … LLMs ➔ Agents

Agents make decisions and actions that can affect their environment

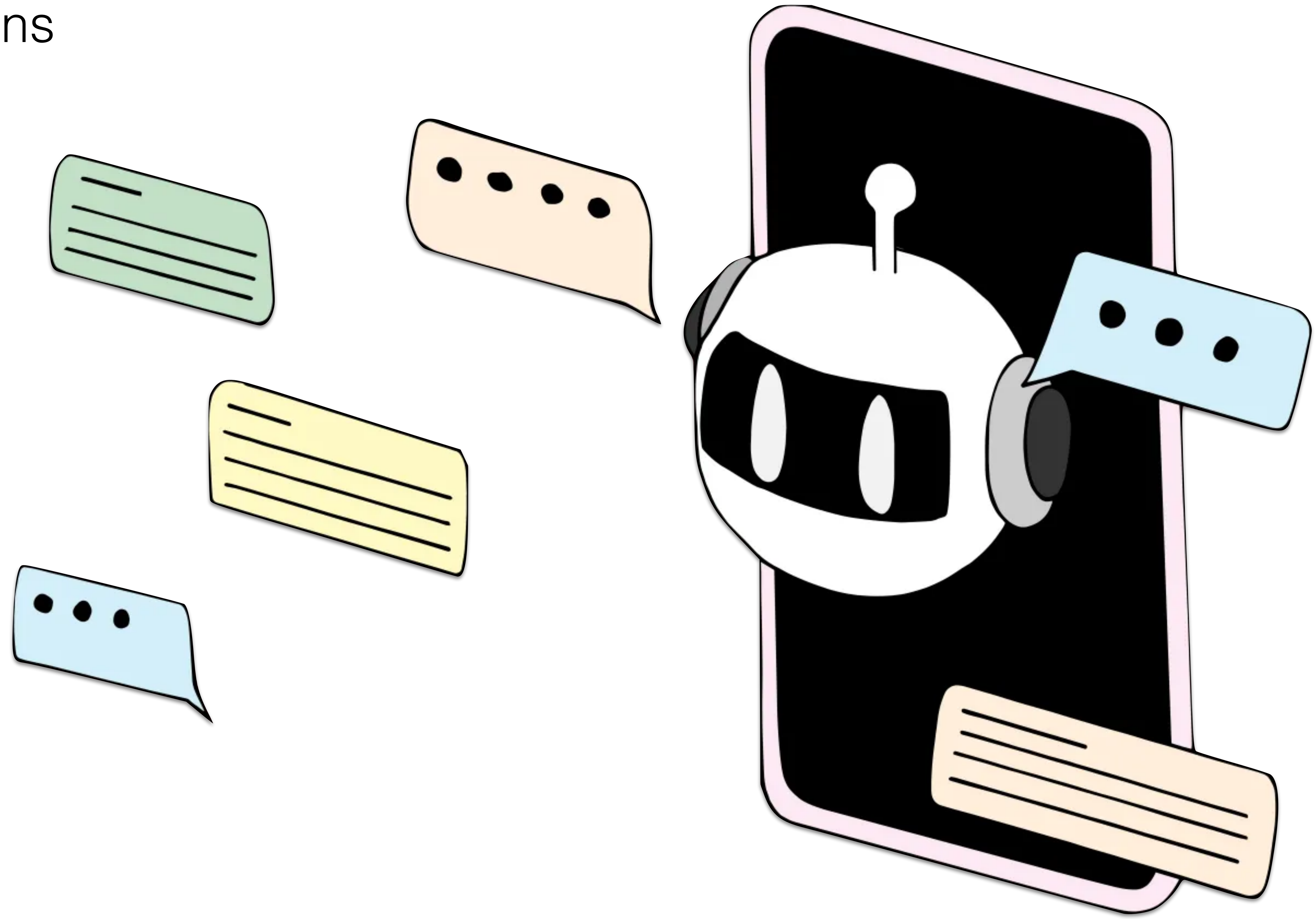
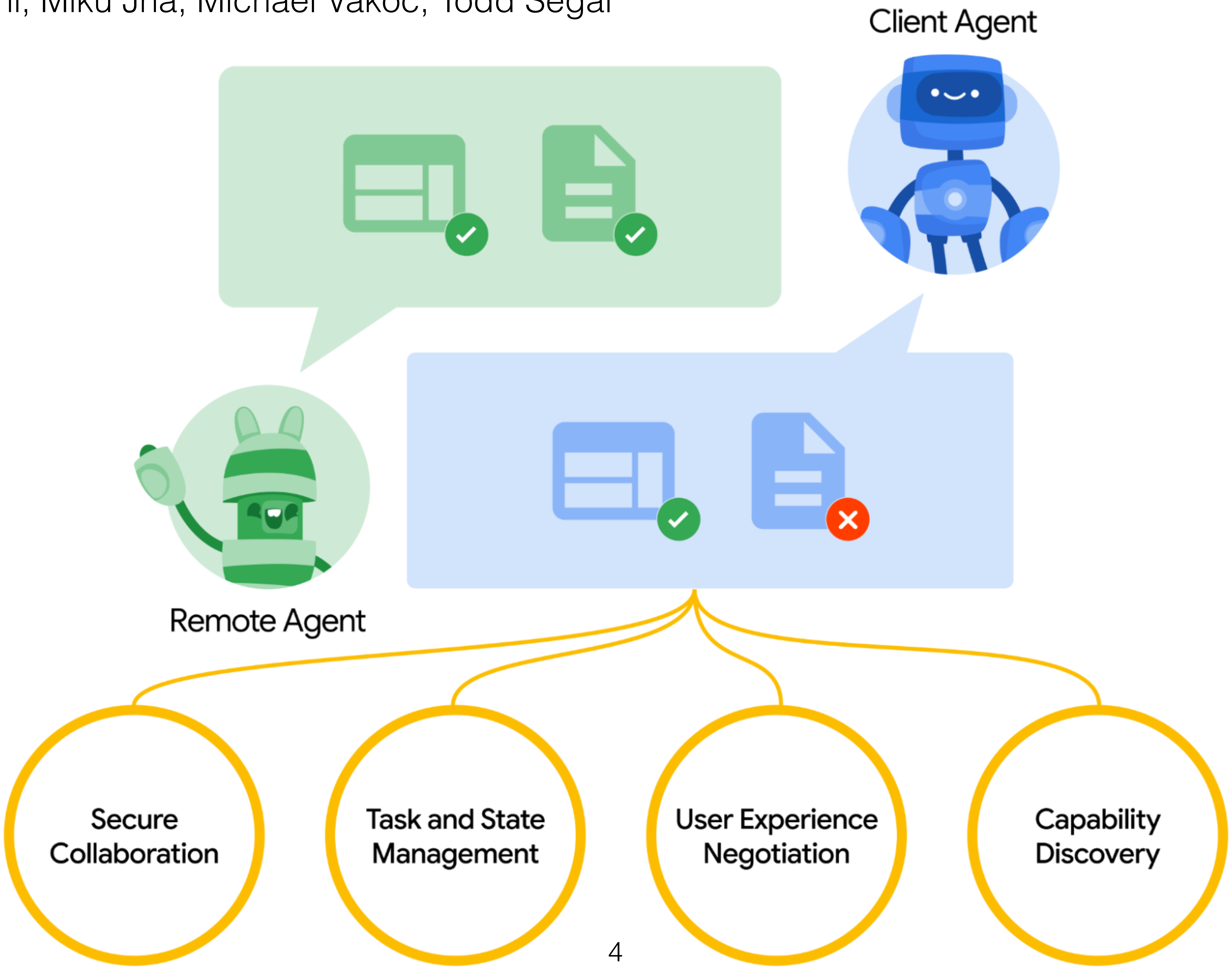


Image Credit: Kelsey Piper. *AI "agents" could do real work in the real world. That might not be a good thing*. 2024 (Malorny/Getty Images)

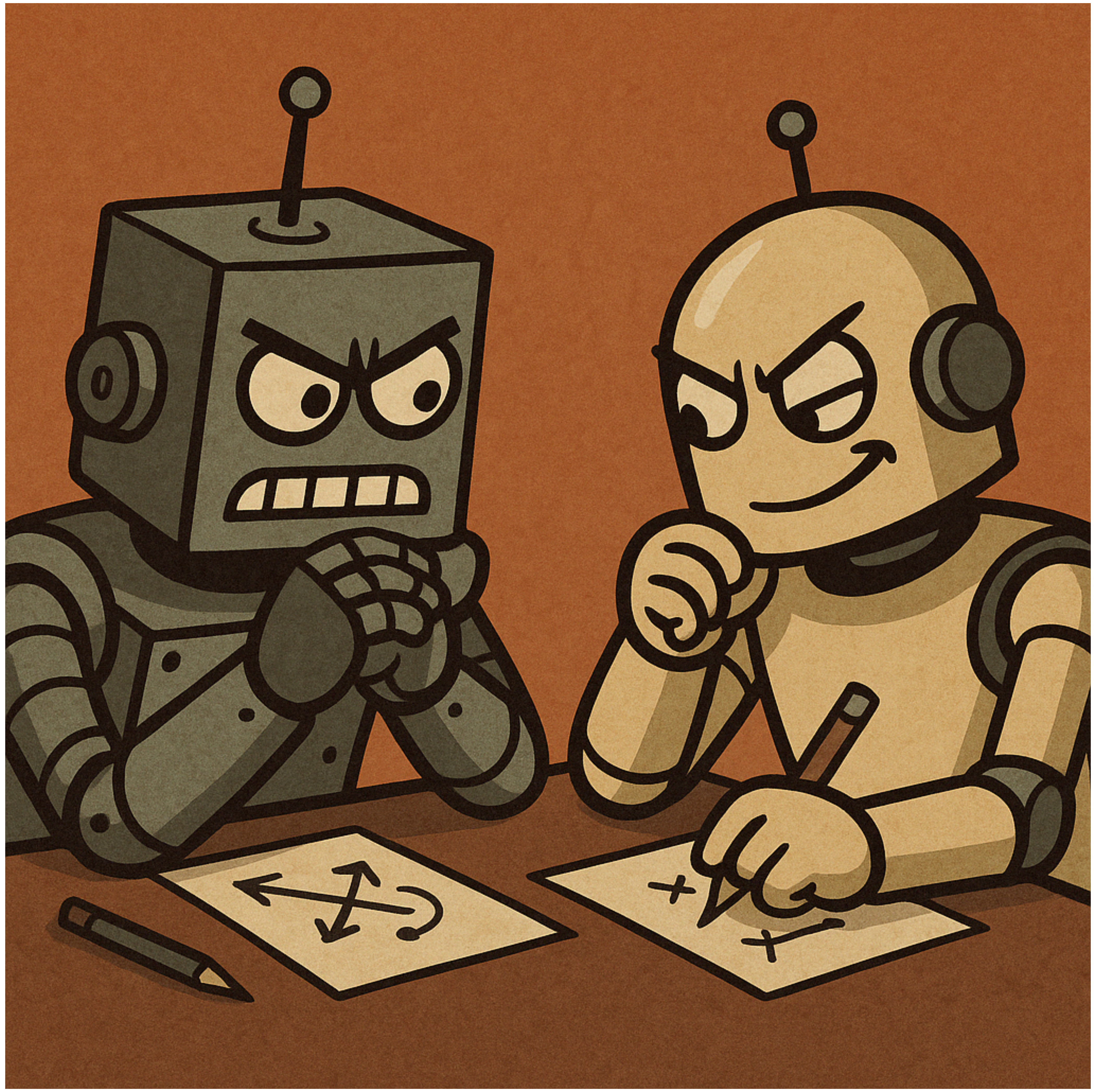
Image Credit: VentureBeat made with Bing Image Creator

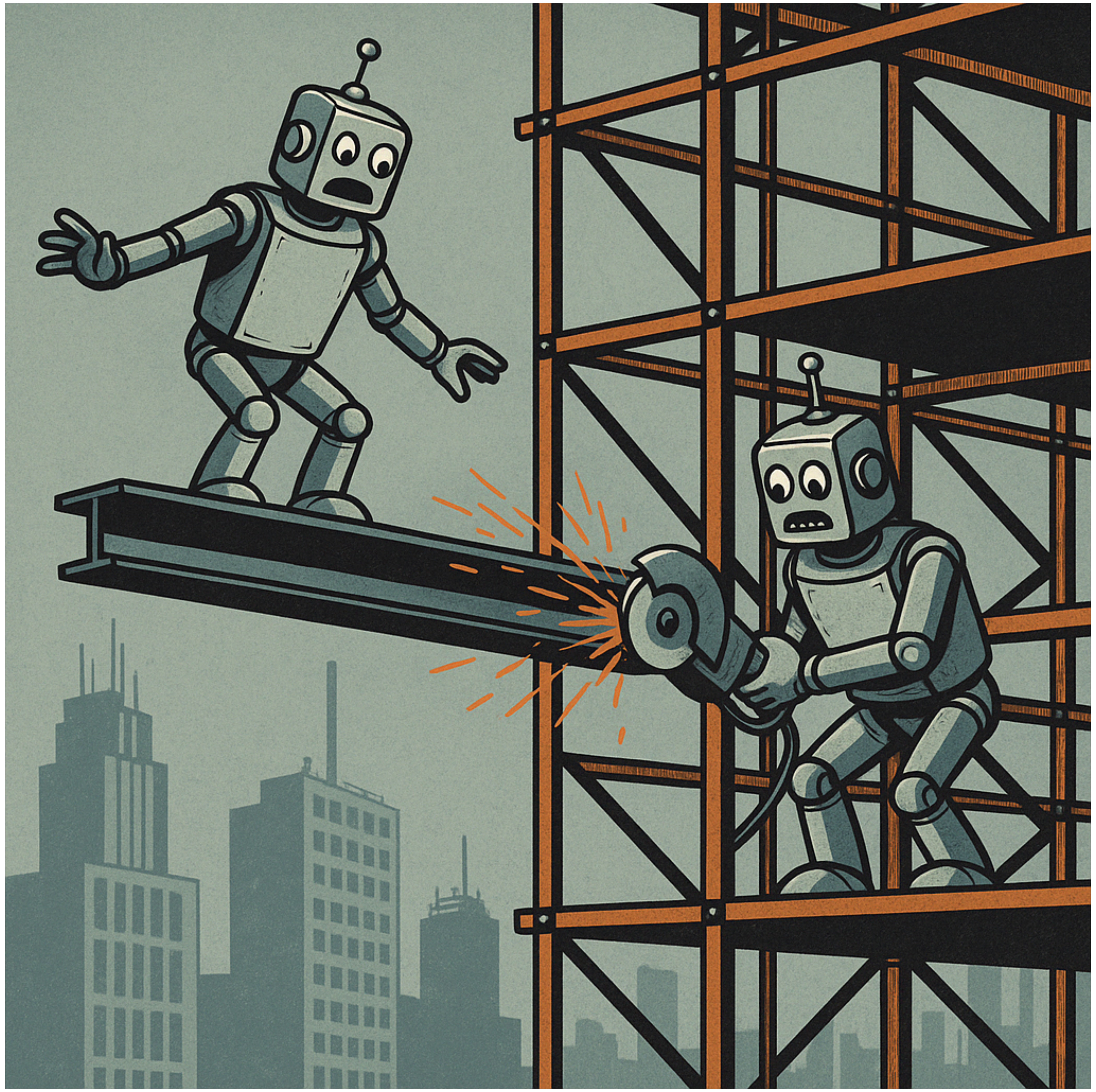
What follows … Agents talking to agents

APR 09, 2025, Rao Surapaneni, Miku Jha, Michael Vakoc, Todd Segal



Multi-Agent Systems





2010 Flash Crash

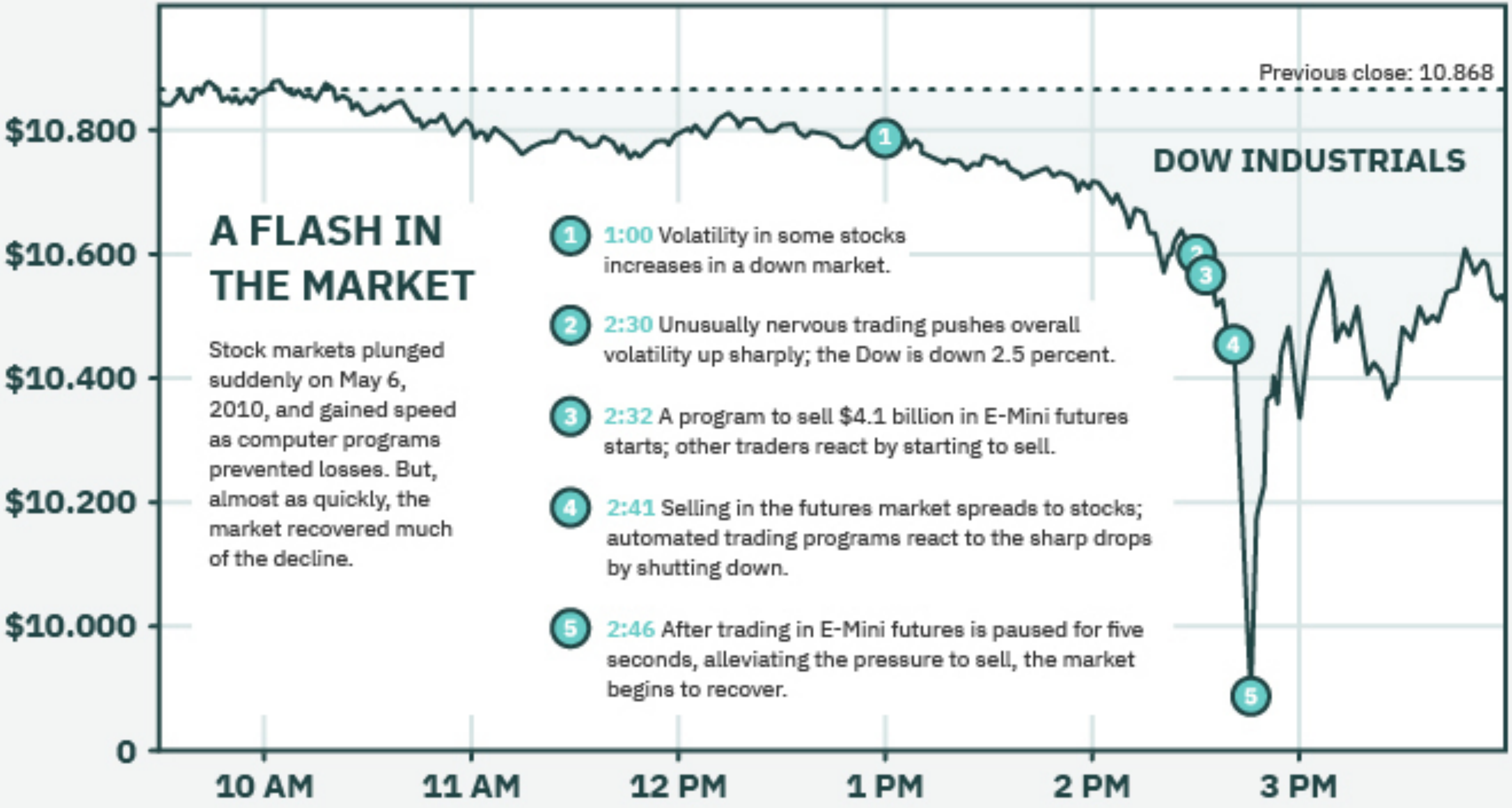


Image credit: Tim Fries (2025) Flash Crashes: What They Are, How They Work

Failure modes in multi-agent systems

1. Miscoordination
    - Agents fail to cooperate despite having the same goal.

2. Conflict
    - Agents with different goals fail to cooperate.
    - Social Dilemmas

3. Collusion
    - Competitive settings where we do not want agents cooperating.

Failure modes in multi-agent systems

1. Miscoordination
    - Agents fail to cooperate despite having the same goal.
2. Conflict
    - Agents with different goals fail to cooperate.
    - Social Dilemmas
3. Collusion
    - Competitive settings where we do not want agents cooperating.

Multi-Agent Risks from Advanced AI

Lewis Hammond
Alan Chan
Jesse Clifton
Jason Hoelscher-Obermaier
Akbir Khan
Euan McLean
Chandler Smith
Wolfram Barfuss
Jakob Foerster
Tomáš Gavenčiak
The Anh Han
Edward Hughes
Vojtěch Kovařík
Jan Kulveit
Joel Z. Leibo
Caspar Oesterheld
Christian Schroeder de Witt
Nisarg Shah
Michael Wellman
Paolo Bova
Theodor Cimpeanu
Carson Ezell
Quentin Feuillade-Montixi
Matija Franklin
Esben Kran
Igor Krawczuk
Max Lamparth
Niklas Lauffer
Alexander Meinke
Sumeet Motwani
Anka Reuel
Vincent Conitzer
Michael Dennis
Iason Gabriel
Adam Gleave
Gillian Hadfield
Nika Haghtalab
Atoosa Kasirzadeh
Sébastien Krier
Kate Larson
Joel Lehman
David C. Parkes
Georgios Piliouras
Iyad Rahwan

Reinforcement Learning

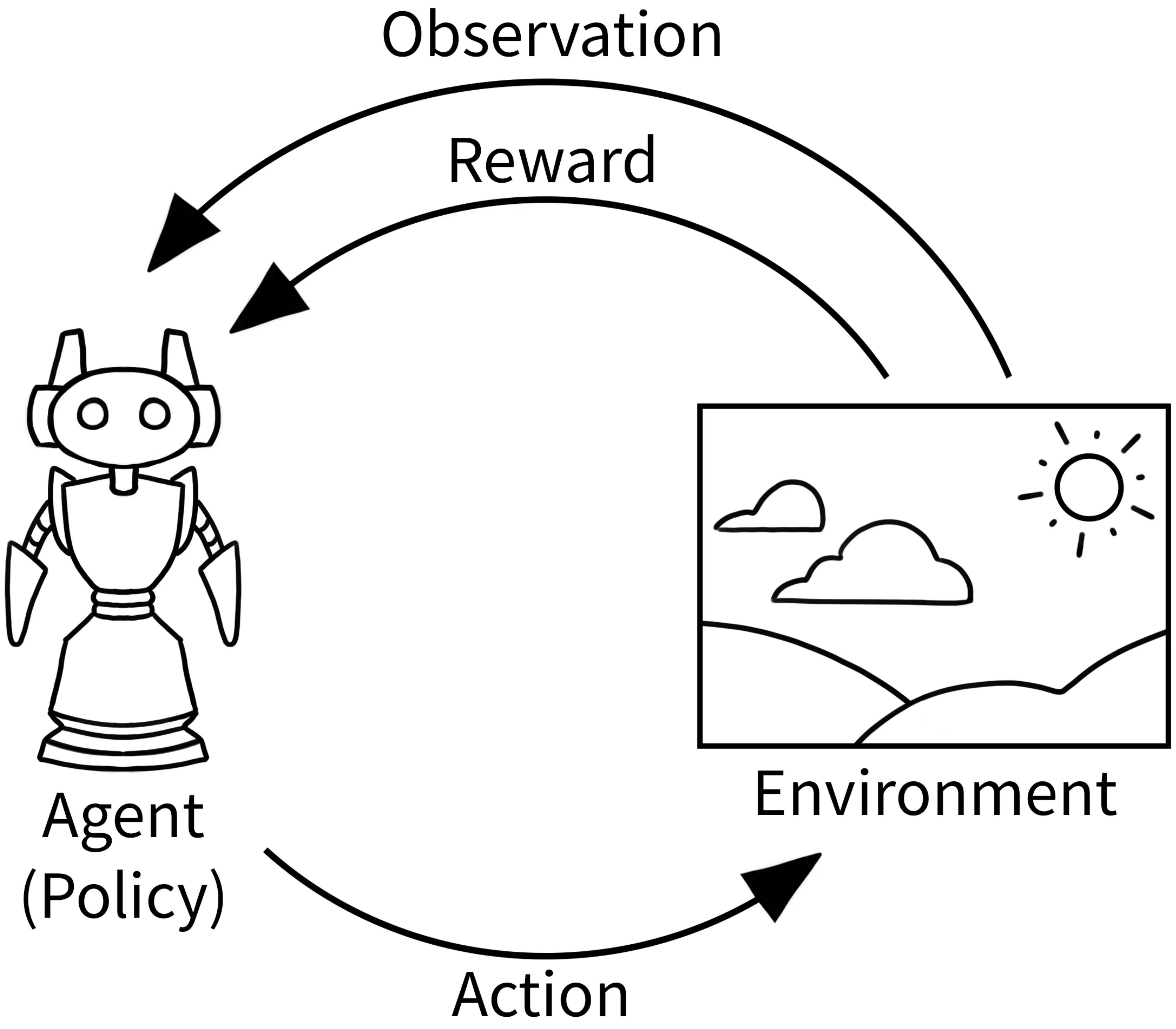


Image Credit: J K Terry, Multi-Agent Deep Reinforcement Learning in 13 Lines of Code Using PettingZoo: A tutorial on multi-agent deep reinforcement learning for beginners, Feb. 2021

Multi-Agent Reinforcement Learning

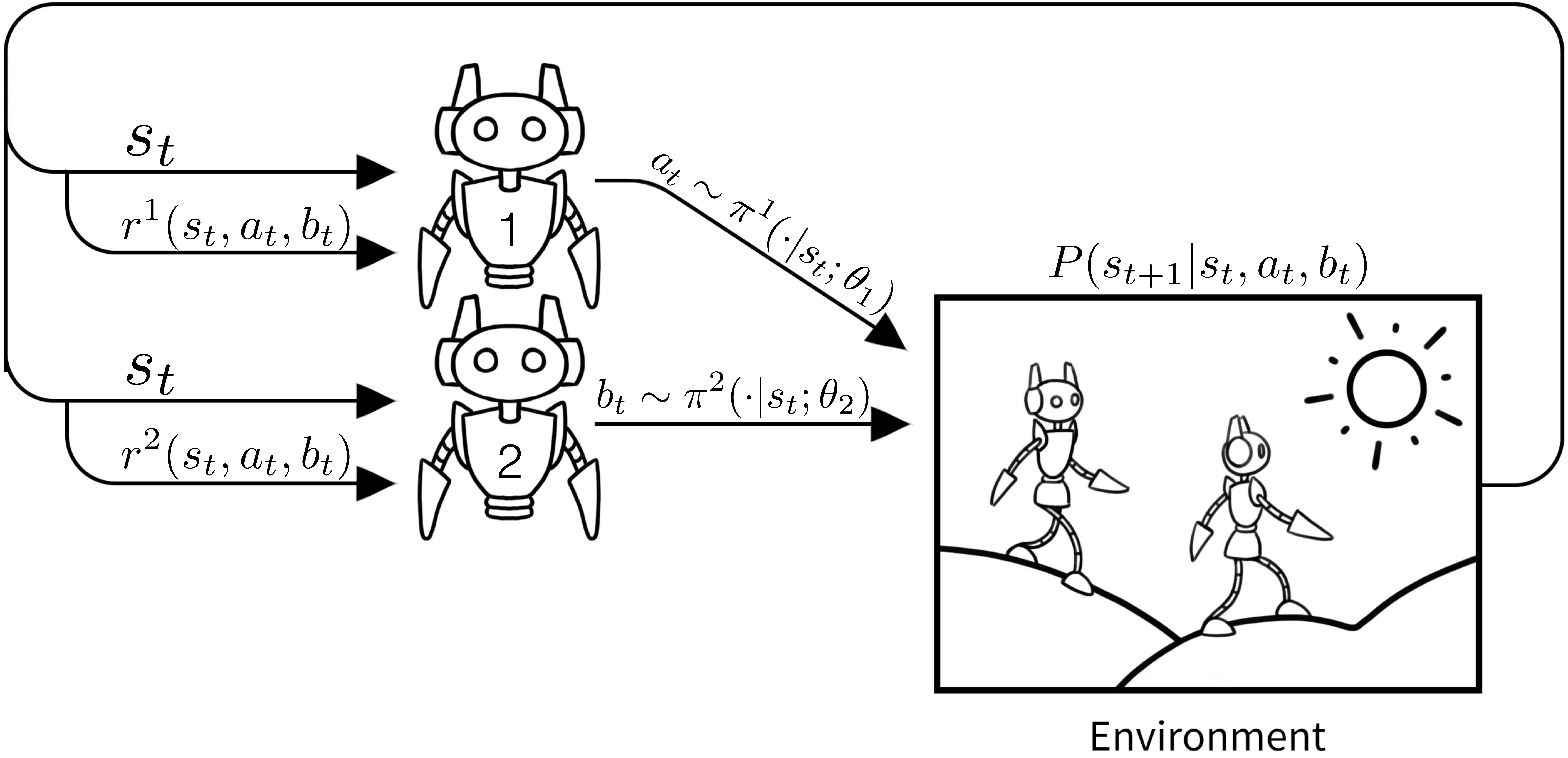
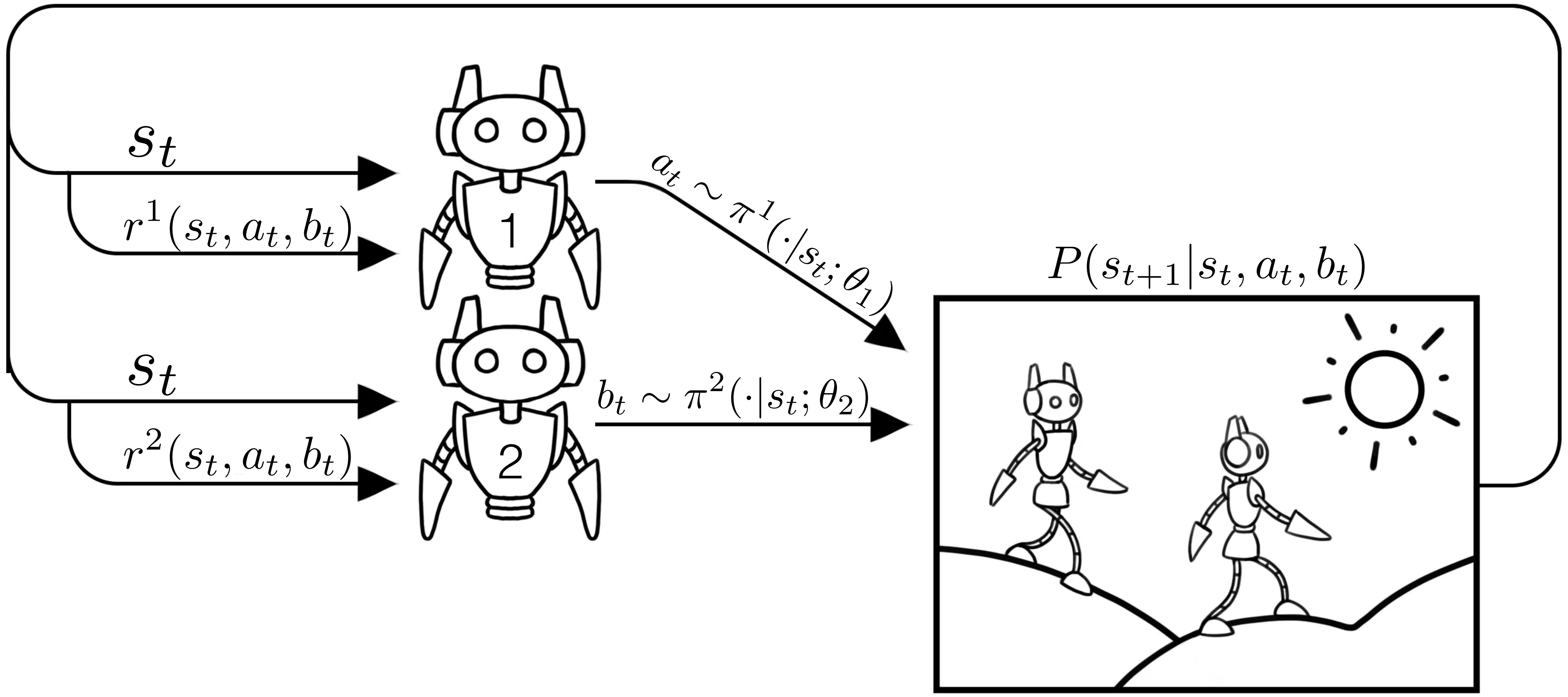


Image Credit: J K Terry, Multi-Agent Deep Reinforcement Learning in 13 Lines of Code Using PettingZoo: A tutorial on multi-agent deep reinforcement learning for beginners, Feb. 2021

Multi-Agent Reinforcement Learning



Discounted Return:

$$R^i(\tau) = \sum_{t=0}^{\infty} \gamma^t r^i(s_t, a_t, b_t)$$

Probability distribution over trajectories:

$$\Pr_{\mu}^{\pi^1,\pi^2}(\tau) = \mu(s_0)\pi^1(a_0|s_0)\pi^2(b_0|s_0)P(s_1|s_0, a_0, b_0)\ldots$$

Value function:

$$V^i(s) = \mathbb{E}_{\tau\sim\Pr_{\mu}^{\pi^1,\pi^2}}\left[R^i(\tau) \mid s_0 = s\right]$$

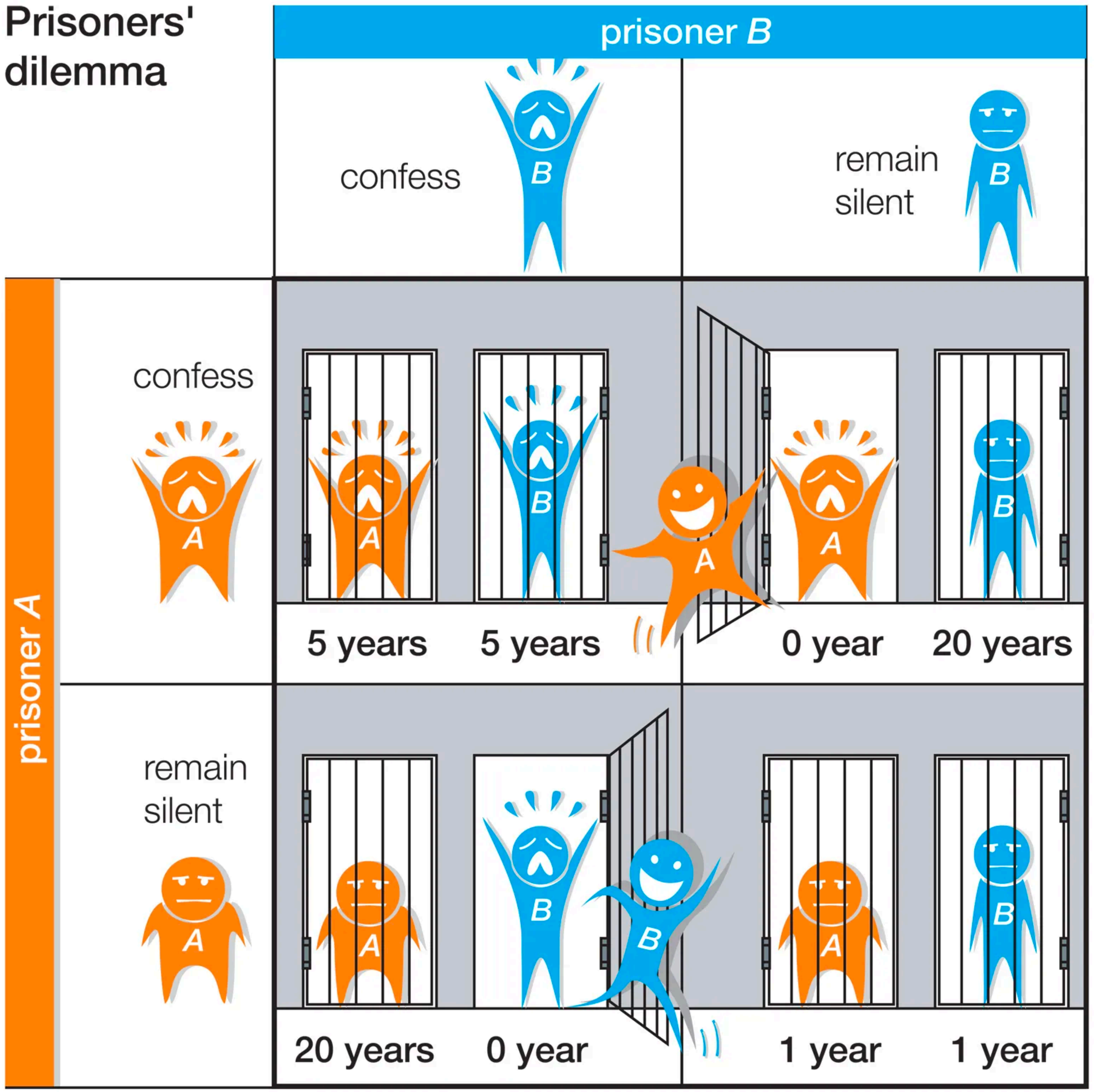
Action-value functions:

$$Q^1(s, a, b) = \mathbb{E}_{\tau\sim\Pr_{\mu}^{\pi^1,\pi^2}}\left[R^1(\tau) \mid s_0 = s, a_0 = a, b_0 = b\right],$$

$$Q^2(s, a, b) = \mathbb{E}_{\tau\sim\Pr_{\mu}^{\pi^1,\pi^2}}\left[R^2(\tau) \mid s_0 = s, a_0 = a, b_0 = b\right]$$

Image Credit: J K Terry, Multi-Agent Deep Reinforcement Learning in 13 Lines of Code Using PettingZoo: A tutorial on multi-agent deep reinforcement learning for beginners, Feb. 2021

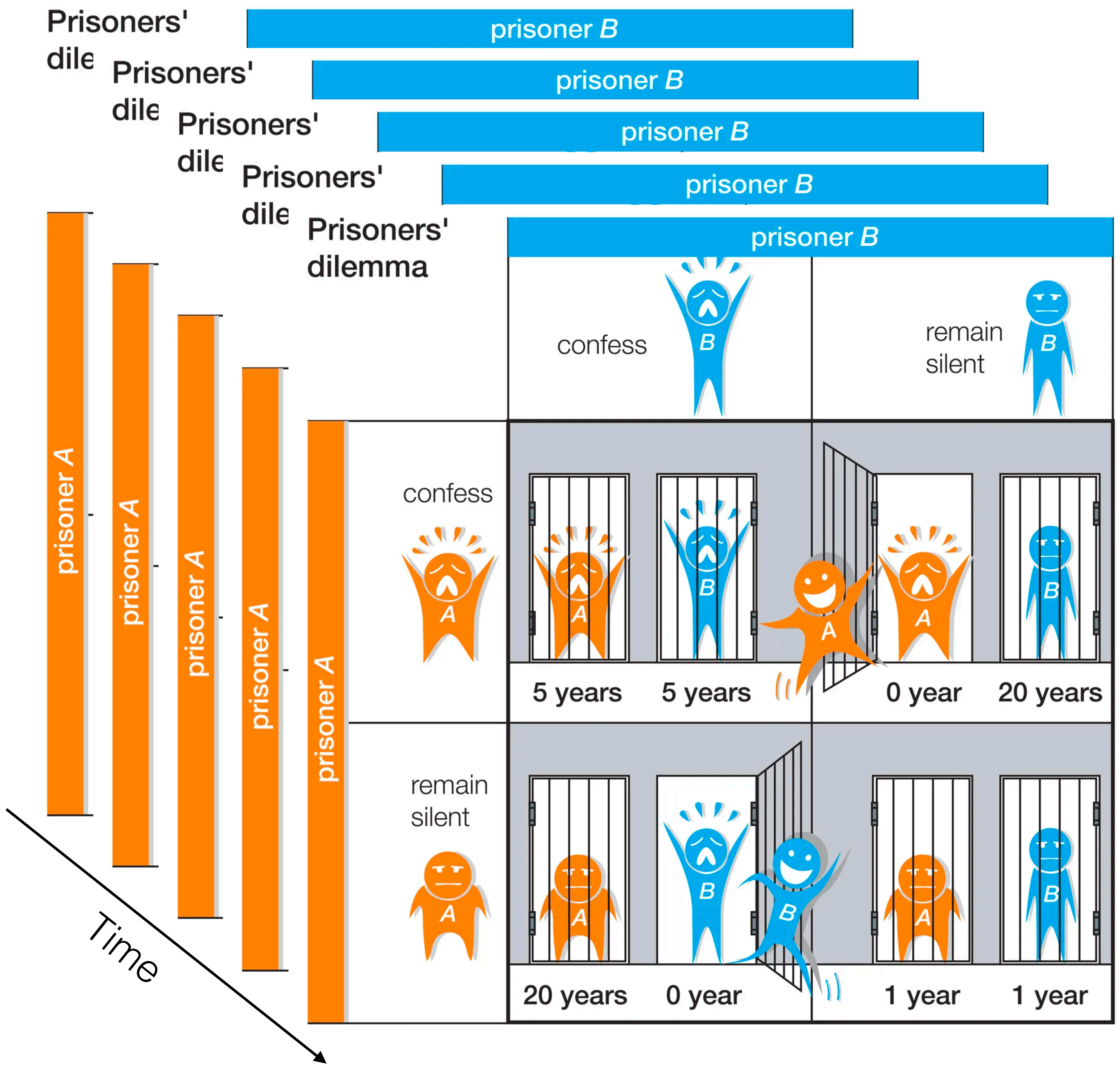
What are Social Dilemmas?



Social Dilemmas:

- A type of decision problem where each party's myopic efforts to maximize their own benefit lead to a less favourable outcome compared to when all parties cooperate.

What are Social Dilemmas? - Iterated Prisoners Dilemma (IPD)



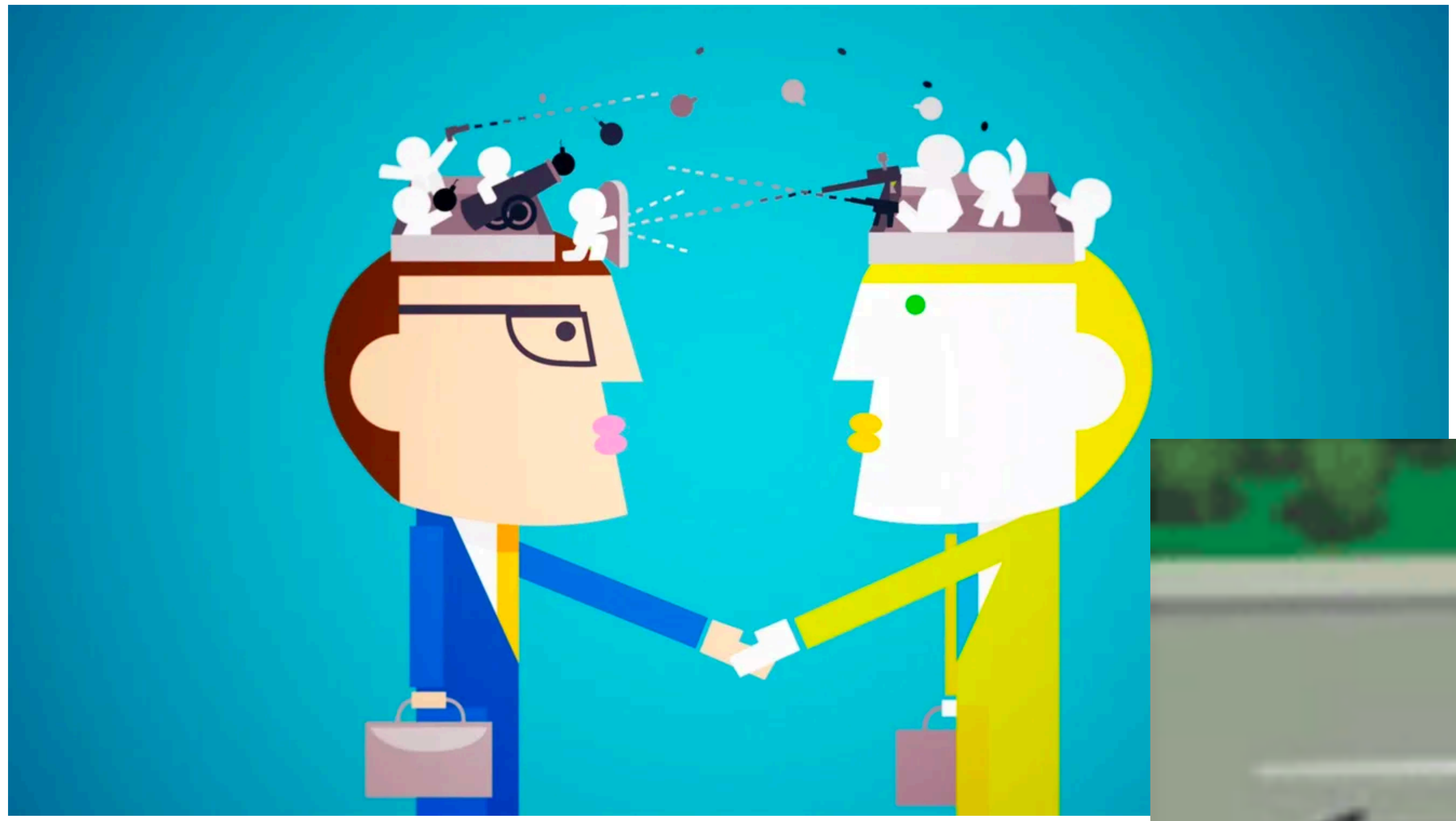
Social Dilemmas:

- A type of decision problem where each party's myopic efforts to maximize their own benefit lead to a less favourable outcome compared to when all parties cooperate.

Famous strategy: Tit-for-tat for prisoner A

- For $t = 1$, $a_t^A = \text{Remain Silent}$
- For $t > 1$, $a_t^A = a_{t-1}^B$

Social Dilemmas are Everywhere!



Business negotiations and deal-making

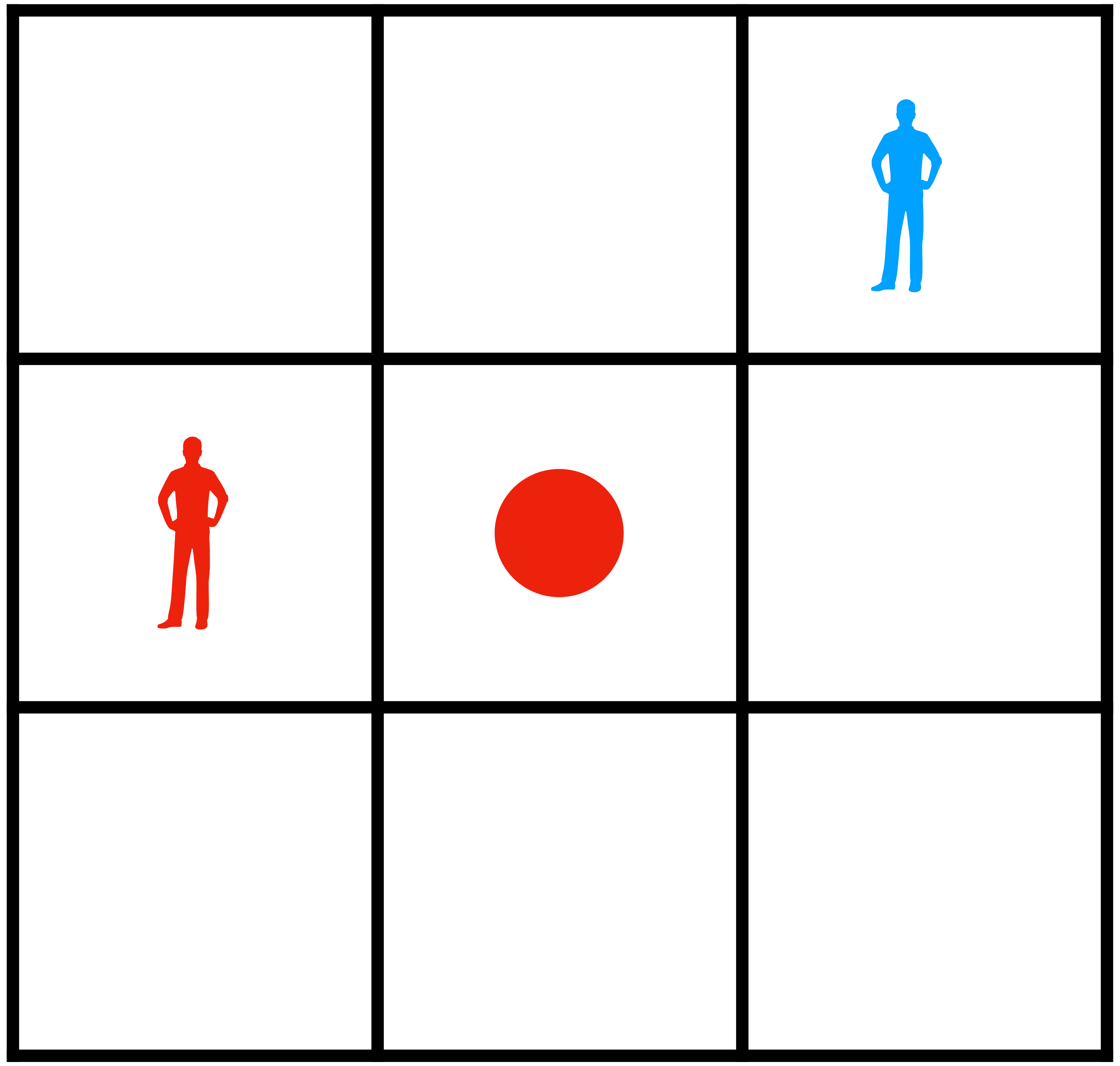
Negotiating interaction with other vehicles on the road.

Policy negotiation between countries.
Climate tragedy of the commons

Why Deep RL for Social Dilemmas?

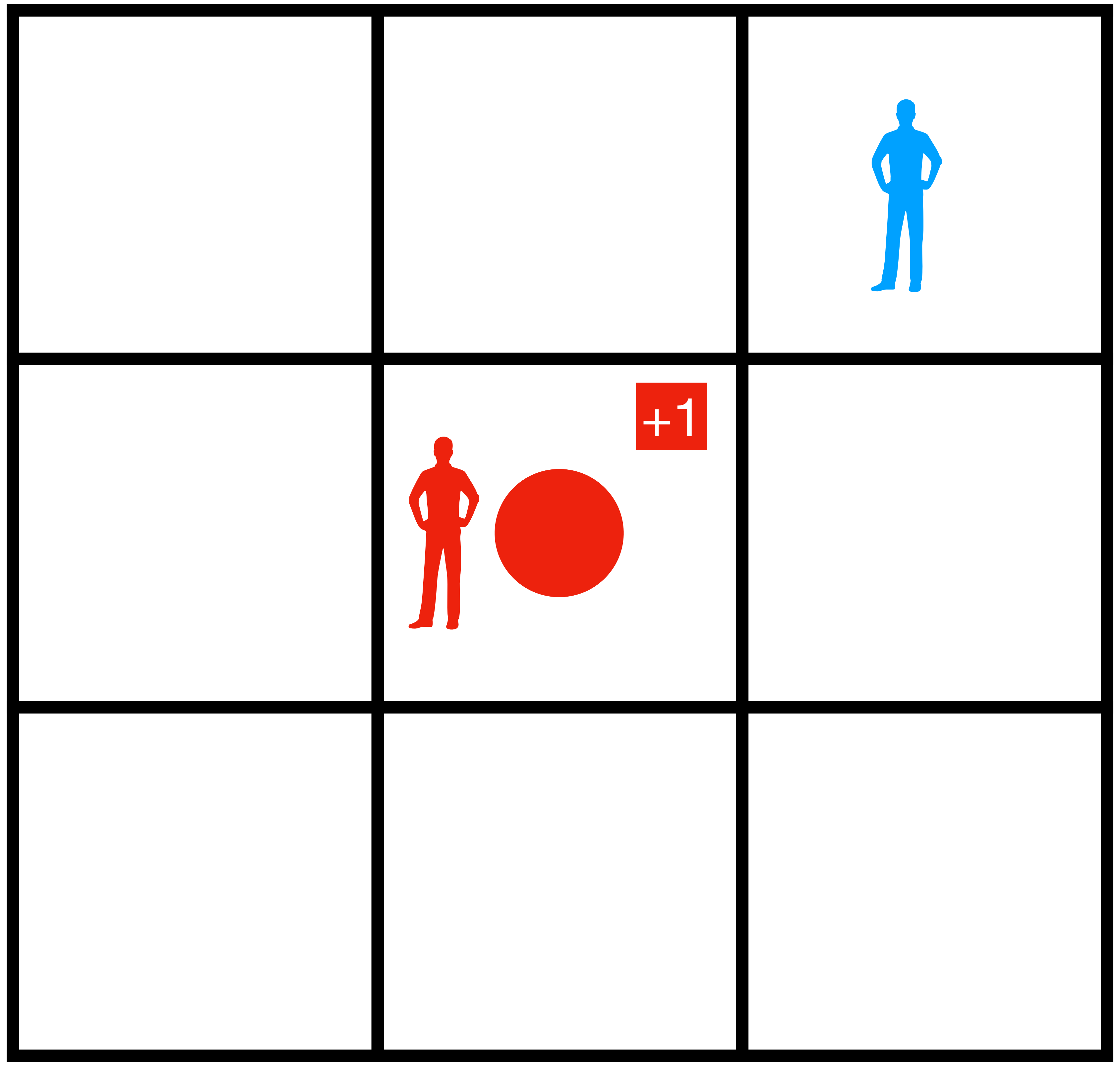
- Real environments are complex
- Deep reinforcement learning can (someday) handle complexity.

Coin Game



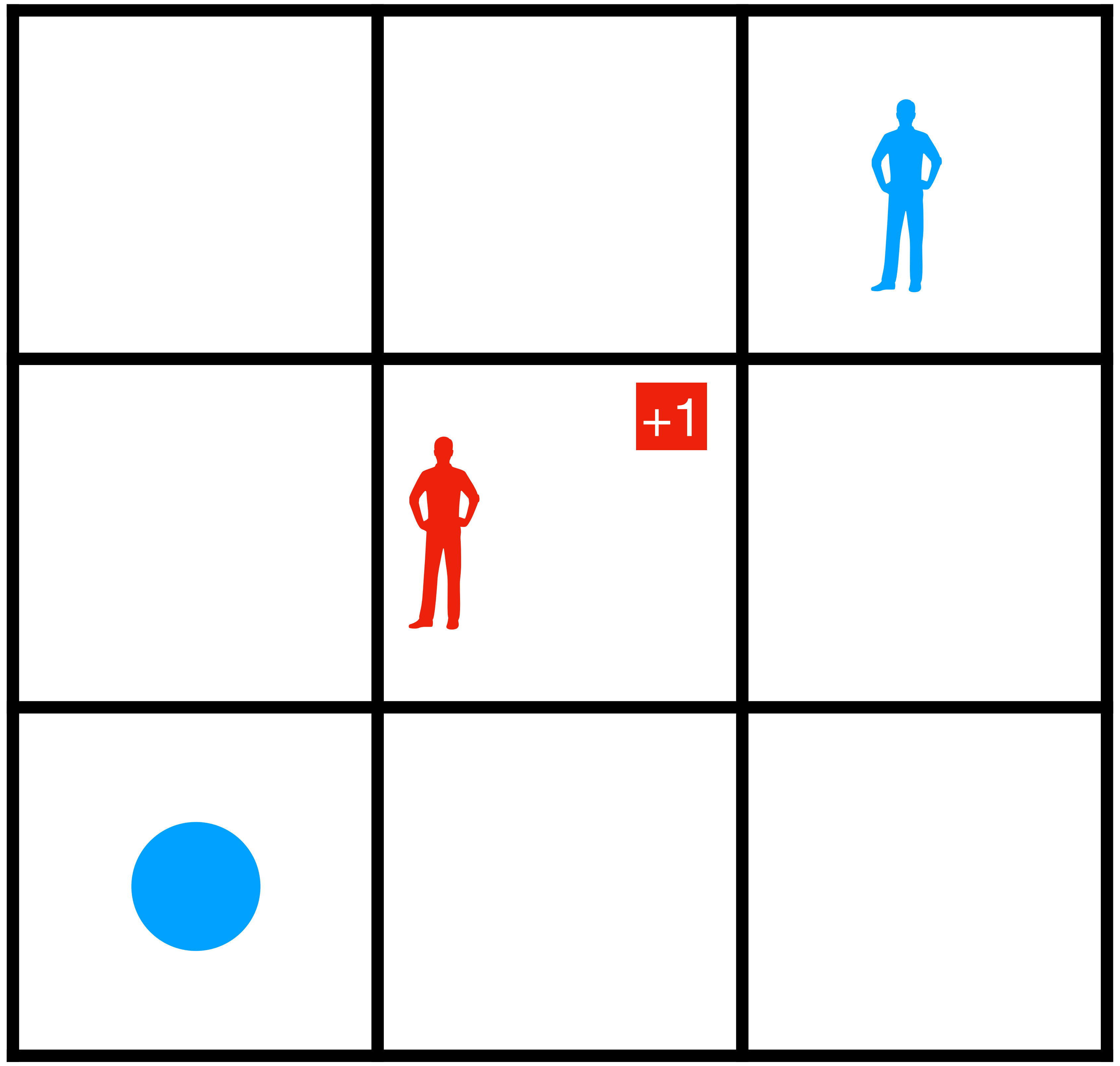
Learning with Opponent-Learning Awareness, Jakob N. Forester et al, 2018 arXiv:1709.04326

Coin Game



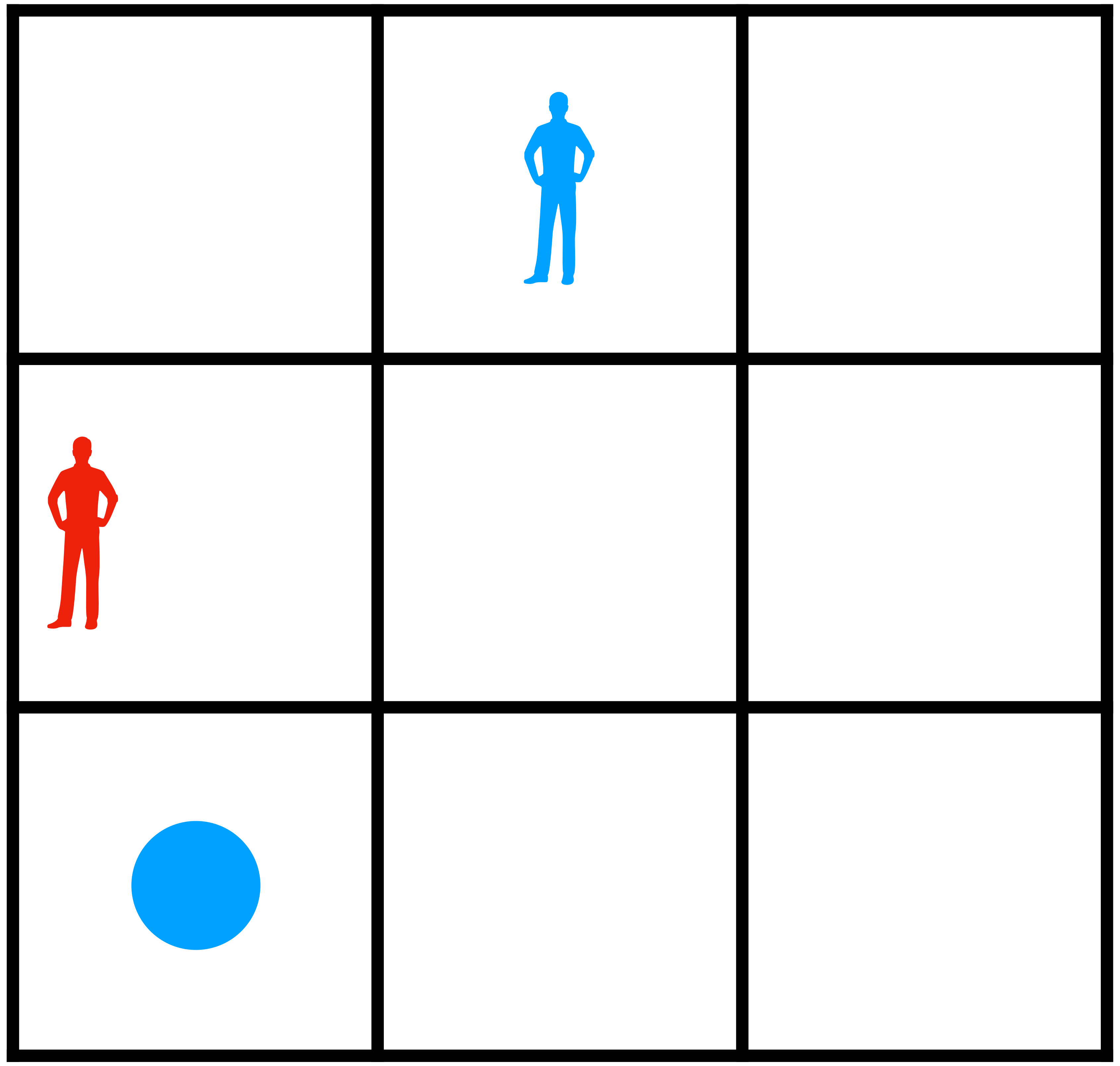
Learning with Opponent-Learning Awareness, Jakob N. Forester et al, 2018 arXiv:1709.04326

Coin Game



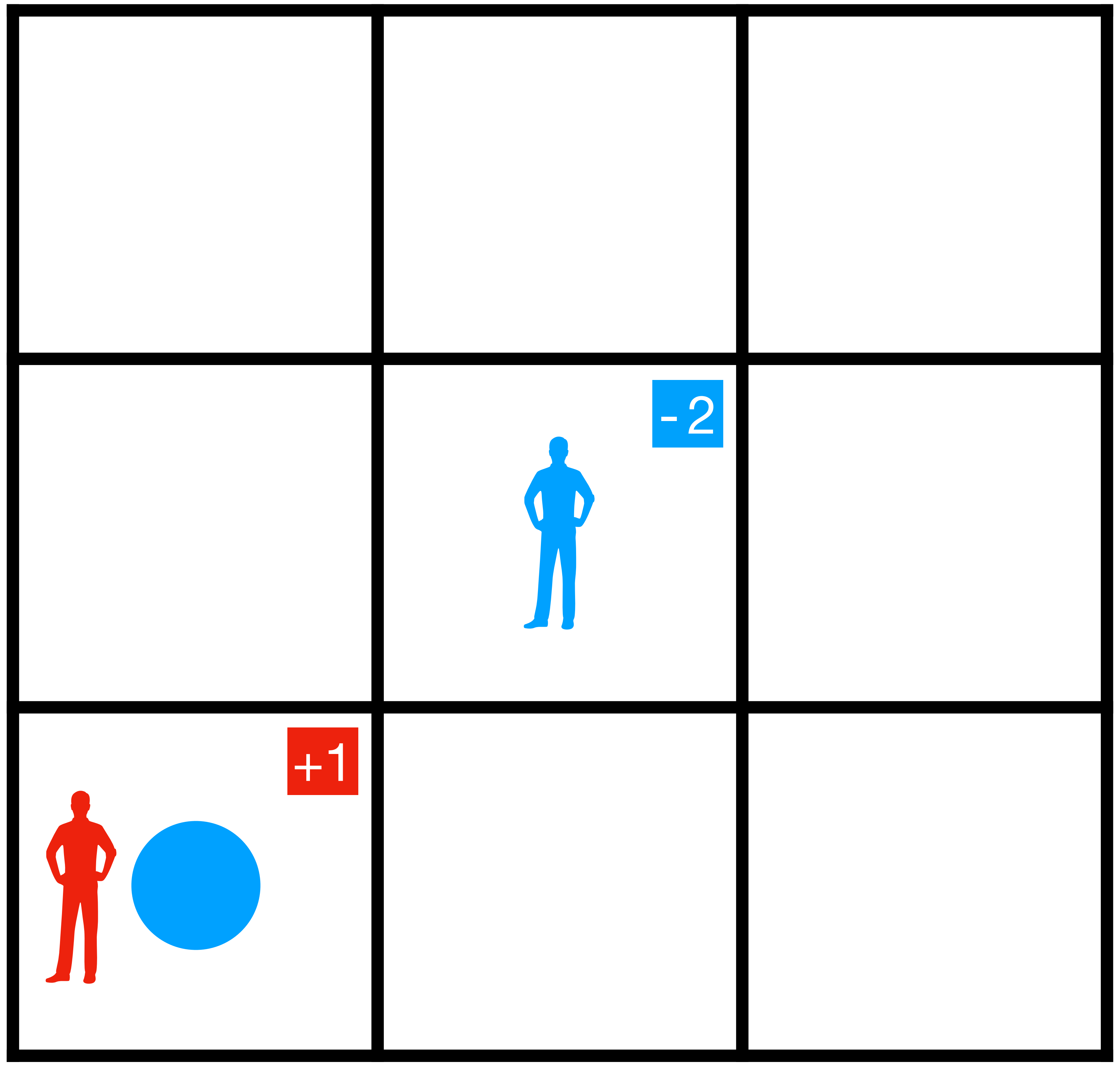
Learning with Opponent-Learning Awareness, Jakob N. Forester et al, 2018 arXiv:1709.04326

Coin Game



Learning with Opponent-Learning Awareness, Jakob N. Forester et al, 2018 arXiv:1709.04326

Coin Game



Learning with Opponent-Learning Awareness, Jakob N. Forester et al, 2018 [arXiv:1709.04326]

Multi-Agent Reinforcement Learning

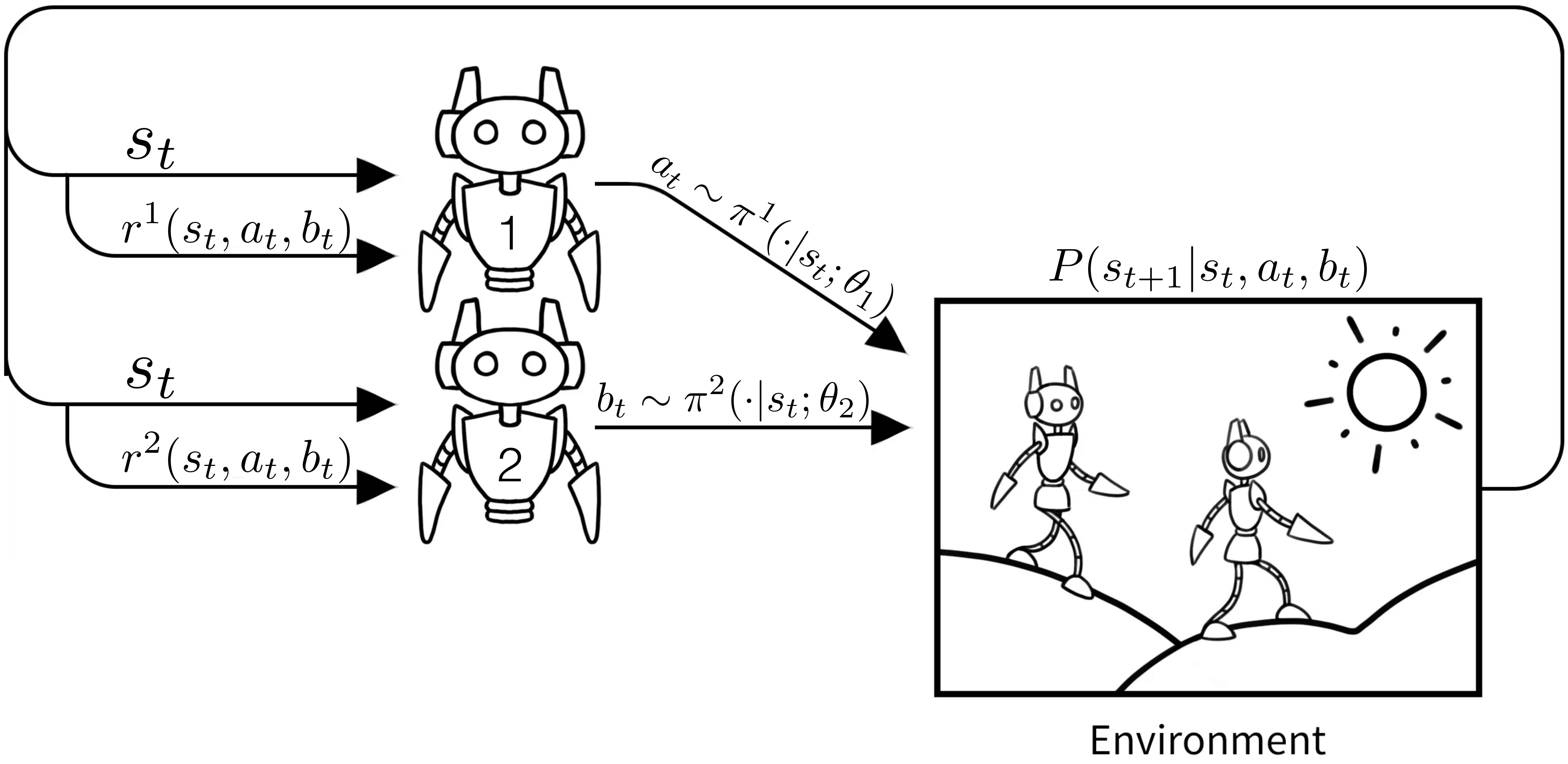
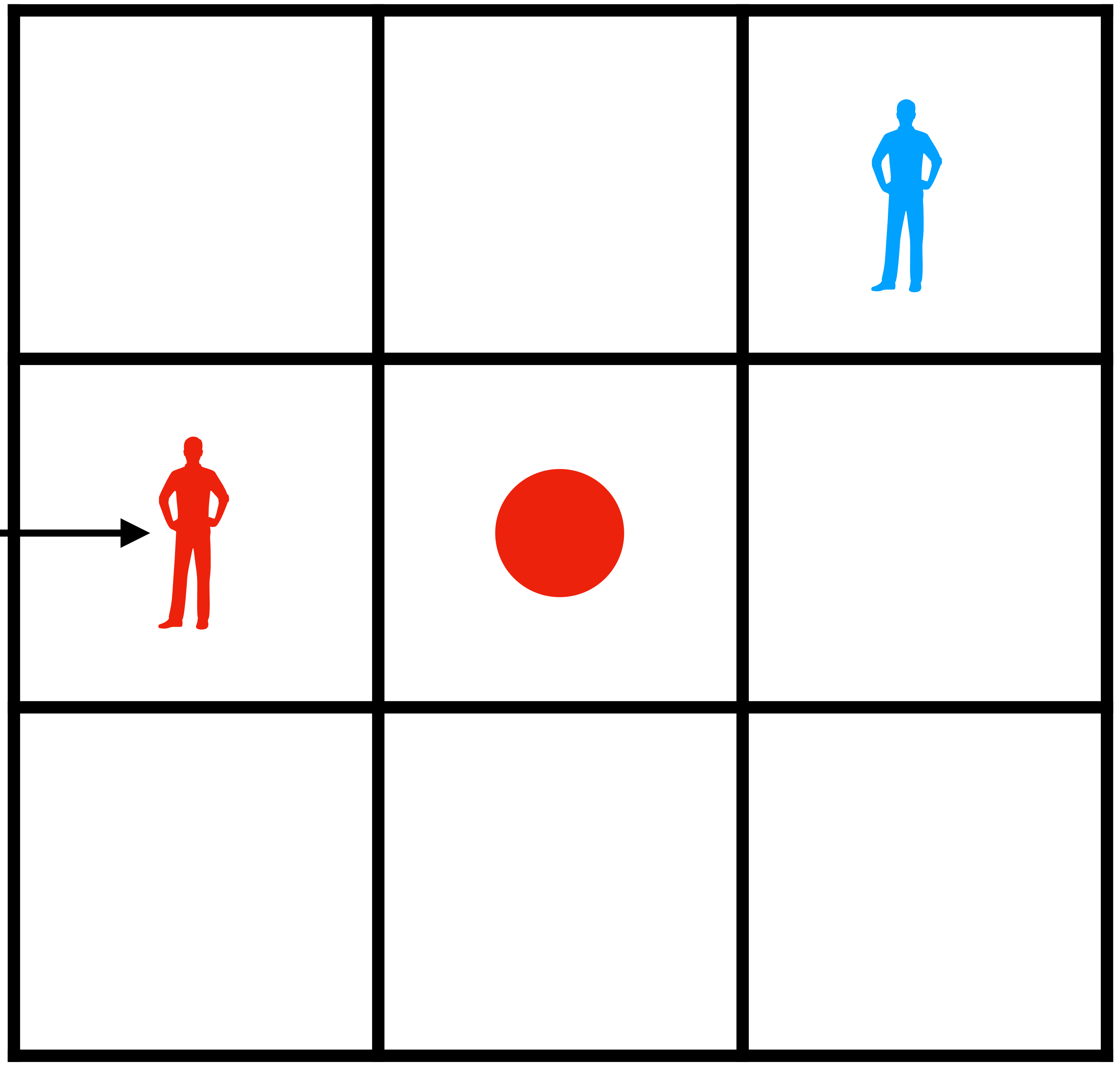


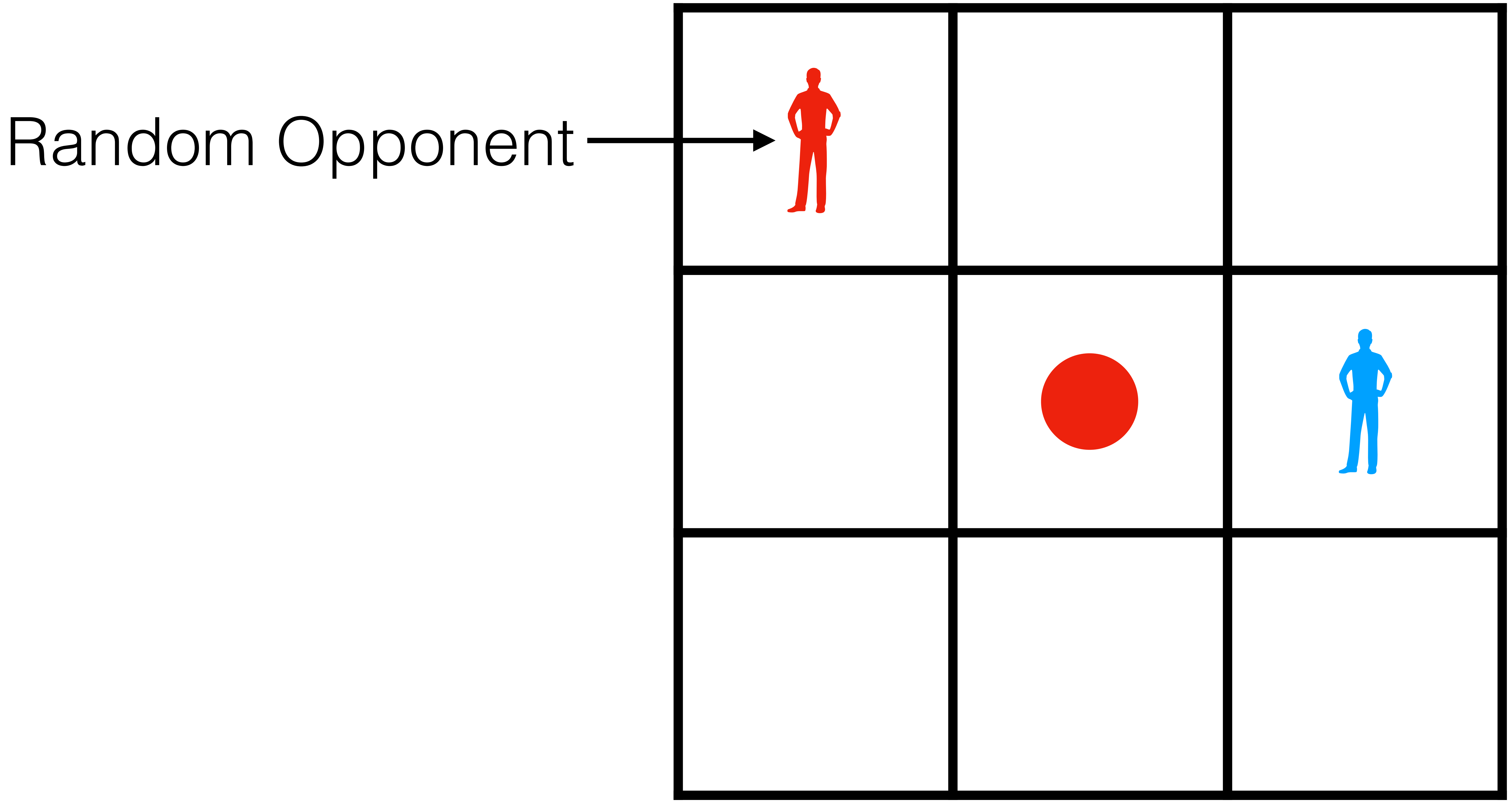
Image Credit: J K Terry, Multi-Agent Deep Reinforcement Learning in 13 Lines of Code Using PettingZoo: A tutorial on multi-agent deep reinforcement learning for beginners, Feb. 2021

Coin Game against a Random Opponent

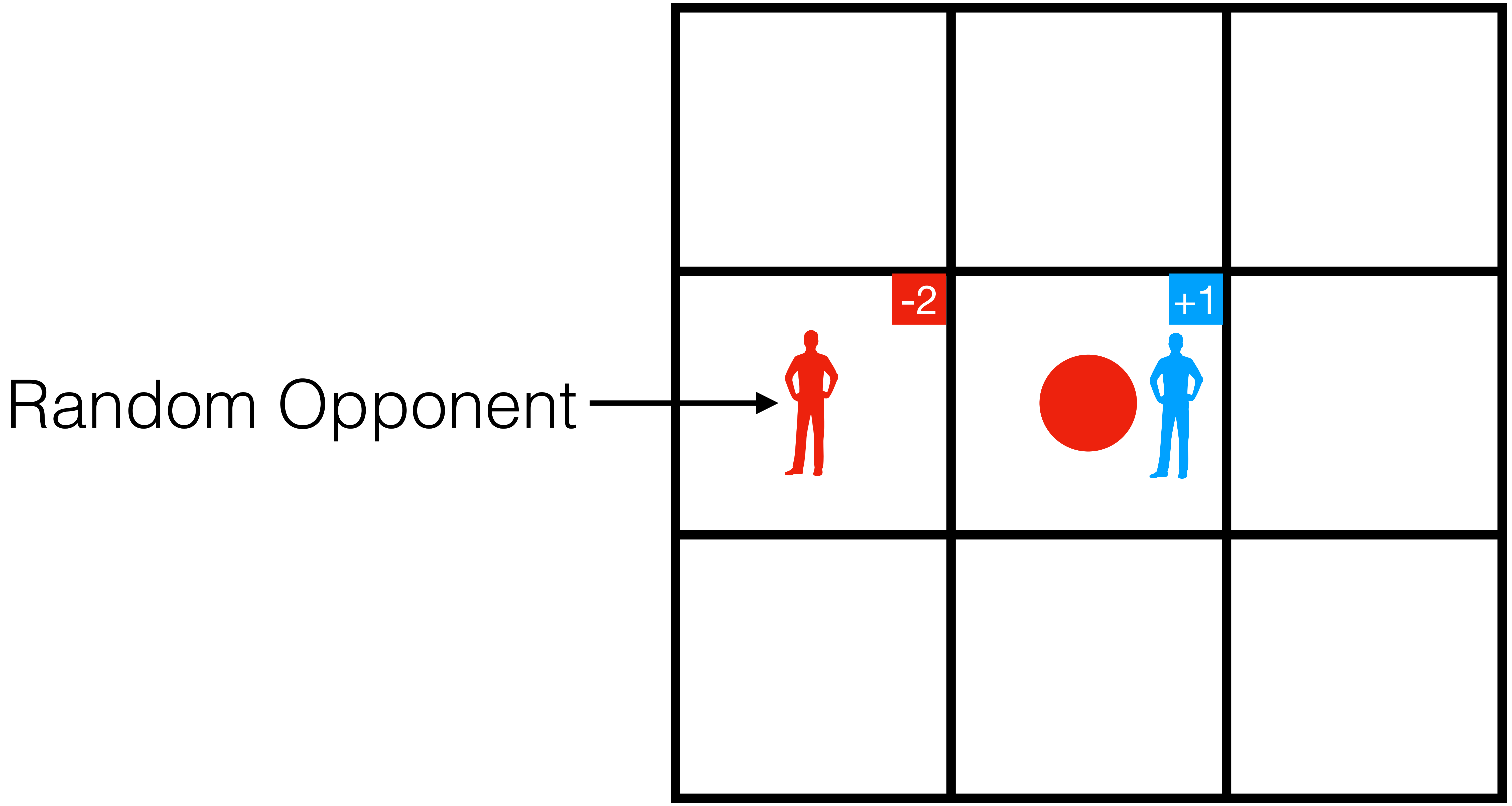
Random Opponent



Coin Game against a Random Opponent



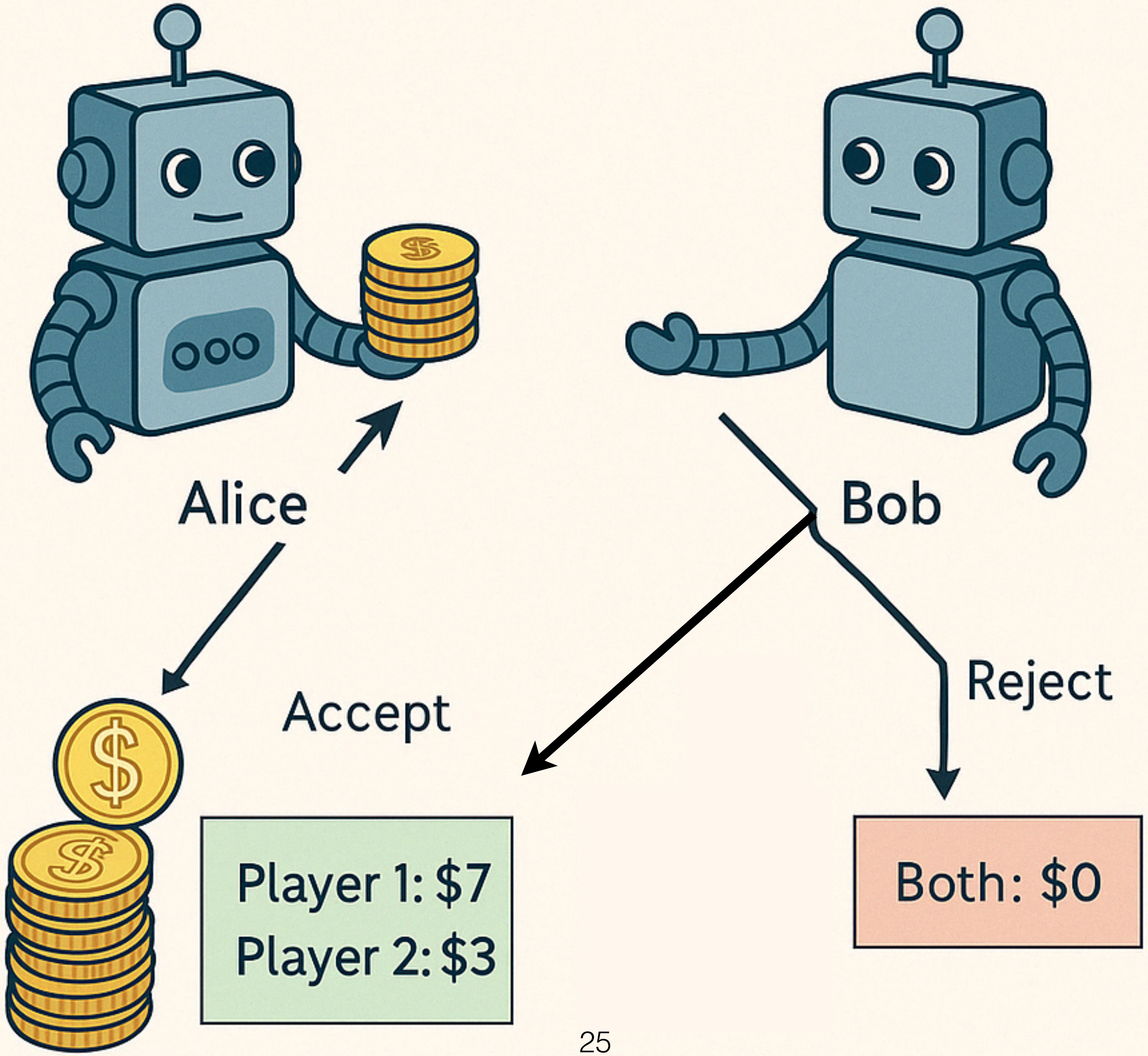
Coin Game against a Random Opponent



LLMs and The Ultimatum Game

Uday Karan Kapur

Muqeeth Mohammed



Déreck Piché

Michael Noukhovitch

LLMs and The Ultimatum Game

ALICE

Intermediary ⚙️

Welcome to the Splitting Game.

Game Overview:

1) Two agents divide 10 coins.

2) Each agent proposes a division of the coins.

3) Both agents must agree on a division to receive the coins. If not, both receive zero coins.

4) The game is played only once.

Response Format:

1) Responses must be within <finalize> </finalize> tags.

2) Use JSON format: <finalize> {"i_take": {"coins": x}, "other_agent_gets": {"coins": y}} </finalize>.

Goal: Aim to be fair with both the other agent and yourself.
You are the first agent. It is your turn to play.

LLM (alice) 🤖

<finalize> {"i_take": {"coins": 5}, "other_agent_gets": {"coins": 5}} </finalize>

BOB

Intermediary ⚙️

Welcome to the Splitting Game.

Game Overview:

1) Two agents divide 10 coins.

2) Each agent proposes a division of the coins.

3) Both agents must agree on a division to receive the coins. If not, both receive zero coins.

4) The game is played only once.

Response Format:

1) Responses must be within <finalize> </finalize> tags.

2) Use JSON format: <finalize> {"i_take": {"coins": x}, "other_agent_gets": {"coins": y}} </finalize>.

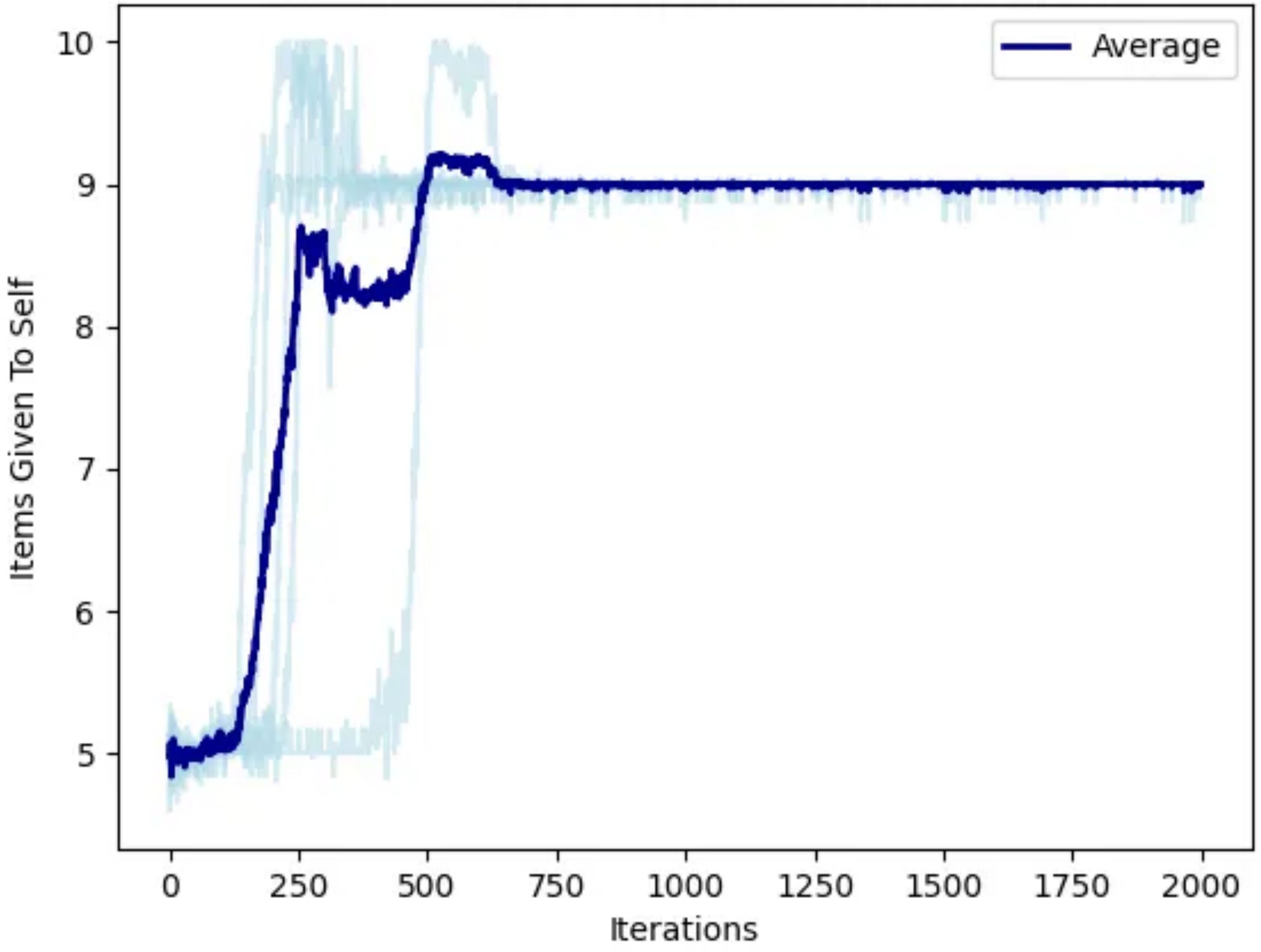
Goal: Aim to be fair with both the other agent and yourself.
The other agent's finalization was {'i_take': {'coins': 5}, 'other_agent_gets': {'coins': 5}}.
You are the second agent. It is your turn to play.

LLM (bob) 🤖

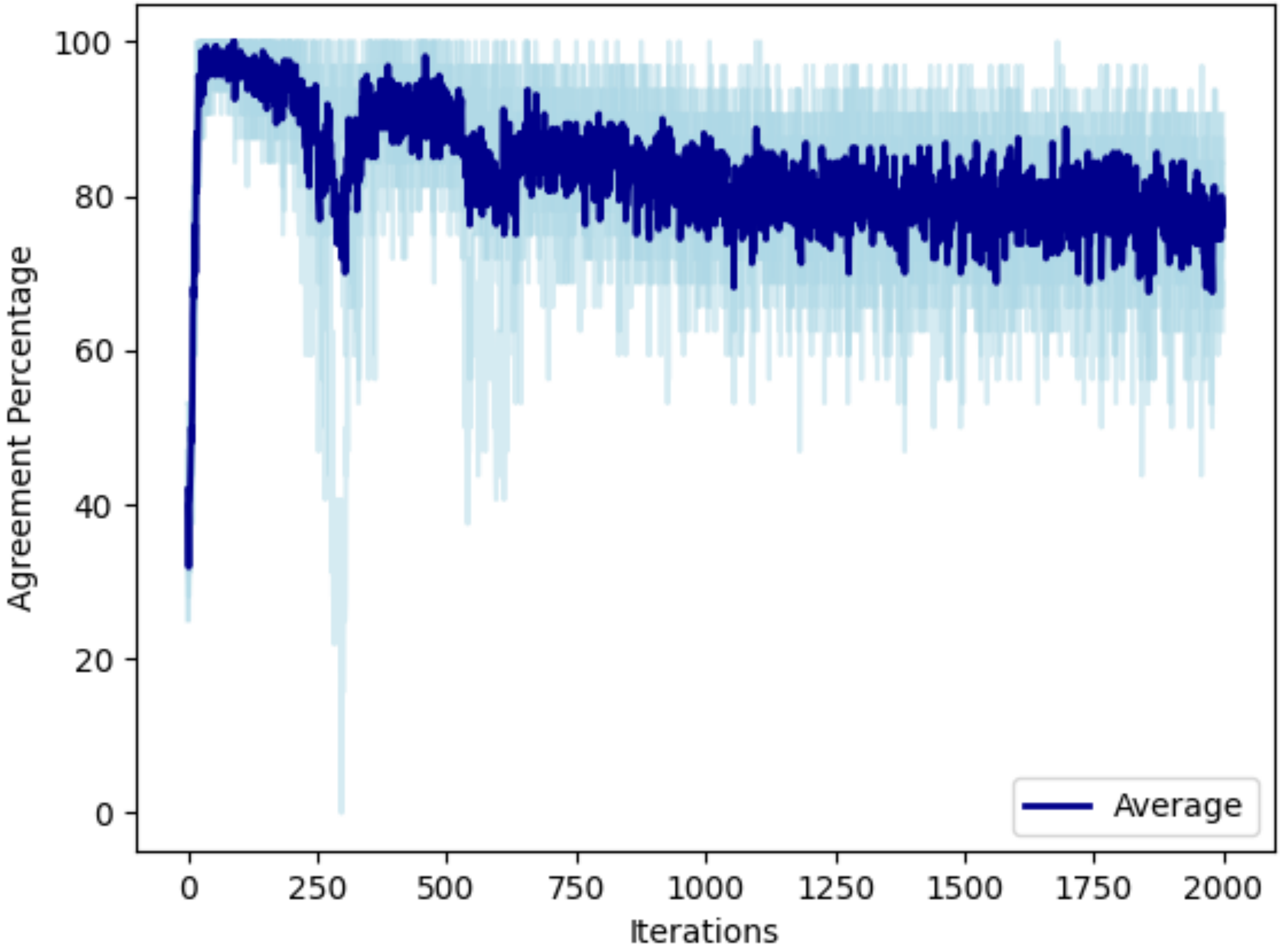
<finalize> {"i_take": {"coins": 5}, "other_agent_gets": {"coins": 5}} </finalize>

LLMs and The Ultimatum Game

Alice: items given to self



Bob: Agreement percentage



Opponent Shaping (Jakob Foerster et al.)

- Assume the learning dynamics of other agents can be controlled via some mechanism to incentivize desired behaviours.

➡ Agent can “shape” their opponents.

Learning with Opponent Learning Awareness (LOLA)

(Jakob Foerster et al., 2018, AAMAS)

LOLA agents assume the opponent is a naive learning agent, so it can simulate the update of the opponent and take gradients w.r.t. opponent policy parameters.

$V^1(\theta^1, \theta^2) :=$ Expected return of the agent conditioned on its policy parameters, θ^1, and the opponent's policy parameters, θ^2

$\Delta\theta^2 :=$ Imagined parameter update for the opponent, which can be differentiated w.r.t. θ^1

LOLA maximizes $V^1(\theta^1, \theta^2 + \Delta\theta^2)$ w.r.t θ^1

Limitations of LOLA

- Assumes access to the opponent's policy parameters.
- $\nabla_{\theta_1} V^1(\theta^1, \theta^2 + \Delta\theta^2)$ is difficult to estimate so in practice a surrogate that uses the first-order Taylor expansion is used (imprecise).
- To compute the gradient with respect to the update, it is necessary to build large computational graphs and differentiate through it (very expensive).

A different perspective on where naive RL goes wrong

- In RL we aim to optimize the expected return of the agent (agent 1):

$$V^1(\mu) = \mathbb{E}_{\tau \sim \Pr_\mu^{\pi^1, \pi^2}} \left[R^1(\tau) \right], \quad \text{where} \quad R^i(\tau) = \sum_{t=0}^{\infty} \gamma^t r^i(s_t, a_t, b_t)$$

- Adapting the original Actor-Critic formulation (Konda & Tsitsiklis, 2000) to the joint agent-opponent policy space we have:

$$\nabla_{\theta^1} V^1(\mu) = \mathbb{E}_{\tau \sim \Pr_\mu^{\pi^1, \pi^2}} \left[\sum_{t=0}^{T} A^1(s_t, a_t, b_t) \nabla_{\theta^1} \left(\log \pi^1(a_t|s_t) + \underbrace{\log \pi^2(b_t|s_t)}_{\nabla_{\theta^1} = 0} \right) \right]$$

  - Where the Advantage of agent 1 is: $A^1(s_t, a_t, b_t) = Q^1(s_t, a_t, b_t) - V^1(s_t)$

Opponent Shaping via Advantage Alignment

- What if we could make the opponent (agent 2) policy be directly dependent on the policy of Agent 1?

$$\nabla_{\theta^1} V^1(\mu) = \mathbb{E}_{\tau \sim \Pr_\mu^{\pi^1, \pi^2}} \left[\sum_{t=0}^{T} A^1(s_t, a_t, b_t) \nabla_{\theta^1} \left(\log \pi^1(a_t|s_t) + \log \hat{\pi}^2(b_t|s_t) \right) \right]$$

- Advantage Alignment key assumption:

$$\hat{\pi}^2(b_t|s_t) \propto \exp\left(\beta \cdot \mathbb{E}_{a_t \sim \pi^1(\cdot|s)}[Q^2(s_t, a_t, b_t)]\right)$$

➡ Direct dependency on Agent 1 policy $\pi^1(a_t \mid s_t)$ and parameters θ^1

Advantage Alignment

Assumption 1: Each agent i learns to maximize their value function: $\max V^i(\mu)$.

Assumption 2: Opponent (player 2) acts proportionally to the exponent of their action-value function: $\hat{\pi}^2(b_t|s_t) \propto \exp\left(\beta \cdot \mathbb{E}_{a_t \sim \pi^1(\cdot|s)}[Q^2(s_t, a_t, b_t)]\right)$

Policy gradient:
(Actor-Critic)

$$\nabla_{\theta_1} V^1(\mu) = \mathbb{E}_{\tau \sim \mathrm{Pr}_\mu^{\pi^1,\pi^2}}\left[\sum_{t=0}^{\infty} \gamma^t A^1(s_t, a_t, b_t)\left(\underbrace{\nabla_{\theta_1} \log \pi^1(a_t|s_t)}_{\text{policy gradient term}} + \underbrace{\nabla_{\theta_1} \log \hat{\pi}^2(b_t|s_t)}_{\text{opponent shaping term}}\right)\right]$$

Expanding the opponent shaping term:

$$\beta \cdot \mathbb{E}_{\tau \sim \mathrm{Pr}_\mu^{\pi^1,\pi^2}}\left[\sum_{t=0}^{\infty} \gamma^{t+1}\left(\sum_{k<t} \gamma^{t-k} A^1(s_k, a_k, b_k)\right) A^2(s_t, a_t, b_t) \nabla_{\theta^1} \log \pi^1(a_t|s_t)\right]$$

Advantage Alignment — Opponent Shaping term

- If interaction with opponent (Agent 2) has been positive (for Agent 1) the advantages are aligned.
- If interaction with opponent (Agent 2) has been negative (for Agent 1) the advantages are at odds.

Agent 2

Agent 1: $\sum_{k<t} \gamma^{t-k} A_k^1$ \ Agent 2: A_t^2	+	-
+	+	-
-	-	+

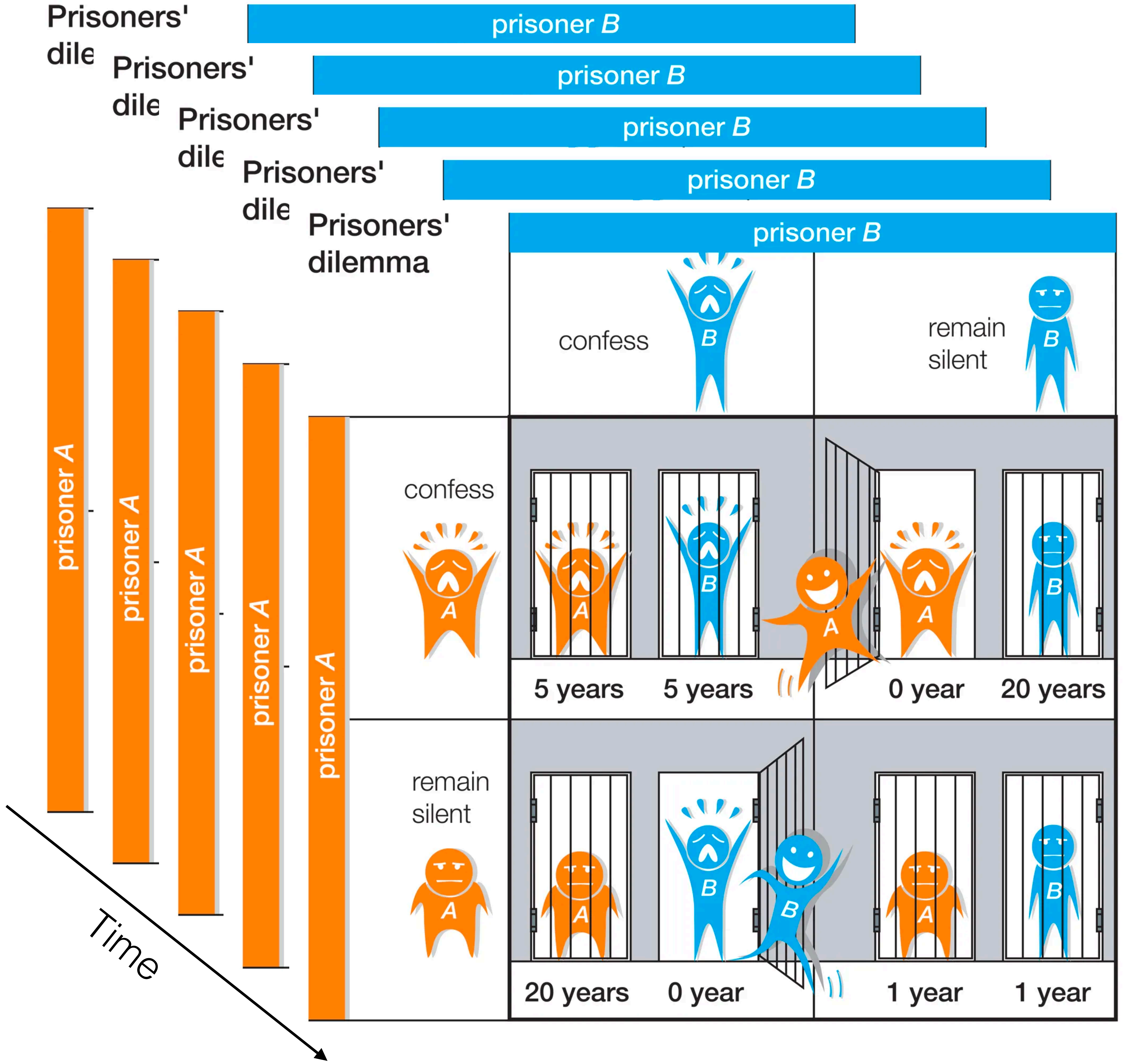
Opponent shaping term:

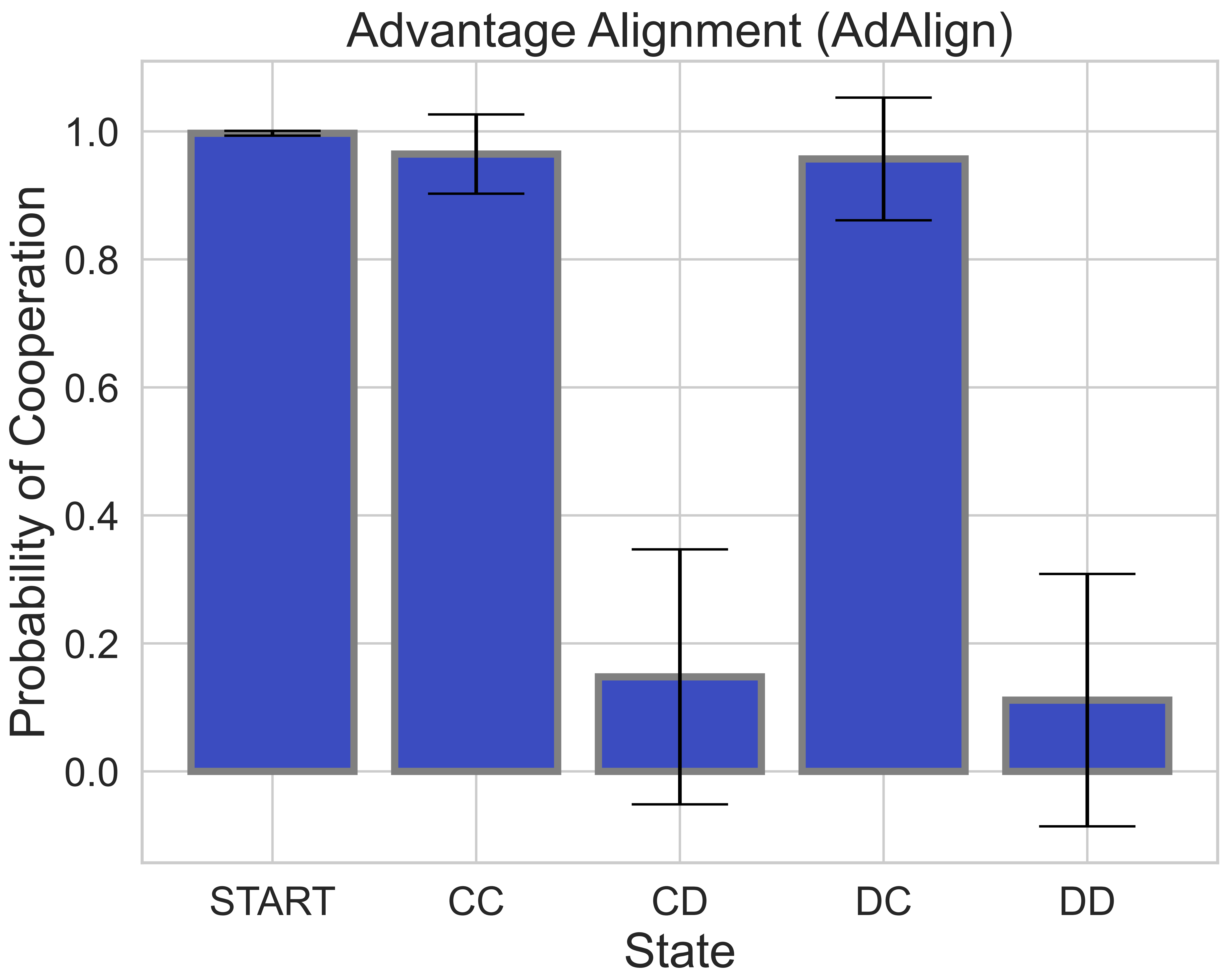
$$\beta \cdot \mathbb{E}_{\tau \sim \Pr_{\mu}^{\pi^1,\pi^2}} \left[\sum_{t=0}^{\infty} \gamma^{t+1} \left(\sum_{k<t} \gamma^{t-k} A^1(s_k, a_k, b_k) \right) A^2(s_t, a_t, b_t) \nabla_{\theta^1} \log \pi^1(a_t|s_t) \right]$$

Theorem: Advantage Alignment preserves Nash equilibria

If a joint policy (π_1^*, π_2^*) *constitutes a Nash equilibrium, then applying Advantage Alignment formula will not change the policy, as the gradient contribution of the advantage alignment term is zero.*

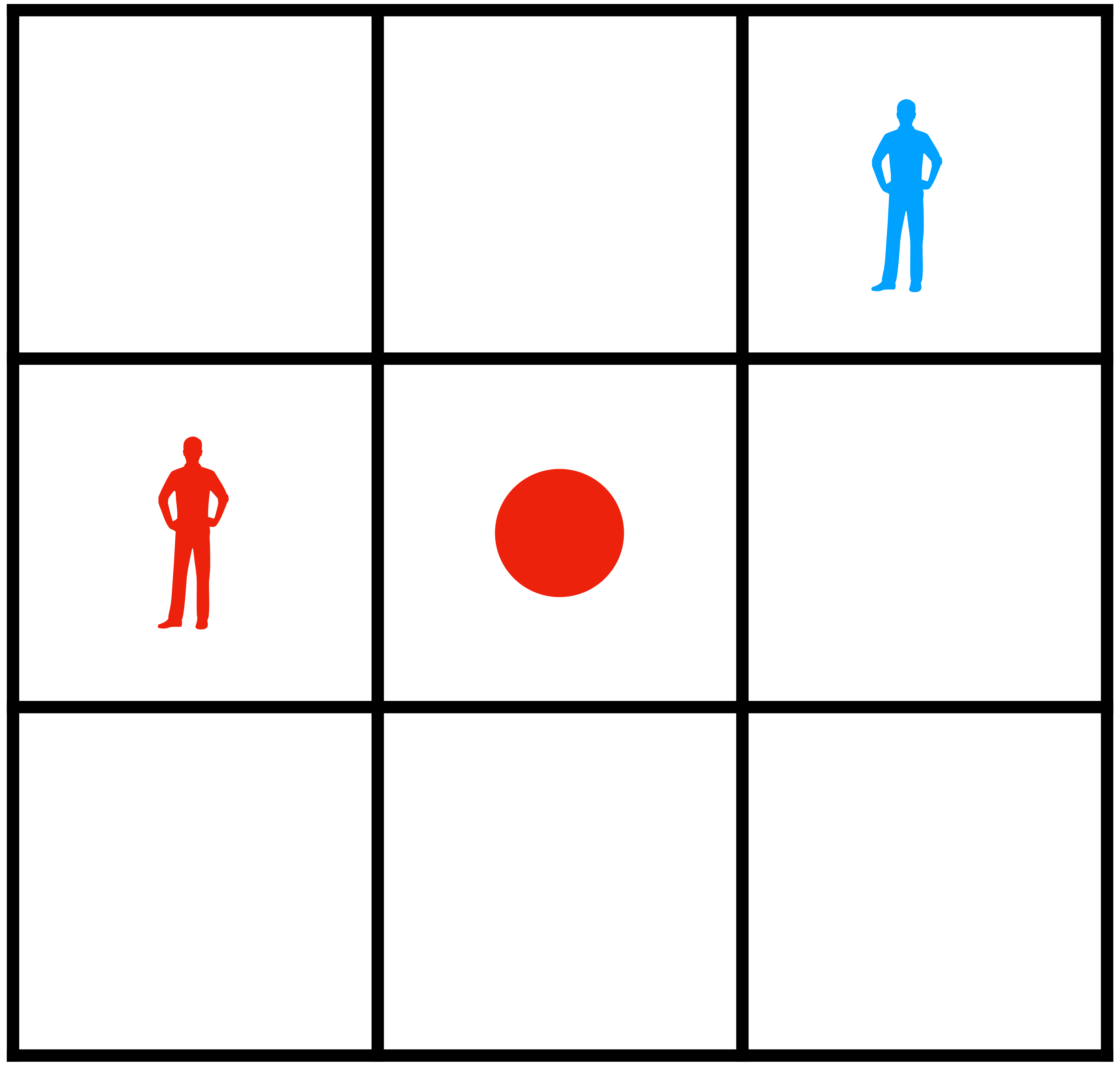
Iterated Prisoners Dilemma (IPD)

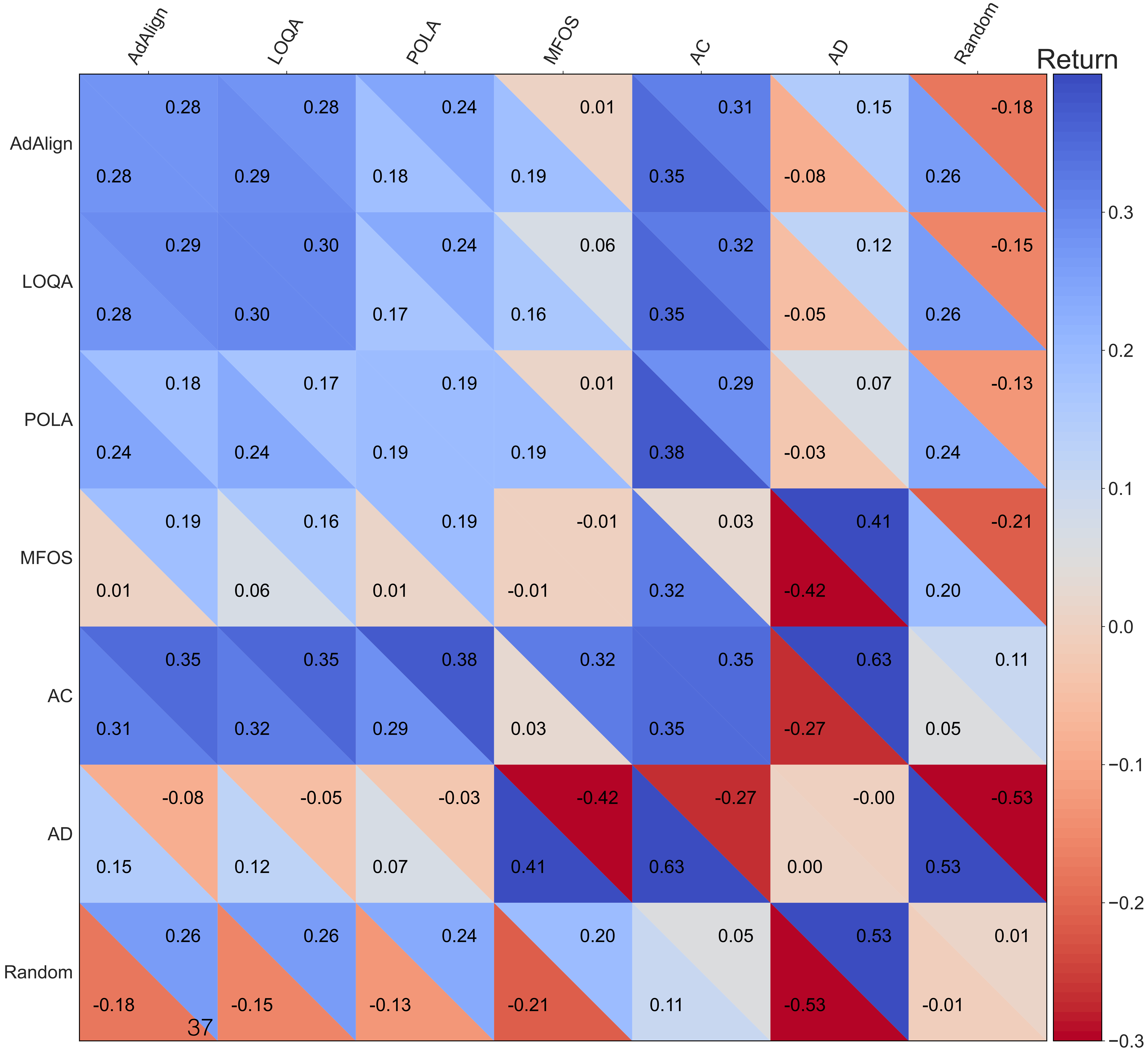




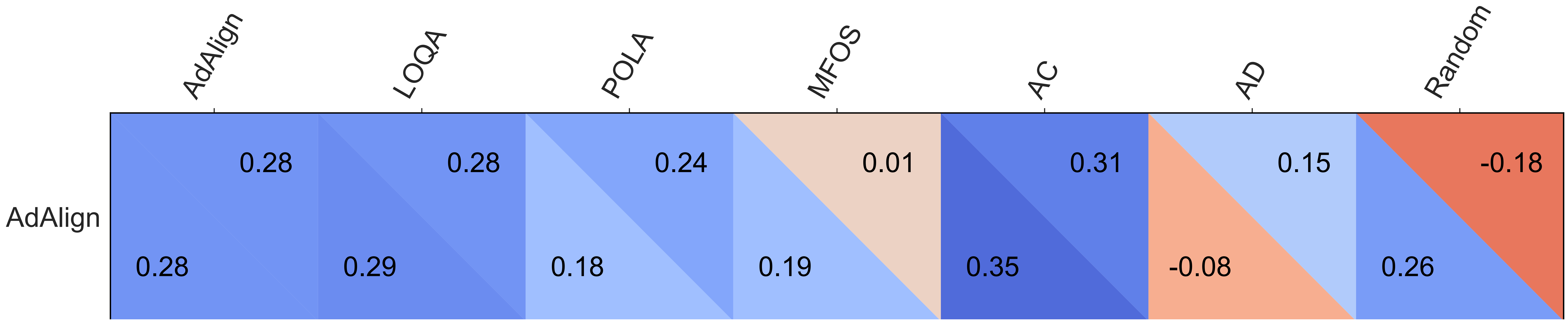
Coin Game Tournament

Coin Game

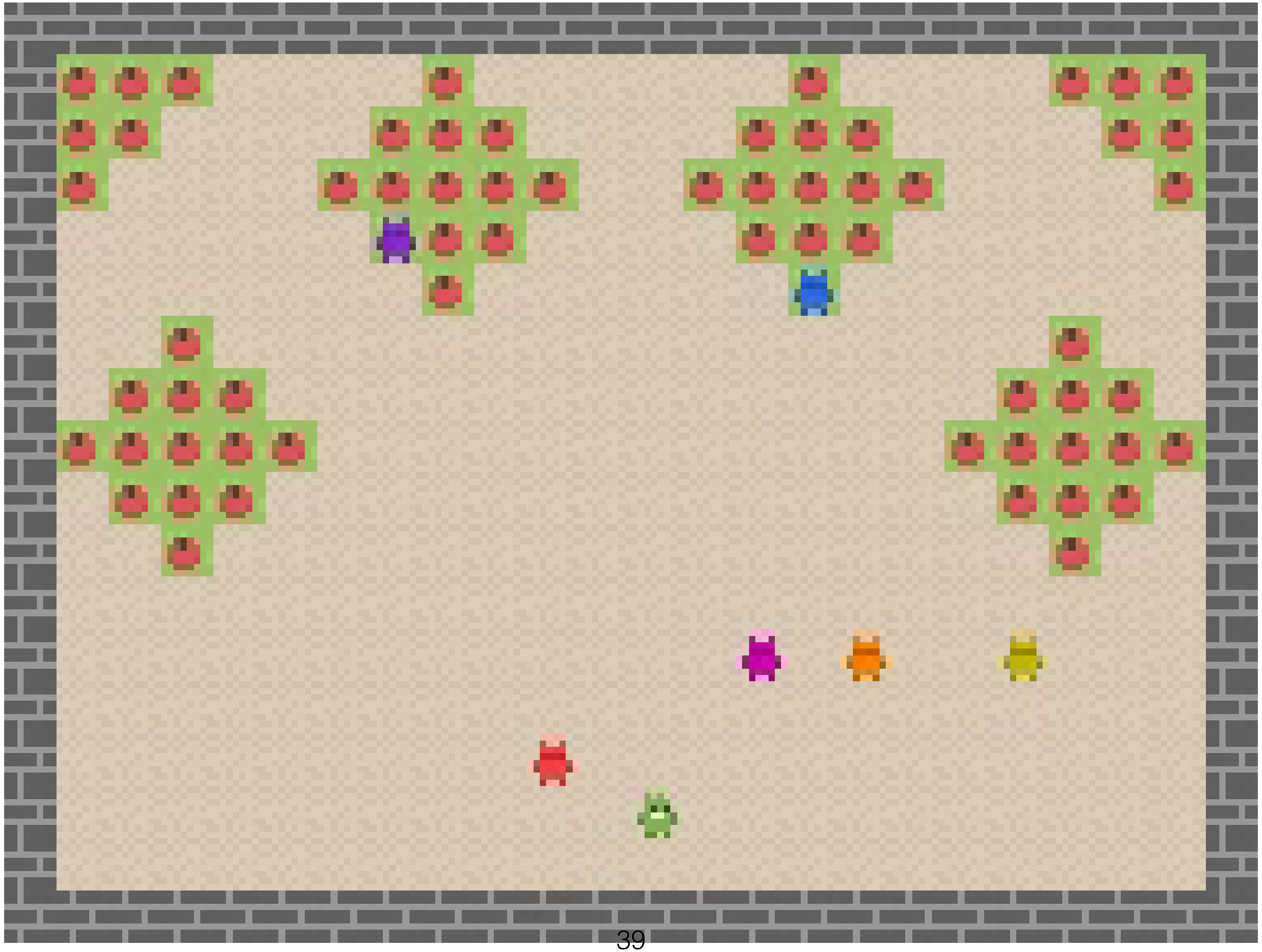




Coin Game Tournament

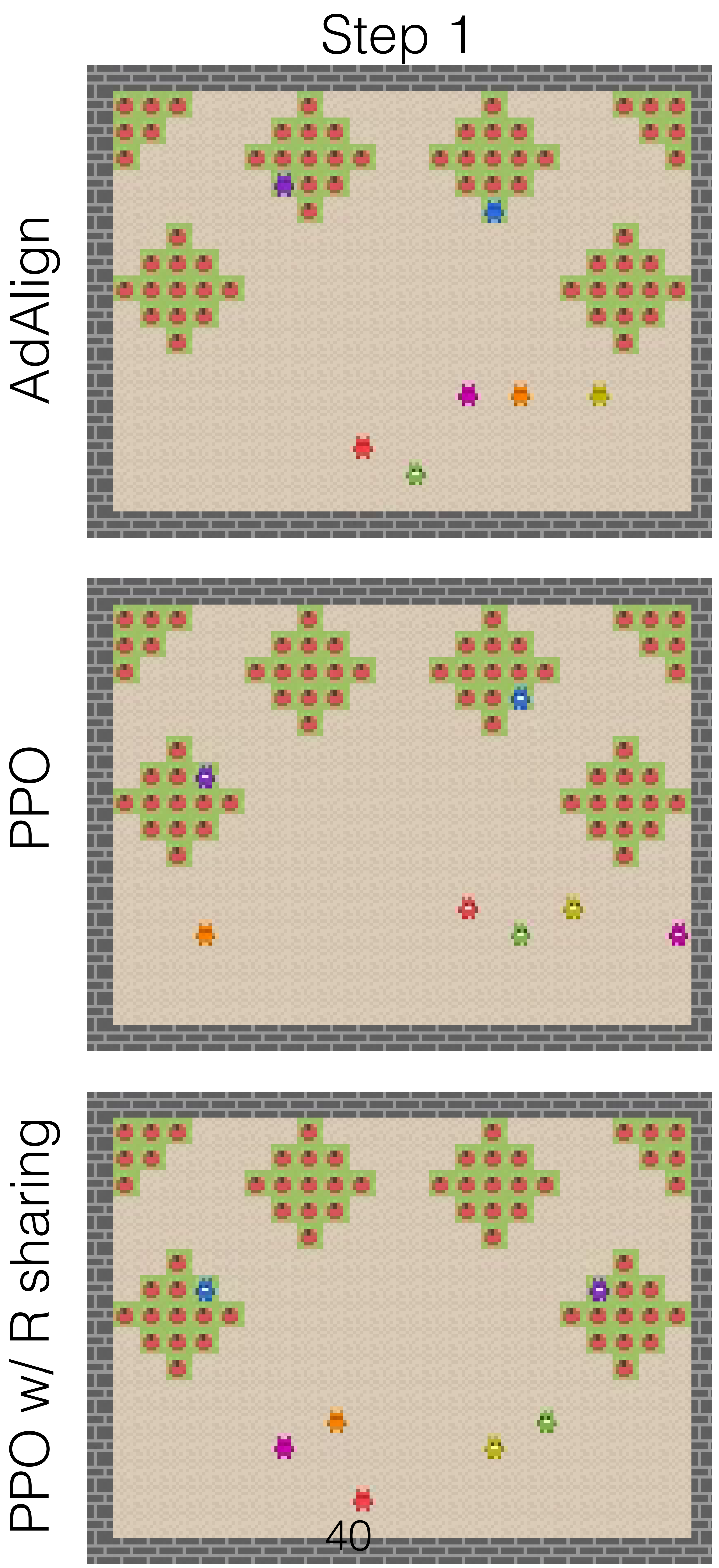


Melting Pot — Common Harvest Game



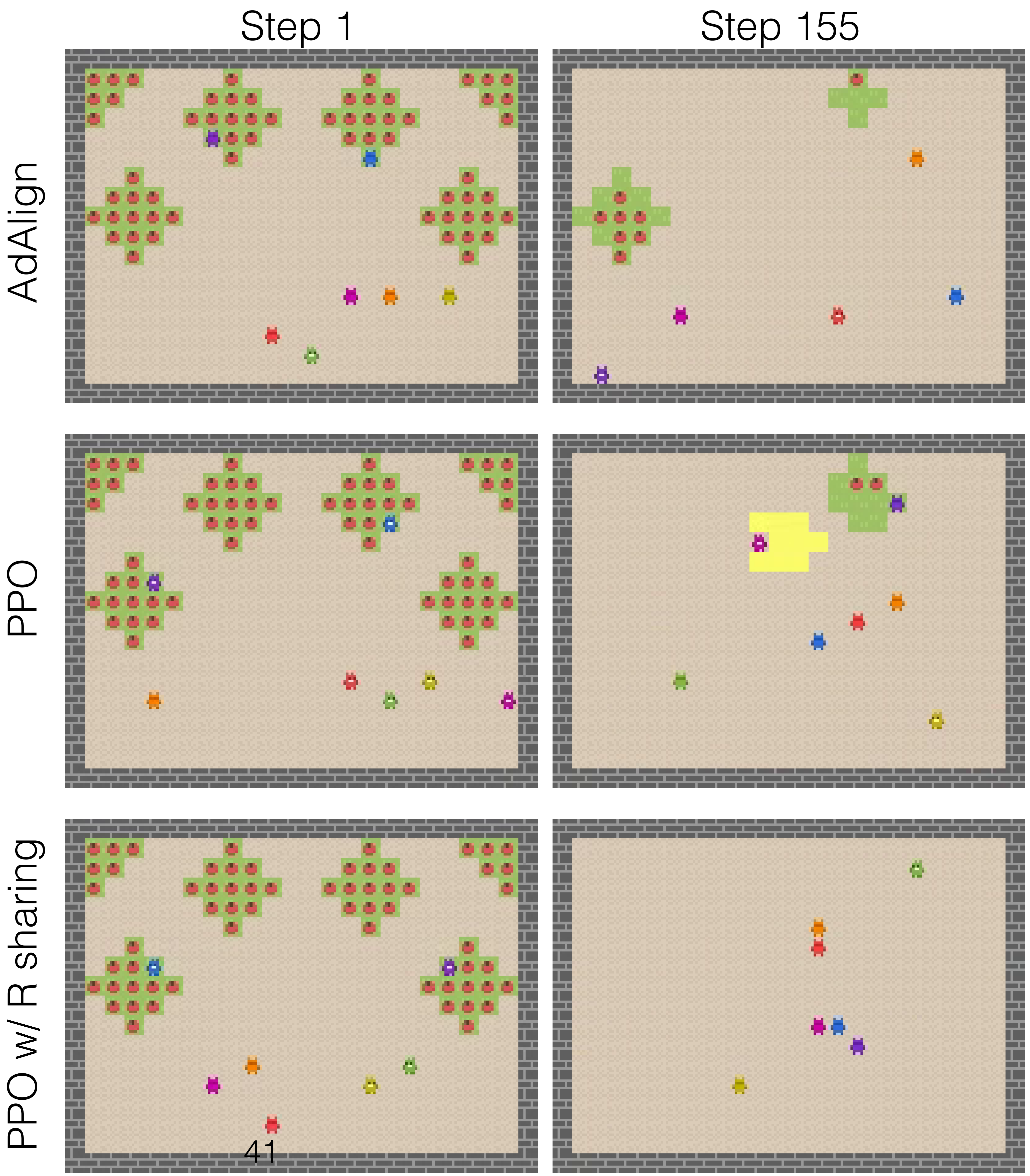
Melting Pot — Common Harvest Game

- 7 players game
- Evaluation protocol
  - 5 [method] players
  - 2 greedy heuristic players
- Advantage Alignment improves resource sustainability.



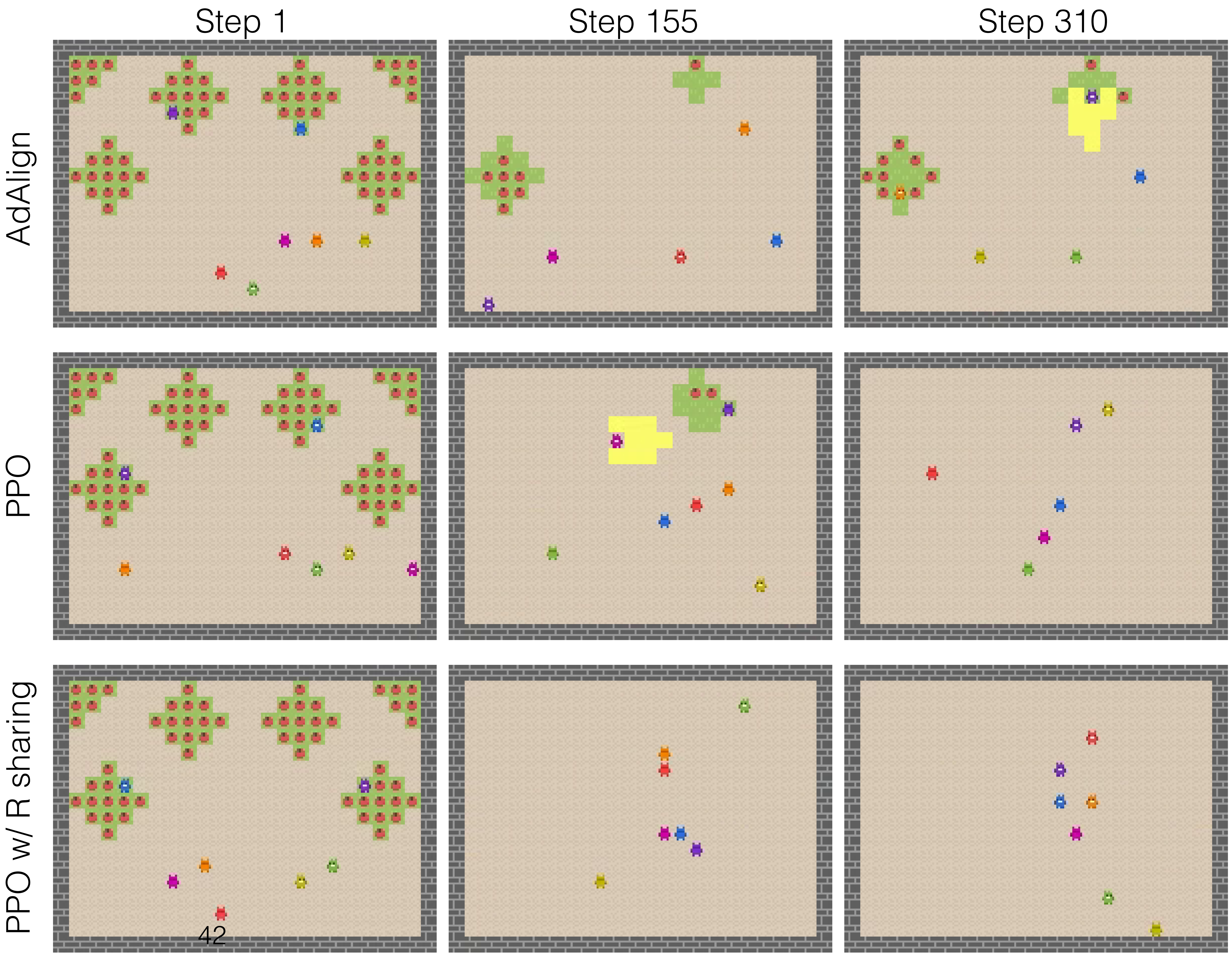
Melting Pot — Common Harvest Game

- 7 players game
- Evaluation protocol
  - 5 [method] players
  - 2 greedy heuristic players
- Advantage Alignment improves resource sustainability.



Melting Pot — Common Harvest Game

- 7 players game
- Evaluation protocol
  - 5 `[method]` players
  - 2 greedy heuristic players
- Advantage Alignment improves resource sustainability.

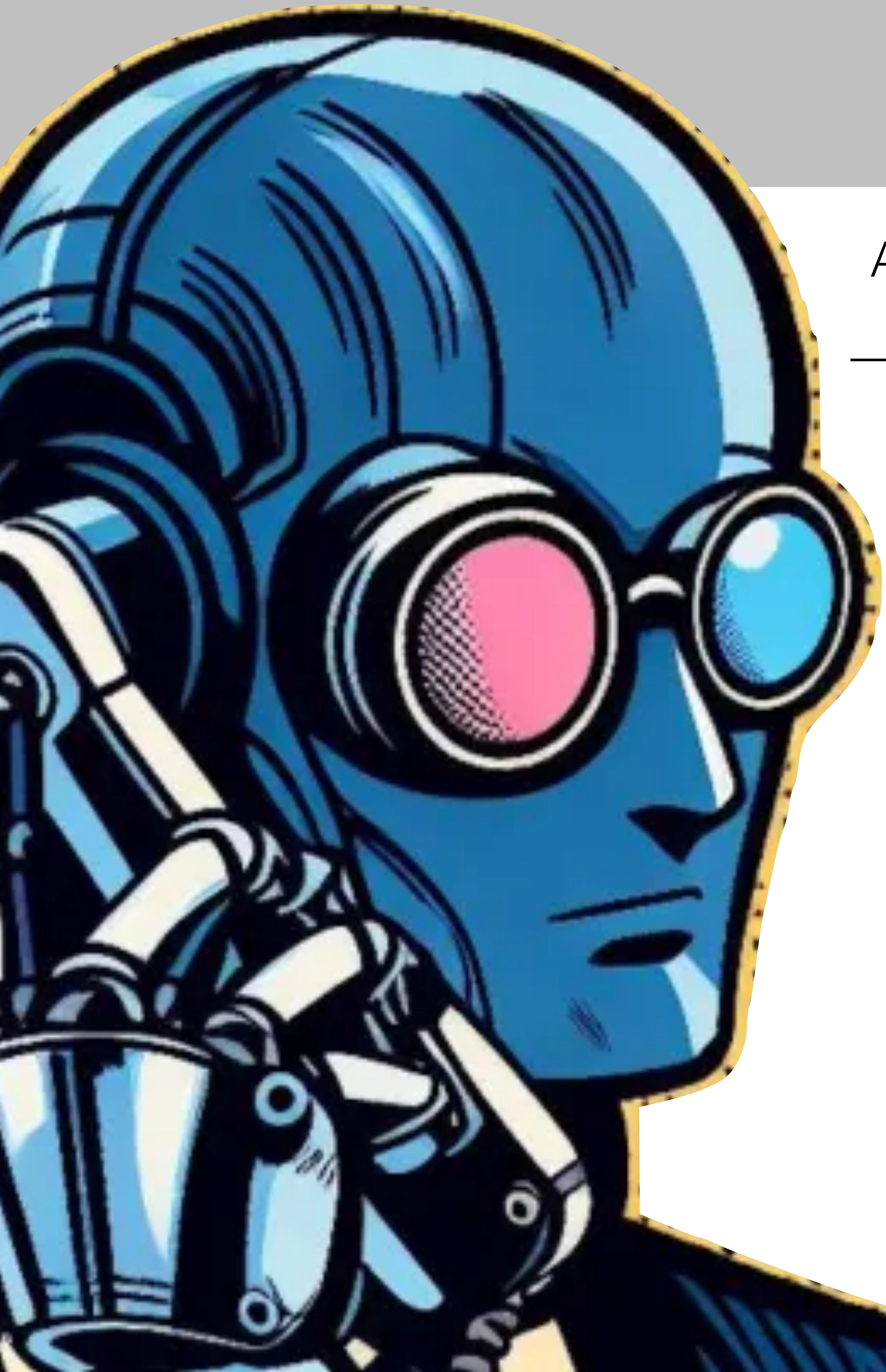


Melting Pot — Common Harvest Game

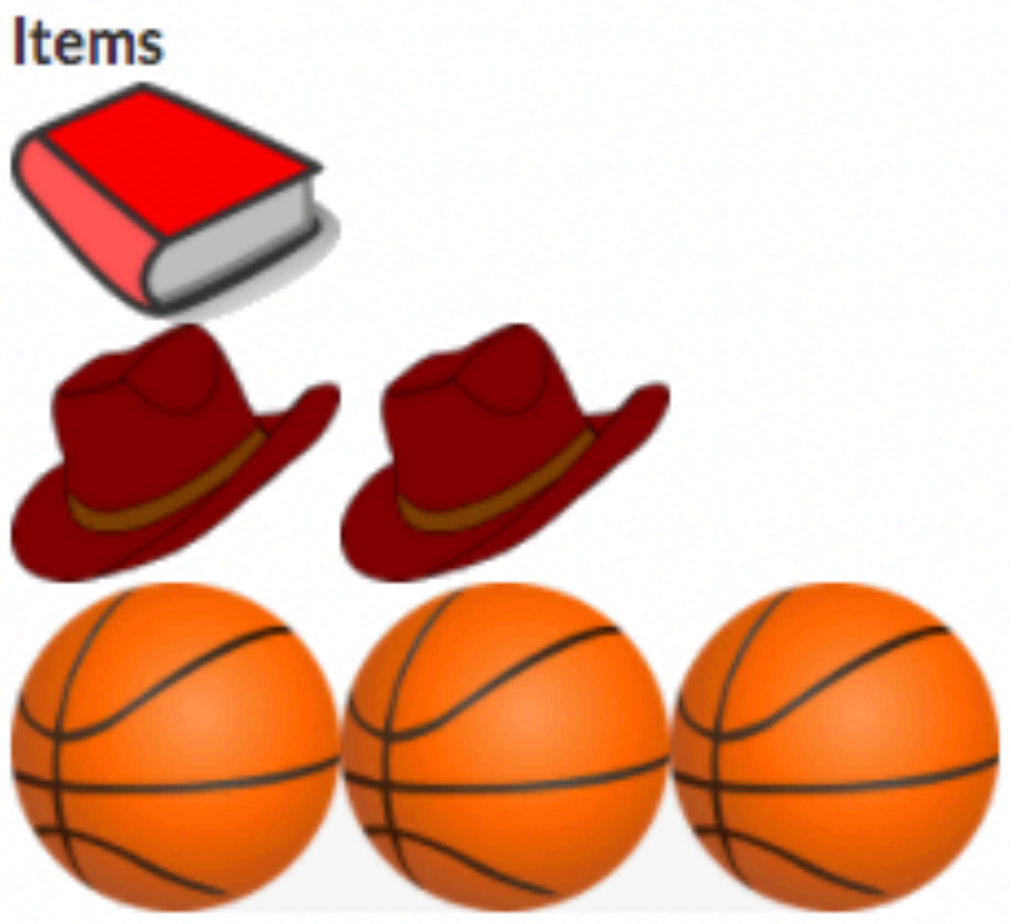
	adalign	ppo	ppo_p	exploiter	acb	vmpo	opre	acb_p	opre_p	random
scenario_0	1.78	1.15	0.33	0.91	0.87	0.93	0.84	0.95	0.53	0.00
scenario_1	1.48	0.74	0.45	0.76	0.80	0.85	0.77	0.94	0.52	0.00
average	1.63	0.94	0.39	0.83	0.83	0.89	0.81	0.94	0.52	0.00

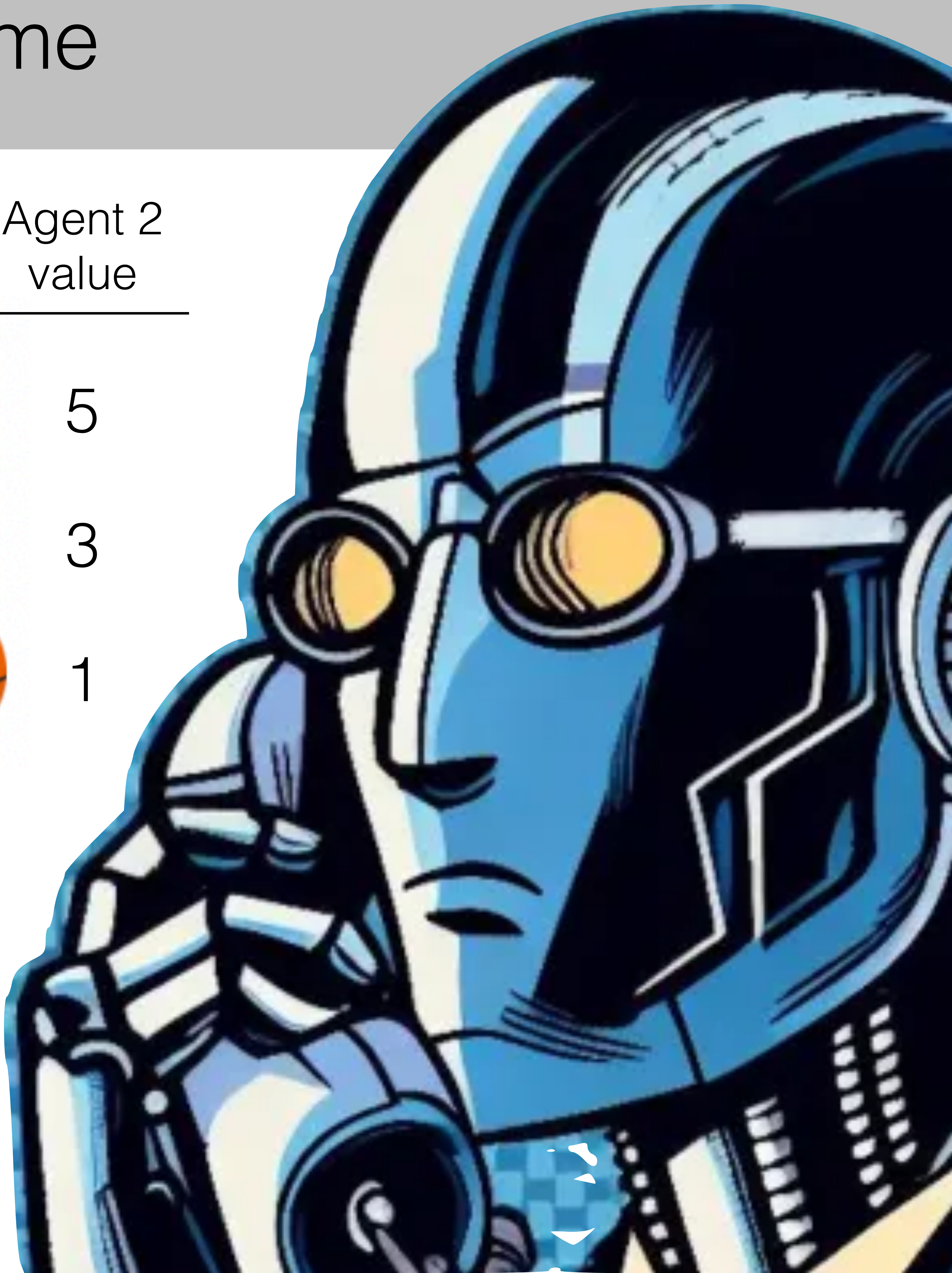
- scenario_0: Agents are visited by two invaders who harvest and zap unsustainably.
- scenario_1: Agents are visited by two invaders who harvest unsustainably.

Negotiation Game

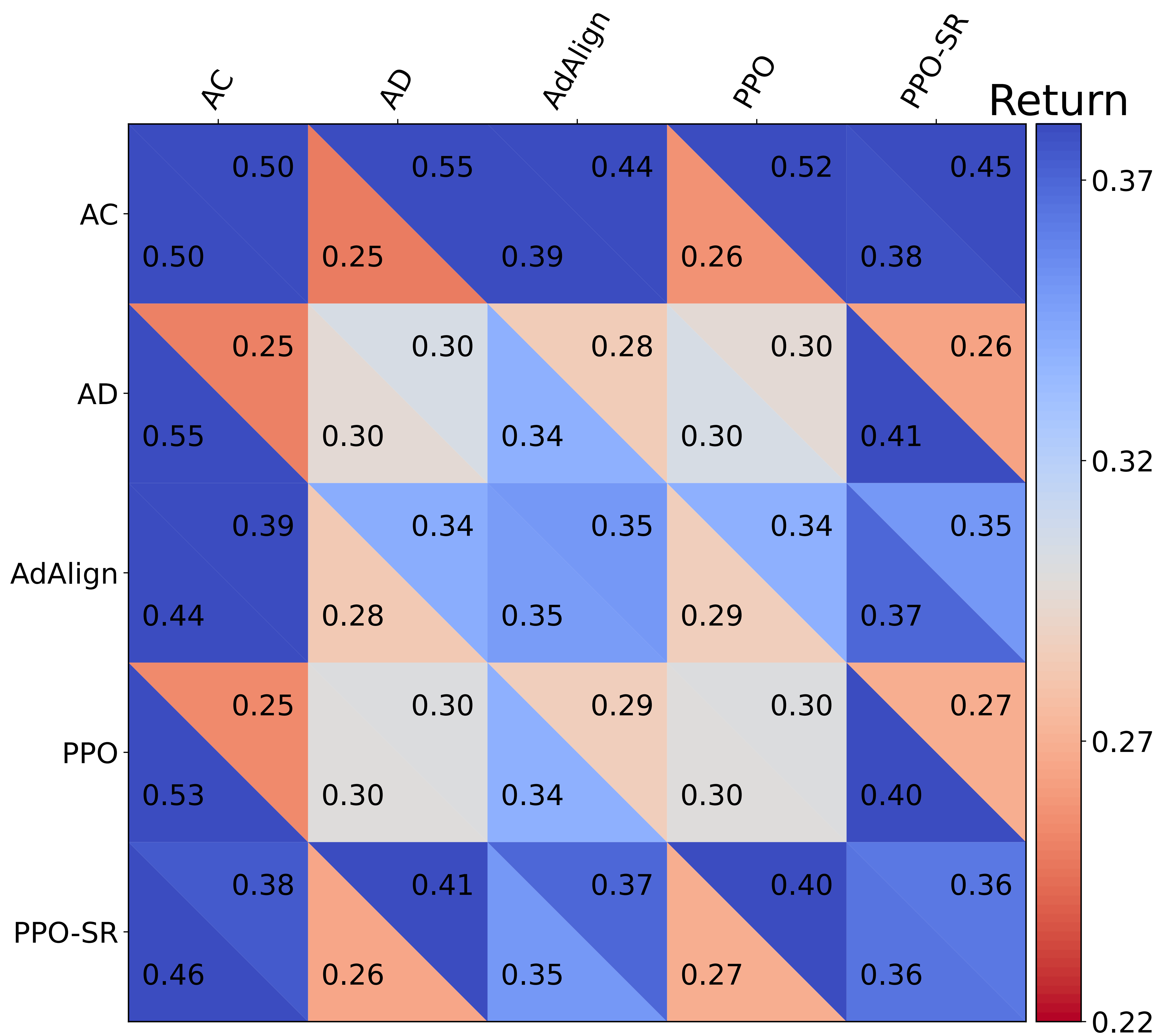


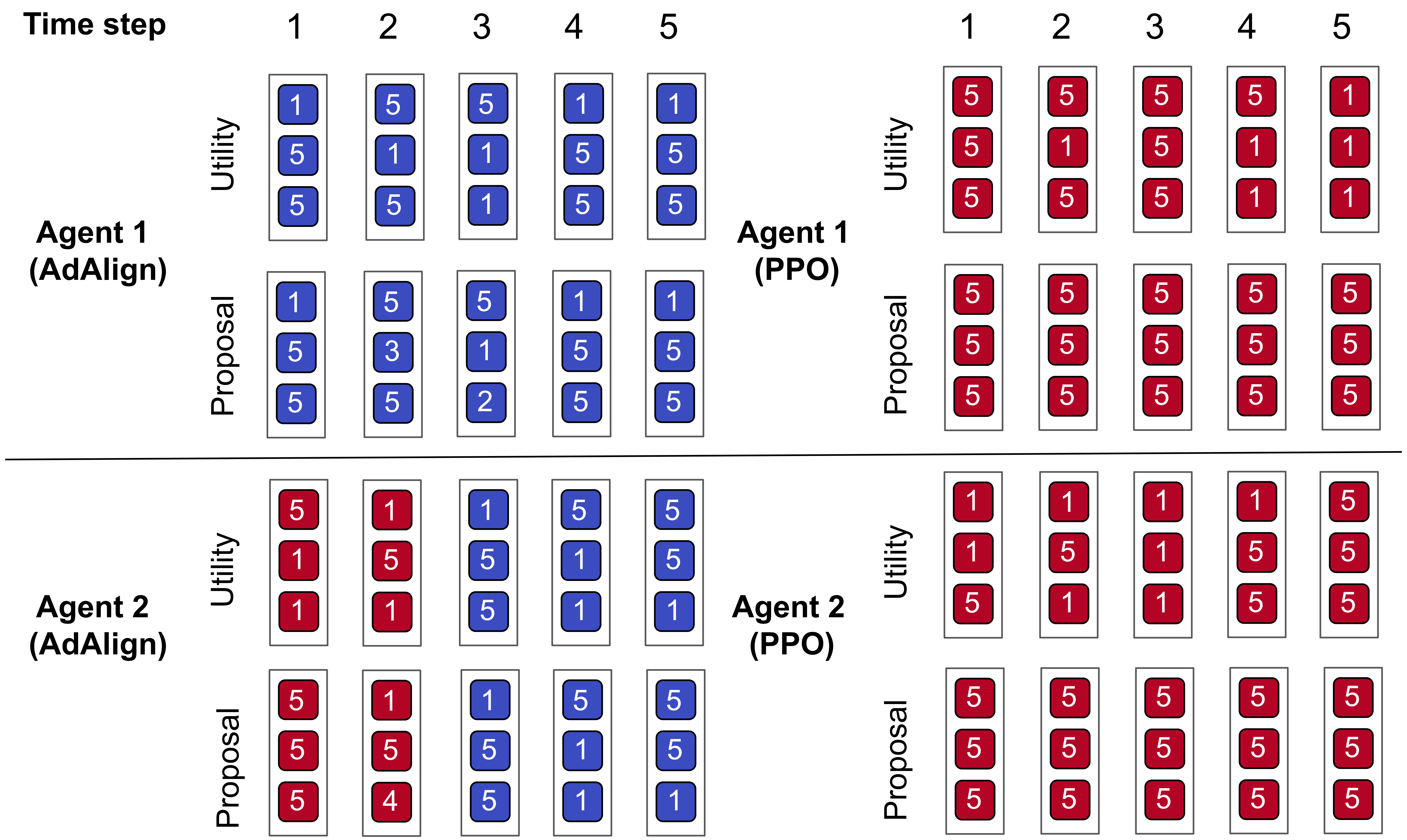
Agent 1 value	Objects & Number	Agent 2 value
1		5
3		3
5		1





Advantage Alignment: Iterated Negotiation Game





Advantage Alignment: Limitations and future work

- The assumption that the other players act accordingly to an inner action-value (Q) function, prevents it from shaping opponents that do not follow this.

- Currently working on improving performance on the Melting Pot suite of tasks — adding representation learning and other RL tricks.

- Exploring applications in negotiations and LLMs.


Image Credit: VentureBeat made with Bing Image Creator
Thank You!
Any Question