Caltech

Multi-Agent Reinforcement Learning

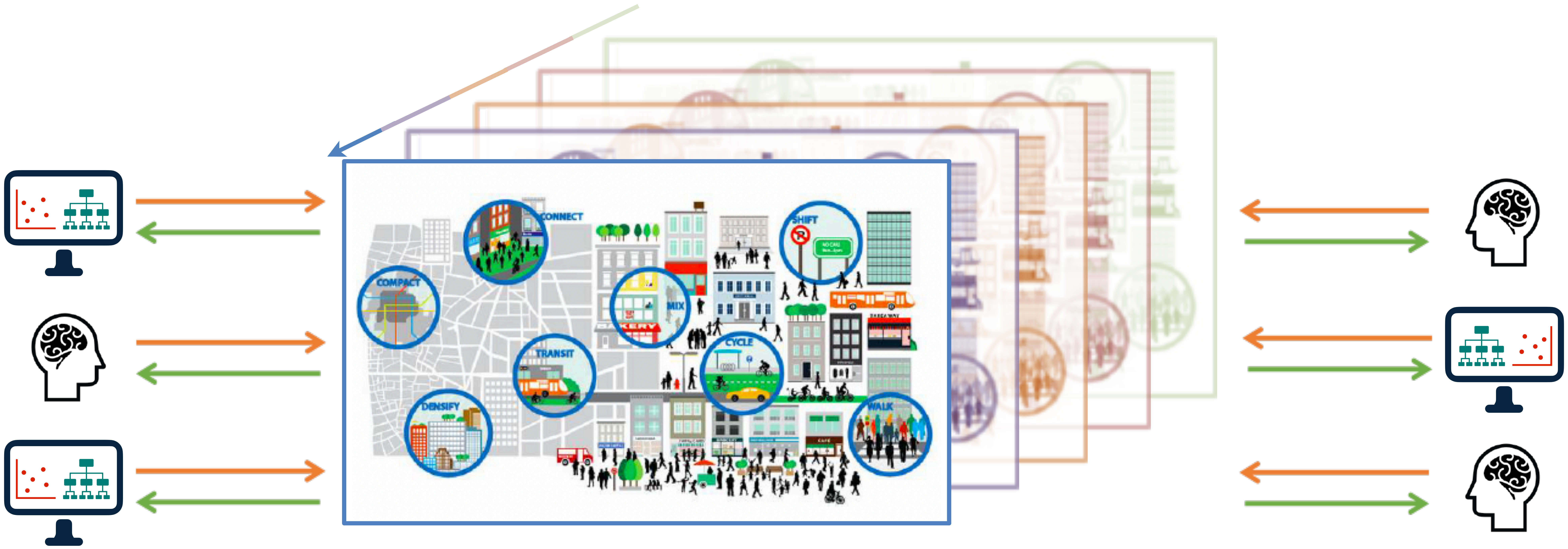
Theory, Algorithms, and Future Directions

Part II.

Eric Mazumdar

Computing + Mathematical Sciences and Economics

Challenges: Strategic interactions vastly complicate the task of learning



Opportunities: Require a careful rethinking of algorithm design.

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Opportunities: Require a careful rethinking of algorithm design.

Though it is less well understood, we can build on foundations from game theory and reinforcement learning to explore and design new algorithmic principles.

Main Question:

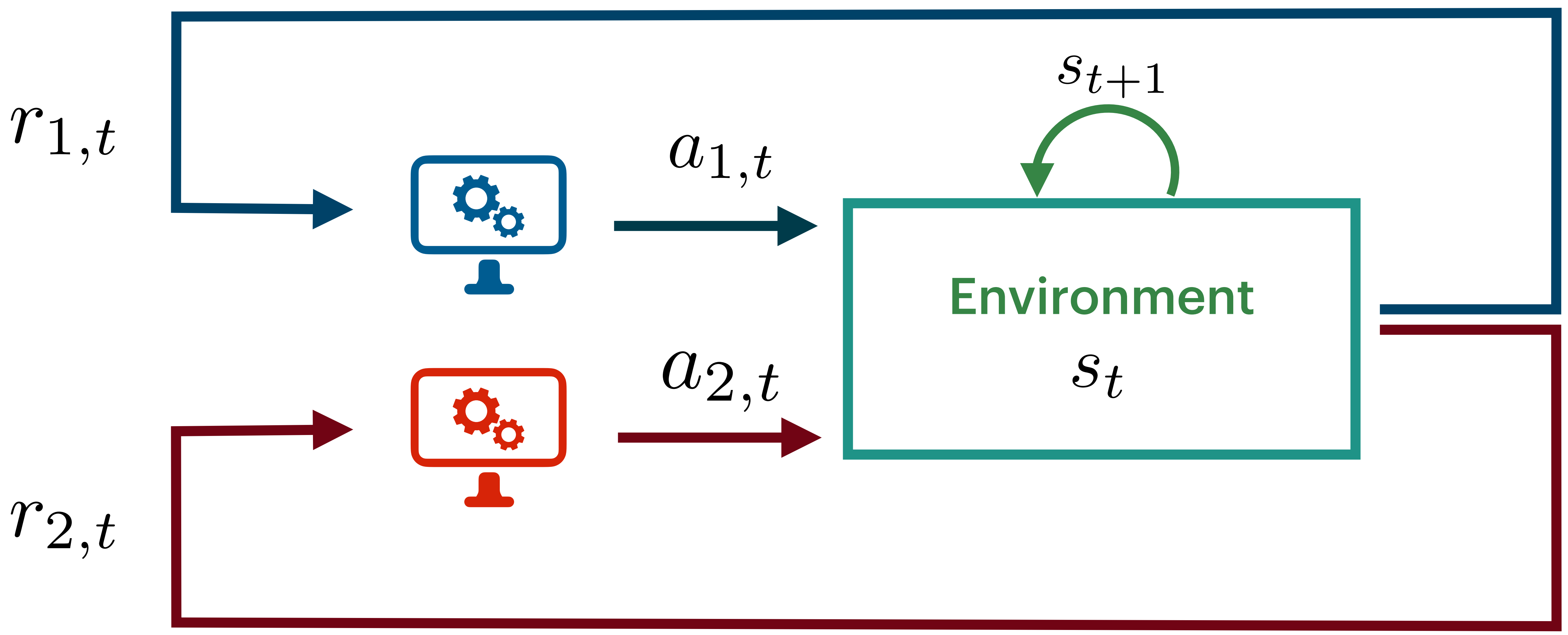
How do we design *principled algorithms* for multi-agent problems?

We will focus on theoretical foundations.

Markov Games

Generalization of a Markov Decision Process introduced by Shapley (1953)

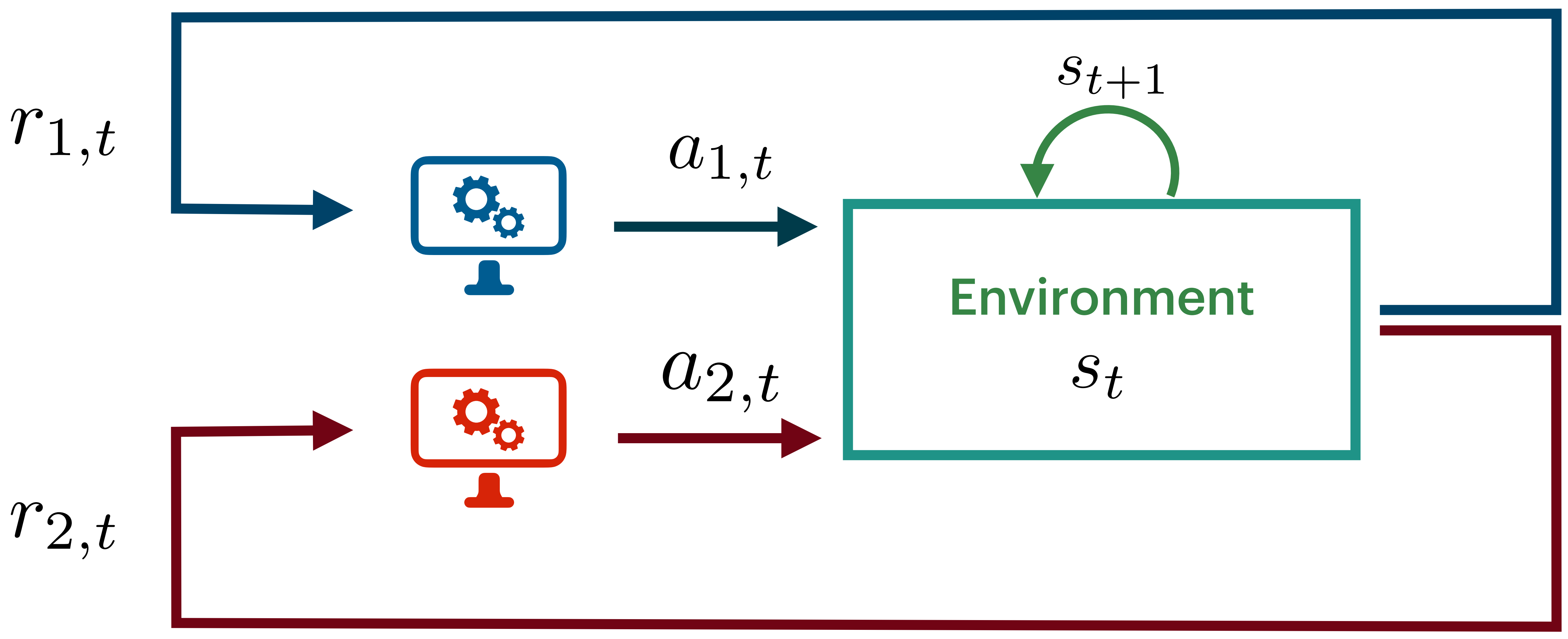
- Action Spaces: $\mathcal{A}_1, \ldots, \mathcal{A}_n, \quad \mathcal{A} = \prod_{i=1}^{n} \mathcal{A}_i$
- State Spaces: $\mathcal{S}$
- Dynamics: $P(s' \mid s, a_1, \ldots, a_n)$
- Reward functions: $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
- Horizon: H or ∞
- Initial state distribution: ρ_0



Markov Games

Interaction Protocol:

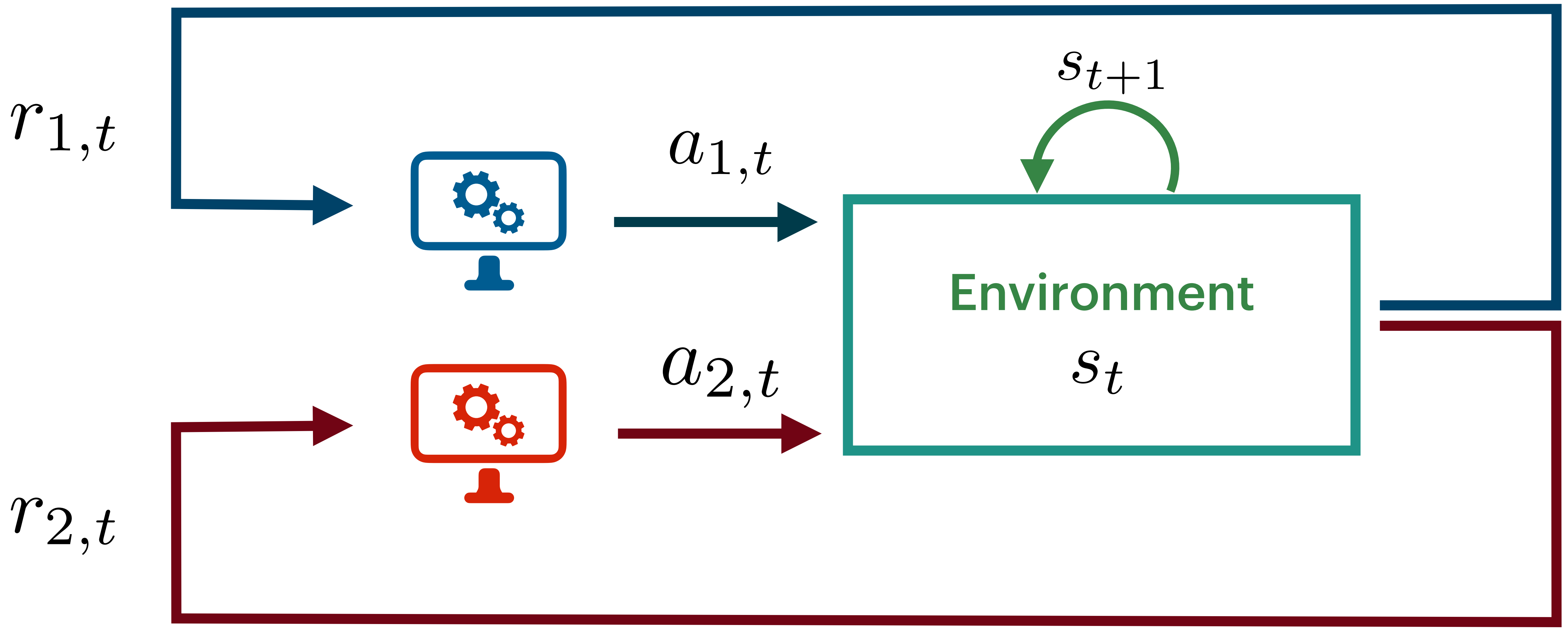
- Environment samples initial state: $s_0 \sim \rho_0$
- For step t=0,1,2,...
  - Each agent plays an action $a_{i,t}$ *simultaneously* $a_t = (a_{1,t}, ...a_{n,t})$
  - Agents receive their immediate reward: $r_{i,t} = R_i(s_t, a_t)$
  - Environment transitions to the next state: $s_{t+1} \sim P(\cdot \mid s_t, a_t)$



Markov Games

In this overview we will focus mainly on fully observable, tabular Markov Games

Fully observable: joint actions and states observed by all agents
Tabular: Finite State and Action Spaces



Policies

Players strategy spaces are spaces of policies (distributions over actions):

General Policy: Depends on the entire history of play:

$$\Pi_i = \{\pi_i : (\mathcal{S}, \times \mathcal{A})^{t-1} \times \mathcal{S} \rightarrow \Delta_{\mathcal{A}_i}\}$$

Policies

Players strategy spaces are spaces of policies (distributions over actions):

General Policy: Depends on the entire history of play:

$$\Pi_i = \{\pi_i : (\mathcal{S}, \times \mathcal{A})^{t-1} \times \mathcal{S} \to \Delta_{\mathcal{A}_i}\}$$

Non-stationary Markov Policy: Depends only on the current state and time

$$\Pi_i = \{\pi_i : \mathbb{R}_+ \times \mathcal{S} \to \Delta_{\mathcal{A}_i}\}$$

Policies

Players strategy spaces are spaces of policies (distributions over actions):

General Policy: Depends on the entire history of play:

$$\Pi_i = \{\pi_i : (\mathcal{S}, \times \mathcal{A})^{t-1} \times \mathcal{S} \to \Delta_{\mathcal{A}_i}\}$$

Non-stationary Markov Policy: Depends only on the current state and time

$$\Pi_i = \{\pi_i : \mathbb{R}_+ \times \mathcal{S} \to \Delta_{\mathcal{A}_i}\}$$

Stationary Markov Policy: Depends only on the current state

$$\Pi_i = \{\pi_i : \mathcal{S} \to \Delta_{\mathcal{A}_i}\}$$

Utilities

To evaluate the quality of their strategies, we assume that players seek to maximize their cumulative reward:

Finite Horizon:

$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{H} r_{i,t}\right]$$

Utility of agent $\boldsymbol{i}$ depends on the policy of agent $\boldsymbol{i}$ as well as the policies of all other agents $\boldsymbol{\pi_{-i}}$

Utilities

To evaluate the quality of their strategies, we assume that players seek to maximize their cumulative reward:

Finite Horizon:

$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{H} r_{i,t}\right]$$

Utility of agent $\boldsymbol{i}$ depends on the policy of agent $\boldsymbol{i}$ as well as the policies of all other agents $\boldsymbol{\pi_{-i}}$

Infinite Horizon:

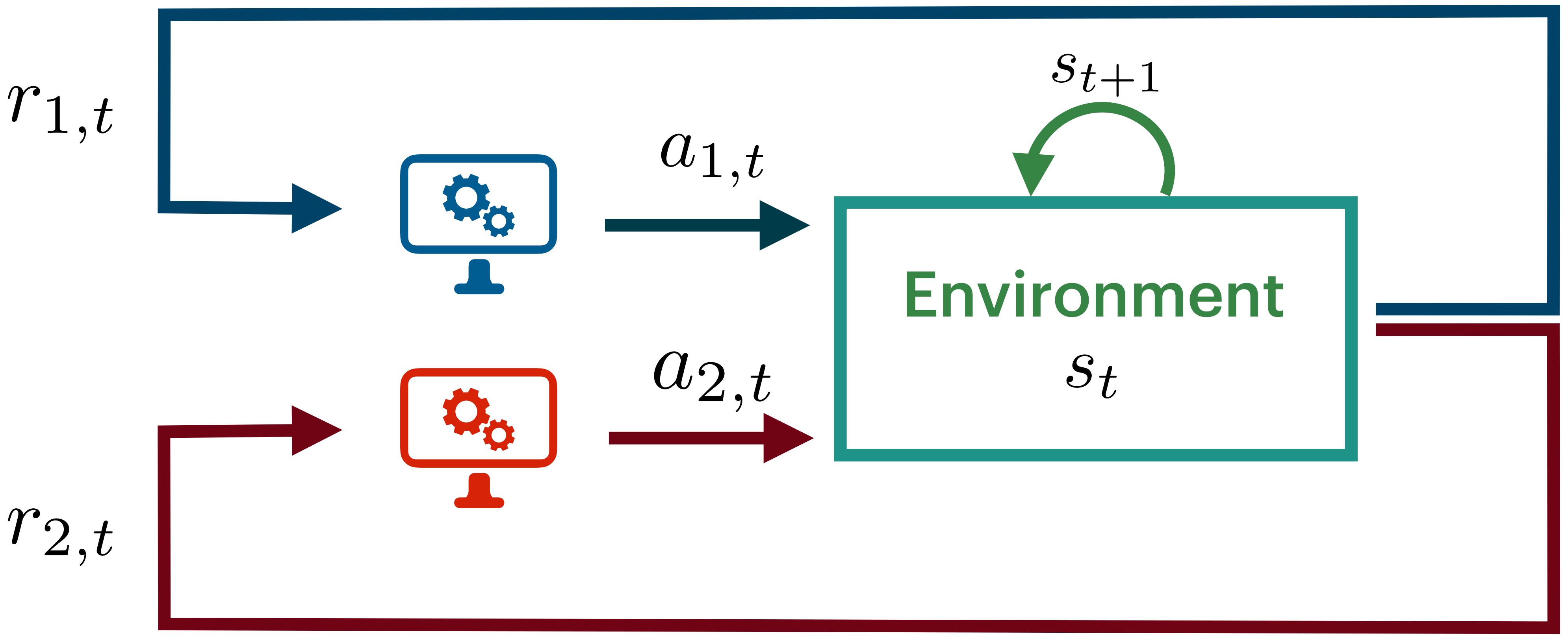
$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_{i,t}\right]$$

Utility is discounted cumulative reward
(each player with their own discount factor).

Recap: Markov Games Setup

- Action Spaces: $\mathcal{A}_1, \ldots, \mathcal{A}_n, \quad \mathcal{A} = \prod_{i=1}^{n} \mathcal{A}_i$
- State Spaces: $\mathcal{S}$
- Dynamics: $P(s' \mid s, a_1, \ldots, a_n)$
- Reward functions: $R_i : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$
- Horizon: $H \text{ or } \infty$
- Initial state distribution: ρ_0

$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0} \left[\sum_{t=0}^{\infty} \gamma^t r_{i,t} \right]$$



Special Cases:

- Single-agent RL
- Two-player Zero-sum $(R_1 = -R_2)$
- Cooperative $(R_i = R_j \quad \forall i, j)$

Nash Equilibrium

What are good outcomes for Markov Games?

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Pi_i$

Nash Equilibrium

What are good outcomes for Markov Games?

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Pi_i$

- **Each player is at a best-response -> no incentive to unilaterally deviate.**
- **Always guaranteed to exist in Markov policies in Markov games.**
  - In non-stationary Markov policies for *finite horizon* games.
  - In *stationary* Markov policies for *infinite horizon* games.

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms
   3. The role of function approximation

3. **Further directions**
   1. Scalable algorithms for zero-sum games
   2. New equilibrium concepts

A Road Map

1. Normal-form & concave games: equilibrium computation and learning in games

Takeaway: Equilibrium computation (even in normal-form games) is hard.

- Coupling between agents gives rise to non-stationarity and complex dynamics.
- No-regret learning and variational inequality perspectives can help for algorithm design with convergence to game theoretically meaningful solutions e.g., CCE.

2. Algorithmic structures in Multi-Agent Reinforcement Learning

i. Policy-gradient algorithms in games

ii. Value-based algorithms

iii. The role of function approximation

3. Further directions

i. Scalable algorithms for zero-sum games

ii. New equilibrium concepts

A Road Map

1. Normal-form & concave games: equilibrium computation and learning in games

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
    1. Policy-gradient algorithms in games
    2. Value-based algorithms
    3. The role of function approximation

3. **Further directions**
    1. Scalable algorithms for zero-sum games
    2. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games
      - No convergence guarantees or no-regret algorithms in general!
      - Zero-sum games (and similar) allow for some positive results.

   ii. Value-based algorithms

   iii. The role of function approximation

3. **Further directions**

   i. Scalable algorithms for zero-sum games

   ii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

Value-Based Approaches

We've seen that no-regret and policy gradient methods don't generally allow us to develop principled algorithms for learning in Markov Games.

- They are agnostic to the underlying structure of the problem.

Value-Based Approaches

We’ve seen that no-regret and policy gradient methods don’t generally allow us to develop principled algorithms for learning in Markov Games.

- They are agnostic to the underlying structure of the problem.
- RL (and thus MARL) is fundamentally an *optimal control* problem

Value-Based Approaches

We've seen that no-regret and policy gradient methods don't generally allow us to develop principled algorithms for learning in Markov Games.

- They are agnostic to the underlying structure of the problem.
- RL (and thus MARL) is fundamentally an *optimal control* problem
  - Are there dynamic programming (value-based) approaches to MARL?
    - Let's start in ***infinite horizon*** Markov Games

$$U(\pi) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Single-Agent Warmup

State-Action Value Function (Q-function) in Single-Agent RL

Cumulative reward starting from state s and action a, and applying policy π hereafter

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^{t} R_i(s_t, a_t) \middle| s_0 = s, a_0 = a\right]$$

Single-Agent Warmup

State-Action Value Function (Q-function) in Single-Agent RL

Cumulative reward starting from state s and action a, and applying policy π hereafter

$$Q^\pi(s,a) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty}\gamma^t R_i(s_t,a_t)\middle| s_0 = s, a_0 = a\right]$$

Our infinite horizon objective:

$$U(\pi) = \mathbb{E}_{\pi,P,\rho_0}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$$

Equivalent formulation of infinite-horizon problem:

$$U(\pi) = \mathbb{E}_{s\sim d^\pi, a\sim\pi(s)}[Q^\pi(s,a)]\ ; \quad d^\pi = (1-\gamma_i)\sum_{t=0}^{\infty}\gamma_i^t Pr(s_t = s|\pi)$$

Discounted state visitation frequency under $\boldsymbol{\pi}$

Single-Agent Warmup

State-Action Value Function (Q-function) in Single-Agent RL

Cumulative reward starting from state s and action a, and applying policy π hereafter

$$Q^\pi(s,a) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t R_i(s_t,a_t) \middle| s_0 = s, a_0 = a\right]$$

Our infinite horizon objective:

$$U(\pi) = \mathbb{E}_{\pi,P,\rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Equivalent formulation of infinite-horizon problem:

$$U(\pi) = \mathbb{E}_{s\sim d^\pi, a\sim\pi(s)}[Q^\pi(s,a)]\ ; \quad d^\pi = (1-\gamma_i)\sum_{t=0}^{\infty} \gamma_i^t Pr(s_t = s|\pi)$$

Bellman's Principle of Optimality says that optimal policy must satisfy:

$$Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$$

Policy can be recovered as $\pi^*(s) = \arg\max_a Q^*(s,a)$

Single-Agent Warmup

State-Action Value Function (Q-function) in Single-Agent RL

Cumulative reward starting from state s and action a, and applying policy π hereafter

$$Q^\pi(s,a) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t R_i(s_t,a_t)\middle| s_0 = s, a_0 = a\right]$$

Our infinite horizon objective:

$$U(\pi) = \mathbb{E}_{\pi,P,\rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

Equivalent formulation of infinite-horizon problem:

$$U(\pi) = \mathbb{E}_{s\sim d^\pi, a\sim\pi(s)}[Q^\pi(s,a)]\ ; \quad d^\pi = (1-\gamma_i)\sum_{t=0}^{\infty}\gamma_i^t Pr(s_t = s|\pi)$$

Bellman's Principle of Optimality says that optimal policy must satisfy:

$$Q^*(s,a) = R(s,a) + \gamma\mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$$

So finding Q^* is enough to solve the RL problem!

Single-Agent Q-learning

Q-learning in Single-Agent RL

So finding $\boldsymbol{Q^*}$ is enough to solve the RL problem: $Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$

Single-Agent Q-learning

Q-learning in Single-Agent RL

So finding $\boldsymbol{Q^*}$ is enough to solve the RL problem: $Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$

Observe that $\boldsymbol{Q^*}$ is the fixed point of this mapping :

$$T : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \qquad TQ(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[\max_{a'} Q(s',a')\right]$$

Single-Agent Q-learning

Q-learning in Single-Agent RL

So finding Q^* is enough to solve the RL problem: $Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$

Observe that Q^* is the fixed point of this mapping :

$$T : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \qquad TQ(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[\max_{a'} Q(s',a')\right]$$

T can be shown to be a ***contraction mapping*** in the L_∞ norm: $\|TQ - TQ'\|_\infty \leq \gamma \|Q - Q'\|_\infty$

Single-Agent Q-learning

Q-learning in Single-Agent RL

So finding $\boldsymbol{Q^*}$ is enough to solve the RL problem: $Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$

Observe that $\boldsymbol{Q^*}$ is the fixed point of this mapping :

$$T : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \qquad TQ(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[\max_{a'} Q(s',a')\right]$$

T can be shown to be a ***contraction mapping*** in the L_∞ norm: $\|TQ - TQ'\|_\infty \leq \gamma \|Q - Q'\|_\infty$

Which immediately suggests a ***Q-learning*** algorithm (Watkins 1989):

$$\forall (s,a) \;\; \text{do:} \;\; Q_{t+1}(s,a) = (1-\alpha) Q_t(s,a) + \alpha TQ(s,a)$$

Single-Agent Q-learning

Q-learning in Single-Agent RL

So finding Q^* is enough to solve the RL problem: $Q^*(s,a) = R(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_{a'} Q^*(s',a')\right]$

Observe that Q^* is the fixed point of this mapping :

$$T : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \qquad TQ(s,a) = R(s,a) + \gamma \mathbb{E}_{s'}\left[\max_{a'} Q(s',a')\right]$$

T can be shown to be a ***contraction mapping*** in the L_∞ norm: $\|TQ - TQ'\|_\infty \leq \gamma \|Q - Q'\|_\infty$

Which immediately suggests a ***Q-learning*** algorithm (Watkins 1989):

$$\forall (s,a) \ \ \text{do:} \ Q_{t+1}(s,a) = (1-\alpha) Q_t(s,a) + \alpha TQ(s,a)$$

Can perform ***stochastic approximation*** of theses dynamics when using a stochastic estimator of ***T***

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Classic Q-Learning

$$Q^*(s,a) = E_{s,a}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a')\right]$$

Minimax Q-Learning

$$Q^*(s,a_1,a_2) = R(s,a_1,a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Introduced by Littman (1994)

Building off of work by Shapley (1959) who showed the existence of Nash eq. In such games by using a contraction mapping on a state-value function.

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Classic Q-Learning

$$Q^*(s,a) = E_{s,a}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a')\right]$$

Minimax Q-Learning

$$Q^*(s,a_1,a_2) = R(s,a_1,a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

$$\pi^*(s) = \ \text{Nash}\left(Q(s), -Q(s)\right)$$

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1} \max_{\pi_2} \ \pi_1^T Q^*(s')\pi_2\right]$$

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Like single-agent RL, we can see this as the fixed point of a map $\mathcal{H}$

$$\mathcal{H}Q(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q(s')\pi_2\right]$$

Observe: $Q^* = \mathcal{H}(Q^*)$

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1} \max_{\pi_2} \ \pi_1^T Q^*(s')\pi_2\right]$$

Like single-agent RL, we can see this as the fixed point of a map $\mathcal{H}$

$$\mathcal{H}Q(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1} \max_{\pi_2} \ \pi_1^T Q(s')\pi_2\right]$$

Like in the single-agent setting, this is a contraction mapping in the ℓ_∞ norm

$$\|\mathcal{H}Q - \mathcal{H}Q'\|_\infty \leq \gamma\|Q - Q'\|_\infty$$

Value-Base Methods in Zero-Sum Games

Can we derive Q-learning algorithms for Infinite-Horizon Zero-Sum Markov Games?

Minimax Q-Learning

$$Q^*(s,a_1,a_2) = R(s,a_1,a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Like single-agent RL, we can see this as the fixed point of a map $\mathcal{H}$

$$\mathcal{H}Q(s,a_1,a_2) = R(s,a_1,a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q(s')\pi_2\right]$$

Like in the single-agent setting, this is a contraction mapping in the ℓ_∞ norm

Which suggests the following ***minimax Q-learning algorithm:***

$$\forall(s,a)\ \ \text{do: } Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha\mathcal{H}Q(s,a)$$

Requires solving a matrix game at every step!

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Centralized computation of Nash in Zero-Sum Markov Games

- Initializes $Q_0, \pi_{1,0}, \pi_{2,0}$
- For step t=0,1,2,...
  - For each state ***s*** :
    - Update policy using no-regret learning on matrix game:

$$\max_{\pi_1(s)} \min_{\pi_2(s)} \; \pi_1(s)^T Q_t(s) \pi_2(s)$$

e.g.,

$$\pi_{1,t+1}(s) = \mathcal{P}_{\Delta_1}\left(\pi_{1,t} + \alpha Q_t(s)\pi_{2,t}(s)\right)$$
$$\pi_{2,t+1}(s) = \mathcal{P}_{\Delta_2}\left(\pi_{1,t} - \alpha Q_t^T(s)\pi_{1,t}(s)\right)$$

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Centralized computation of Nash in Zero-Sum Markov Games

- Initializes $Q_0, \pi_{1,0}, \pi_{2,0}$
- For step t=0,1,2,…
  - For each state **s** :
    - Update policy using no-regret learning on matrix game:

$$\max_{\pi_1(s)} \min_{\pi_2(s)} \; \pi_1(s)^T Q_t(s) \pi_2(s)$$

e.g.,

$$\pi_{1,t+1}(s) = \mathcal{P}_{\Delta_1}\left(\pi_{1,t} + \alpha Q_t(s)\pi_{2,t}(s)\right)$$
$$\pi_{2,t+1}(s) = \mathcal{P}_{\Delta_2}\left(\pi_{1,t} - \alpha Q_t^T(s)\pi_{1,t}(s)\right)$$

Fast timescale $\alpha >> \beta$

Converges quickly to minimax value

- Update Q-function:

$$\forall (s,a) \;\; \text{do:} \;\; Q_{t+1}(s,a) = (1-\beta)Q_t(s,a) + \beta\left(R(s,a_1,a_2) + \gamma \mathbb{E}_{s'|s,a}\left[\pi_{1,t}(s')^T Q_t(s')\pi_{2,t}(s')\right]\right)$$

Slow timescale

Update Q-function

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Centralized computation of Nash in Zero-Sum Markov Games

```
- Initializes $Q_0, \pi_{1,0}, \pi_{2,0}$
- For step t=0,1,2,...
        - For each state **s** :
            - Update policy using no-regret learning on matrix game:
                  $$\max_{\pi_1(s)} \min_{\pi_2(s)} \; \pi_1(s)^T Q_t(s) \pi_2(s)$$
              e.g.,  $\pi_{1,t+1}(s) = \mathcal{P}_{\Delta_1}\left(\pi_{1,t} + \alpha Q_t(s)\pi_{2,t}(s)\right)$
                     $\pi_{2,t+1}(s) = \mathcal{P}_{\Delta_2}\left(\pi_{1,t} - \alpha Q_t^T(s)\pi_{1,t}(s)\right)$
            - Update Q-function:
              $\forall(s,a)$ do: $Q_{t+1}(s,a) = (1-\beta)Q_t(s,a) + \beta\left(R(s,a_1,a_2) + \gamma\mathbb{E}_{s'|s,a}\left[\pi_{1,t}(s')^T Q_t(s')\pi_{2,t}(s')\right]\right)$
```

Sanity Check

Suppose the fast timescale has converged: this is exactly minimax-Q

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Centralized computation of Nash in Zero-Sum Markov Games

- Initializes $Q_0, \pi_{1,0}, \pi_{2,0}$
- For step t=0,1,2,…
    - For each state **s** :
        - Update policy using no-regret learning on matrix game:

$$\max_{\pi_1(s)} \min_{\pi_2(s)} \; \pi_1(s)^T Q_t(s) \pi_2(s)$$

        - Update Q-function:

$$\forall (s,a) \;\; \text{do:} \;\; Q_{t+1}(s, a_2, a_2) = (1-\beta) Q_t(s, a_1, a_2) + \beta \left(R(s, a_1, a_2) + \gamma \mathbb{E}_{s' | s, a_1, a_2} \left[\min_{\pi_2} \max_{\pi_1} \pi_1^T Q_t(s') \pi_2 \right] \right)$$

Sanity Check

Suppose the fast timescale has converged: this is exactly minimax-Q

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale***
  (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Many papers use this algorithmic structure:

E.g., Use no-regret learning subroutines & analyze full-information algorithms:
[Zhang et al. 2022]
[Cen et al. 2022]
[Cen et al. 2023]

E.g., Use no-regret learning subroutines & analyze sample based algorithms:
[Cai et al. 2023]
[Chen et al. 2023]

General-Algorithmic Ideas for Zero-Sum Markov Games

Unifying idea: Timescale separation

- Solve per-state Matrix Games on a ***fast timescale*** (using e.g., no-regret learning, extra gradient, proximal methods)
- Use solutions in minimax Q-learning on a ***slow timescale***

Many papers use this algorithmic structure:

E.g., Use no-regret learning subroutines & analyze full-information algorithms:
[Zhang et al. 2022]
[Cen et al. 2022]
[Cen et al. 2023]

E.g., Use no-regret learning subroutines & analyze sample based algorithms:
[Cai et al. 2023]
[Chen et al. 2023]

Still an ongoing area of research!

Beyond Zero-Sum Markov Games

Do these ideas extend beyond zero-sum games in infinite horizon Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1} \max_{\pi_2} \ \pi_1^T Q^*(s')\pi_2\right]$$

Beyond Zero-Sum Markov Games

Do these ideas extend beyond zero-sum games in infinite horizon Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Nash Q-Learning [Hu & Wellman 2003]

$$Q_i^*(s, a_1, ..., a_n) = E_{s,a}\left[R_i(s, a_1, ..., a_n) + \gamma_i \mathrm{Nash}_i(Q_1^*(s'), ..., Q_n^*(s'))\right]$$

Each agent keeps track of their Q-function

This is the value at Nash given the state-dependent matrix game:

$$\mathbb{E}_{a\sim\pi^*}[Q_i(s,a)] \quad \text{where} \quad \pi^* = \text{Nash Eq}$$

Beyond Zero-Sum Markov Games

Do these ideas extend beyond zero-sum games in infinite horizon Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Nash Q-Learning [Hu & Wellman 2003]

$$Q_i^*(s, a_1, ..., a_n) = E_{s,a}\left[R_i(s, a_1, ..., a_n) + \gamma_i \mathrm{Nash}_i(Q_1^*(s'), ..., Q_n^*(s'))\right]$$

Reduces to minimax Q learning in zero-sum games

Difficulties in general: e.g., equilibrium selection (if there are multiple Nash which to choose?)

Beyond Zero-Sum Markov Games

Do these ideas extend beyond zero-sum games in infinite horizon Markov Games?

Minimax Q-Learning

$$Q^*(s, a_1, a_2) = R(s, a_1, a_2) + \gamma E_{s'|s,a_1,a_2} \left[\min_{\pi_1} \max_{\pi_2} \ \pi_1^T Q^*(s') \pi_2 \right]$$

Nash Q-Learning [Hu & Wellman 2003]

$$Q_i^*(s, a_1, ..., a_n) = E_{s,a} \left[R_i(s, a_1, ..., a_n) + \gamma_i \text{Nash}_i(Q_1^*(s'), ..., Q_n^*(s')) \right]$$

Unfortunately this is ***not a contraction mapping*** in general

This is true even replacing Nash with CCE, CE

Only known cases where you get contraction is in zero-sum\competitive games and fully cooperative

Recap:
Value-Based Approaches for ***Infinite-Horizon*** Markov Games

- In general, we have no algorithms for infinite horizon Markov games beyond highly structured cases
  - e.g., zero-sum games, cooperative games

- In infinite horizon, zero-sum Markov games we can develop algorithms based on top of the framework of *minimax Q-learning.*
  - Main Idea: Timescale separation
    - Solve matrix games in each state on fast timescales, do Q-learning on a slow timescale.

Recap:
Value-Based Approaches for ***Infinite-Horizon*** Markov Games

- In general, we have no algorithms for infinite horizon Markov games beyond highly structured cases
    - e.g., zero-sum games, cooperative games

- In infinite horizon, zero-sum Markov games we can develop algorithms based on top of the framework of *minimax Q-learning.*
    - Main Idea: Timescale separation
        - Solve matrix games in each state on fast timescales, do Q-learning on a slow timescale.

This should not be surprising given the hardness results we saw yesterday!

Recap: Value-Based Approaches for ***Infinite-Horizon*** Markov Games

- In general, we have no algorithms for infinite horizon Markov games beyond highly structured cases
  - e.g., zero-sum games, cooperative games

- In infinite horizon, zero-sum Markov games we can develop algorithms based on top of the framework of *minimax Q-learning.*
  - Main Idea: Timescale separation
    - Solve matrix games in each state on fast timescales, do Q-learning on a slow timescale.

This should not be surprising given the hardness results we saw yesterday!

Fortunately, in finite-horizon games, there is more to say

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon single-agent problems, the optimal policy can be found via dynamic programming:

Dynamic Programming In Reinforcement Learning

- Initialize $Q_{H+1} = 0$
- For step $t = H, H-1, H-2, \ldots, 1, 0$
    - For each state ***s,a*** :

$$Q_t(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_a Q_{t+1}(s',a)\right]$$

Computes the optimal (time-dependent!) Q-functions which we can use to infer optimal policy:

$$\pi^*(s,t) = \arg\max_a Q_t(s,a)$$

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon single-agent problems, the optimal policy can be found via dynamic programming:

Dynamic Programming In Reinforcement Learning

```
- Initialize $Q_{H+1} = 0$
- For step $t = H, H-1, H-2, ..., 1, 0$    ← Proceed backwards
    - For each state s,a :
        $Q_t(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[\max_a Q_{t+1}(s',a)\right]$
```

Computes the optimal (time-dependent!) Q-functions which we can use to infer optimal policy:

$$\pi^*(s,t) = \arg\max_a Q_t(s,a)$$

- No contraction mapping argument needed!

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon Markov games the Nash policy can be found via dynamic programming:

```
Nash Value Iteration
- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$
- For step $t = H, H - 1, H - 2, \ldots, 1, 0$
    - For each state agent $i$ and all **s,a** :
        $Q_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s'|s,a}\left[V_{i,t+1}(s')\right]$
    - For each agent $i$ and all **s**:
        $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \text{Nash}(Q_{1,t}(s), \ldots, Q_{n,t}(s))$
        $V_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[Q_{i,t}(s)]$
```

Solve a matrix game at each state and time at horizon

Keep track of state value function

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon Markov games the Nash policy can be found via dynamic programming:

```
Nash Value Iteration

- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$
- For step $t = H, H-1, H-2, \ldots, 1, 0$
    - For each state agent $i$ and all s,a :
          $Q_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s'|s,a}\left[V_{i,t+1}(s')\right]$
    - For each agent $i$ and all s:
          $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{Nash}(Q_{1,t}(s), \ldots, Q_{n,t}(s))$
          $V_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[Q_{i,t}(s)]$
```

Computes a Nash of a finite-horizon Markov Game (by definition)

In ***poly(H,S,A,B)*** steps given a Nash oracle

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon Markov games the Nash policy can be found via dynamic programming:

```
*Nash Value Iteration*

- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$
- For step $t = H, H-1, H-2, \ldots, 1, 0$
    - For each state agent $i$ and all **s,a** :
        $Q_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s'|s,a}\left[V_{i,t+1}(s')\right]$
    - For each agent $i$ and all **s**:
        $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \text{Nash}(Q_{1,t}(s), \ldots, Q_{n,t}(s))$
        $V_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[Q_{i,t}(s)]$
```

Computes a Nash of a finite-horizon Markov Game (by definition)

In ***poly(H,S,A,B)*** steps given a Nash oracle

Unfortunately this step is hard in general!

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon Markov games a non stationary CCE can be found via dynamic programming:

```
CCE Value Iteration

Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$
For step $t = H, H-1, H-2, \ldots, 1, 0$
    For each state agent $i$ and all s,a :
        $Q_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s'|s,a}\left[V_{i,t+1}(s')\right]$
    For each agent $i$ and all s:
        $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(Q_{1,t}(s), \ldots, Q_{n,t}(s))$
        $V_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[Q_{i,t}(s)]$
```

Can be solved via linear programming or no-regret learning

Dynamic Programming Approaches to Nash in Finite-Horizon Markov Games

- In finite-horizon Markov games a non stationary CCE can be found via dynamic programming:

CCE Value Iteration

```
- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$
- For step $t = H, H - 1, H - 2, \ldots, 1, 0$
    - For each state agent $i$ and all **s,a** :
        $Q_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s'|s,a}\left[V_{i,t+1}(s')\right]$
    - For each agent $i$ and all **s**:
        $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(Q_{1,t}(s), \ldots, Q_{n,t}(s))$
        $V_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[Q_{i,t}(s)]$
```

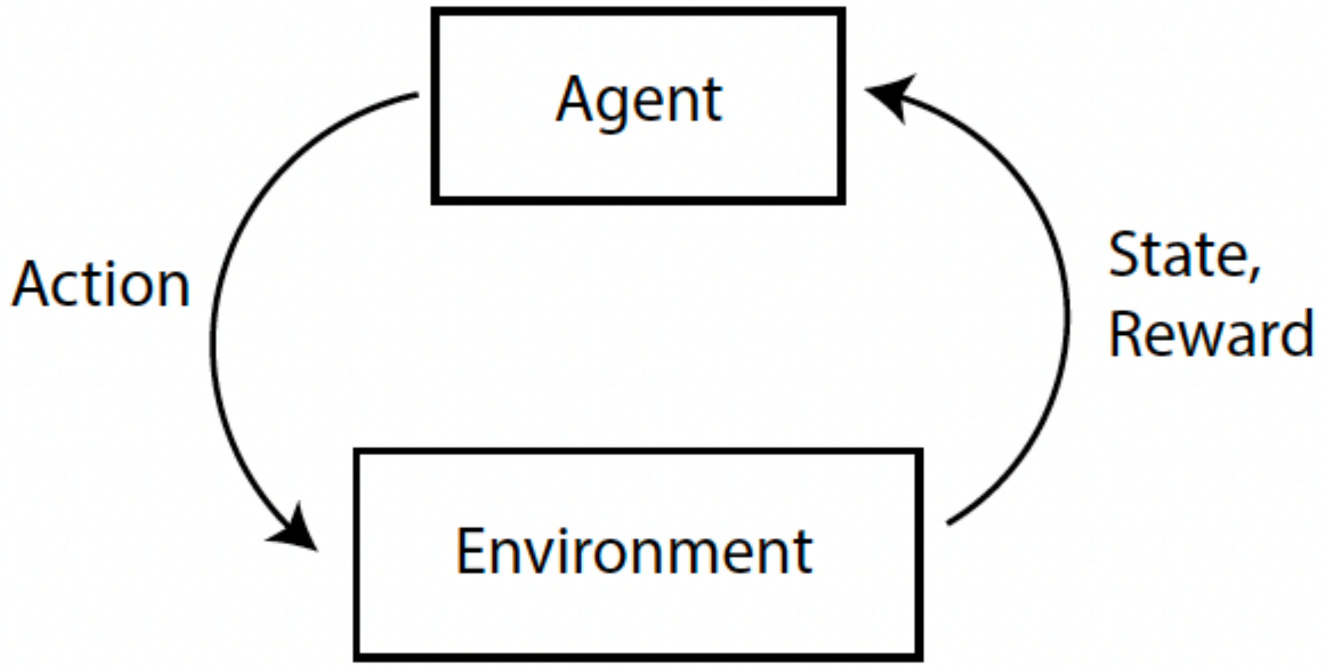
Computes a ***CCE*** of a finite-horizon Markov Game (by definition)

In ***poly(H,S,A,B)*** steps

From full information to learning

- So far we have assumed that you know the Markov Game perfectly and only looked at ***computation***

- In RL you need to explore to ***learn :***
  1. The dynamics of the environment
  2. The reward function

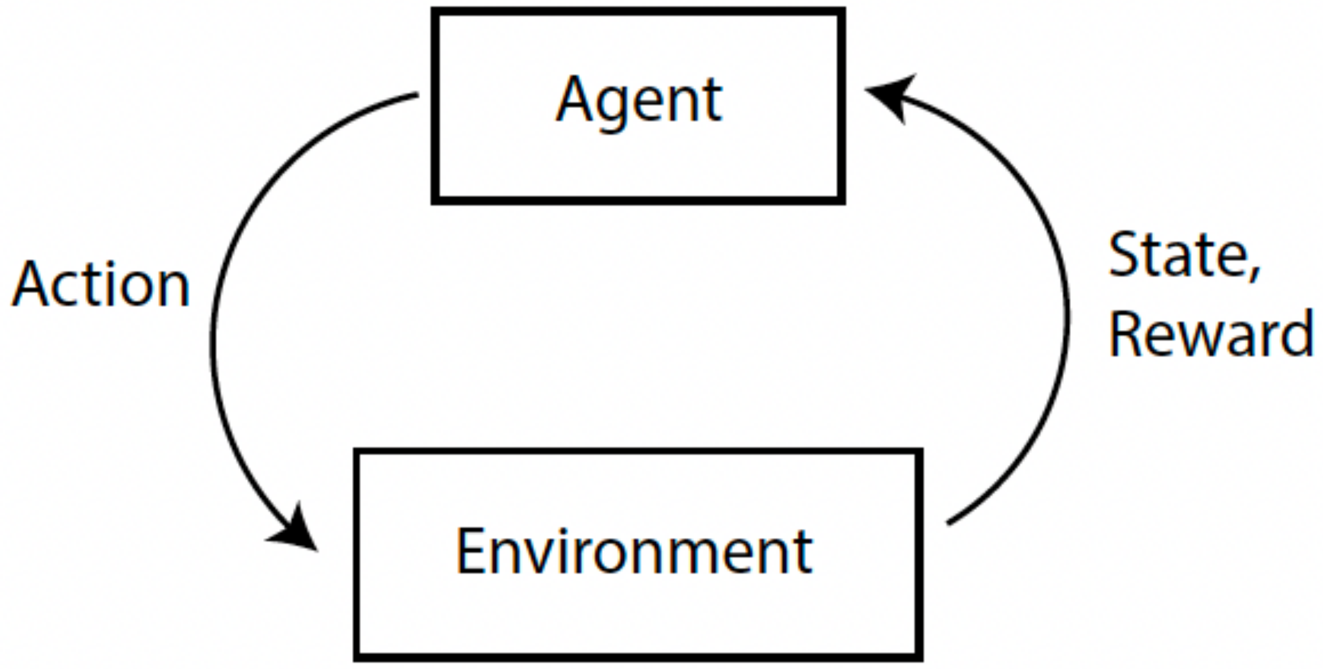


- Trade-off ***exploration*** (of the environment and reward function) with ***exploitation*** to accumulate reward

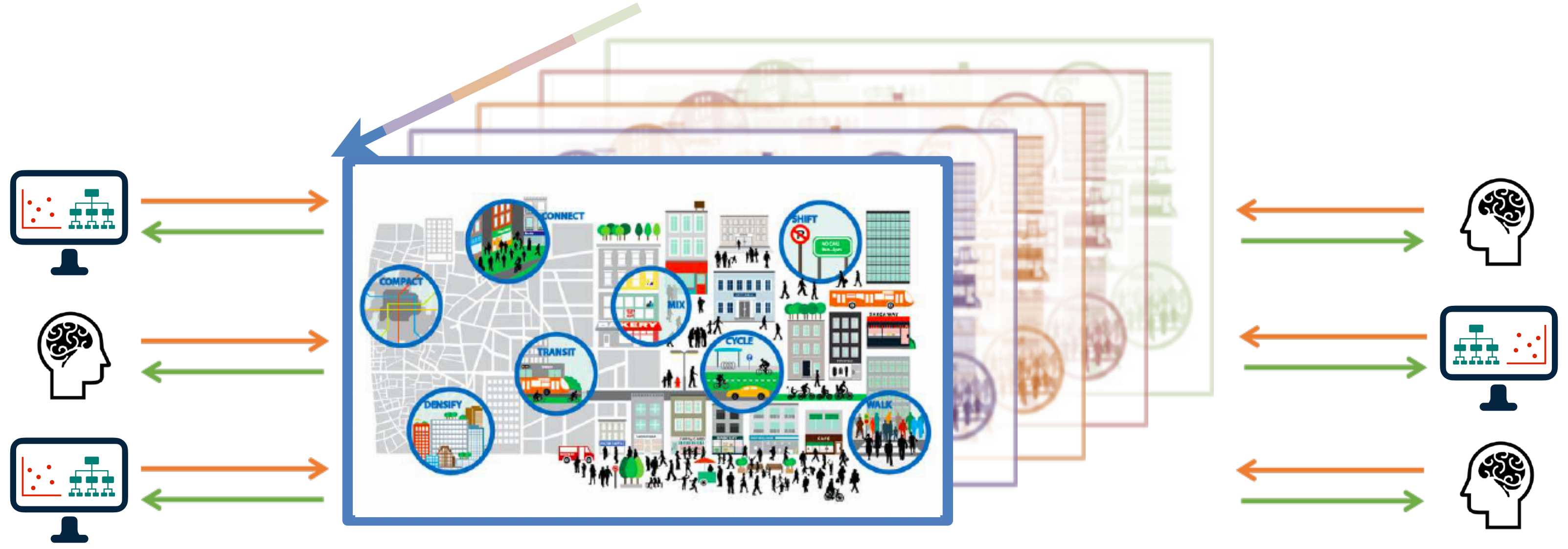
From full information to learning

- So far we have assumed that you know the Markov Game perfectly and only looked at ***computation***

- In RL you need to explore to ***learn*** **:**
  1. The dynamics of the environment
  2. The reward function



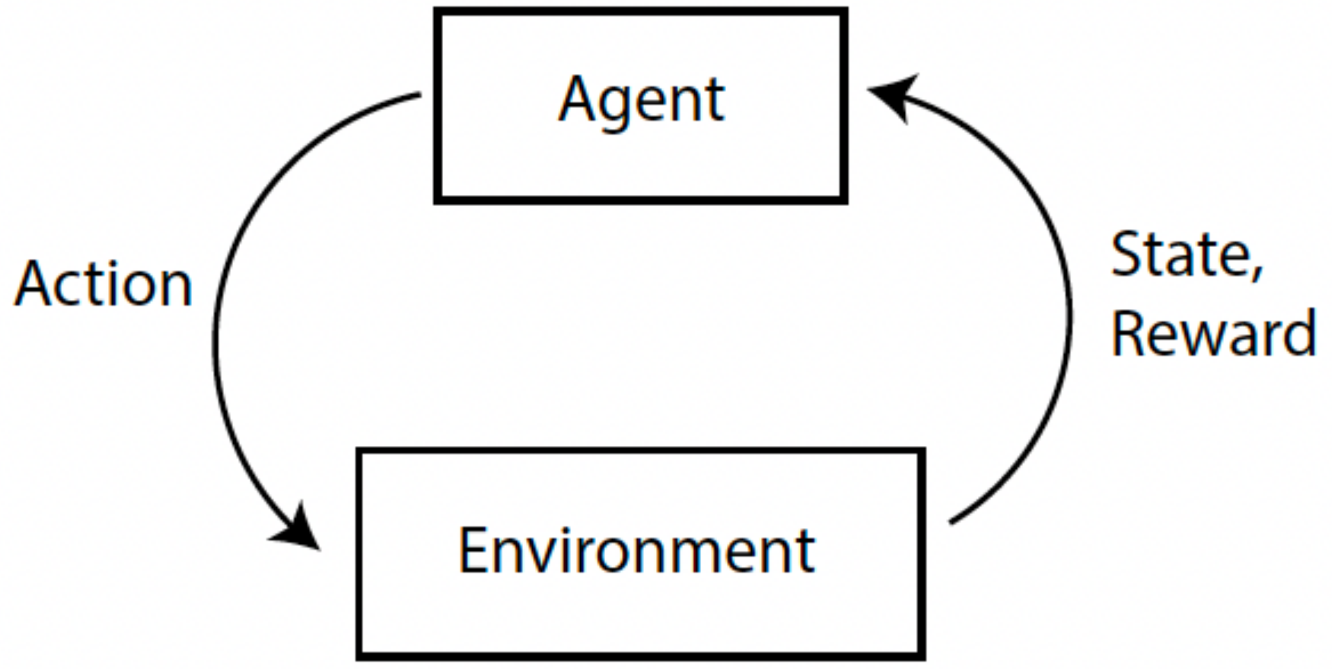
- In MARL you need to explore to ***learn*** **:**
  1. The dynamics of the environment
  2. The reward function
  3. How to compete



Exploration in MDPs and Markov Games

- In reinforcement learning there has been a lot of recent progress on how to ***explore*** MDPS

- Naive exploration: ϵ-**greedy**: take $\begin{cases} \text{random action}, & \text{with probability } \epsilon \\ \text{greedy action}, & \text{otherwise} \end{cases}$



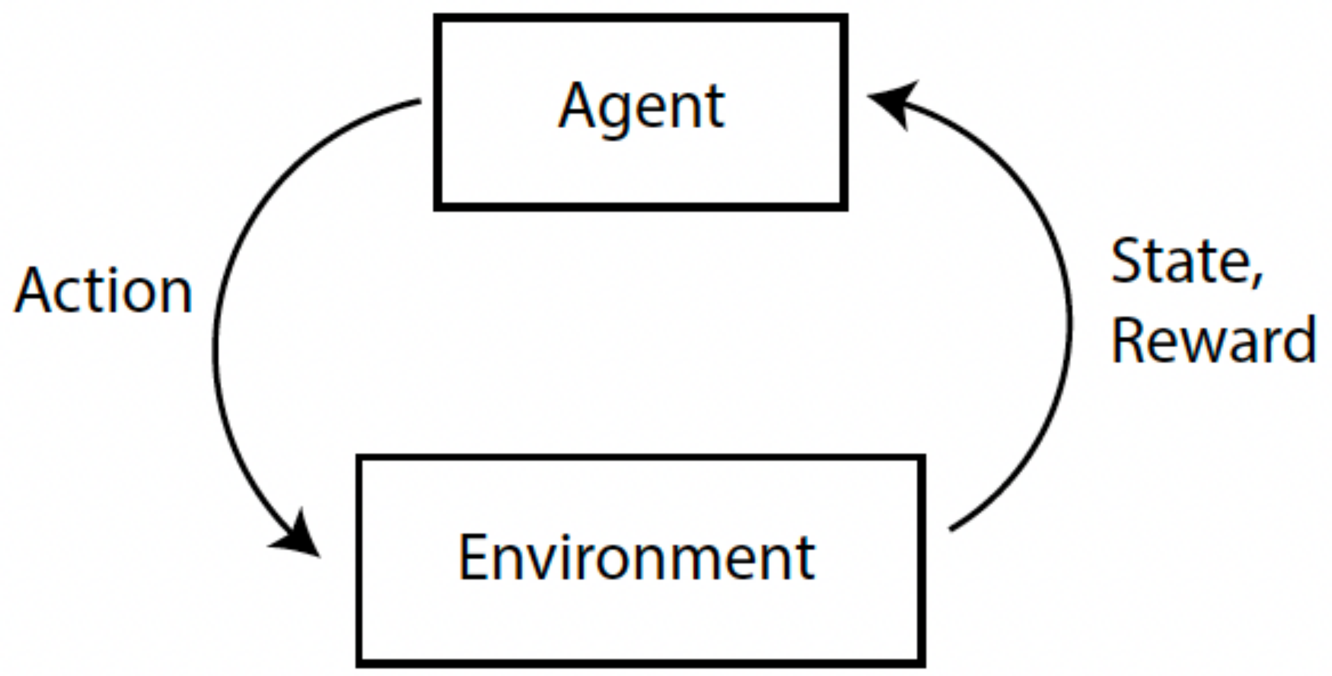
- Can be shown that in the worst case, this can take an exponential in the number of samples to learn an optimal policy!

Exploration in MDPs and Markov Games

- In reinforcement learning there has been a lot of recent progress on how to ***explore*** MDPS

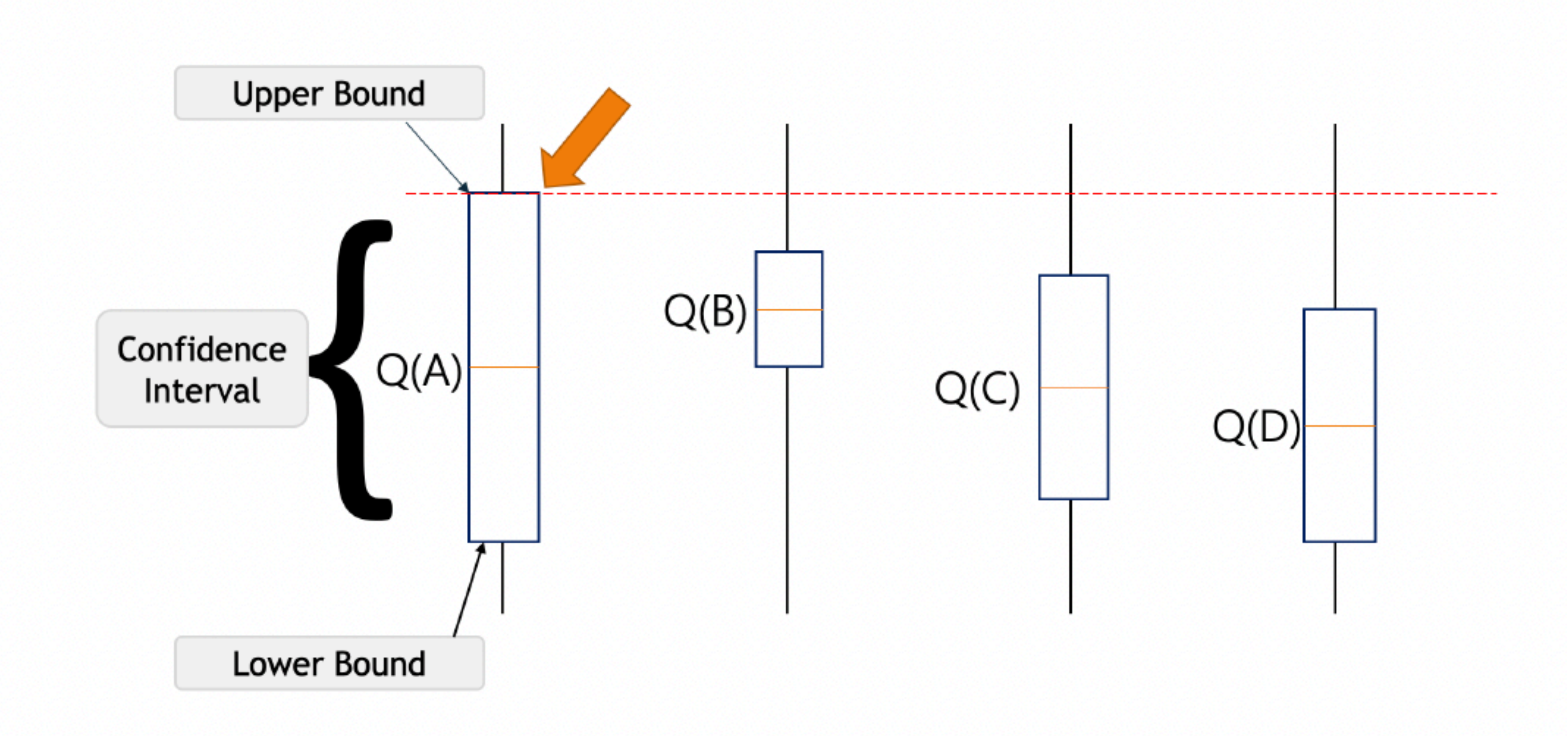
- Naive exploration:

$$\epsilon\text{-}\textbf{greedy}\text{: take } \begin{cases} \text{random action,} & \text{with probability } \epsilon \\ \text{greedy action,} & \text{otherwise} \end{cases}$$



- ***Optimistic exploration:*** *be optimistic* pick action with the largest Upper confidence bound

Strategy yields a statistically optimal way of exploring bandits and MDPs



Optimistic Value Iteration in Markov Games

Optimistic CCE Value Iteration

- Initialize, for all agents i: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
- For step $k = 1,2,\ldots$
  - Execute policies, collect rollouts, estimate transition matrix $\hat{P}$

$$\hat{P}(s'|s,a) = \frac{N(s',s,a)}{N(s,a)}$$

Optimistic Value Iteration in Markov Games

```
*Optimistic CCE Value Iteration*

- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0$, $V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
- For step $k = 1,2,\ldots$
    - Execute policies, collect rollouts, estimate transition matrix $\hat{P}$
    - For $t = H, H-1,\ldots,0$
        - For each state agent $i$ and all **s,a** :
              $\bar{Q}_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s' \sim \hat{P}(\cdot|s,a)}\left[\bar{V}_{i,t+1}(s')\right] + \mathrm{UCB}$
        - For each agent $i$ and all **s**:
              $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(\bar{Q}_{1,t}(s), \ldots, \bar{Q}_{n,t}(s))$
              $\bar{V}_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[\bar{Q}_{i,t}(s)]$
```

$$\hat{P}(s'|s,a) = \frac{N(s',s,a)}{N(s,a)}$$

Optimistic Value Iteration in Markov Games

```
Optimistic CCE Value Iteration

- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
- For step $k = 1,2,...$
    - Execute policies, collect rollouts, estimate transition matrix $\hat{P}$
    - For $t = H, H-1,...,0$
        - For each state agent $i$ and all s,a :
              $\bar{Q}_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s' \sim \hat{P}(\cdot|s,a)}\left[\bar{V}_{i,t+1}(s')\right] + \mathrm{UCB}$
        - For each agent $i$ and all s:
              $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(\bar{Q}_{1,t}(s), \ldots, \bar{Q}_{n,t}(s))$
              $\bar{V}_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[\bar{Q}_{i,t}(s)]$
```

Optimistic estimator of Q

High probability ***over-estimate*** given properly chosen UCB term

Normally chosen via concentration inequalities

Optimistic Value Iteration in Markov Games

Optimistic CCE Value Iteration

```
- Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
- For step $k = 1,2,...$
    - Execute policies, collect rollouts, estimate transition matrix $\hat{P}$
    - For $t = H, H-1,...,0$
        - For each state agent $i$ and all **s,a** :
              $\bar{Q}_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s' \sim \hat{P}(\cdot|s,a)} \left[\bar{V}_{i,t+1}(s')\right] + \mathrm{UCB}$
        - For each agent $i$ and all **s**:
              $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(\bar{Q}_{1,t}(s), \ldots, \bar{Q}_{n,t}(s))$
              $\bar{V}_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[\bar{Q}_{i,t}(s)]$
```

Optimistic estimator of Q

High probability ***over-estimate*** given properly chosen UCB term

Gives exploration of Markov Game

Optimistic Value Iteration in Markov Games

Optimistic CCE Value Iteration

```
Initialize, for all agents $i$: $Q_{i,H+1}(s, a_i, a_{-i}) = 0, V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
For step $k = 1,2,...$
    Execute policies, collect rollouts, estimate transition matrix $\hat{P}$
    For $t = H, H-1,...,0$
        For each state agent $i$ and all s,a :
            $\bar{Q}_{i,t}(s, a_i, a_{-i}) \leftarrow R_i(s, a_i, a_{-i}) + \mathbb{E}_{s' \sim \hat{P}(\cdot|s,a)} \left[\bar{V}_{i,t+1}(s')\right] + \mathrm{UCB}$
        For each agent $i$ and all s:
            $(\pi^*_{1,t}, \ldots, \pi^*_{n,t}) \leftarrow \mathrm{CCE}(\bar{Q}_{1,t}(s), \ldots, \bar{Q}_{n,t}(s))$
            $\bar{V}_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[\bar{Q}_{i,t}(s)]$
```

Using CCE oracle

Optimistic Value Iteration in Markov Games

Optimistic CCE Value Iteration

```
Initialize, for all agents $i$: $Q_{i,H+1}(s,a_i,a_{-i}) = 0, V_{i,H+1} = 0$, policies $\{\pi_{i,t}\}_{t=0}^{H}$
For step $k = 1,2,...$
    Execute policies, collect rollouts, estimate transition matrix $\hat{P}$
    For $t = H, H-1,...,0$
        For each state agent $i$ and all **s,a** :
            $$\bar{Q}_{i,t}(s,a_i,a_{-i}) \leftarrow R_i(s,a_i,a_{-i}) + \mathbb{E}_{s'\sim\hat{P}(\cdot|s,a)}\left[\bar{V}_{i,t+1}(s')\right] + \text{UCB}$$
        For each agent $i$ and all **s**:
            $$(\pi^*_{1,t},...,\pi^*_{n,t}) \leftarrow \text{CCE}(\bar{Q}_{1,t}(s),...,\bar{Q}_{n,t}(s))$$
            $$\bar{V}_{i,t}(s) \leftarrow \mathbb{E}_{\pi^*_t}[\bar{Q}_{i,t}(s)]$$
```

This algorithm can be seen as a multi-agent extension of UCB-VI (Azar et al. 2017) from single-agent reinforcement learning

Optimistic Value Iteration in Markov Games

Thm: Finite Horizon Zero-Sum Markov Games [Liu et al. 2020]

With high probability, optimistic Value-iteration finds an ϵ-approximate Nash equilibrium in:

$$\tilde{O}\left(\underset{\uparrow}{\frac{H^3 S |A_1||A_2|}{\epsilon^2}}\right) \text{ episodes.}$$

Polynomial-time and polynomial number of samples required

Optimistic Value Iteration in Markov Games

Thm: Finite Horizon Zero-Sum Markov Games [Liu et al. 2020]

With high probability, optimistic Value-iteration finds an ϵ-approximate Nash equilibrium in:

$$\tilde{O}\left(\frac{H^3 S |A_1| |A_2|}{\epsilon^2}\right) \text{ episodes.}$$

They also show an information theoretic lower bound that learning Nash requires at least:

$$\tilde{O}\left(\frac{H^3 S \ \max\{ |A_1|, |A_2| \}}{\epsilon^2}\right) \text{ episodes.}$$

Optimistic Value Iteration in Markov Games

Can Extend the Results to CCE in general-sum Markov games!

Thm: Finite Horizon General-Sum Markov Games [Liu et al. 2020]

With high probability, optimistic Value-iteration finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^4 S \prod_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Note that optimistic Value-iteration is not no-regret, but we can still find a non-stationary CCE

Optimistic Value Iteration in Markov Games

Can Extend the Results to CCE in general-sum Markov games!

Thm: Finite Horizon General-Sum Markov Games [Liu et al. 2020]

With high probability, optimistic Value-iteration finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^4 S \prod_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

They also show an information theoretic lower bound that learning Nash requires at least:

$$\tilde{O}\left(\frac{H^3 S \ \max_i \ |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Curse-of Multi-Agency

Consider the gap:

Lower bound: $\tilde{O}\left(\frac{H^3 S \ \max_i \ |A_i|}{\epsilon^2}\right).$ vs. $\tilde{O}\left(\frac{H^4 S \prod_i |A_i|}{\epsilon^2}\right)$ episodes

Lower bound suggests that we should pay only for the ***largest*** action space

Curse-of Multi-Agency

Consider the gap:

Lower bound: $\tilde{O}\left(\frac{H^3 S \ \max_i \ |A_i|}{\epsilon^2}\right).$ vs. $\tilde{O}\left(\frac{H^4 S \prod_i |A_i|}{\epsilon^2}\right)$ episodes

Lower bound suggests that we should pay only for the ***largest*** action space

Optimistic VI pays for the ***product***

Worst case: $\prod_i |A_i| = |A|^n$

Exponential dependence on number of agents $\Longrightarrow$ ***"Curse of multi-agency"***

(Analog to curse of dimensionality)

Curse-of Multi-Agency

Consider the gap:

Lower bound: $\tilde{O}\left(\frac{H^3 S \ \max_i \ |A_i|}{\epsilon^2}\right).$ vs. $\tilde{O}\left(\frac{H^4 S \prod_i |A_i|}{\epsilon^2}\right)$ episodes

Lower bound suggests that we should pay only for the ***largest*** action space

Optimistic VI pays for the ***product***

Worst case: $\prod_i |A_i| = |A|^n$

Exponential dependence on number of agents $\Longrightarrow$ ***"Curse of multi-agency"*** (Analog to curse of dimensionality)

Can we overcome this?

Overcoming the curse of multi-agency

Problem:

The size of the Q-function itself is $S \times \prod_{i} A_i$

Any algorithm based on **Q** functions has to pay this.

Overcoming the curse of multi-agency

Problem:

The size of the Q-function itself is $S \times \prod_i A_i$

Idea:

We only need to estimate the ***Value*** at a CCE/Nash.

Use
timescale separation. $\Longrightarrow$

Fast timescale: compute optimistic value using e.g., no-regret learning

Slow timescale: update values using dynamic programming

Overcoming the curse of multi-agency

Problem:

The size of the Q-function itself is $S \times \prod_i A_i$

Idea:

We only need to estimate the ***Value*** at a CCE/Nash.

Use
timescale separation.

$\Longrightarrow$

Fast timescale: compute optimistic value using e.g., no-regret learning

Slow timescale: update values using dynamic programming

Crux of idea:
Sample complexity of no-regret learning in games scales with
$\max_i |A_i|$

Overcoming the curse of multi-agency

Use
timescale separation. $\Longrightarrow$

Fast timescale: compute optimistic value using e.g., no-regret learning

Slow timescale: update values using dynamic programming

Crux of idea:
Sample complexity of no-regret learning in games scales with

$$\max_i |A_i|$$

Use payoff-based (bandit feedback) no-regret learning:
(e.g., Exp3 algorithm, stochastic mirror descent)

$$\max_{\pi} \sum_{t=0}^{T} \langle \pi, \ell_t \rangle - \langle \pi_t, \ell_t \rangle \leq o(T)$$

Against arbitrary sequences of rewards $\{r_t \in \mathbb{R}^{|A|}\}_{t=0}^{T}$

while only playing actions $a_t \sim \pi_t$ at each round and observing $r_t(a_t)$

V-Learning in Markov Games

V-learning (PoV of agent i)

```
Initialize policies $\pi_{i,t}^{0}$
For episode $k = 1,2,...$
    Receive initial state $s_0$
    For $t = 0,1,2,...,H$  ← Proceed forwards in time
        Sample action $a_{i,t} \sim \pi_{i,t}^{k}(s_t)$ , observe reward $r_t = R_i(s_t, a_t)$ , next state $s_{t+1}$
        Keep track of number of times each state has been visited:
            $N(s_{t+1}) \leftarrow N(s_{t+1}) + 1$
```

V-Learning in Markov Games

```
V-learning (PoV of agent i)
- Initialize policies $\pi_{i,t}^0$
- For episode $k = 1,2,...$
    - Receive initial state $s_0$
    - For $t = 0,1,2,...,H$
        - Sample action $a_{i,t} \sim \pi_{i,t}^k(s_t)$ , observe reward $r_t = R_i(s_t, a_t)$ , next state $s_{t+1}$
        - Keep track of number of times each state has been visited:
              $N(s_{t+1}) \leftarrow N(s_{t+1}) + 1$
        - Update optimistic value estimate
              $V_{i,t}(s_t) \leftarrow (1 - \alpha_k)V_{i,t}(s_t) + \alpha_k(r_t + V_{i,t}(s_{t+1}) + UCB(N(s_{t+1}))$
        - Update policy using no-regret learning:
              $\pi_{i,t}^{k+1}(s_t) \leftarrow \text{No Regret Step}(a_{i,t}, r_t + V_{i,t}(s_{t+1}) + UCB(N(s_{t+1}), \beta_k)$
```

Optimistic estimate of V
(In high probability with
carefully chosen UCB)

V-Learning in Markov Games

V-learning (PoV of agent i)

```
Initialize policies $\pi_{i,t}^{0}$
For episode $k = 1,2,\ldots$
    Receive initial state $s_0$
    For $t = 0,1,2,\ldots,H$
        Sample action $a_{i,t} \sim \pi_{i,t}^{k}(s_t)$ , observe reward $r_t = R_i(s_t, a_t)$ , next state $s_{t+1}$
        Keep track of number of times each state has been visited:
            $N(s_{t+1}) \leftarrow N(s_{t+1}) + 1$
        Update optimistic value estimate
            $V_{i,t}(s_t) \leftarrow (1-\alpha_k)V_{i,t}(s_t) + \alpha_k(r_t + V_{i,t}(s_{t+1}) + UCB(N(s_{t+1}))$
        Update policy using no-regret learning:
            $\pi_{i,t}^{k+1}(s_t) \leftarrow \text{No Regret Step}(a_{i,t}, r_t + V_{i,t}(s_{t+1}) + UCB(N(s_{t+1}), \beta_k)$
```

← Adversarial bandit step on fast timescale

V-Learning in Markov Games

V-learning (PoV of agent i)

- Initialize policies $\pi^0_{i,t}$
- For episode $k = 1,2,\ldots$
    - Receive initial state s_0
    - For $t = 0,1,2,\ldots,H$
        - Sample action $a_{i,t} \sim \pi^k_{i,t}(s_t)$, observe reward $r_t = R_i(s_t, a_t)$, next state s_{t+1}
        - Keep track of number of times each state has been visited:

$$N(s_{t+1}) \leftarrow N(s_{t+1}) + 1$$

        - Update optimistic value estimate

$$V_{i,t}(s_t) \leftarrow (1 - \alpha_k)V_{i,t}(s_t) + \alpha_k \text{CCE Value}_i \left(\bar{Q}_{1,t}(s_{t+1}), \ldots, \bar{Q}_{n,t}(s_{t+1})\right)$$

To see why this works, suppose the fast timescale has converged

This is just stochastic approximations of the dynamic programming algorithm

V-Learning in Markov Games

```
V-learning (PoV of agent i)
Initialize policies $\pi_{i,t}^0$
For episode $k = 1,2,\ldots$
    Receive initial state $s_0$
    For $t = 0,1,2,\ldots,H$
        Sample action $a_{i,t} \sim \pi_{i,t}^k(s_t)$ , observe reward $r_t = R_i(s_t, a_t)$ , next state $s_{t+1}$
        Keep track of number of times each state has been visited:
            $N(s_{t+1}) \leftarrow N(s_{t+1}) + 1$
        Update optimistic value estimate
            $V_{i,t}(s_t) \leftarrow (1-\alpha_k)V_{i,t}(s_t) + \alpha_k \text{CCE Value}_i\left(\bar{Q}_{1,t}(s_{t+1}), \ldots, \bar{Q}_{n,t}(s_{t+1})\right)$
```

To see why this works,
suppose the fast timescale
has converged

Note that this algorithm is independent! Players can do this separately.
(Under the same caveat that the step sizes are synced)

V-Learning in Markov Games

Thm: Finite Horizon General-Sum Markov Games [Jin et al. 2021]

With high probability, optimistic Value-iteration with ***Follow-the-Regularized-Leader (FTRL)*** as the no-regret learning algorithm finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^5 S \ \max_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Which overcame the curse of multi-agents
(albeit with a worse dependence on horizon)

V-Learning in Markov Games

Thm: Finite Horizon General-Sum Markov Games [Jin et al. 2021]

With high probability, optimistic Value-iteration with ***Follow-the-Regularized-Leader (FTRL)*** as the no-regret learning algorithm finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^5 S \ \max_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Similar result with a slightly worse rate was derived around the same time by Mao & Basar, 2021 using online mirror descent instead of FTRL

V-Learning in Markov Games

Thm: Finite Horizon General-Sum Markov Games
[Jin et al. 2021]

With high probability, optimistic Value-iteration with ***Follow-the-Regularized-Leader (FTRL)*** as the no-regret learning algorithm finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^5 S \ \max_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Extended to approximate
correlated eq. by Song et al. 2021
by using a no-swap regret algorithm

V-Learning in Markov Games

Thm: Finite Horizon General-Sum Markov Games [Jin et al. 2021]

With high probability, optimistic Value-iteration with ***Follow-the-Regularized-Leader (FTRL)*** as the no-regret learning algorithm finds an ϵ-approximate ***CCE*** equilibrium in:

$$\tilde{O}\left(\frac{H^5 S \ \max_i |A_i|}{\epsilon^2}\right) \text{ episodes.}$$

Still an ongoing area of research!

Faster convergence rates, specific structures (e.g., extensive form games, cooperative games, zero-sum games), equilibrium selection, multi-objective optimization

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

    i. Policy-gradient algorithms in games

    ii. Value-based algorithms

3. **Further directions**

    i. The role of function approximation

    ii. Scalable algorithms for zero-sum games

    iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

      - Takeaways:

         - Q-learning algorithms for ***infinite horizon zero-sum*** Markov games, not in general.

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

   - Takeaways:

     - In ***finite-horizon Markov games***, can efficiently learn Nash though you have to be careful about the ***curse of multi-agency.***

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

      - Takeaways:

         - In both of these cases ***timescale separation*** is key to simplify the problem.

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions (Work from my group)**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

Sample efficiency is crucial

So far we have seen that we have algorithms for ***efficiently*** learning in tabular finite-horizon Markov Games

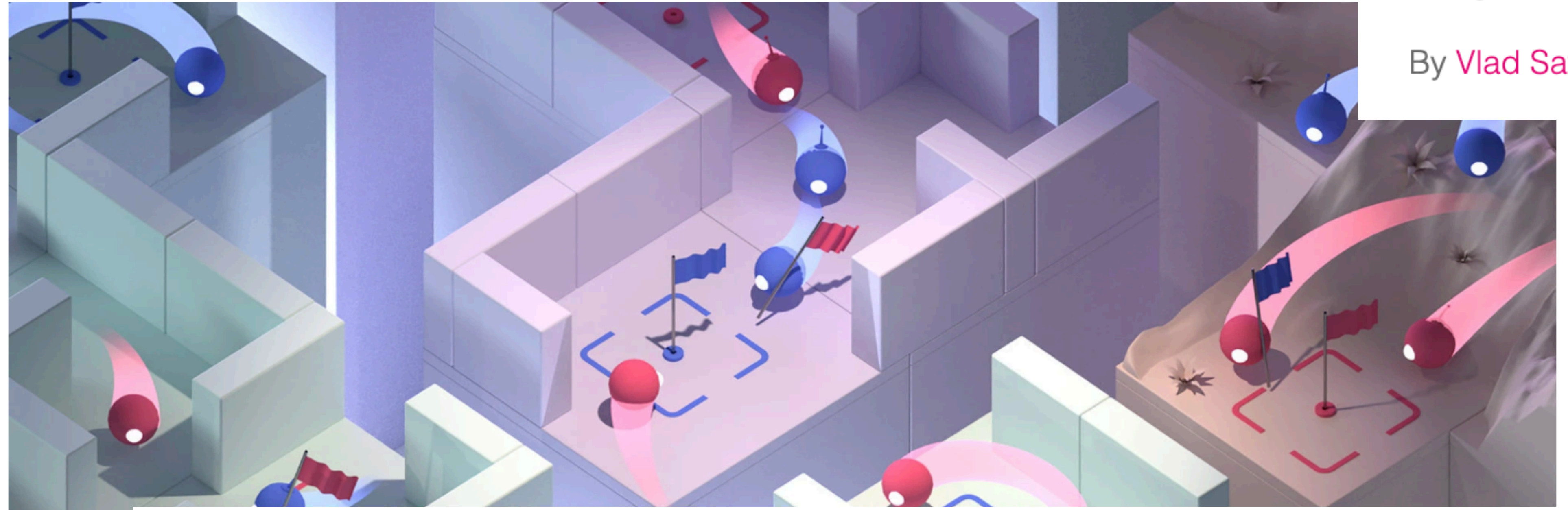
Meaning it achieves good sample complexities

$>10^7$ games of Go
>1 month of training time on dedicated servers

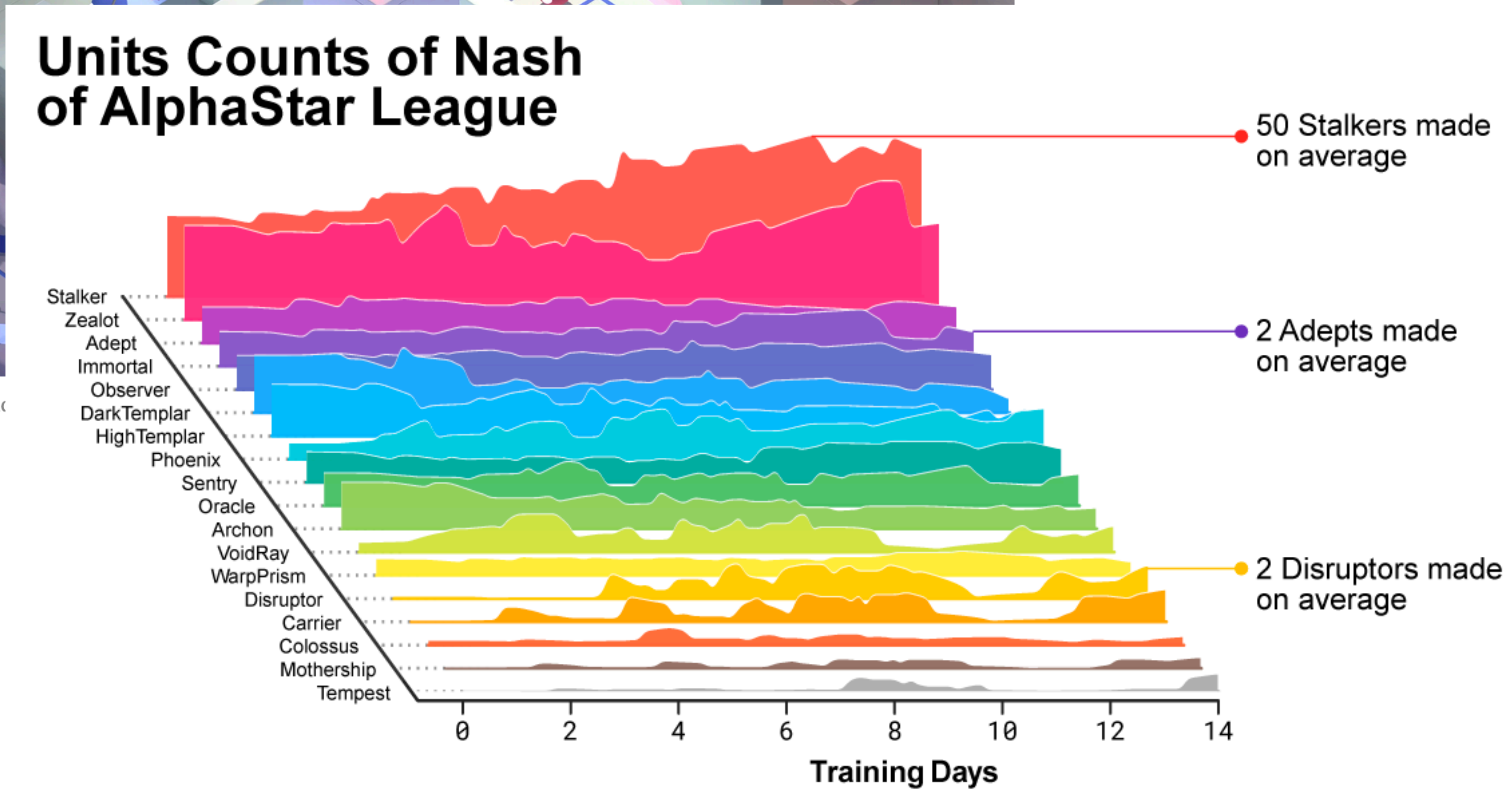
200 years of real-time StarCraft games
>1 month of training time on dedicated servers

DeepMind Can Now Beat Us at Multiplayer Games, Too

Chess and Go were child's play. Now A.I. is winning at capture the flag. Will such skills translate to the real world?



DeepMinc



Units Counts of Nash of AlphaStar League

50 Stalkers made on average

2 Adepts made on average

2 Disruptors made on average

GAMING ENTERTAINMENT TECH

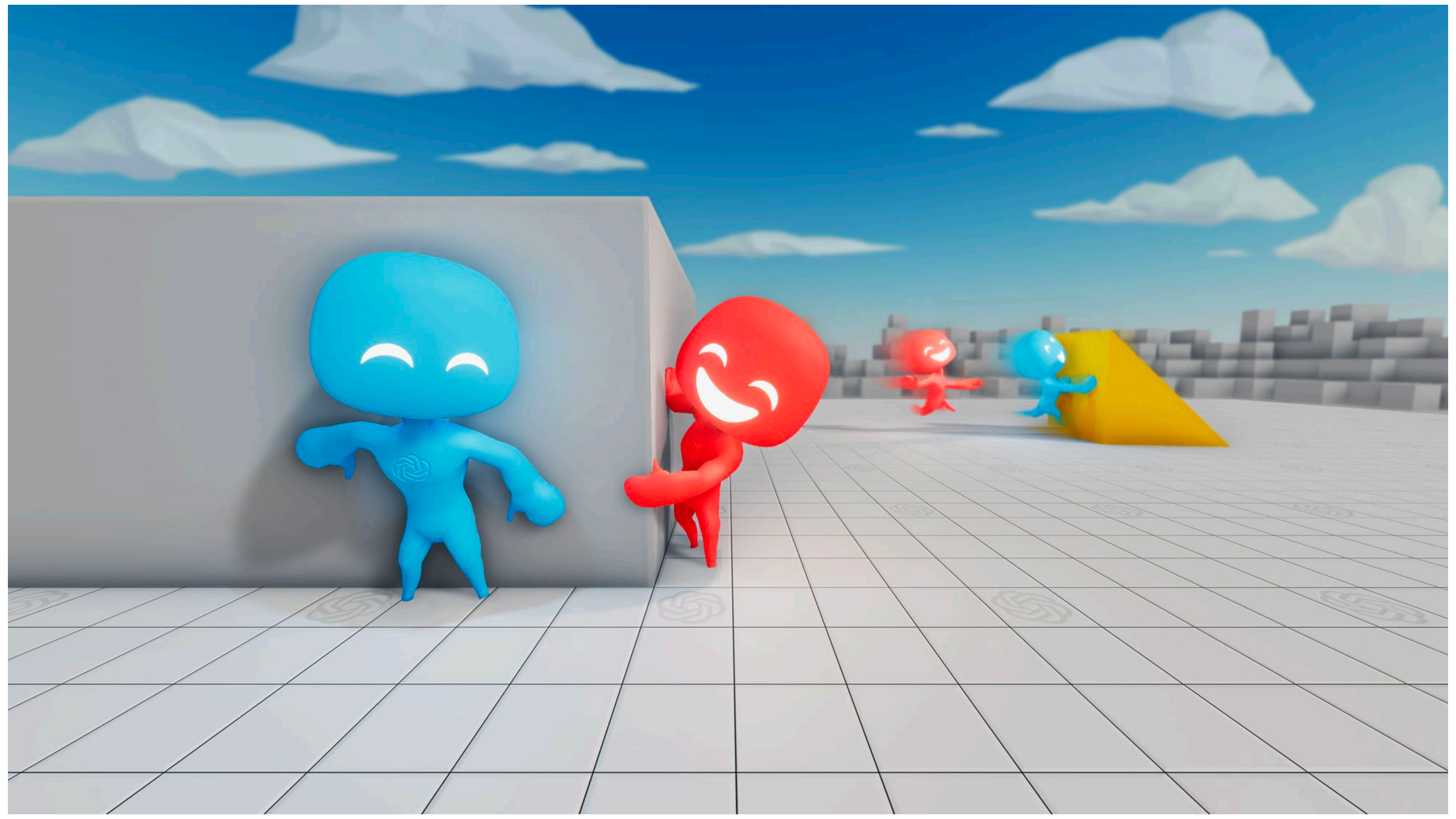
Feeble humans prove no match for OpenAI's Dota 2 gods

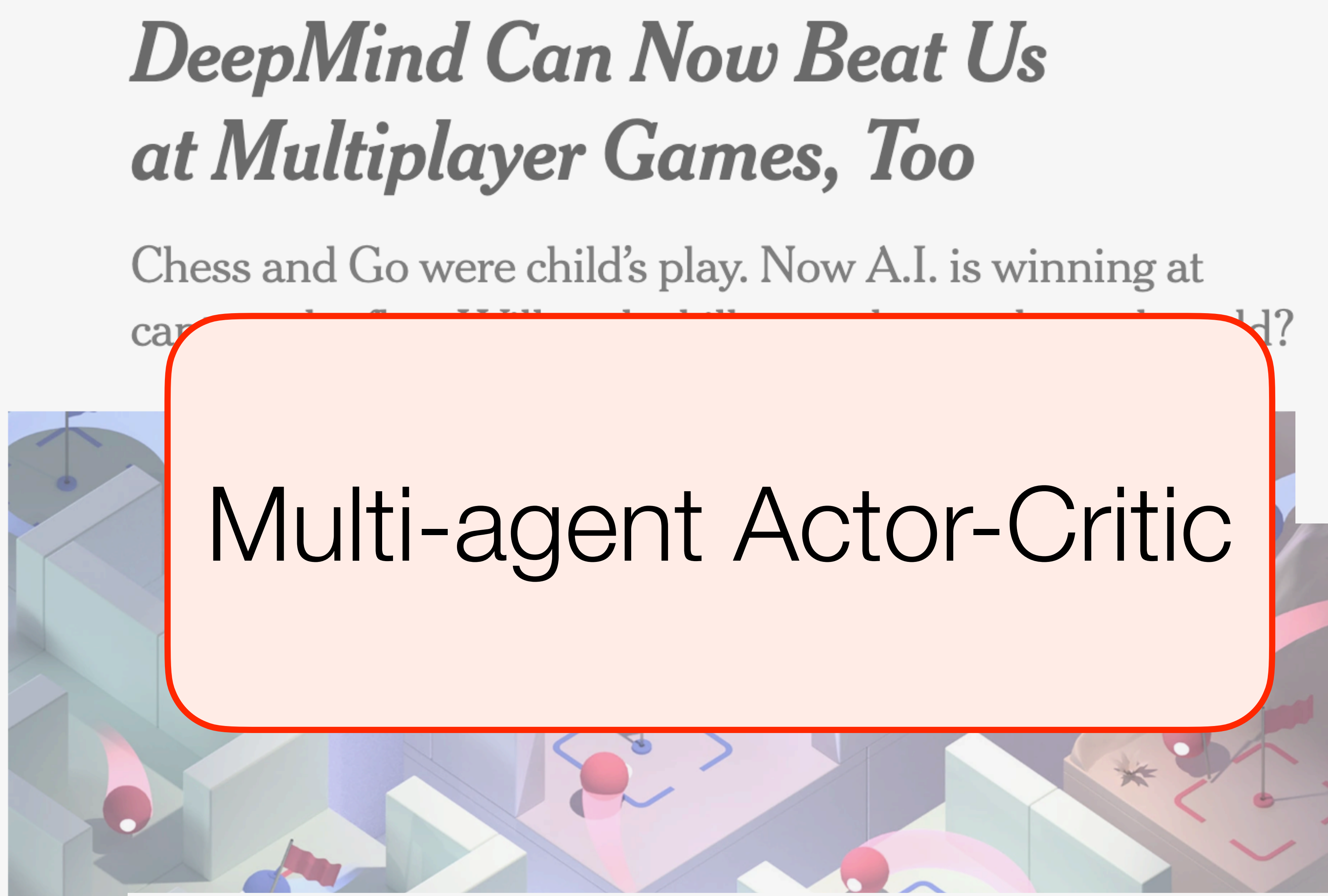
The AI won 7,215 matches against humans, losing only 42 in the process

By Vlad Savov | @vladsavov | Apr 23, 2019, 9:25am EDT

JUNE 8, 2017 • 5 MINUTE READ

Learning to Cooperate, Compete, and Communicate





DeepMind Can Now Beat Us at Multiplayer Games, Too

Chess and Go were child's play. Now A.I. is winning at ca ... ld?

DeepMinc

Multi-agent Actor-Critic

GAMING ENTERTAINMENT TECH

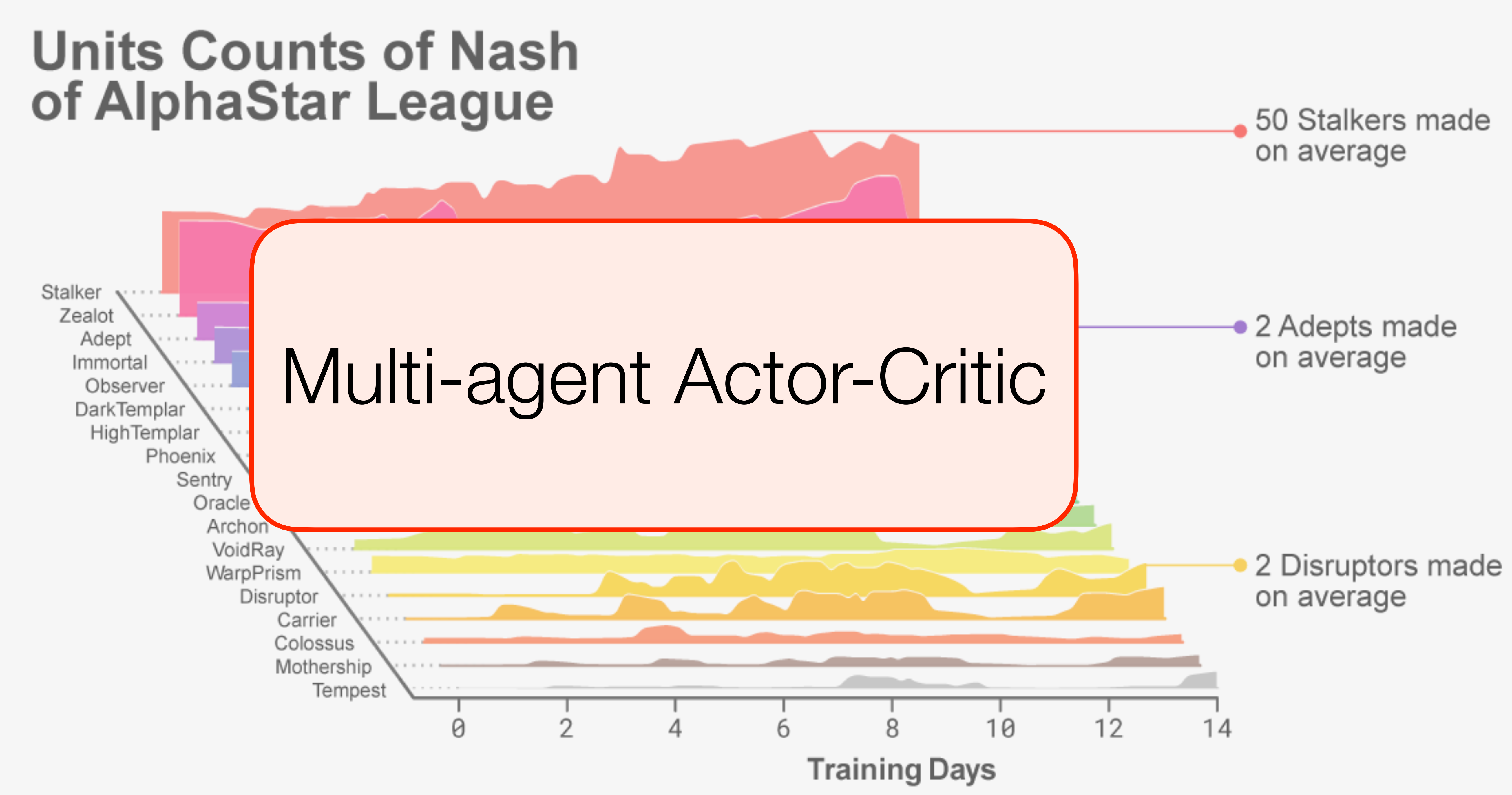
Feeble humans penAI's Dota 2 gods

The AI won 7,215 matches against

By Vlad Savov | @vladsavov | Apr 23, 2019, 9:25am EDT

Multi-agent PPO

(Proximal Policy Optimization)



Units Counts of Nash of AlphaStar League

Multi-agent Actor-Critic

JUNE 8, 2017 • 5 MINUTE READ

Learning to Cooperate, Compete, and Communicate

Multi-agent DDPG

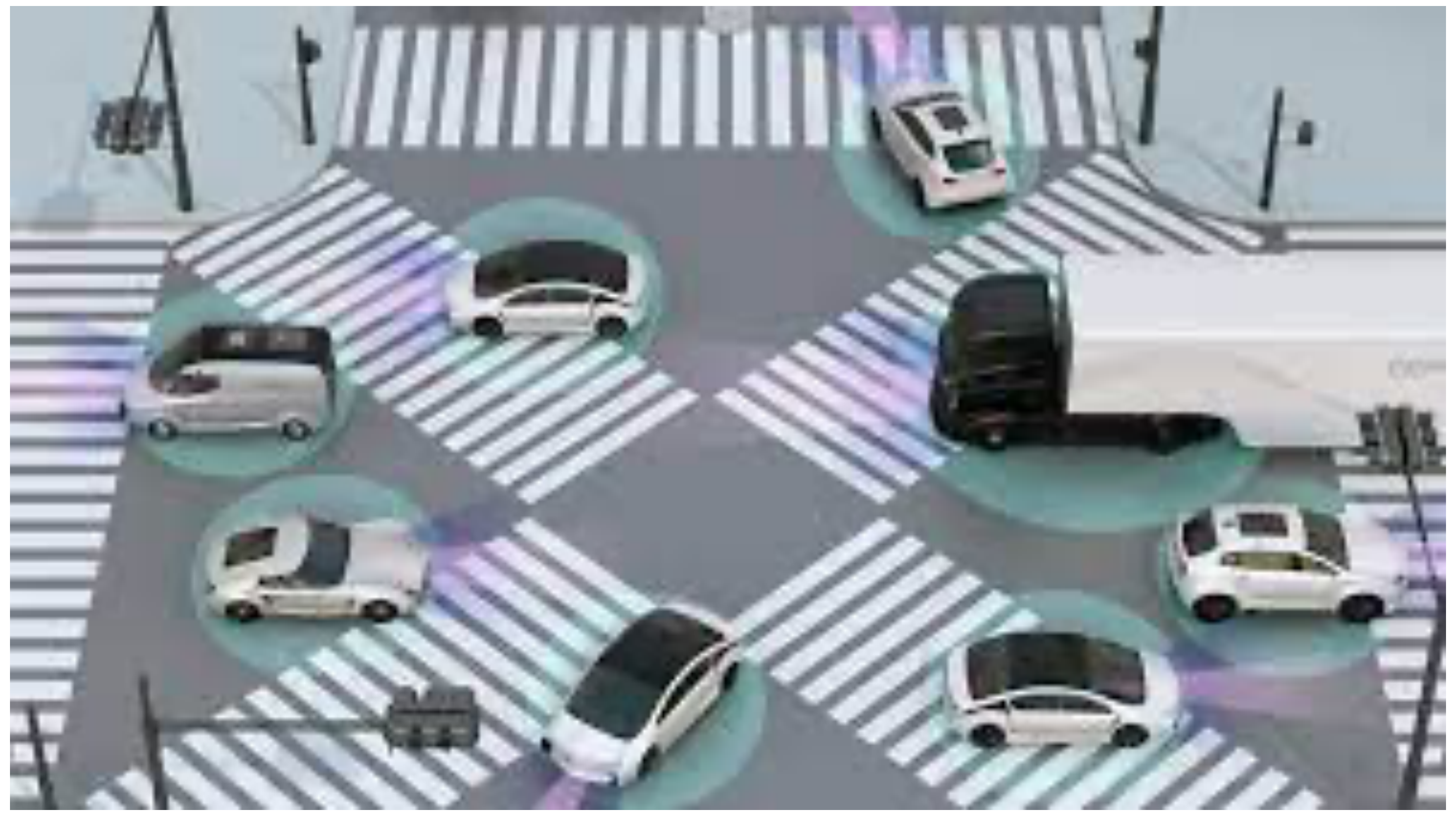
(Deep Deterministic Policy Gradient)

Learning in Games

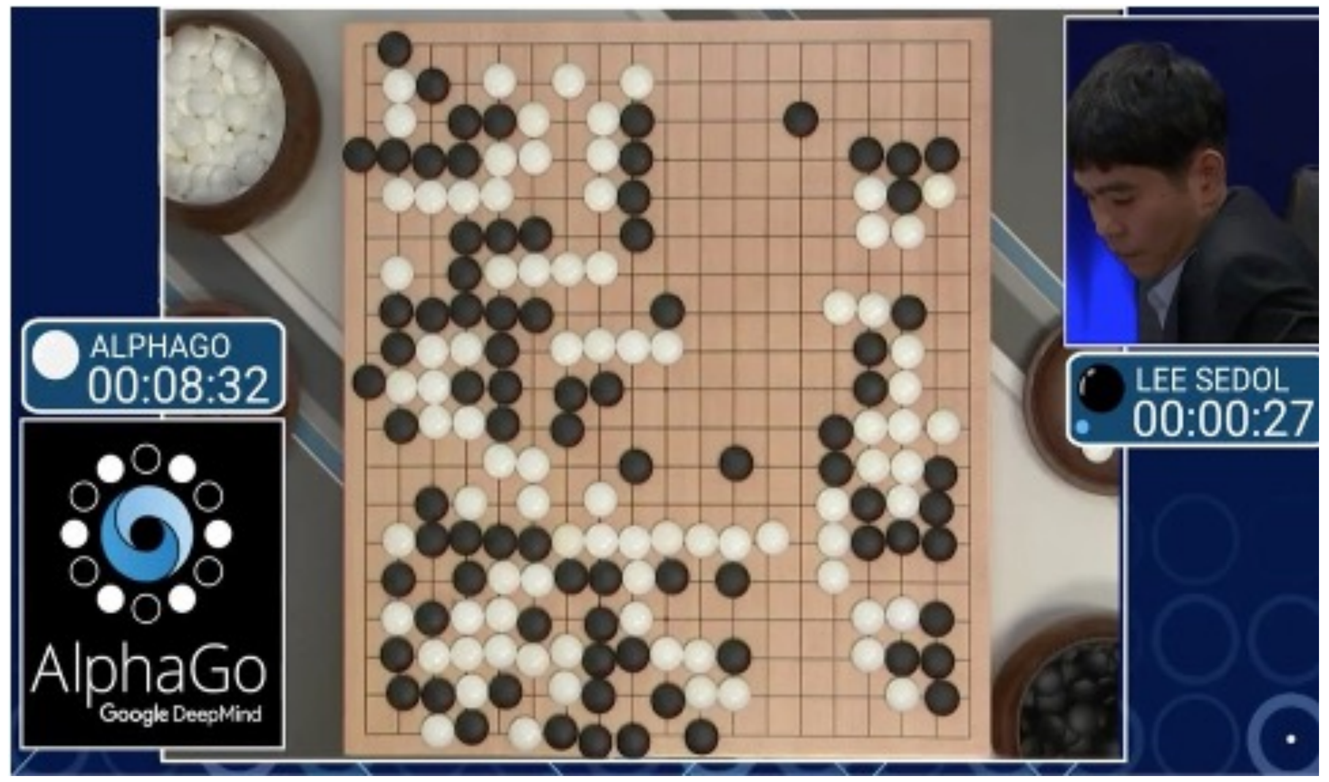
We need scalable algorithms as well:

What makes good learning algorithms in games?

- Convergent
  last-iterate, ergodic, best iterate convergence to Nash
- Independent learning
  agents should not know anything about their opponents utility
- No-regret
  agents should be "rational" (e.g., take advantage of naive opponents)
- **Scalable**
  Many real world applications have large state-spaces!



3^{361} states!



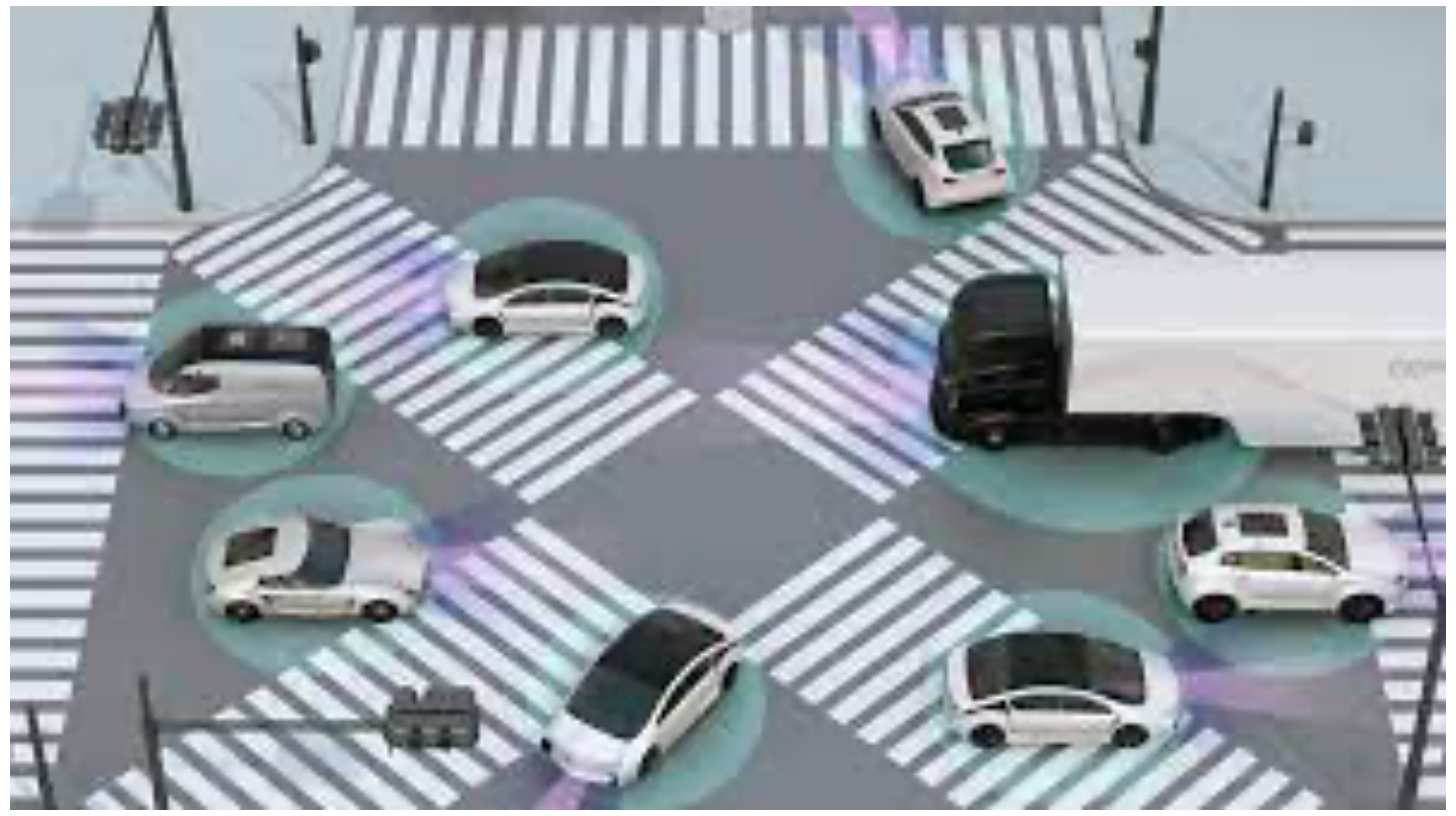
Learning in Games

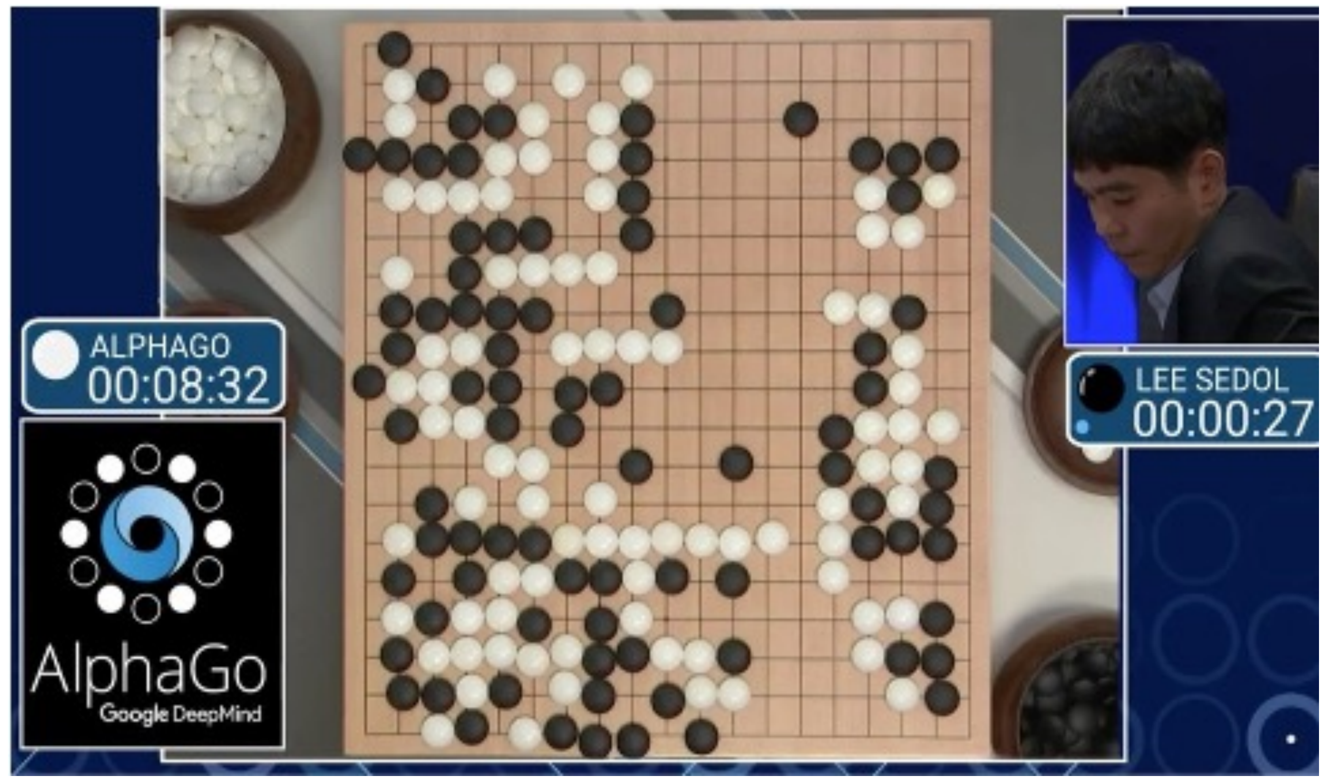
We need scalable algorithms as well:

What makes good learning algorithms in games?

- Convergent
  last-iterate, ergodic, best iterate convergence to Nash
- Independent learning
  agents should not know anything about their opponents utility
- No-regret
  agents should be “rational” (e.g., take advantage of naive opponents)
- **Scalable**
  Many real world applications have large state-spaces!

This requires using function approximation





3^{361} states!

Learning in Games

We need scalable algorithms as well:

What makes good learning algorithms in games?

- Conver
  last-it

- Indeper
  agent

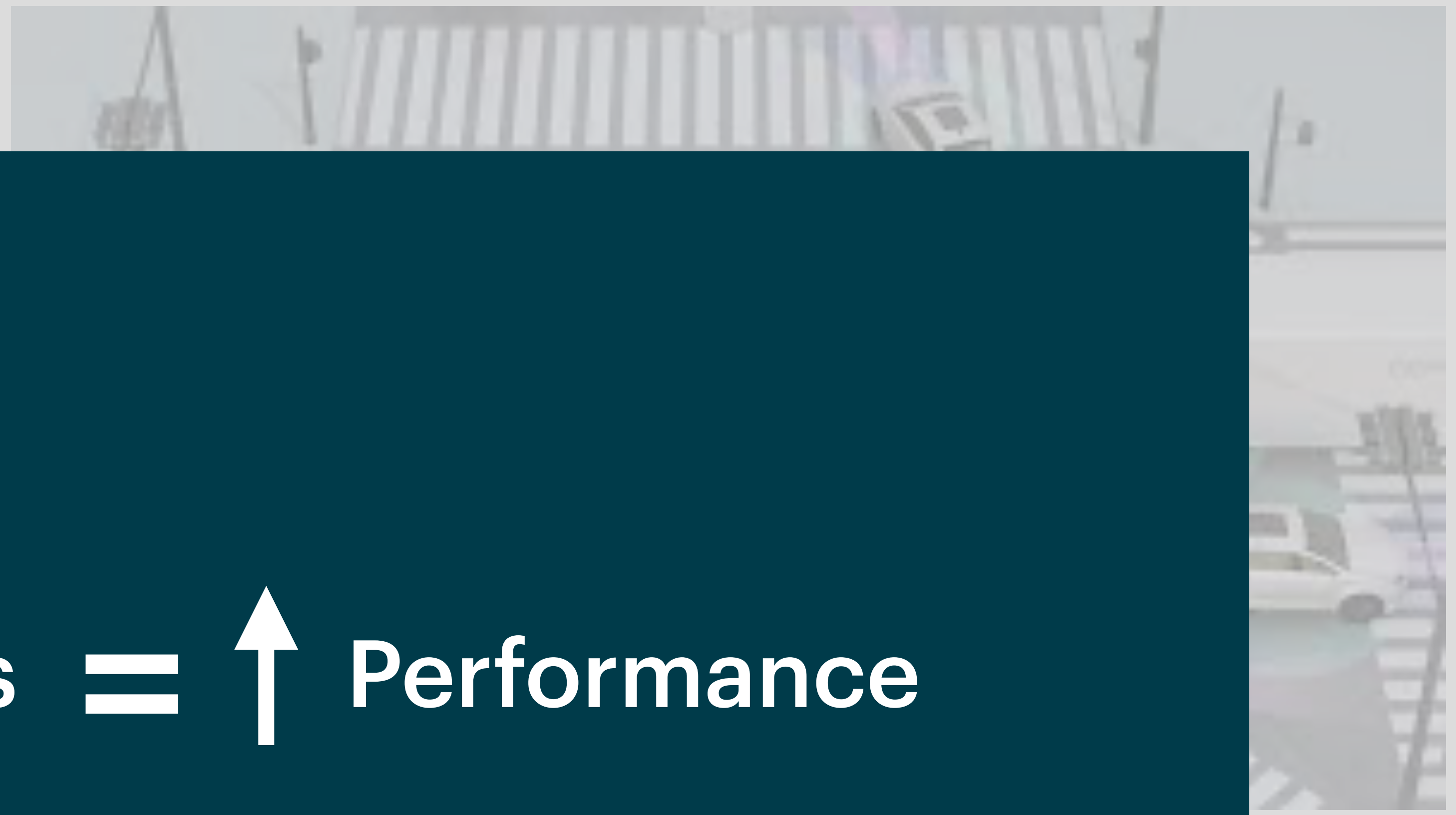
- No-regret
  agents should be “rational” (e.g., take advantage of naive opponents)

- **Scalable**
  Many real world applications have large state-spaces!

This requires using function approximation

Deep RL

↑ **Data** ↑ **Compute** + ↑ **Complex Models** = ↑ **Performance**



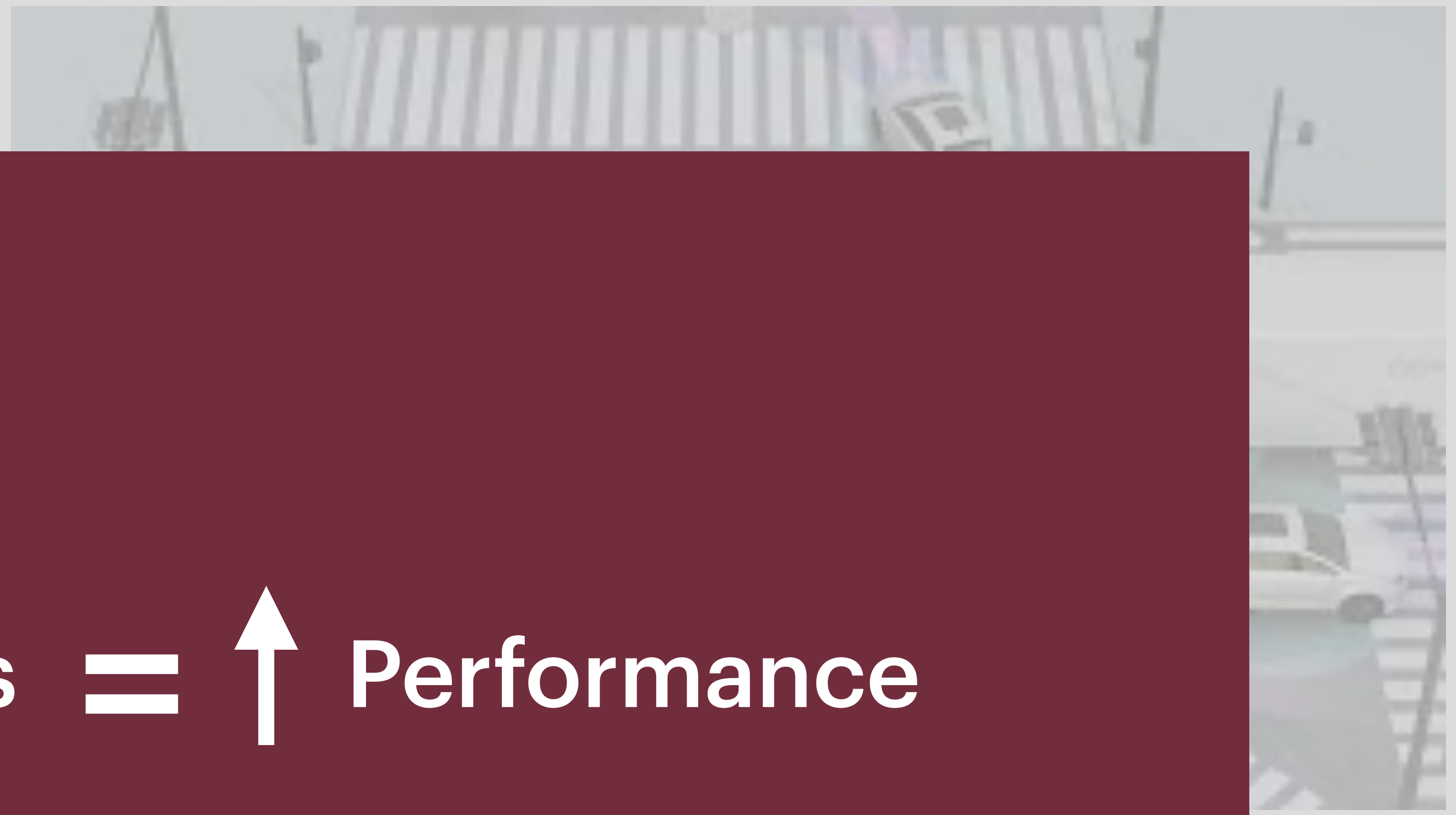
3^{361} states!

Learning in Games

We need scalable algorithms as well:

What makes good learning algorithms in games?

- Conver
  last-it
- Indeper
  agent
- No-regr
  agents
- **Scalabl**
  Many



3^{361} states!

This requires using function approximation

Deep RL

↑ Data + ↑ Compute + ↑ Complex Models = ↑ Performance

Does this still hold true when algorithms are used in MARL?

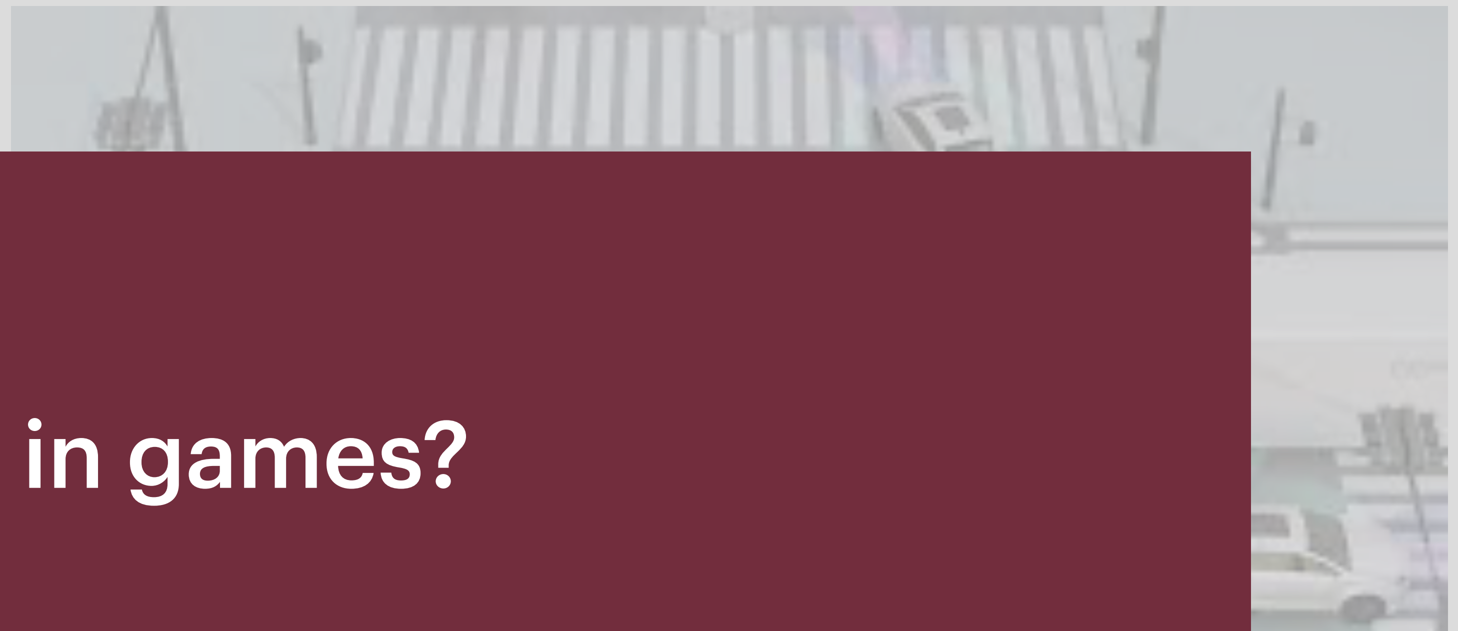
Learning in Games

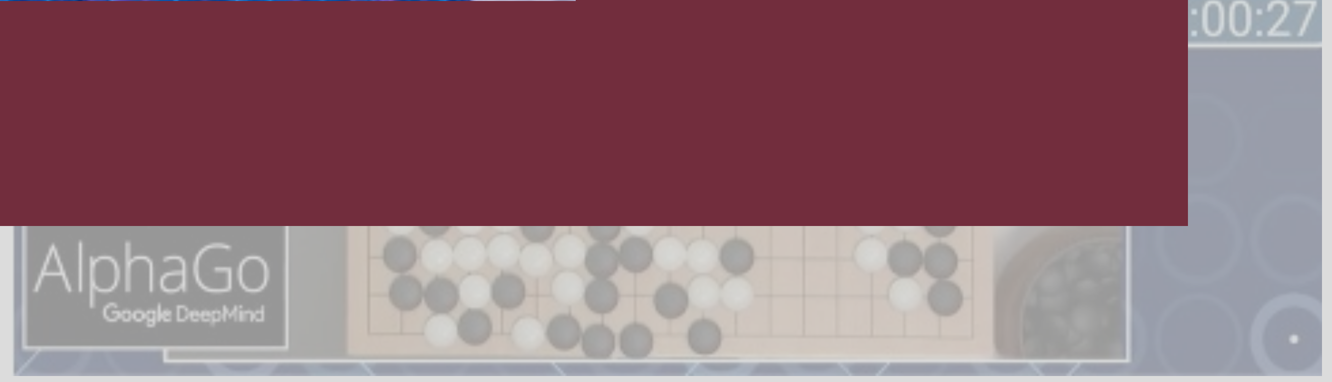
We need scalable algorithms as well:

What makes good learning algorithms in games?

- Conver
  last-it
- Indepen
  agent
- No-regr
  agents
- **Scalabl**
  Many

3^{361} states!





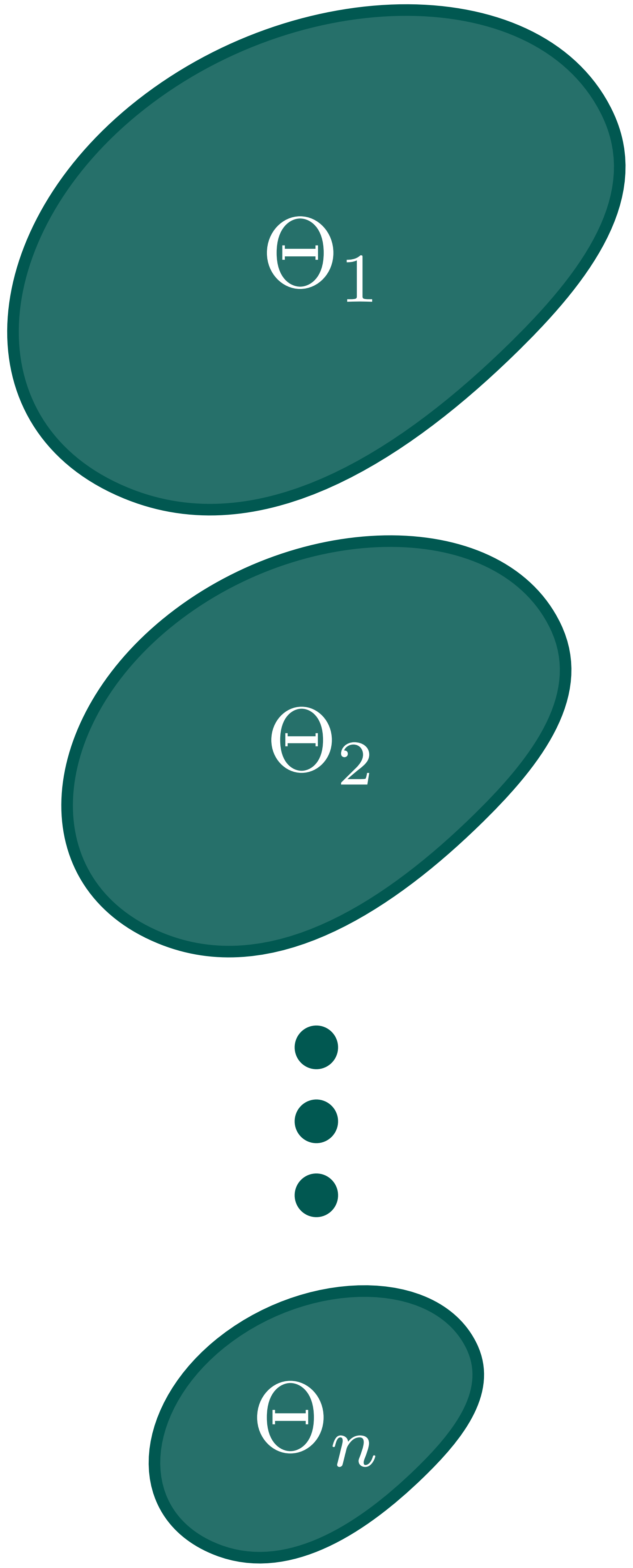
This requires using function approximation

How should we use function approximation in games?

Joint work with Tinashe Handina (Caltech)

Understanding Model Selection in Strategic Environments
NeurIPs 2024

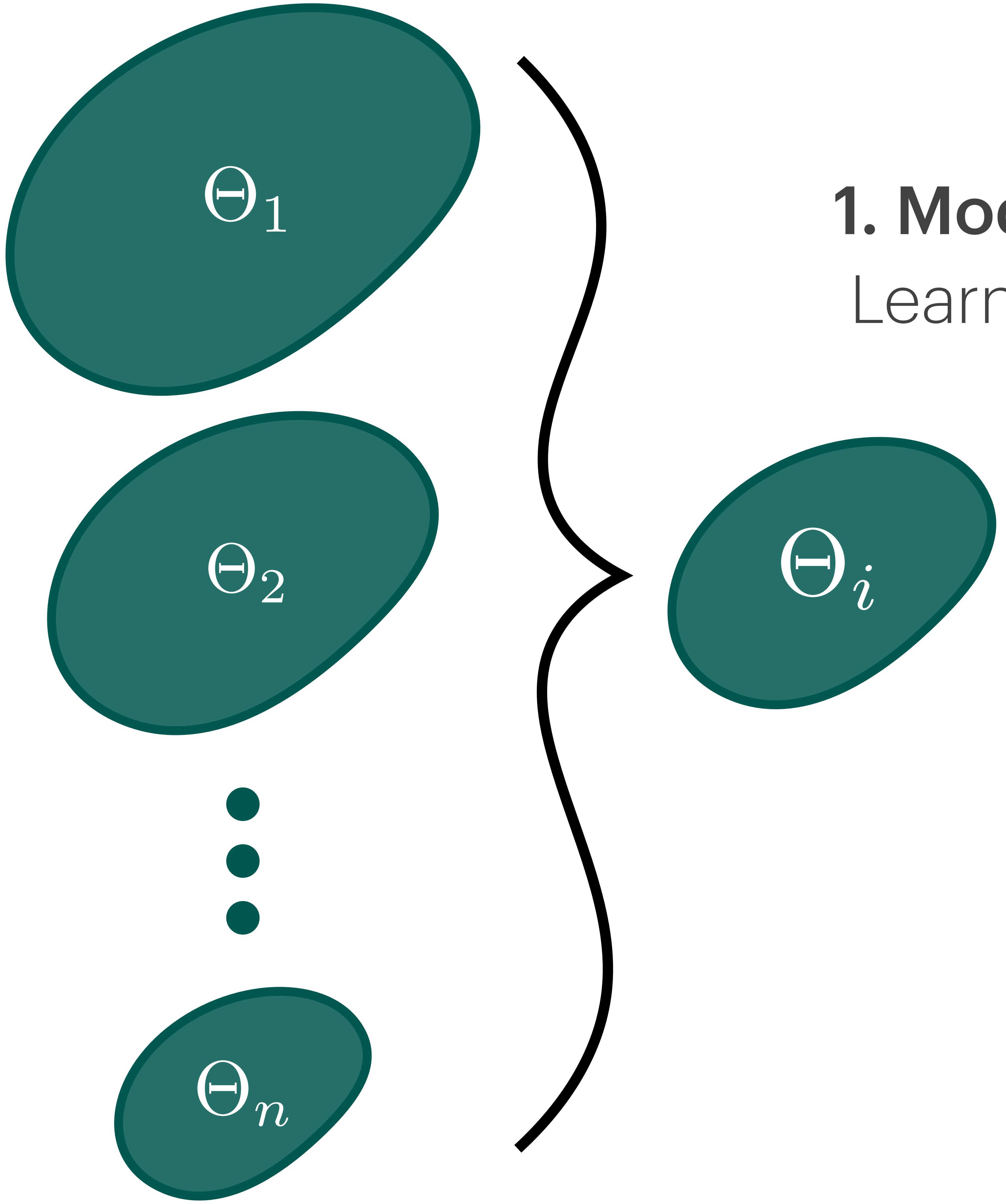
Deep MARL Pipeline



Learner has a menu of policy classes of decreasing expressivity

$$\Theta_1 \supset \Theta_2 \supset \ldots \supset \Theta_n$$

Deep MARL Pipeline



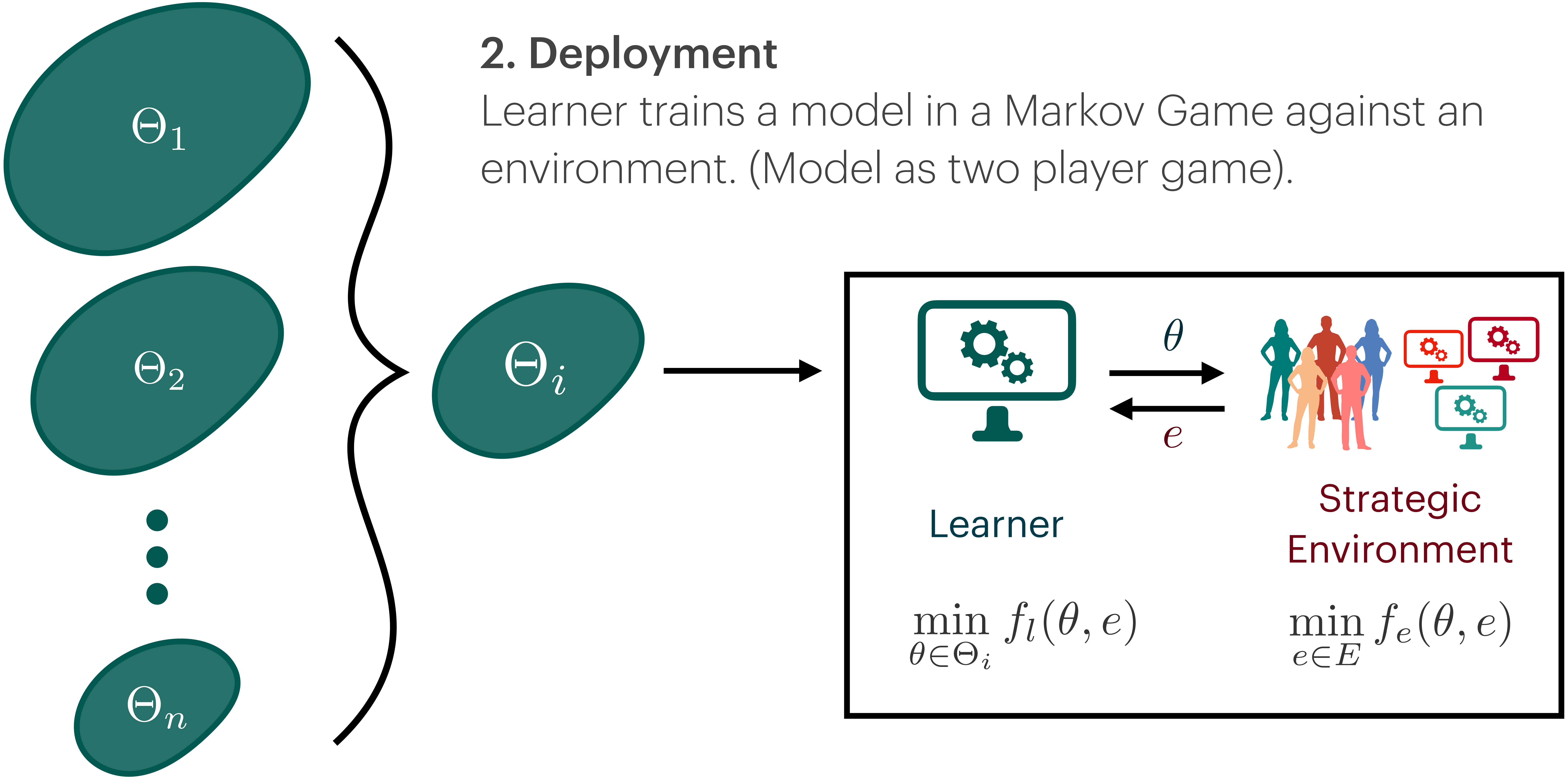
1. Model selection

Learner chooses a policy class

Deep MARL Pipeline

2. Deployment

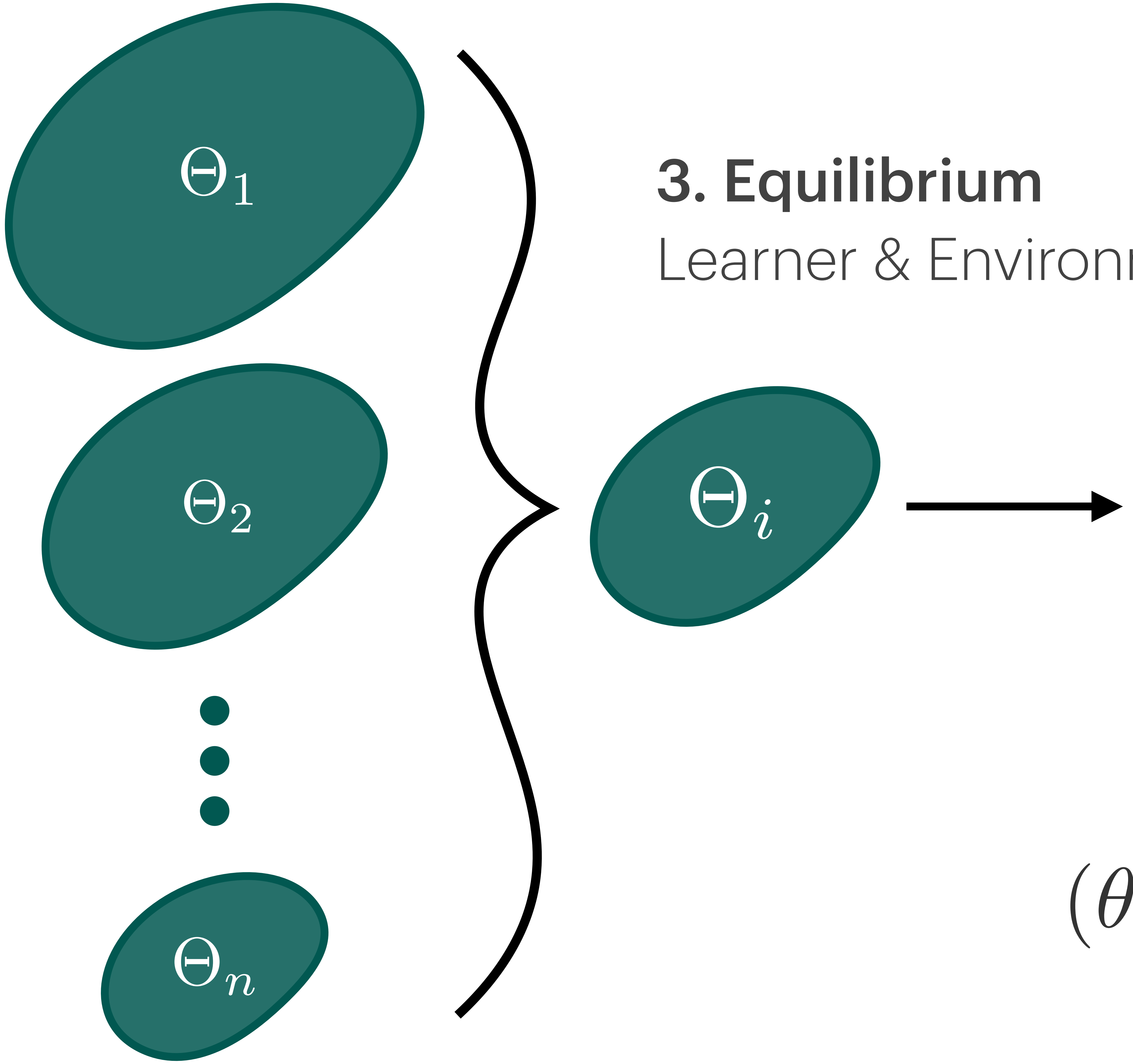
Learner trains a model in a Markov Game against an environment. (Model as two player game).

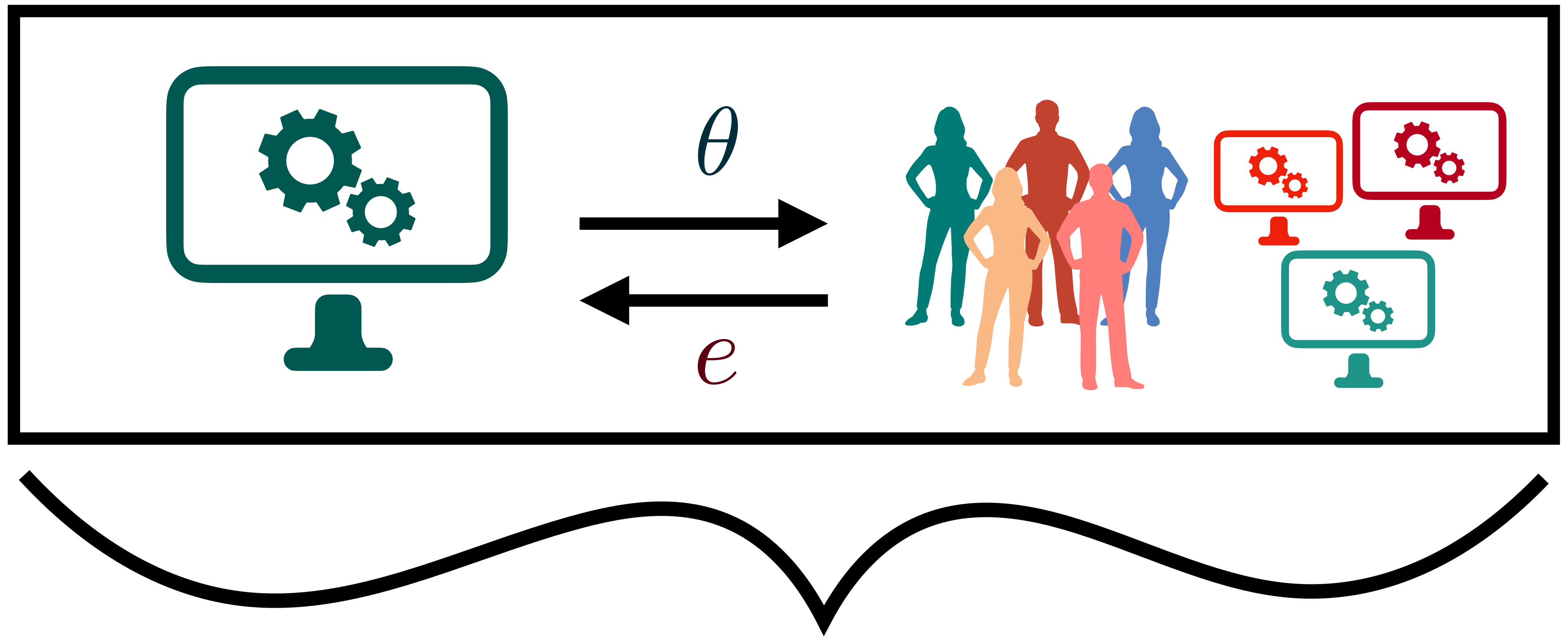


ML Deployment Pipeline

3. Equilibrium

Learner & Environment settle into a *Nash equilibrium.*



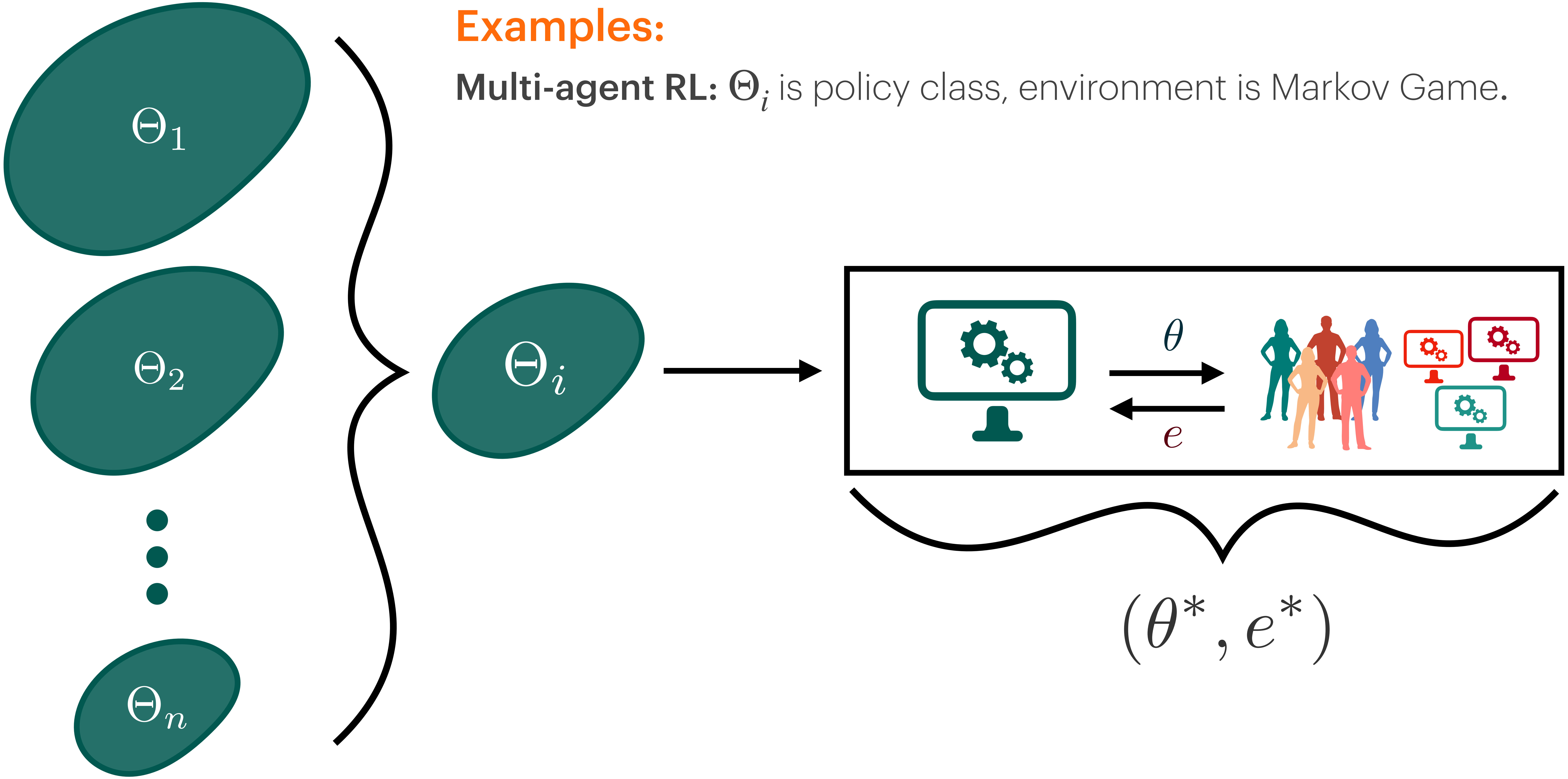


$$(\theta^*, e^*): \quad \begin{matrix} f_l(\theta^*, e^*) \leq f_l(\theta, e^*) & \forall \theta \in \Theta_i \\ f_e(\theta^*, e^*) \leq f_e(\theta^*, e) & \forall e \in E \end{matrix}$$

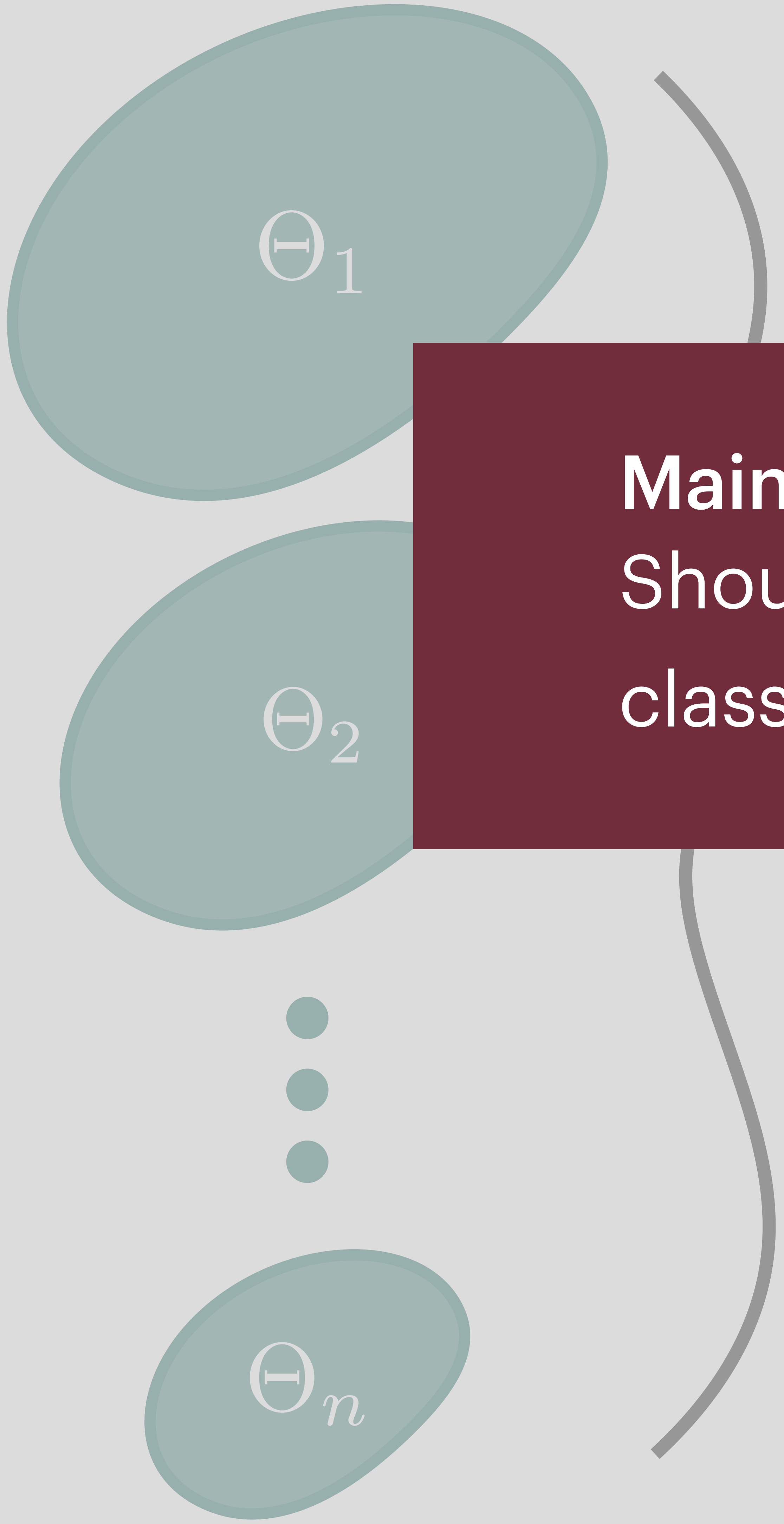
ML Deployment Pipeline

Examples:

Multi-agent RL: Θ_i is policy class, environment is Markov Game.



ML Deployment Pipeline

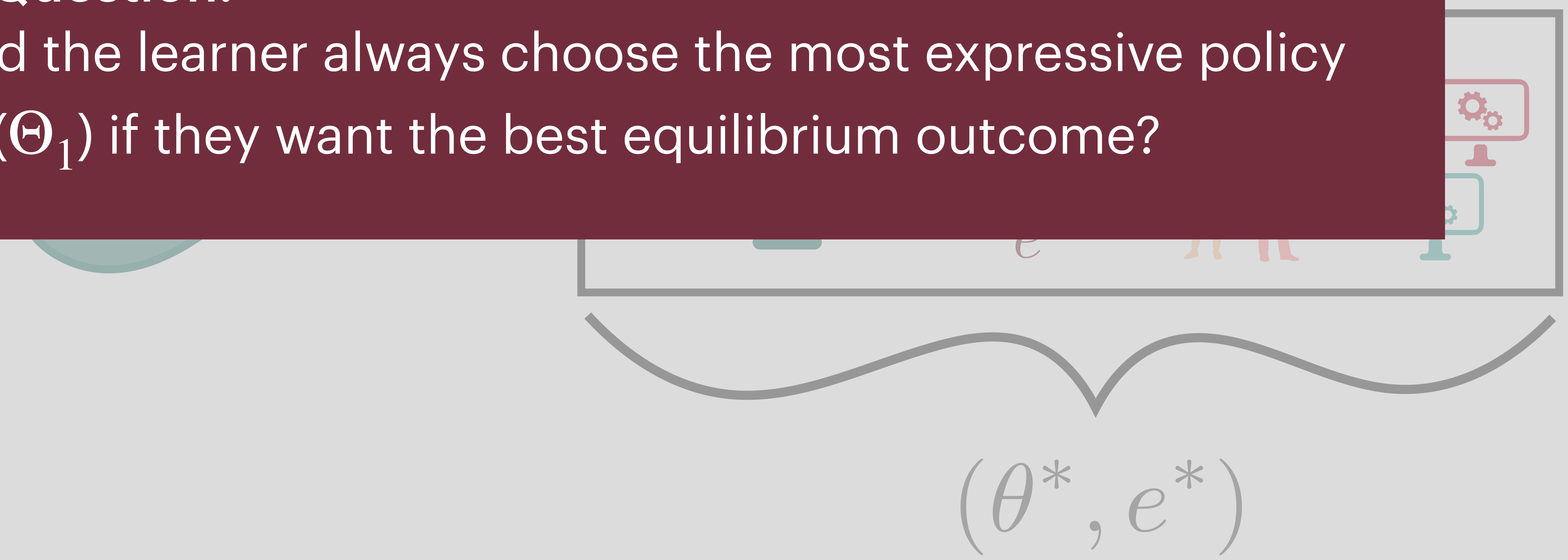


Examples:

Multi-agent RL: Θ_i is policy class, environment is Markov Game

Strategic Classification: Θ_i is a classifier, environment is a population of

Main Question:
Should the learner always choose the most expressive policy class (Θ_1) if they want the best equilibrium outcome?



Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

1. If the environment is **static** $(E = \{e\})$ — i.e., single-agent RL: **yes**

$$\min_{\theta \in \Theta_1} f_l(\theta, e) \leq \min_{\theta \in \Theta_i} f_l(\theta, e) \quad \text{if} \quad \Theta_i \subset \Theta_1$$

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

1. If the environment is **static** $(E = \{e\})$ — i.e., single-agent RL: **yes**

$$\min_{\theta \in \Theta_1} f_l(\theta, e) \leq \min_{\theta \in \Theta_i} f_l(\theta, e) \quad \text{if} \quad \Theta_i \subset \Theta_1$$

2. If the environment is **adversarial** $(f_e(\theta, e) = -f_l(\theta, e))$ — i.e., zero-sum Markov Game: ***yes***

$$\min_{\theta \in \Theta_1} \max_{e \in E} f_l(\theta, e) \leq \min_{\theta \in \Theta_i} \max_{e \in E} f_l(\theta, e) \quad \text{if} \quad \Theta_i \subset \Theta_1$$

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

1. If the environment is **static** $(E = \{e\})$: **yes**

$$\min_{\theta \in \Theta_1} f_l(\theta, e) \leq \min_{\theta \in \Theta_i} f_l(\theta, e) \quad \text{if} \quad \Theta_i \subset \Theta_1$$

2. If the environment is **adversarial** $(f_e(\theta, e) = -f_l(\theta, e))$ — i.e., zero-sum Markov Game: *yes*

$$\min_{\theta \in \Theta_1} \max_{e \in E} f_l(\theta, e) \leq \min_{\theta \in \Theta_i} \max_{e \in E} f_l(\theta, e) \quad \text{if} \quad \Theta_i \subset \Theta_1$$

3. The answer is ***no*** even in ***"well-behaved" general-sum games.***

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

Theorem.

If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

Highly structured games with a unique Nash equilibrium

There exists a coordinated change that improves upon both players outcomes (Not true in zero-sum games by definition)

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

Theorem.

If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

There ***always exists a unilateral restriction of their action set*** $\Theta' \subset \Theta_1$ such that the Nash equilibrium $(\theta', e') \in \Theta' \times E$ yields a better payoff.

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

Theorem.

If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

There ***always exists a unilateral restriction of their action set*** $\Theta' \subset \Theta_1$ such that the Nash equilibrium $(\theta', e') \in \Theta' \times E$ yields a better payoff.

Proof Idea:

Choosing a model class to optimize over is a form of ***commitment,*** which gives you a first mover advantage

Theorem.

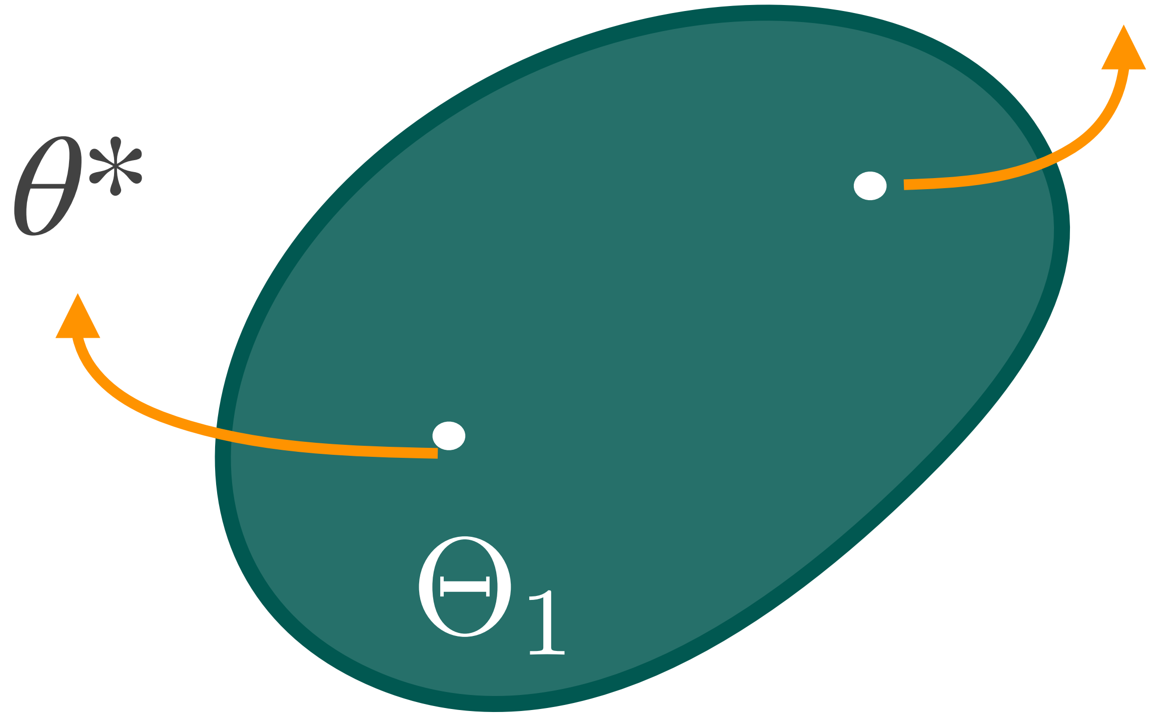
If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

There ***always exists a unilateral restriction of their action set*** $\Theta' \subset \Theta_1$ such that the Nash equilibrium $(\theta', e') \in \Theta' \times E$ yields a better payoff.

Proof Sketch:

Choosing a model class to optimize over is a form of ***commitment***

$$\theta_{SE} = \arg\min_{\theta \in \Theta_1} f_l(\theta, b(\theta)) : \ b(\theta) = \arg\min_{e \in E} f_e(\theta, e)$$



Theorem.

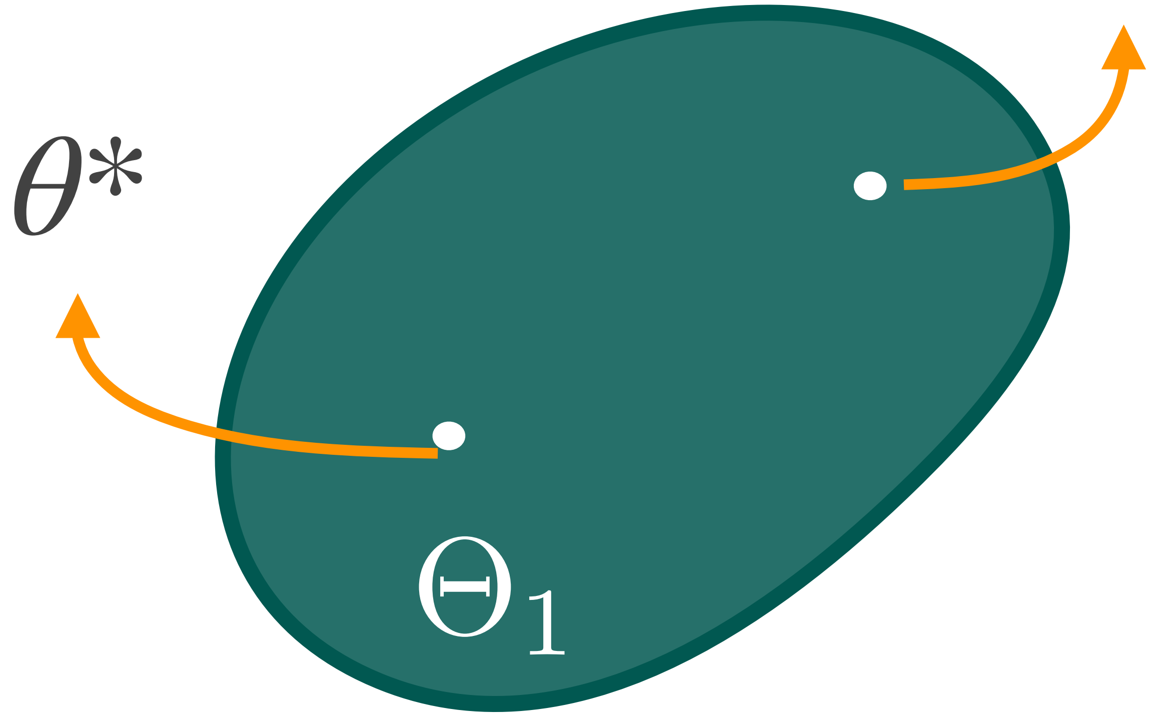
If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

There ***always exists a unilateral restriction of their action set*** $\Theta' \subset \Theta_1$ such that the Nash equilibrium $(\theta', e') \in \Theta' \times E$ yields a better payoff.

Proof Sketch:

Choosing a model class to optimize over is a form of ***commitment***

$$\theta_{SE} = \arg\min_{\theta \in \Theta_1} f_l(\theta, b(\theta)) : \; b(\theta) = \arg\min_{e \in E} f_e(\theta, e)$$



By definition: $f_l(\theta_{SE}, b(\theta_{SE})) \leq f(\theta^*, e^*)$

since $e* = b(\theta*)$

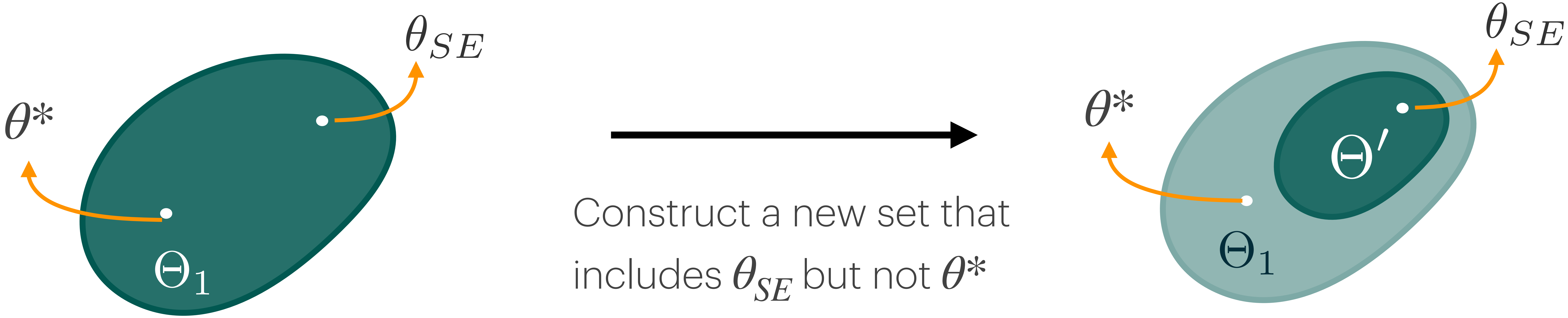
Theorem.

If the game is ***strongly monotone*** but the Nash equilibrium $(\theta^*, e^*) \in \Theta_1 \times E$ is ***not Pareto optimal***:

There ***always exists a unilateral restriction of their action set*** $\Theta' \subset \Theta_1$ such that the Nash equilibrium $(\theta', e') \in \Theta' \times E$ yields a better payoff.

Proof Sketch:

Choosing a model class to optimize over is a form of ***commitment***



Construct a new set that includes θ_{SE} but not θ^*

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

1. If the environment is **static** $(E = \{e\})$: **yes**

2. If the environment is **adversarial** $(f_e(\theta, e) = -f_l(\theta, e))$: ***yes***

3. The answer is ***no*** even in ***"well-behaved" general-sum games.***

Even with ***infinite data and infinite compute***, less expressive model classes can give better equilibrium outcomes.

Should not be too surprising: game theory is full of such examples!
e.g., *Braess' Paradox*, *LQG control*, *comparative statics*...

Main Question:

Should the learner always choose Θ_1 if they want the best equilibrium outcome?

1. If the environment is **static** $(E = \{e\})$: **yes**

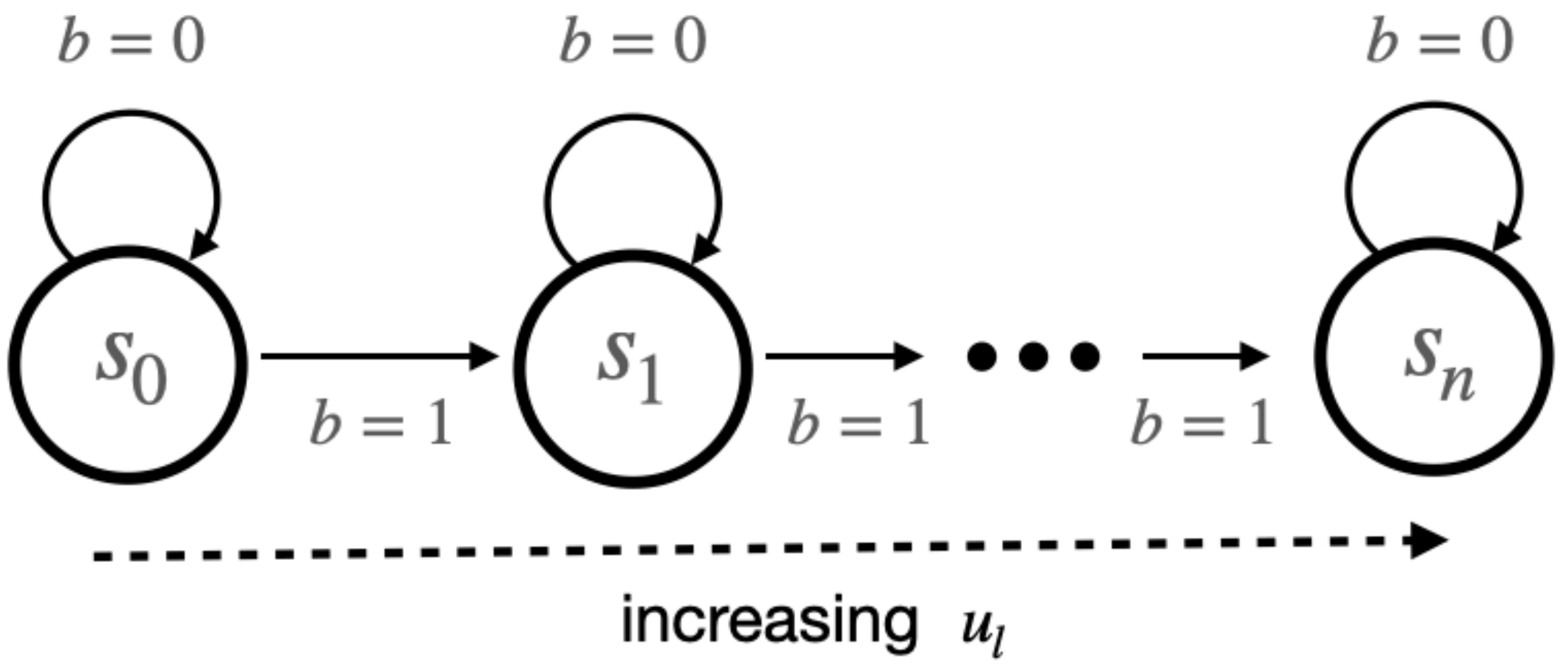
2. If the environment is **adversarial** $(f_e(\theta, e) = -f_l(\theta, e))$: ***yes***

3. The answer is ***no*** even in ***"well-behaved" general-sum games.***

Even with ***infinite data and infinite compute***, less expressive model classes can give better equilibrium outcomes.

An extreme example in MARL

2 players, ***n states***, *2 actions for each player {0,1}*



- Player 2 controls the state transitions
- Player 1 has increasing payoff in larger states

An extreme example in MARL

Player 1's menu of policy classes:

(for ease of visualization consider the 3-simplex)

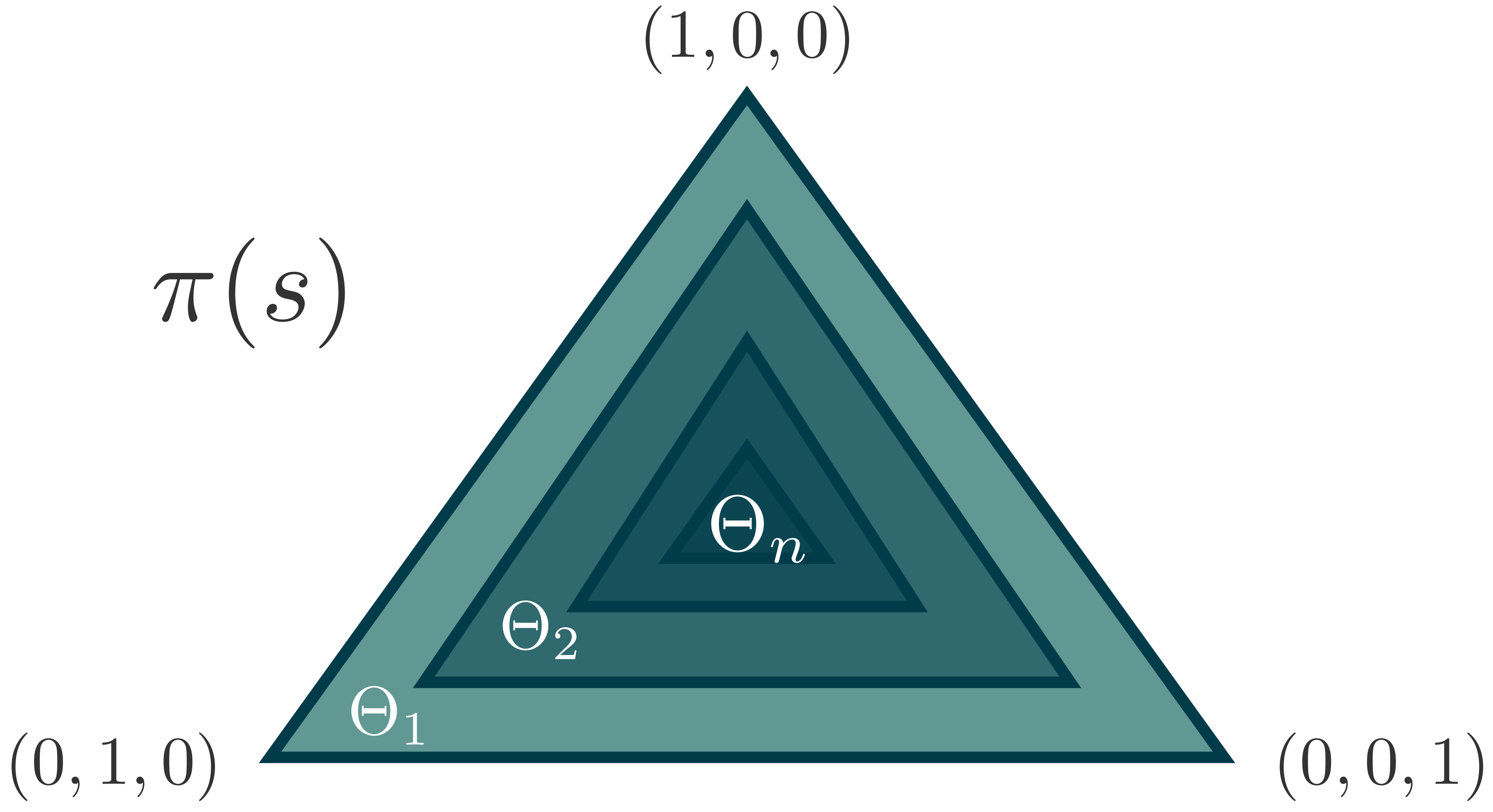
$$\Theta_1 : 1 - \bar{p}_1 \leq \pi_(s) \leq \bar{p}_1$$

$$\Theta_2 : 1 - \bar{p}_2 \leq \pi_(s) \leq \bar{p}_2$$

$$\vdots$$

$$\Theta_n : 1 - \bar{p}_n \leq \pi_(s) \leq \bar{p}_n$$

$$0.5 \leq \bar{p}_i \leq 1$$



An extreme example in MARL

Player 1’s menu of policy classes:

(for ease of visualization consider the 3-simplex)

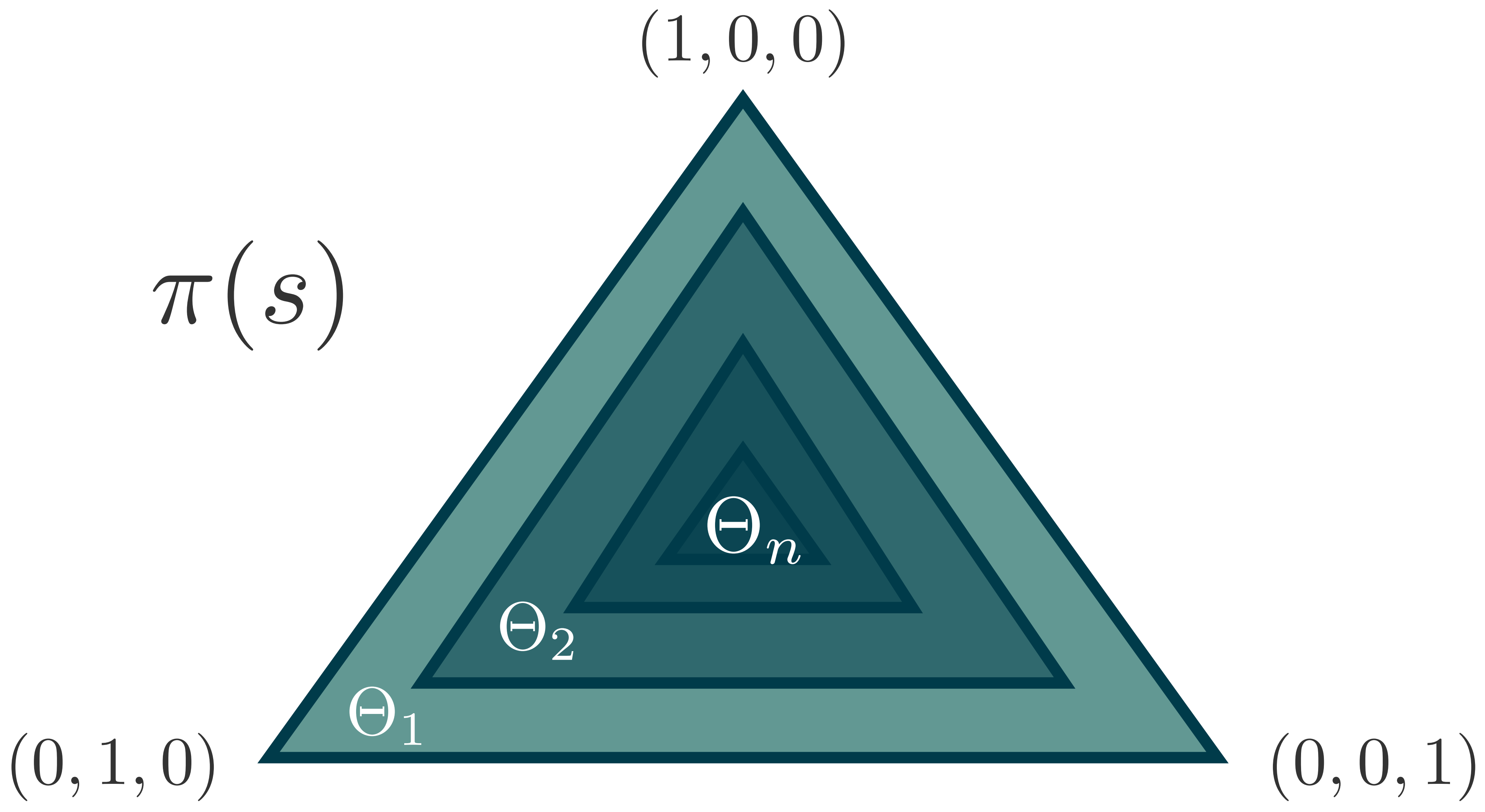
$$\Theta_1 : 1 - \bar{p}_1 \leq \pi_{(}s) \leq \bar{p}_1$$

$$\Theta_2 : 1 - \bar{p}_2 \leq \pi_{(}s) \leq \bar{p}_2$$

$$\vdots$$

$$\Theta_n : 1 - \bar{p}_n \leq \pi_{(}s) \leq \bar{p}_n$$

$$0.5 \leq \bar{p}_i \leq 1$$



- Player 1 has a ***dominant strategy*** in every state (a=0).

An extreme example in MARL

Player 1's menu of policy classes:

(for ease of visualization consider the 3-simplex)

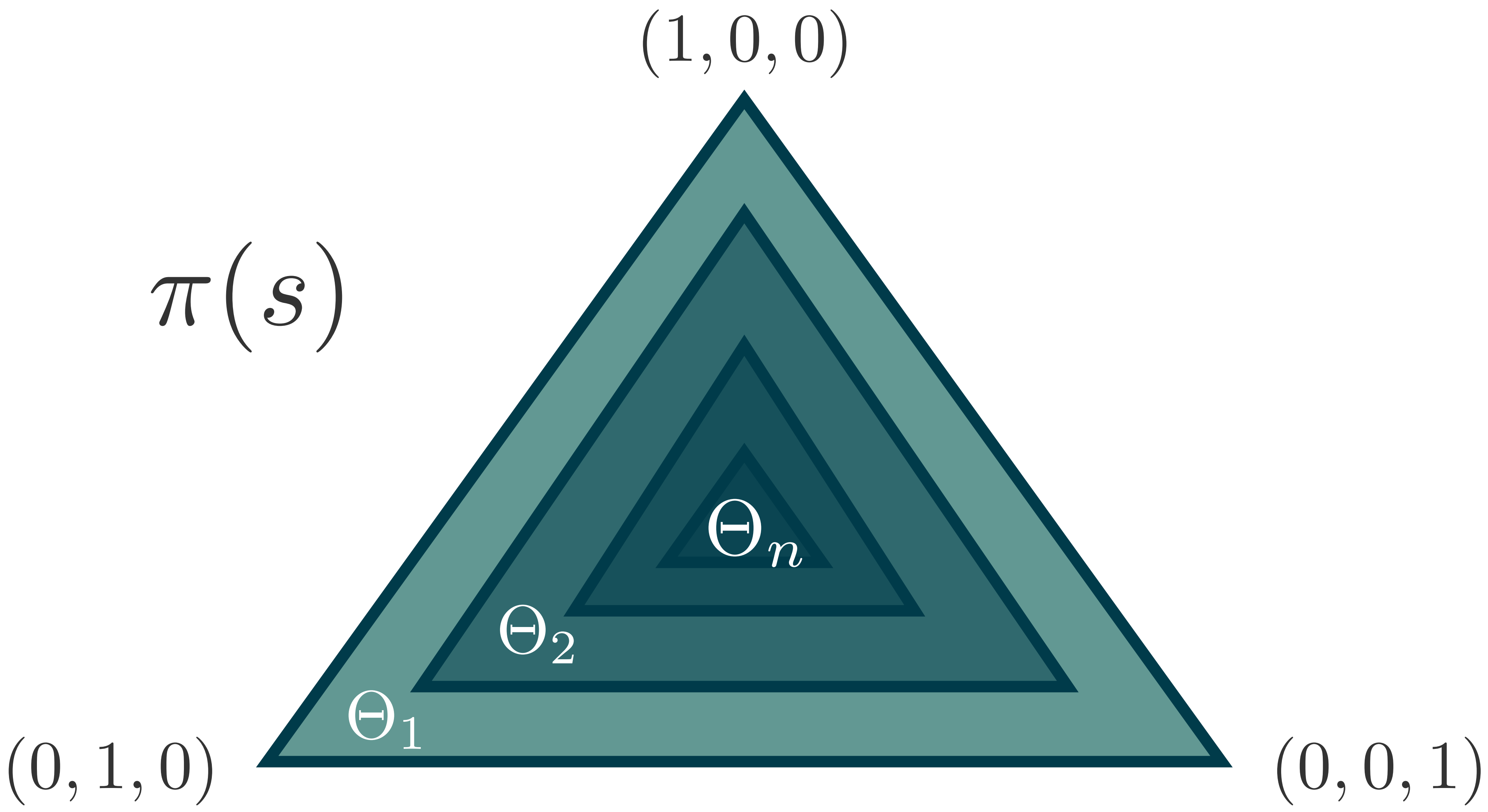
$$\Theta_1 : 1 - \bar{p}_1 \leq \pi_(s) \leq \bar{p}_1$$

$$\Theta_2 : 1 - \bar{p}_2 \leq \pi_(s) \leq \bar{p}_2$$

$$\vdots$$

$$\Theta_n : 1 - \bar{p}_n \leq \pi_(s) \leq \bar{p}_n$$

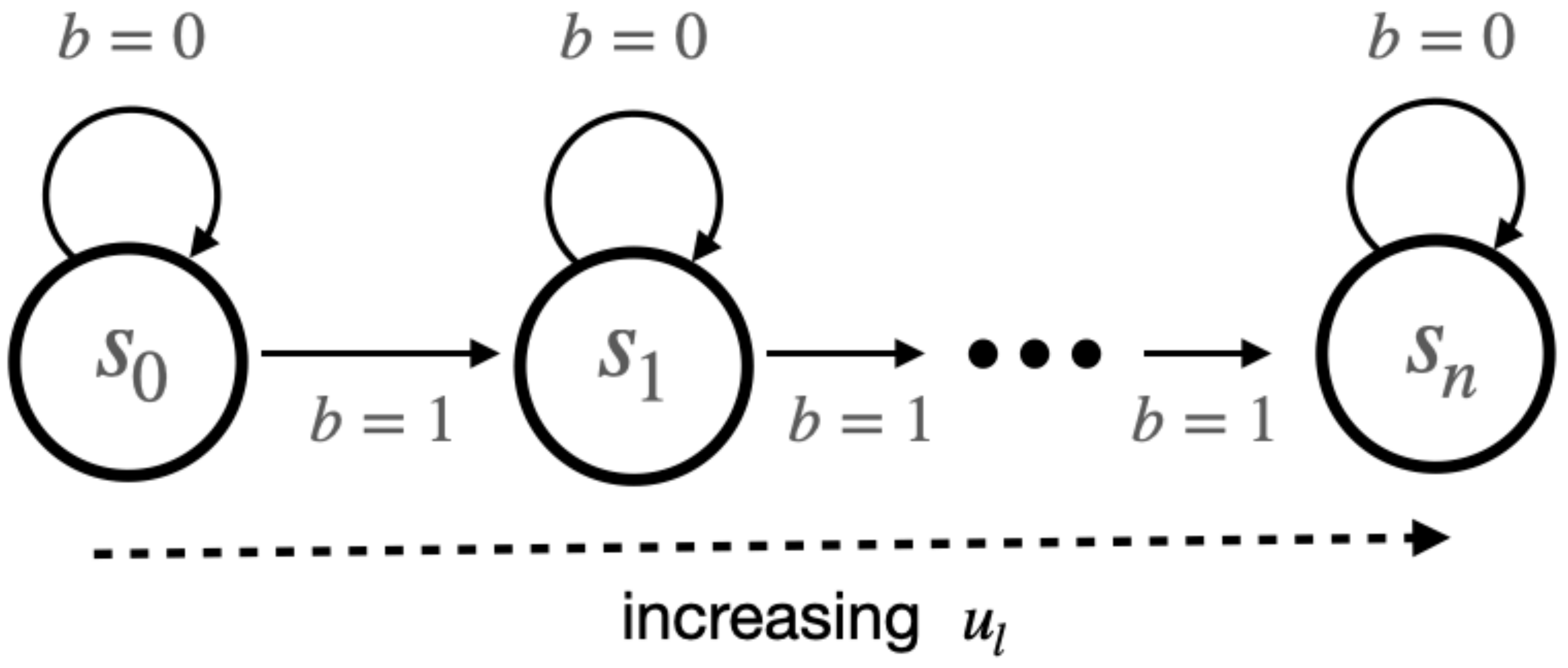
$$0.5 \leq \bar{p}_i \leq 1$$

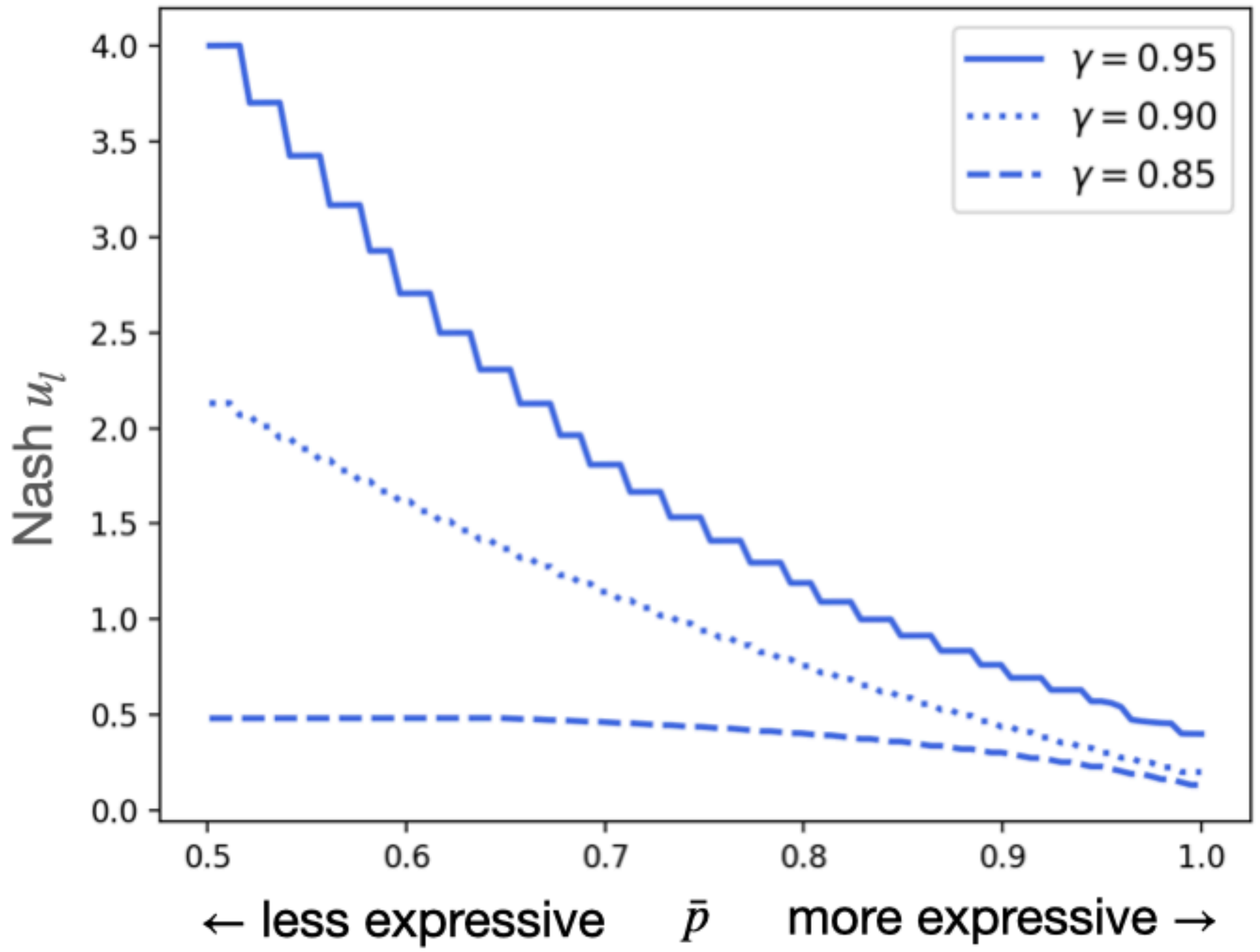


- Player 1 has a ***dominant strategy*** in every state (a=0).
- Their ***Nash eq strategy*** in each state for a policy class Θ_i is: $\pi_1^*(s) = (\bar{p}_i, 1 - \bar{p}_i)$

An extreme example in MARL

We construct the payoffs for player 2 such that the following scaling law holds for this game





We see reverse scaling: ↑ expressivity = ↓ performance

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms

3. **Further directions**
   1. The role of function approximation
      - Takeaway:
        - Choosing good function approximations in general-sum Markov games is ***non-trivial.***
          - Could hurt equilibrium performance by choosing a more expressive class (even in a full-information regime).
   2. Scalable algorithms for zero-sum games
   3. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms

3. **Further directions**
   1. The role of function approximation
      - Takeaway:
        - Choosing good function approximations in zero-sum Markov games is ***similar to RL.***
          - More expressive models will always help!
   2. Scalable algorithms for zero-sum games
   3. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

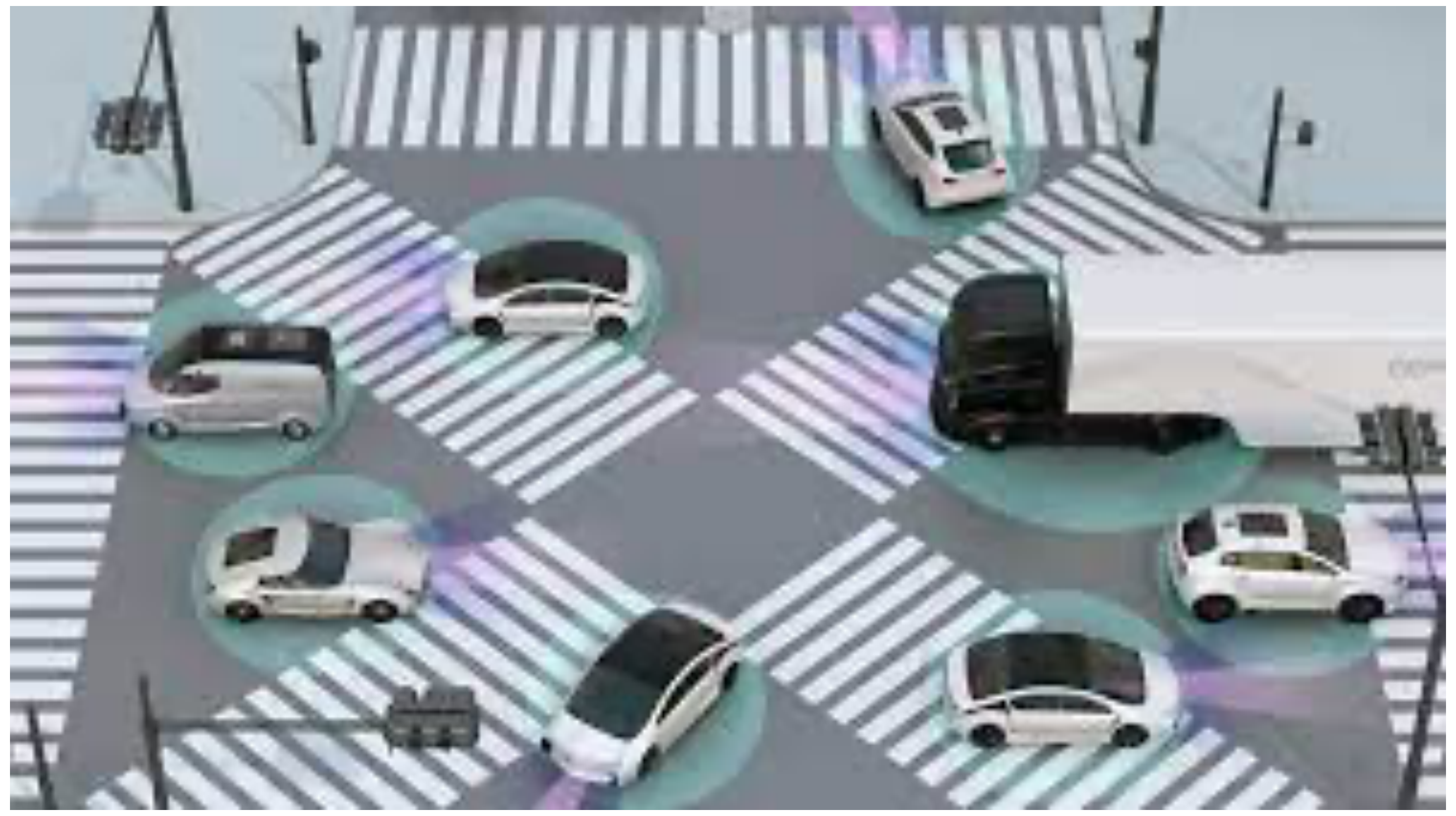
   iii. New equilibrium concepts

Function Approximation in Zero-Sum Markov Games

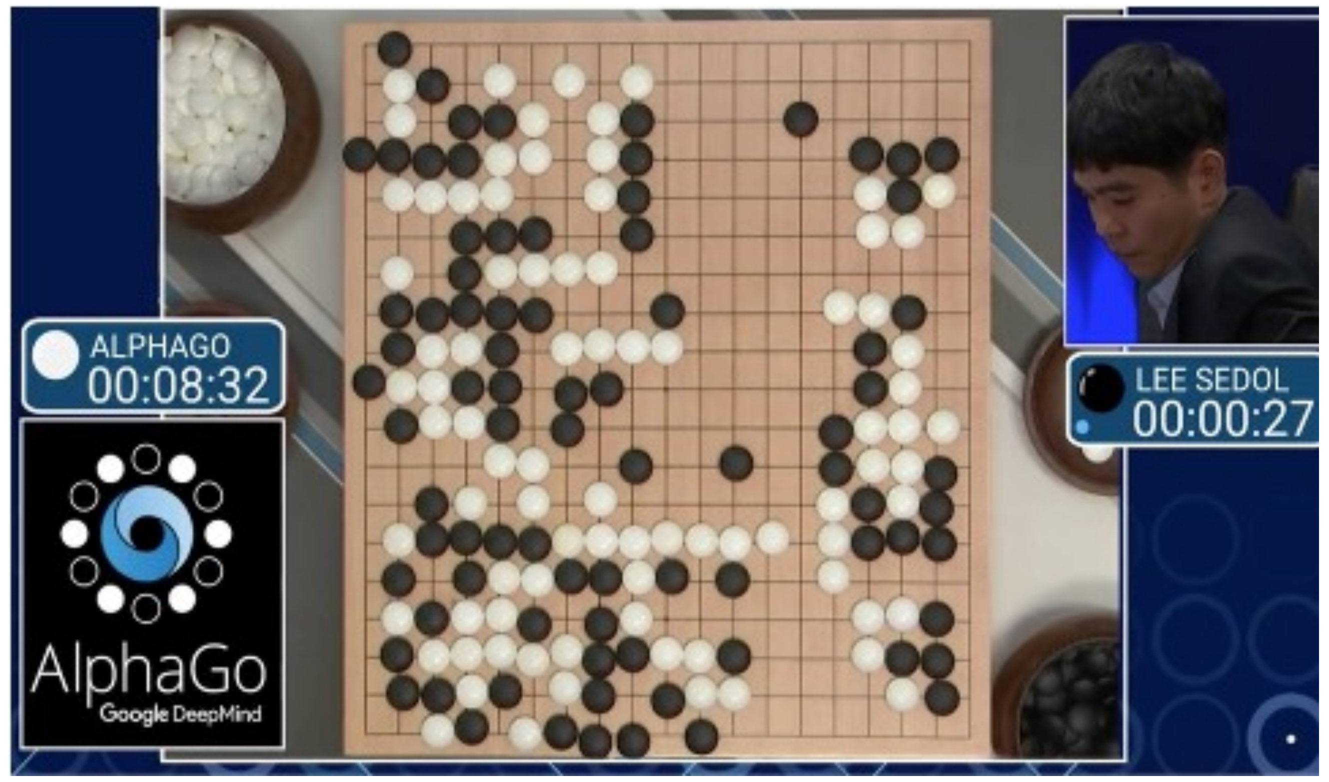
Can we design principled algorithms that ***use function approximation*** in zero-sum Markov Games?

What makes good learning algorithms in games?

- Convergent
  last-iterate, ergodic, best iterate convergence to Nash
- Independent learning
  agents should not know anything about their opponents utility
- No-regret
  agents should be “rational” (e.g., take advantage of naive opponents)
- **Scalable**
  Many real world applications have large state-spaces!



3^{361} states!



But, there's no need to reinvent the wheel!

Classical algorithms for learning in games in Economics for independent learning have strong convergence guarantees:

Smooth Fictitious-Play [Fudenberg & Kreps 1993]

Independent, last-iterate convergent in finite-action zero-sum games, no-regret[1]

- Trembling hand version of Fictitious-Play
  [Brown 1949]

- Well analyzed (asymptotically) via stochastic approximations
  [Hofbauer et al. 2002, Benaim & Hirsch 1999]

Last Iterate Convergence of Generalized Frank Wolfe in Monotone Variational Inequalities, Chen, Mazumdar (Under Submission 2023)

Today:

We develop a payoff-based version of smoothed Fictitious-Play that is:

1. ***Rational -*** can take advantage of stationary opponents
2. ***Independent -*** requires no knowledge of the opponents's policy
3. ***Convergent in the last iterate sense in two-player zero-sum games -*** not on average
4. ***Can be adjusted to work with function approximation -*** retains convergence guarantees with linear function approximation

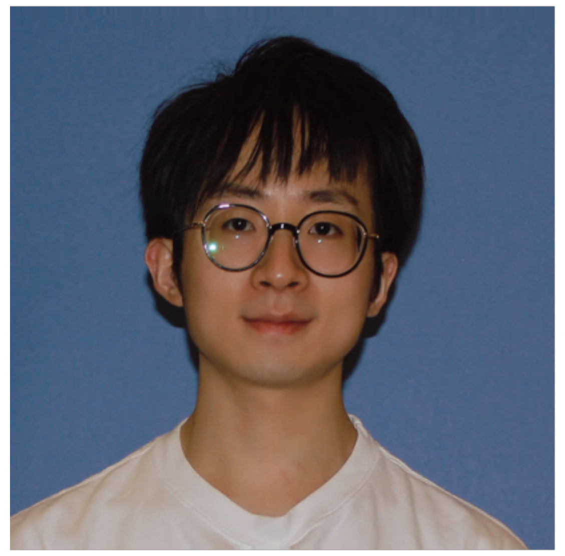
Key Idea: *two-timescale analyses of smoothed fictitious-play*

Today:

We develop a payoff-based version of smoothed Fictitious-Play that is:

1. ***Rational -*** can take advantage of stationary opponents
2. ***Independent -*** requires no knowledge of the opponents's policy
3. ***Convergent in the last iterate sense in two-player zero-sum games -*** not on average
4. ***Can be adjusted to work with function approximation -*** retains convergence guarantees with linear function approximation

Zaiwei Chen
(Caltech)



Kaiqing Zhang
(UMD)

Asu Ozdaglar
(MIT)

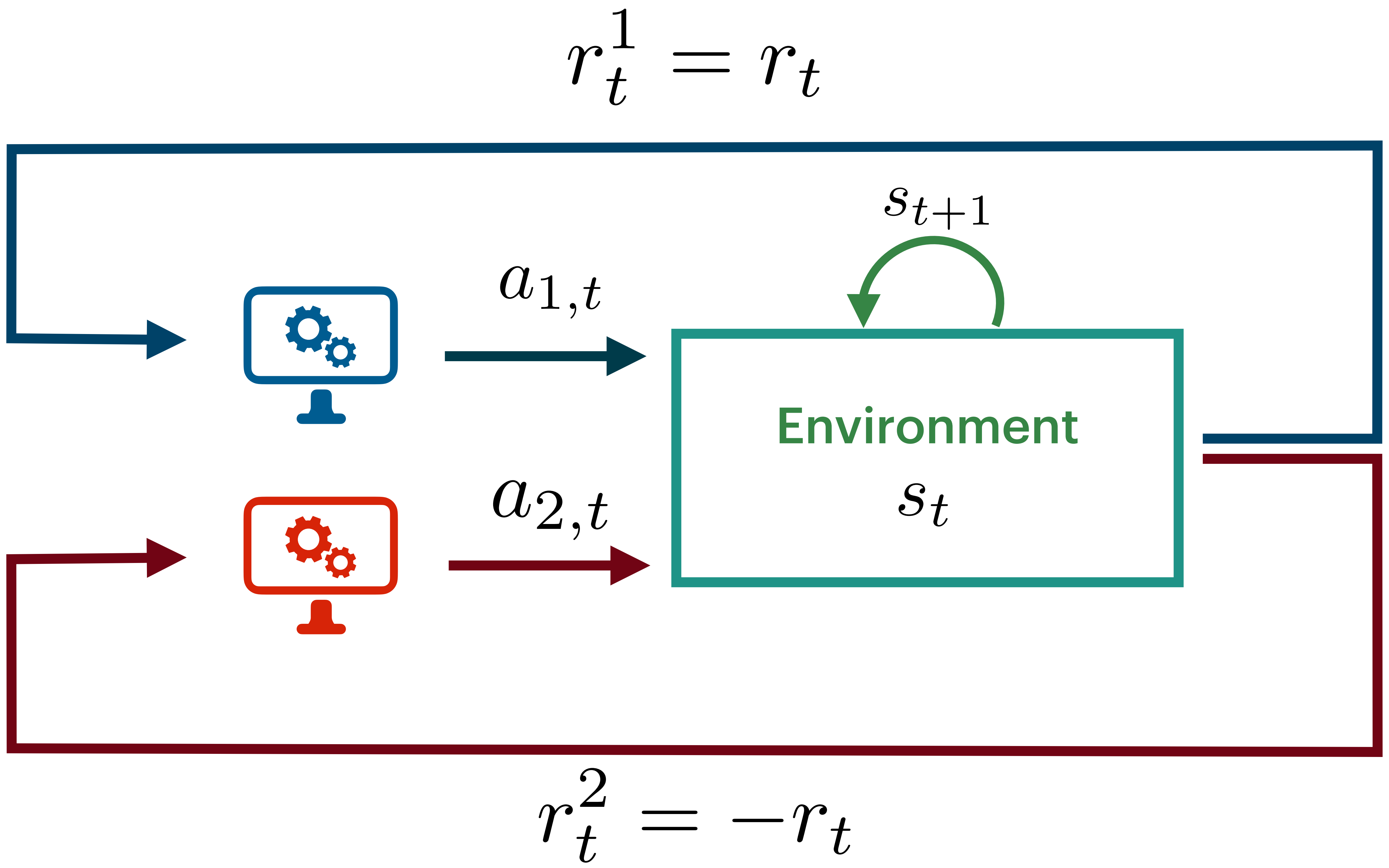
Adam Wierman
(Caltech)

Zero-Sum Markov Games

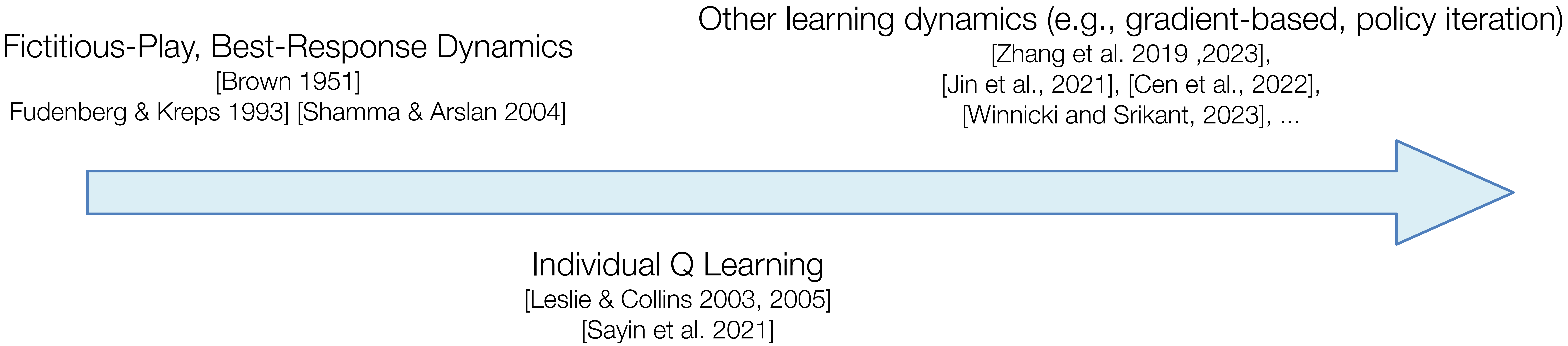
- Finite Action Spaces: $\mathcal{A}_1, \mathcal{A}_2$
- Finite (but large) State Spaces: $\mathcal{S}$
- Players only observe rewards and states

$$U_1(\pi_1, \pi_2) = \mathbb{E}_{\pi_1, \pi_2, P}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

$$U_2(\pi_1, \pi_2) = \mathbb{E}_{\pi_1, \pi_2, P}\left[\sum_{t=0}^{\infty} -\gamma^t r_t\right]$$



A brief look at prior work on infinite horizon zero-sum Markov Games



Fictitious-Play, Best-Response Dynamics

[Brown 1951]
Fudenberg & Kreps 1993] [Shamma & Arslan 2004]

Other learning dynamics (e.g., gradient-based, policy iteration)

[Zhang et al. 2019 ,2023],
[Jin et al., 2021], [Cen et al., 2022],
[Winnicki and Srikant, 2023], ...

Individual Q Learning

[Leslie & Collins 2003, 2005]
[Sayin et al. 2021]

All are missing at least one of the following

- Convergence in Markov Games
- Payoff-Based Independent Learning
- Function Approximation
- Last-Iterate Convergence Guarantees

Warm-Up: Smooth Fictitious-Play in Matrix Games

Smoothed fictitious-play (SFP). [Fudenberg & Kreps 1993]

Player's *approximately best-respond* to their opponents empirical history of play

At round t:

Belief over opponent's strategy

Compute strategy: $x_{1,t} = \arg\max_{\pi \in \Delta} \ \pi^T A \pi_{2,t} + \tau \nu(\pi)$

Play $a_{1,t} \sim x_{1,t}$ observe $a_{2,t}$

Update your belief: $\pi_{2,t+1} = \pi_{2,t} + \frac{1}{t+1}(e(a_{2,t}) - \pi_{2,t})$

Warm-Up: Smooth Fictitious-Play in Matrix Games

Smoothed fictitious-play (SFP). [Fudenberg & Kreps 1993]

Player's *approximately best-respond* to their opponents empirical history of play

At round t:

known payoffs

entropy

Compute strategy: $x_{1,t} = \arg\max_{\pi \in \Delta} \; \pi^T A \pi_{2,t} + \tau \nu(\pi)$

Play $a_{1,t} \sim x_{1,t}$ observe $a_{2,t}$

Update your belief: $\pi_{2,t+1} = \pi_{2,t} + \frac{1}{t+1}(e(a_{2,t}) - \pi_{2,t})$

empirical history of play of opponent

Requires observation of opponents' actions!

This is a case with neither bandit or full information feedback but somewhere in between

Smoothed Fictitious-Play and Bandit Feedback

Implementing SFP requires observation of your opponent's actions

At round t:

Compute strategy: $x_{1,t} = \arg\max_{\pi \in \Delta} \; \pi^T A \pi_{2,t} + \tau \nu(\pi)$

Play $a_{1,t} \sim x_{1,t}$ observe $a_{2,t}$

Update your belief: $\pi_{2,t+1} = \pi_{2,t} + \frac{1}{t+1}(e(a_{2,t}) - \pi_{2,t})$

Suppose you do not know A and cannot observe $a_{2,t}$

How should a player learn an estimate of $A\pi_2$ using only their own payoffs?

Smoothed Fictitious-Play and Bandit Feedback

Implementing SFP requires observation of your opponent's actions

At round t:

Compute strategy: $x_{1,t} = \arg\max_{\pi \in \Delta} \quad \pi^T A \pi_{2,t} + \tau \nu(\pi)$

Play $a_{1,t} \sim x_{1,t}$ observe $a_{2,t}$

Update your belief: $\pi_{2,t+1} = \pi_{2,t} + \frac{1}{t+1}(e(a_{2,t}) - \pi_{2,t})$

Suppose you do not know A and cannot observe $a_{2,t}$

How should a player learn an estimate of $A\pi_2$ using only their own payoffs?

Use Stochastic Approximation (TD-Learning) on $A\pi_2 - q = 0$

Doubly-Smoothed Best-Response Dynamics

At round t:

Play $a_{1,t} \sim \pi_{1,t}$ observe r_t

Update your belief: $q_{t+1}(a) = q_t(a) + \alpha_t \mathbb{I}_{a=a_{1,t}}(r_t - q_t(a))$

Compute smooth best-response: $x_{1,t} = \arg\max_{\pi \in \Delta} \ \pi^T q_t + \tau\nu(\pi)$

Update strategy: $\pi_{1,t+1} = \pi_{1,t} - \beta_t(\pi_{1,t} - x_{1,t})$

← Bandit feedback: $\mathbb{E}\left[r_t\right] = [A]_{a_{1,t},a_{2,t}}$

← TD Learning on fast time-scale

← Update policy on slow time-scale

Main Idea: If $q_t \to A\pi_{2,t}$ on a fast timescale this algorithm mimics best-response dynamics on the slow timescale.

Not a new idea... using multiple timescales to stabilize Q learning was proposed in [Leslie & Collins 2003]!

Convergent Multiple-Timescales Reinforcement Learning Algorithms in Normal Form Games, Leslie & Collins [2003]

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics

```
At round $t$:
    Play $a_{1,t} \sim \pi_{1,t}$ observe $r_t$                                          ← Bandit feedback: $\mathbb{E}[r_t] = [A]_{a_{1,t},a_{2,t}}$
    Update your belief: $q_{t+1}(a) = q_t(a) + \alpha_t \mathbb{I}_{a=a_{1,t}}(r_t - q_t(a))$   ← TD Learning on fast time-scale
    Compute smooth best-response: $x_{1,t} = \arg\max_{\pi \in \Delta} \pi^T q_t + \tau \nu(\pi)$
    Update strategy: $\pi_{1,t+1} = \pi_{1,t} - \beta_t(\pi_{1,t} - x_{1,t})$                ← Update policy on slow time-scale
```

Main Idea: If $q_t \to A\pi_{2,t}$ on a fast timescale this algorithm mimics best-response dynamics on the slow timescale.

We show that this is achievable using a single timescale: $\alpha_t = C\beta_t$

Convergent Multiple-Timescales Reinforcement Learning Algorithms in Normal Form Games, Leslie & Collins [2003]

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics

```
At round $t$:
    Play $a_{1,t} \sim \pi_{1,t}$ observe $r_t$
    Update your belief: $q_{t+1}(a) = q_t(a) + \alpha_t \mathbb{I}_{a=a_{1,t}}(r_t - q_t(a))$
    Compute smooth best-response: $x_{1,t} = \arg\max_{\pi \in \Delta} \pi^T q_t + \tau \nu(\pi)$
    Update strategy: $\pi_{1,t+1} = \pi_{1,t} - \beta_t(\pi_{1,t} - x_{1,t})$
```

This algorithm is:

- Independent
- Payoff-Based
- No Markov Games
- No Function Approximation

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (NeurIPS 2023)

Last-Iterate Convergence of Doubly Smoothed BRD

For a matrix game, define the regularized Nash Gap as:

$$V_\tau(\pi_1, \pi_2) = \max_{\bar{\pi}_2 \in \Delta} \pi_1^T A \bar{\pi}_2 - \tau \nu(\bar{\pi}_2) - \left(\min_{\bar{\pi}_1 \in \Delta} \bar{\pi}_1^T A \pi_2 + \tau \nu(\bar{\pi}_1) \right)$$

Entropy

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Last-Iterate Convergence of Doubly Smoothed BRD

For a matrix game, define the regularized Nash Gap as:

$$V_\tau(\pi_1, \pi_2) = \max_{\bar{\pi}_2 \in \Delta} \pi_1^T A \bar{\pi}_2 - \tau \nu(\bar{\pi}_2) - \left(\min_{\bar{\pi}_1 \in \Delta} \bar{\pi}_1^T A \pi_2 + \tau \nu(\bar{\pi}_1) \right)$$

Theorem. Doubly Smoothed BRD with $\alpha_t = O\left(\frac{1}{t}\right)$ and $\beta_t = C\alpha_t$ and $\tau > 0$ satisfies:

$$\mathbb{E}\left[V_\tau(\pi_{1,T}, \pi_{2,T})\right] = O\left(\frac{1}{T}\right)$$

Finite-time last iterate convergence with only Bandit Feedback

Last iterate convergence to Nash in space of policies achievable with τ-softmax policies (Nash distribution).

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Last-Iterate Convergence of Doubly Smoothed BRD

For a matrix game, define the regularized Nash Gap as:

$$V_\tau(\pi_1, \pi_2) = \max_{\bar{\pi}_2 \in \Delta} \pi_1^T A \bar{\pi}_2 - \tau\nu(\bar{\pi}_2) - \left(\min_{\bar{\pi}_1 \in \Delta} \bar{\pi}_1^T A \pi_2 + \tau\nu(\bar{\pi}_1) \right)$$

Theorem. Doubly Smoothed BRD with $\alpha_t = O\left(\frac{1}{t}\right)$ and $\beta_t = C\alpha_t$ and $\tau > 0$ satisfies:

$$\mathbb{E}\left[V_\tau(\pi_{1,T}, \pi_{2,T})\right] = O\left(\frac{1}{T}\right)$$

Proof sketch:

1. View the fast timescale as constructing a variance-reduced estimator of the player's marginalized payoffs for use in smoothed Fictitious-Play.
2. Show that smoothed Fictitious-play has last-iterate convergence in the regularized Nash gap.

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Last-Iterate Convergence of Doubly Smoothed BRD

For a matrix game, define the regularized Nash Gap as:

$$V_\tau(\pi_1, \pi_2) = \max_{\bar{\pi}_2 \in \Delta} \pi_1^T A \bar{\pi}_2 - \tau \nu(\bar{\pi}_2) - \left(\min_{\bar{\pi}_1 \in \Delta} \bar{\pi}_1^T A \pi_2 + \tau \nu(\bar{\pi}_1) \right)$$

Theorem. Doubly Smoothed BRD with $\alpha_t = O\left(\frac{1}{t}\right)$ and $\beta_t = C\alpha_t$ and τ appropriately chosen satisfies:

$$\mathbb{E}\left[V_0(\pi_{1,T}, \pi_{2,T})\right] = O\left(\frac{1}{T^{1/8}}\right)$$

The convergence to the true Nash is much slower due to trade offs in stepsize selection
(but still converges at the fastest known rate for last-iterate convergence with bandit feedback)

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Last-Iterate Convergence of Doubly Smoothed BRD

For a matrix game, define the regularized Nash Gap as:

$$V_\tau(\pi_1, \pi_2) = \max_{\bar{\pi}_2 \in \Delta} \pi_1^T A \bar{\pi}_2 - \tau\nu(\bar{\pi}_2) - \left(\min_{\bar{\pi}_1 \in \Delta} \bar{\pi}_1^T A \pi_2 + \tau\nu(\bar{\pi}_1) \right)$$

Theorem. Do ppriately

chosen satisfi

$\left(T^{1/8}\right)$

Convergence in Matrix Games but how about MARL?

The convergence to the true Nash is much slower due to trade offs in stepwise selection (but still converges at the fastest known rate for last-iterate convergence with bandit feedback)

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

From Matrix to Markov Games

Classic Q-Learning

$$Q^*(s,a) = E_{s,a}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a')\right]$$

Minimax Q-Learning

$$Q^*(s,a_1,a_2) = R(s,a_1,a_2) + \gamma E_{s'|s,a_1,a_2}\left[\min_{\pi_1}\max_{\pi_2}\ \pi_1^T Q^*(s')\pi_2\right]$$

Requires solving a zero-sum game

Like in the single-agent setting, the optimal Q-function is a fixed point of a contraction mapping in the ℓ_∞ norm

$$Q_1^* = \mathcal{H}(Q_1^*)$$

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

Main Idea:

Use SBRD to solve—online— a matrix game in each state but adjust payoffs with slowly updated estimates of future discounted rewards.

At round t:

Play $a_{1,t} \sim \pi_{1,t}(\cdot|s_t)$ observe r_t move to new state s_{t+1}

Update your belief: $q_{t+1}(s,a) = q_t(s,a) + \alpha_t \mathbb{I}_{s=s_t, a=a_{1,t}}(r_t - q_t(s,a) + \gamma v_t(s_{t+1})) \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}$

future payoffs

Compute smooth best-response: $x_{1,t} = \arg\max_{\pi \in \Delta} \ \pi^T q_t + \tau \nu(\pi)$

Update strategy: $\pi_{1,t+1} = \pi_{1,t} - \beta_t(\pi_{1,t+1} - x_{1,t})$

Every K steps update value estimate: $v_{t+1} = \begin{cases} \pi_{1,t+1}(\cdot|s)^T q_{t+1}(s,\cdot) \ \forall s \in \mathcal{S} \ ; & (t+1) \bmod K = 0 \\ v_t \quad \text{otherwise} & \end{cases}$

min-max value iteration on a slower timescale.

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

At round t:

Play $a_{1,t} \sim \pi_{1,t}(\cdot|s_t)$ observe r_t move to new state s_{t+1}

Update your belief: $q_{t+1}(s,a) = q_t(s,a) + \alpha_t \mathbb{I}_{s=s_t, a=a_{1,t}}(r_t - q_t(s,a) + \gamma v_t(s_{t+1})) \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}$

future payoffs

Compute smooth best-response: $x_{1,t} = \arg\max_{\pi \in \Delta} \; \pi^T q_t + \tau \nu(\pi)$

Update strategy: $\pi_{1,t+1} = \pi_{1,t} - \beta_t(\pi_{1,t+1} - x_{1,t})$

Every K steps update value estimate: $v_{t+1} = \begin{cases} \pi_{1,t+1}(\cdot|s)^T q_{t+1}(s,\cdot) \; \forall s \in \mathcal{S} \;\; ; & (t+1) \bmod K = 0 \\ v_t \quad \text{otherwise} \end{cases}$

Algorithm is independent and payoff-based

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} U(\pi_1, \bar{\pi}_2)$$

Captures the distance from “Nash”

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} \; U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} \; U(\pi_1, \bar{\pi}_2)$$

Theorem. Suppose there exists a pair of policies (π_1, π_2) such that the induced Markov Chain is irreducible and ergodic, then Doubly Smoothed BRD satisfies:

$$\mathbb{E}\left[NG(\pi_{1,T}, \pi_{2,T})\right] \leq O\left(\frac{1}{T^{1/8}}\right)$$

Last iterate convergence

No slower than the matrix case.
Due to geometric convergence of value iteration

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} U(\pi_1, \bar{\pi}_2)$$

Theorem. Suppose there exists a pair of policies (π_1, π_2) such that the induced Markov Chain is irreducible and ergodic, then Doubly Smoothed BRD satisfies:

$$\mathbb{E}\left[NG(\pi_{1,T}, \pi_{2,T})\right] \leq O\left(\frac{1}{T^{1/8}}\right)$$

Payoff-based (i.e. bandit feedback), independent algorithm with ***finite-time iterate-convergence*** to Nash equilibrium in zero-sum Markov Games.

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Doubly-Smoothed Best-Response Dynamics in MARL

Theorem. Suppose there exists a pair of policies (π_1, π_2) such that the induced Markov Chain is irreducible and ergodic, then Doubly Smoothed BRD satisfies:

$$\mathbb{E}\left[NG(\pi_{1,T}, \pi_{2,T})\right] \leq O\left(\frac{1}{T^{1/8}}\right)$$

Proof sketch:

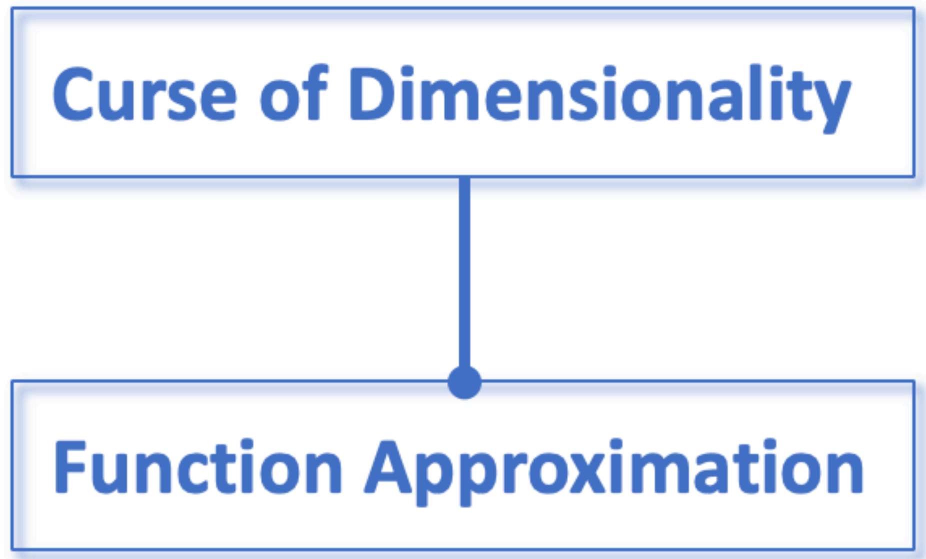
1. Solve matrix game on a fast timescale using the Matrix-game version of the algorithm. Becomes highly nontrivial due to time inhomogeneous Markovian Noise, and the loss of zero-sum structure since agents may have different beliefs over future payoffs.
2. Use regularized Nash gap as Lyapunov function for the slow timescale.
3. timescale of value iteration to help stabilize the learning at the fast timescale.

A Finite-Sample Analysis of Payoff-Based Independent Learning in Zero-Sum Stochastic Games, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

How to incorporate function approximation?

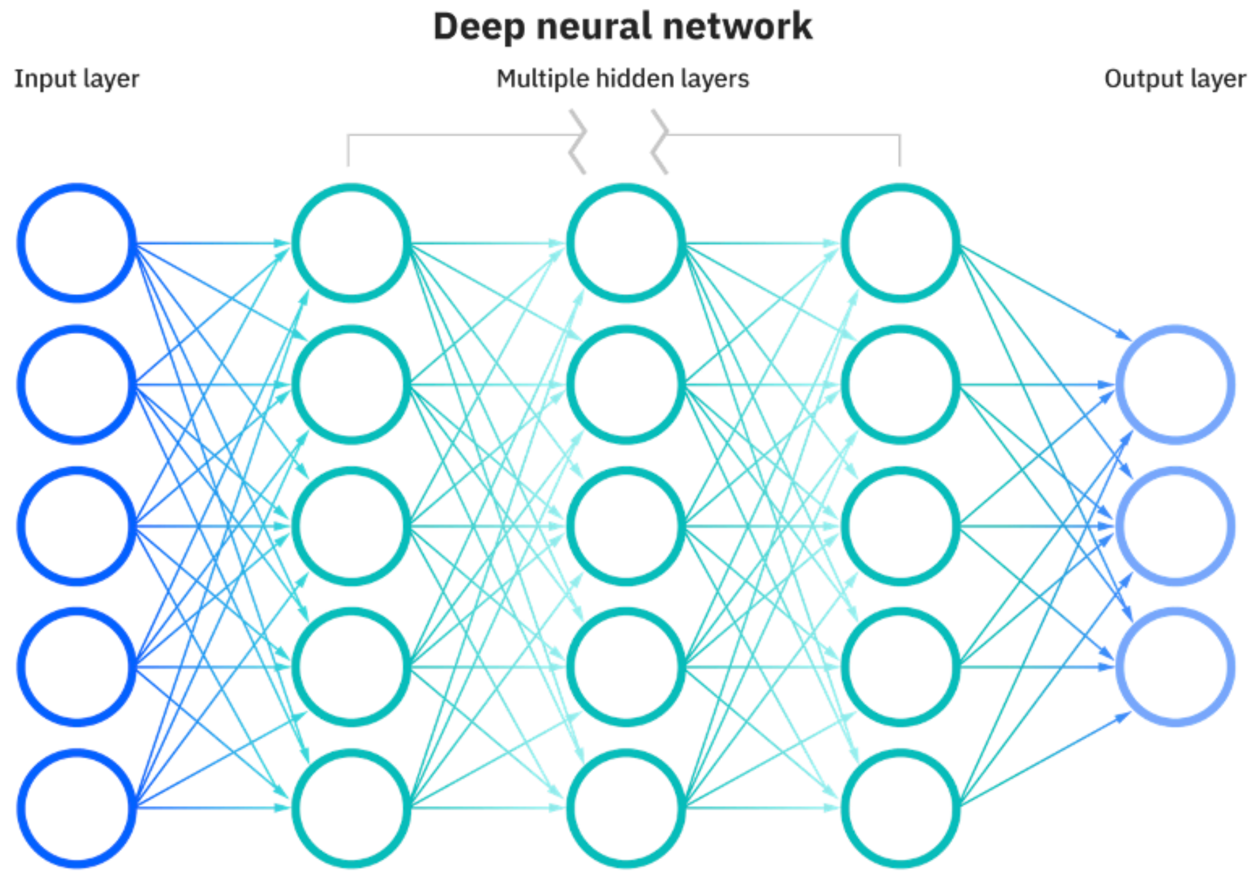
Scalability: algorithms for deep reinforcement learning

Deep Q-Networks [Mnih et al. 2015]



Curse of Dimensionality

Function Approximation

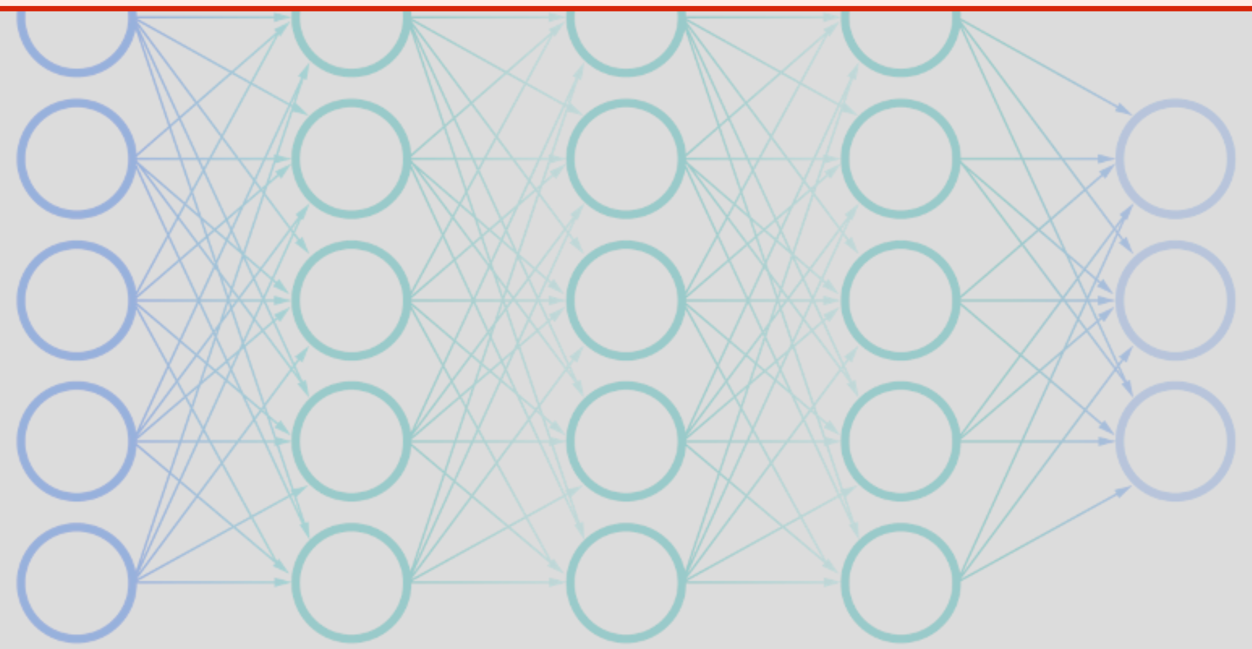


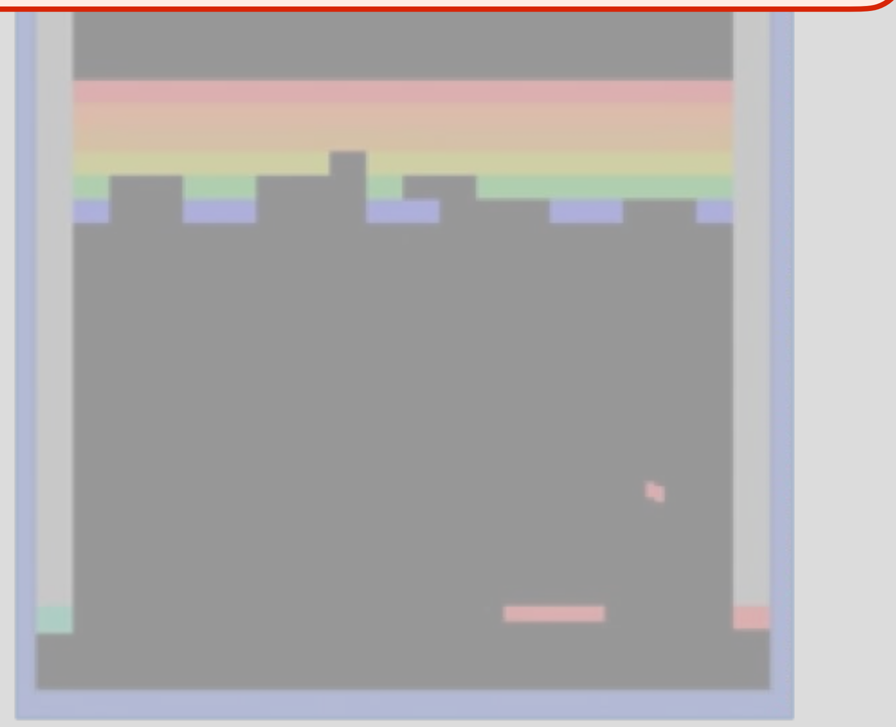
How to incorporate function approximation?

Scalability: algorithms for deep reinforcement learning

Deep Q-Networks [Mnih et al. 2015]

Key Idea: *DQN* can be viewed as smoothed fictitious-play under a basis transform*





Incorporating Function Approximation

Suppose we have a respecified functions class parametrized by weights w: $\{Q_w \in \mathcal{S} \times \mathcal{A}\}$

At round t:

Play $a_{1,t} \sim \pi_{1,t}(\cdot|s_t)$ observe r_t move to new state s_{t+1}

target network

Update your belief: update weights via SGD: $w_{t+1} = \mathcal{P}\left(w_t - \alpha_t \nabla_w \|\mathcal{H}_{\bar{\pi}}(Q_{\bar{w}}) - Q_{w_t}\|^2\right)$

Update strategy:

$$\theta_{t+1} = \theta_t + \beta_t(w_{t+1} - \theta_t) \qquad \pi_{t+1} = \arg\max_{\pi \in \Delta} \pi^T Q_{\theta_{t+1}} + \tau\nu(\pi)$$

Every K steps synchronize policy, weights: $\bar{\pi} = \pi_t \quad \bar{w} = w_t$

Note: $\beta_t << \alpha_t$

If $\beta = 1$ then we recover vanilla DQN

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation, Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Relationship with Smoothed Fictitious-Play

Suppose we are in a state-less regime and the fast timescales have converged:

$$\gamma = 0 \qquad w_t = \arg\min_w \|\mathcal{H}_{\bar{\pi}}(Q_{\bar{w}}) - Q_w\|^2$$

$$x_t = \arg\max_{\pi \in \Delta} \pi^T Q_t^2 + \tau\nu(\pi)$$

$$Q^1_{w_{t+1}} = Q^1_{w_t} + \beta_t(Rx_t - Q^1_{\theta_t})$$

DQN

$$x_t = \arg\max_{\pi \in \Delta} \pi^T R\pi_t^2 + \tau\nu(\pi)$$

$$\pi^1_{t+1} = \pi^1_t + \beta_t(x_t - \pi_t)$$

Smoothed Best-Reponse Dynamics

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Relationship with Smoothed Fictitious-Play

Suppose we are in a state-less regime and the fast timescales have converged:

$$\gamma = 0 \qquad\qquad w_t = \arg\min_{w} \|\mathcal{H}_{\bar{\pi}}(Q_{\bar{w}}) - Q_w\|^2$$

Perform a change of basis: $u_i = R^{-1}Q_i$

$$x_t = \arg\max_{\pi \in \Delta} \pi^T R R^{-1} Q_t^2 + \tau \nu(\pi)$$

$$R^{-1}Q^1_{w_{t+1}} = R^{-1}Q^1_{w_t} + \beta_t(x_t - R^{-1}Q^1_{\theta_t})$$

DQN

$$x_t = \arg\max_{\pi \in \Delta} \pi^T R \pi_t^2 + \tau \nu(\pi)$$

$$\pi^1_{t+1} = \pi^1_t + \beta_t(x_t - \pi_t)$$

Smoothed Best-Reponse Dynamics

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Relationship with Smoothed Fictitious-Play

Suppose we are in a state-less regime and the fast timescales have converged:

$$\gamma = 0 \qquad w_t = \arg\min_w \|\mathcal{H}_{\bar{\pi}}(Q_{\bar{w}}) - Q_w\|^2$$

Perform a change of basis: $u_i = R^{-1} Q_i$

$$x_t = \arg\max_{\pi \in \Delta} \pi^T R u_t^2 + \tau \nu(\pi)$$

$$u_{t+1}^1 = u_t^1 + \beta_t (x_t - u_t^1)$$

DQN

$$x_t = \arg\max_{\pi \in \Delta} \pi^T R \pi_t^2 + \tau \nu(\pi)$$

$$\pi_{t+1}^1 = \pi_t^1 + \beta_t (x_t - \pi_t)$$

Smoothed Best-Reponse Dynamics

Under a change of basis the two algorithms have the same limiting dynamics

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Finite-Time Last Iterate Convergence

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} U(\pi_1, \bar{\pi}_2)$$

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Finite-Time Last Iterate Convergence

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} \ U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} \ U(\pi_1, \bar{\pi}_2)$$

Theorem. Suppose there exists a pair of policies (π_1, π_2) such that the induced Markov Chain is irreducible and ergodic, and both agents use DQN* with *linear function approximation* then DQN* satisfies:

$$\mathbb{E}\left[NG(\pi_{1,T}, \pi_{2,T})\right] \le \underbrace{O\left(\frac{\gamma^T}{c_1(\tau)}\right)}_{\text{Convergence of min-max value iteration}} + \underbrace{O\left(\frac{1}{c_2(\tau)K}\right)}_{\text{Convergence of inner loop}} + \underbrace{O(\tau)}_{\text{Error due to soft-max policies}} + \underbrace{\epsilon_{\text{approx}}}_{\text{Function approximation error}}$$

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

Finite-Time Last Iterate Convergence

Consider the Nash-Gap:

$$NG(\pi_1, \pi_2) = \max_{\bar{\pi}_1} \; U(\bar{\pi}_1, \pi_2) - \min_{\bar{\pi}_2} \; U(\pi_1, \bar{\pi}_2)$$

Theorem. Suppose there exists a pair of policies (π_1, π_2) such that the induced Markov Chain is irreducible and ergodic, and both agents use DQN* with *linear function approximation* then DQN* satisfies:

$$\mathbb{E}\left[NG(\pi_{1,T}, \pi_{2,T})\right] \leq O\left(\frac{\gamma^T}{c_1(\tau)}\right) + O\left(\frac{1}{c_2(\tau)K}\right) + O(\tau) + \epsilon_{\text{approx}}$$

Independent, payoff-based, algorithm that provably incorporates function approximation in zero-sum Markov Games while having last-iterate convergence

Last Iterate Convergence of Deep Q Networks in Zero-Sum Markov Games with Linear Function Approximation , Chen, Zhang, Mazumdar, Ozdaglar, Wierman (2023)

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms

3. **Further directions**
   1. The role of function approximation
   2. Scalable algorithms for zero-sum games
   3. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   - Takeaway: Small tweaks to the ***DQN algorithm*** allow it to have strong convergence guarantees in zero-sum games under ***linear function approximation.***

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms

3. **Further directions**
   1. The role of function approximation
   2. Scalable algorithms for zero-sum games
   3. New equilibrium concepts

Learning in Games

How should agents learn in ***dynamic*** game theoretic settings?

- Convergent

- Independent learning

- Individually Rationalizable

Is this too much to ask for in general-sum multi-agent RL?

Learning in Games

How should agents learn in ***dynamic*** game theoretic settings?

- Convergent
- Independent learning
- Individually Rationalizable

Is this too much to ask for in general-sum multi-agent RL?

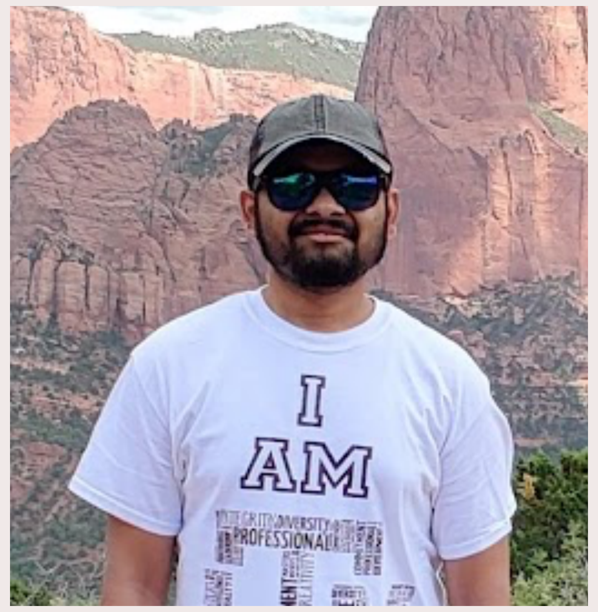
- Computational hardness of Nash eq. in finite games
  [Daskalakis & Papadimitriou '09]
- Computational hardness of stationary CCE
  [Daskalakis et al. 22, Jin et al. '22]
- Non-trivial dynamics from no-regret algorithms
  [Palaiopanos et al 2017]
- Curse of multi-agency
  [Bai et al. '20]
- Strong conditions for dynamic programming to work in infinite-horizon MARL
  [Hu & Wellman '03]

Learning in Games

How should agents learn in ***dynamic*** game theoretic settings?

- Convergent
- Independent learning
- Individually Rationalizable

Today: Tractable MARL through behavioral economics.



Kishan Panaganti
(Caltech)

Laixi Shi
(Caltech)

Today:

We define a computationally tractable* class of equilibria for all normal form (and finite-horizon stochastic games) that is

(1). independent of the underlying game structure
(2). can recreate human-play in experimental data.

*The equilibria can be computed through no-regret learning on a related convex game.

Today:

We define a computationally tractable* class of equilibria for all normal form (and finite-horizon stochastic games) that is

(1). independent of the underlying game structure
(2). can recreate human-play in experimental data.

This arises from assuming:

1. Risk Aversion: Agents are risk-averse to the randomness introduced by their opponents and the environment.

2. Bounded Rationality: Agents have a systematic failure to perfectly optimize.
(i.e., they optimize over quantal responses)

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Warm-Up: n-player finite-action games

- N players
- Finite Action Spaces: $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_N$
- Each player seeks to maximize their expected payoff over mixed strategies $\pi_i \in \Delta(\mathscr{A}_1)$:

$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{a \sim \pi}[R_i(a)]$$

Warm-Up: n-player finite-action games

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Warm-Up: n-player finite-action games

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Intractable to compute outside of highly structured games (e.g., zero-sum).

[Daskalakis & Papadimitriou 2008]

- **Led to focus on:** correlated eq [Moulin & Vial 1978] , coarse correlated eq. [Aumann 1974], …, smoothed Nash [Daskalakis et al. 2023]

Warm-Up: n-player finite-action games

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Intractable to compute outside of highly structured games (e.g., zero-sum).

[Daskalakis & Papadimitriou 2008]

- **Led to focus on:** correlated eq [Moulin & Vial 1978] , coarse correlated eq. [Aumann 1974], …, smoothed Nash [Daskalakis et al. 2023]

Computationally tractable but still have drawbacks
(e.g., eq. selection, support on dominated strategies)

Warm-Up: n-player finite-action games

Nash Eq: Natural solution concept for individually rational agents.

π^* is Nash if for each player i: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Intractable to compute outside of highly structured games (e.g., zero-sum).

[Daskalakis & Papadimitriou 2008]

Not predictive of human play in games

[Selten 1975], [Myerson 1978],[Mckelvey & Palfrey 1995],[Burchardi & Pencyniski 2012], [Wright & Leighton-Brown 2013]....

- More predictive eq. concepts based around ideas that people fail to perfectly optimize (but often not computationally tractable in general-sum games)

Beyond Nash Eq.

A computationally tractable equilibrium concept that arises when agents have natural features of human decision-making:

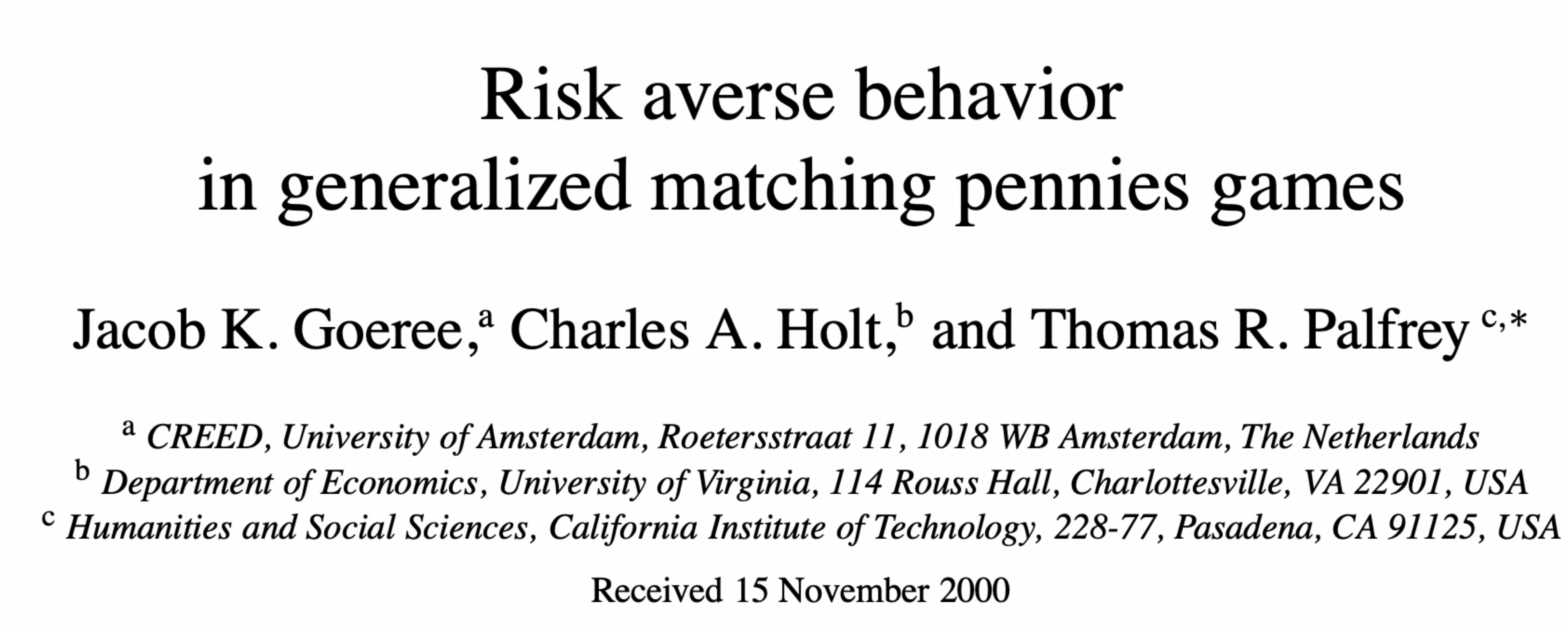
1. Risk Aversion ***2. Bounded Rationality***

Beyond Nash Eq.

A computationally tractable equilibrium concept that arises when agents have natural features of human decision-making:

1. Risk Aversion ***2. Bounded Rationality***

GAMES and Economic Behavior



Risk averse behavior
in generalized matching pennies games

Jacob K. Goeree,[a] Charles A. Holt,[b] and Thomas R. Palfrey[c,*]

[a] CREED, University of Amsterdam, Roetersstraat 11, 1018 WB Amsterdam, The Netherlands
[b] Department of Economics, University of Virginia, 114 Rouss Hall, Charlottesville, VA 22901, USA
[c] Humanities and Social Sciences, California Institute of Technology, 228-77, Pasadena, CA 91125, USA

Received 15 November 2000

Quantal response alone is not enough to recreate human-play in matching pennies.

Risk-aversion is a crucial feature of human decision-making.

Classic finding in behavioral economics
(e.g., [Kahneman & Tversky 1979])

Risk-adjusted Matrix Games

- To introduce risk-aversion into games we make use of a general class of ***convex risk metrics.***

Convex Risk Metric:

Generalization of the expectation ρ such that satisfies:

Sign difference because risk is ***minimized***

1. *Monotonicity:* If $X \leq Y$ almost surely, then $\rho(X) \geq \rho(Y)$.
2. *Translation Invariance:* If $m \in \mathbb{R}$ then $\rho(X+m) = \rho(X) - m$.
3. *Convexity:* For all $\lambda \in (0,1)$, $\rho(\lambda X + (1-\lambda)Y) \leq \lambda\rho(X) + (1-\lambda)\rho(Y)$.

E.g., entropic risk, CVAR, ϕ-divergence based risk metrics, shortfall risks,…

Risk-adjusted Matrix Games

- Assume that players are risk-averse to their opponent's randomness in some *convex risk-metric.*

Maximize payoff $$\max_{\pi_i \in \Delta_i} U_i(\pi_i, \pi_{-i}) = \max_{\pi_i \in \Delta_i} \mathbb{E}_{a \sim \pi}[R_i(a)]$$

Risk-adjusted Matrix Games

- Assume that players are risk-averse to their opponent's randomness in some *convex risk-metric.*

Maximize payoff

$$\max_{\pi_i \in \Delta_i} U_i(\pi_i, \pi_{-i}) = \max_{\pi_i \in \Delta_i} \mathbb{E}_{a \sim \pi}[R_i(a)]$$

1. Evaluate risk associated with each pure strategy

$$\rho_{\pi_{-i}}(R_i(a_i, a_{-i}))$$

Risk level associated with playing pure strategy a_i for agent i

Risk-adjusted Matrix Games

- Assume that players are risk-averse to their opponent’s randomness in some *convex risk-metric.*

Maximize payoff

$$\max_{\pi_i \in \Delta_i} U_i(\pi_i, \pi_{-i}) = \max_{\pi_i \in \Delta_i} \mathbb{E}_{a \sim \pi}[R_i(a)]$$

1. Evaluate risk associated with each pure strategy

$$\rho_{\pi_{-i}}(R_i(a_i, a_{-i}))$$

2. Minimize risk

$$\min_{\pi_i \in \Delta_i} f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \mathbb{E}_{\pi_i}\left[\rho_{\pi_{-i}}(R_i(a))\right]$$

Indifferent amongst pure strategies
that yield the same risk level.

Risk-adjusted Matrix Games

- Assume that players are risk-averse to their opponent’s randomness in some *convex risk-metric.*

Maximize payoff

$$\max_{\pi_i \in \Delta_i} U_i(\pi_i, \pi_{-i}) = \max_{\pi_i \in \Delta_i} \mathbb{E}_{a \sim \pi}[R_i(a)]$$

1. Evaluate risk associated with each pure strategy

$$\rho_{\pi_{-i}}(R_i(a_i, a_{-i}))$$

2. Minimize risk

$$\min_{\pi_i \in \Delta_i} f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \mathbb{E}_{\pi_i}\left[\rho_{\pi_{-i}}(R_i(a))\right]$$

Reduces to well studied risk-averse control and risk averse RL paradigms in single-agent settings

Risk-adjusted Matrix Games

- Assume that players are risk-averse to their opponent's randomness in some *convex risk-metric.*

Maximize payoff

$$\max_{\pi_i \in \Delta_i} U_i(\pi_i, \pi_{-i}) = \max_{\pi_i \in \Delta_i} \mathbb{E}_{a \sim \pi}[R_i(a)]$$

More conservative: Minimize Aggregate Risk

$$\min_{\pi_i \in \Delta_i} f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \rho_{\pi_{-i}}\left(\mathbb{E}_{\pi_i}[R_i(a)]\right)$$

Minimize Action-dependent Risk

$$\min_{\pi_i \in \Delta_i} f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \mathbb{E}_{\pi_i}\left[\rho_{\pi_{-i}}(R_i(a))\right]$$

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Risk-averse Nash equilibria

$$\min_{\pi_i \in \Delta_i} \; f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \; \rho_{\pi_{-i}} \left(\mathbb{E}_{\pi_i}[R_i(a)] \right)$$

Risk-averse Nash Eq: Natural solution concept for individually rational and risk-averse agents.

π^* is RNE if for each player i: $\quad f_i(\pi_i^*, \pi_{-i}^*) \leq f_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Risk-averse Nash equilibria

$$\min_{\pi_i \in \Delta_i} \; f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \; \rho_{\pi_{-i}} \left(\mathbb{E}_{\pi_i}[R_i(a)] \right)$$

Risk-averse Nash Eq: Natural solution concept for individually rational and risk-averse agents.

π^* is RNE if for each player i: $f_i(\pi_i^*, \pi_{-i}^*) \leq f_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Delta_i$

Do these always exist and are they computationally tractable to compute?

Risk-averse Nash equilibria

$$\min_{\pi_i \in \Delta_i} \; f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \; \rho_{\pi_{-i}} \left(\mathbb{E}_{\pi_i}[R_i(a)] \right)$$

Theorem: Existence of risk-averse Nash eq.

If agents' risk aversion can be captured by convex risk metrics then a risk-averse Nash equilibrium exists.

Agents can only be risk averse to their opponents and the environments

(if they are risk averse to their own randomness then this result does not hold [Fiat & Papadimitriou 2010])

Risk-averse Nash equilibria

$$\min_{\pi_i \in \Delta_i} \; f_i(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \; \rho_{\pi_{-i}} \left(\mathbb{E}_{\pi_i}[R_i(a)] \right)$$

Theorem: Existence of risk-averse Nash eq.

If agents' risk aversion can be captured by convex risk metrics then a risk-averse Nash equilibrium exists.

Risk aversion isn't enough to ensure

1. Computational tractability.

2. The equilibrium is predictive of human-play [Goeree, Holt, Palfrey. 2002], [Goeree, Holt, Palfrey 2003].

Bounded Rationality in Risk-adjusted Matrix Games

- To have computational tractability we optimize over quantal responses.

Quantal Response Function: a quantal response function is a continuous function $\sigma : \mathbb{R}^n \to \Delta_n$ such that if x>y, $\boldsymbol{\sigma}(\boldsymbol{y}) > \boldsymbol{\sigma}(\boldsymbol{x})$

Canonical example is space of softmax policies: $\sigma_\epsilon(x) = \frac{1}{\sum_{i=1}^n e^{-\epsilon x_i}} \begin{bmatrix} e^{-\epsilon x_1} \\ \vdots \\ e^{-\epsilon x_n} \end{bmatrix}$

Bounded Rationality in Risk-adjusted Matrix Games

- To have computational tractability we optimize over quantal responses.

Quantal Response Function: a quantal response function is a continuous function $\sigma : \mathbb{R}^n \to \Delta_n$ such that if x>y, $\sigma(y) > \sigma(x)$

$$\min_{\pi_i \in \Delta_i} f_i(\pi_i, \pi_{-i}) \longrightarrow \min_{\pi_i \in \sigma_i(\Delta_i)} f(\pi_i, \pi_{-i})$$

Optimize over space of quantal best responses.

Bounded Rationality in Risk-adjusted Matrix Games

- To have computational tractability we optimize over quantal responses.

Quantal Response Function: a quantal response function is a continuous function $\sigma : \mathbb{R}^n \to \Delta_n$ such that if x>y, $\sigma(y) > \sigma(x)$

$$\min_{\pi_i \in \Delta_i} \; f_i(\pi_i, \pi_{-i}) \longrightarrow \min_{\pi_i \in \Delta_i} f_i^\epsilon(\pi_i, \pi_{-i}) = \rho_{\pi_{-i}}\left(\mathbb{E}_{\pi_i}[R_i(a)]\right) + \epsilon_i \nu_i(\pi_i)$$

Can be captured with a strictly convex regularizer

ϵ_i captures an agents’ degree of bounded rationality.

Bounded Rationality in Risk-adjusted Matrix Games

$$\min_{\pi_i \in \sigma_i(\Delta_i)} \quad f_i^{\epsilon}(\pi_i, \pi_{-i}) = \min_{\pi_i \in \Delta_i} \quad \rho_{\pi_{-i}}\left(\mathbb{E}_{\pi_i}[R_i(a)]\right) + \epsilon_i \nu_i(\pi_i)$$

Risk-averse Quantal Response Eq (RQE): Natural solution concept for *risk averse* and individually *boundedly rational* agents.

π^* is RQE if for each player i: $\quad f(\pi_i^*, \pi_{-i}^*) \leq f(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \sigma_i(\Delta_i)$

Given these definitions, we can show that a class of RQE is computationally tractable in all games.

Will focus on 2-player for today
paper has n-player results

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

- Using a dual representation theorem, we can write any convex risk metric in a variational form [Follmer & Shied 2002]

$$\min_{\pi_i \in \Delta_i} \rho_{\pi_{-i}}\left(\mathbb{E}_{\pi_i}[R_i(a)]\right) + \epsilon_i \nu_i(\pi_i) = \min_{\pi_i \in \Delta_i} \max_{p_i \in \Delta_{-i}} \mathbb{E}_{\pi_i, p_i}[R_i(a)] - \frac{1}{\tau_i} D(p_i, \pi_{-i}) + \epsilon_i \nu_i(\pi_i)$$

Risk metric can be fully defined by a penalty function D that is convex in its first argument.
e.g., when D is the KL divergence we recover the entropic risk

τ_i captures agent i's degree of risk aversion.

$\tau_i \to 0$ yields risk-neural game

$\tau_i \to \infty$ yields security strategy

Computationally Tractability of RQE in Matrix Games

- Using a dual representation theorem, we can write any convex risk metric in a variational form [Follmer & Shied 2002]

$$\min_{\pi_i \in \Delta_i} \; \rho_{\pi_{-i}}\left(\mathbb{E}_{\pi_i}[R_i(a)]\right) + \epsilon_i \nu_i(\pi_i) = \min_{\pi_i \in \Delta_i} \max_{p_i \in \Delta_{-i}} \mathbb{E}_{\pi_i, p_i}\left[R_i(a)\right] - \frac{1}{\tau_i} D(p_i, \pi_{-i}) + \epsilon_i \nu_i(\pi_i)$$

Risk metric can be fully defined by a penalty function D that is convex in its first argument.
e.g., when D is the KL divergence we recover the entropic risk

τ_i captures agent i's degree of risk aversion.

$\tau_i \to 0$ yields risk-neural game

$\tau_i \to \infty$ yields security strategy

***Interpretation:** Introduce an adversary for each player that is fully adversarial but is penalized from deviations from the opponents strategy*

Computationally Tractability of RQE in Matrix Games

Theorem: Computational Tractability of RQE (2-player)

Suppose. $\epsilon_1\tau_1 \geq 1/\epsilon_2\tau_2$ then resulting RQE can be computed using no-regret learning on a related 4-player convex game.

This result doesn't depend on the structure of the underlying game!
Only on players' relative degrees of risk-aversion/bounded rationality

e.g., policy gradients, multiplicative weights

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

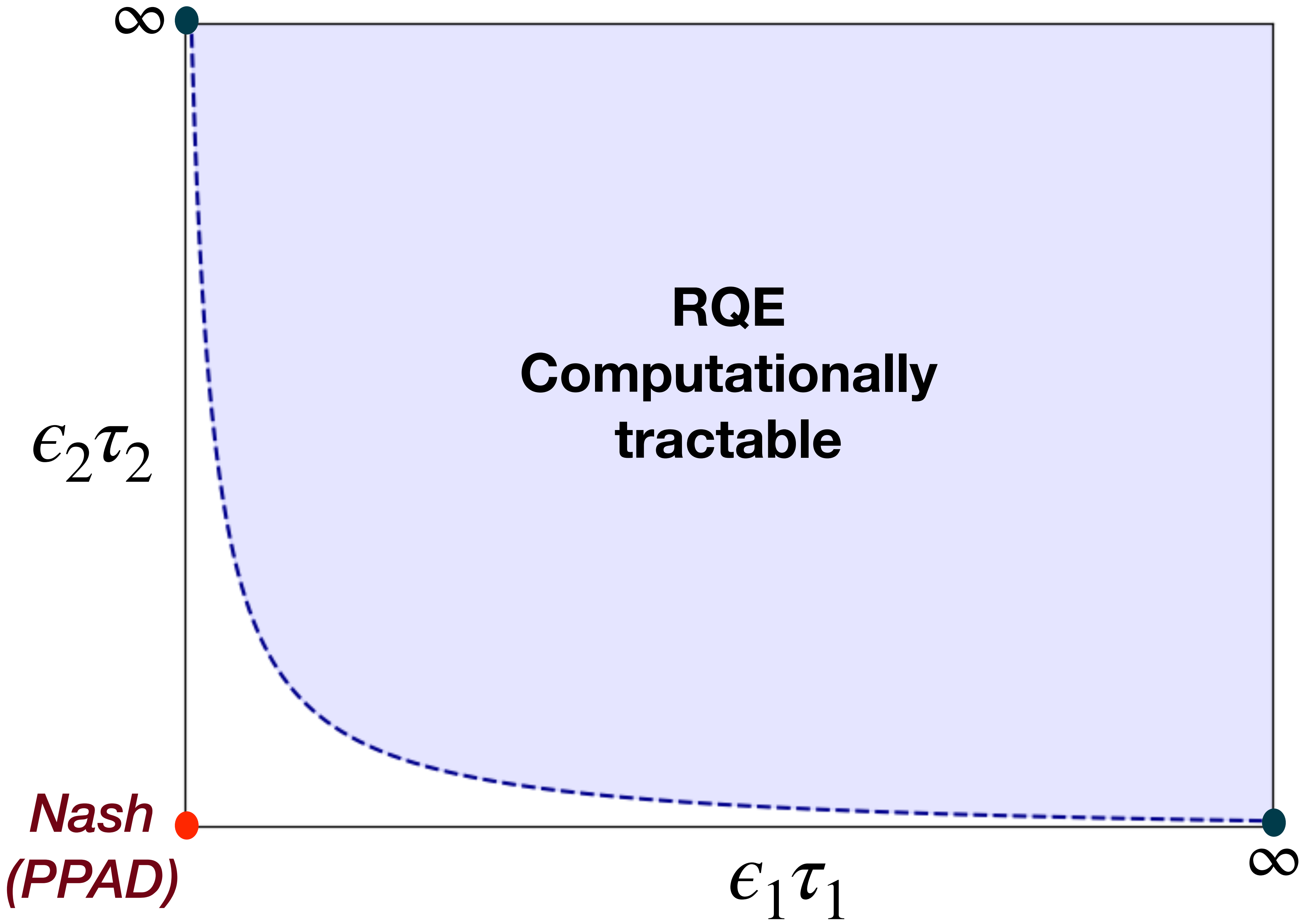
Theorem: Computational Tractability of RQE (2-player)

Suppose. $\epsilon_1 \tau_1 \geq 1/\epsilon_2 \tau_2$ then a resulting RQE can be computed using no-regret learning on a related 4-player convex game.

$\tau, \epsilon = 0$
Agents are risk-neutral.

↓

general-sum game where approximating Nash Eq is known to be hard in general



Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

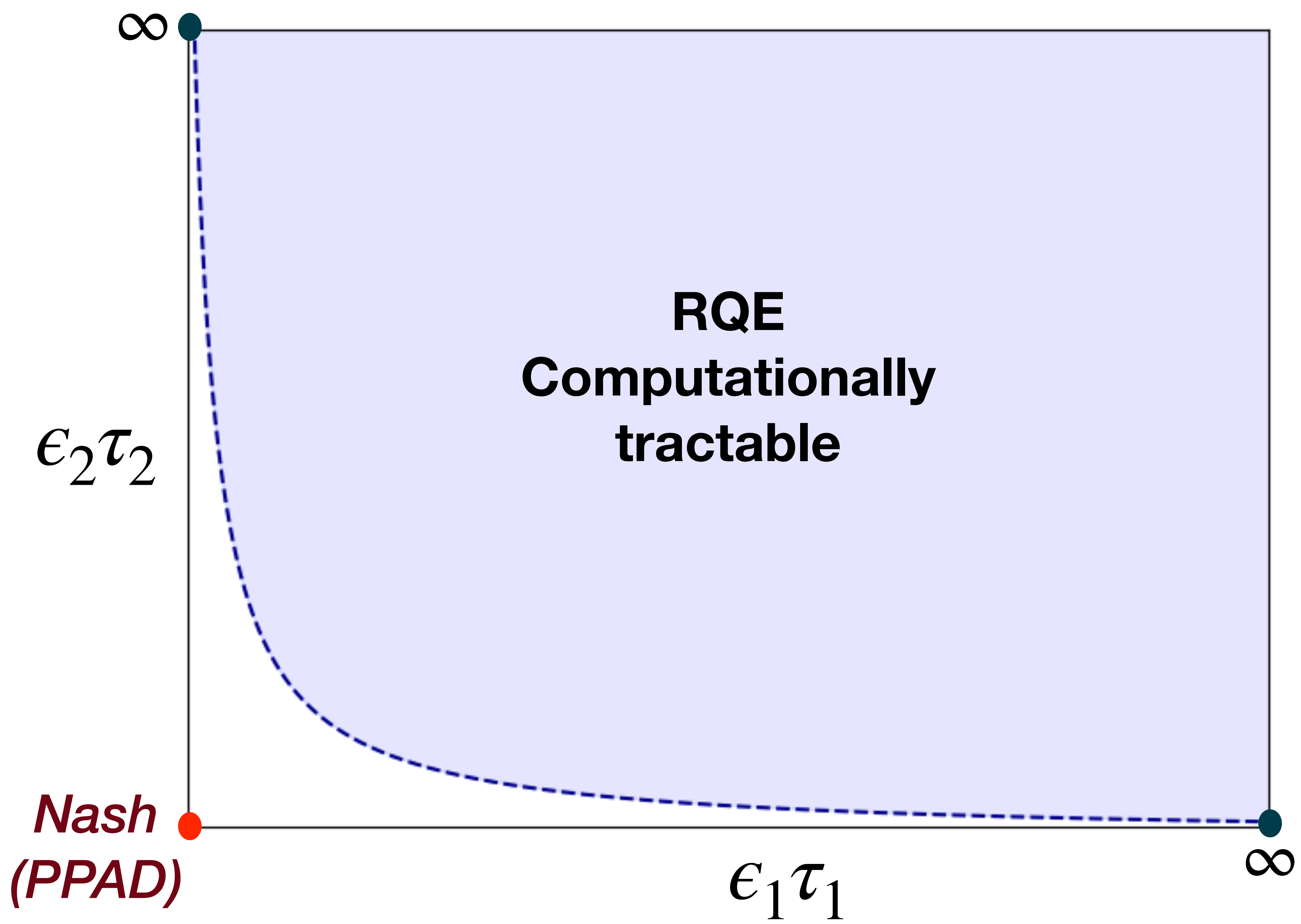
Theorem: Computational Tractability of RQE (2-player)

Suppose. $\epsilon_1 \tau_1 \geq 1/\epsilon_2 \tau_2$ then a resulting RQE can be computed using no-regret learning on a related 4-player convex game.

$\tau, \epsilon = 0$
Agents are risk-neutral.

↓

general-sum game where approximating Nash Eq is known to be hard in general



$\tau = \infty, \epsilon \geq 0$
Player is so risk averse they treat their opponent as adversarial

↓

zero-sum game for agent computationally tractable for any $\epsilon \geq 0$

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

Proof sketch:

- *Start with 2 player game*

$$\min_{\pi_1 \in \sigma_1(\Delta_1)} \mathbb{E}_{\pi_1}[\rho_{\pi_2}(R_1(a))]$$

$$\min_{\pi_2 \in \sigma_2(\Delta_2)} \mathbb{E}_{\pi_2}[\rho_{\pi_1}(R_2(a))]$$

Computationally Tractability of RQE in Matrix Games

Proof sketch:

- *Start with 2 player game*

$$\min_{\pi_1 \in \Delta_1} \max_{p_1 \in \Delta_2} \ \mathbb{E}_{\pi_1, p_1}[R_1(a)] - \frac{1}{\tau_1} D(p_1, \pi_2) + \epsilon_1 \nu(\pi_1)$$

$$\min_{\pi_2 \in \Delta_2} \max_{p_2 \in \Delta_1} \ \mathbb{E}_{\pi_2, p_2}[R_2(a)] - \frac{1}{\tau_2} D(p_2, \pi_1) + \epsilon_2 \nu(\pi_2)$$

Computationally Tractability of RQE in Matrix Games

Proof sketch:

- *Lift the game to a 4-player game by introducing adversaries for each player (who are penalized from deviations from opponents).*

$$\min_{\pi_1 \in \Delta_1} \max_{p_1 \in \Delta_2} \; \mathbb{E}_{\pi_1, p_1}[R_1(a)] - \frac{1}{\tau_1} D(p_1, \pi_2) + \epsilon_1 \nu(\pi_1)$$

$$\min_{\pi_2 \in \Delta_2} \max_{p_2 \in \Delta_1} \; \mathbb{E}_{\pi_2, p_2}[R_2(a)] - \frac{1}{\tau_2} D(p_2, \pi_1) + \epsilon_2 \nu(\pi_2)$$

$$\min_{\pi_1 \in \Delta_1} \mathbb{E}_{\pi_1, p_1}[R_1(a)] - \frac{1}{\tau_1} D(p_1, \pi_2) + \epsilon_1 \nu(\pi_1)$$

$$\min_{p_1 \in \Delta_2} -\mathbb{E}_{\pi_1, p_1}[R_1(a)] + \frac{1}{\tau_1} D(p_1, \pi_2) - \epsilon_2 \nu(\pi_2)$$

$$\min_{p_2 \in \Delta_1} -\mathbb{E}_{\pi_2, p_2}[R_2(a)] + \frac{1}{\tau_2} D(p_2, \pi_1) - \epsilon_1 \nu(\pi_1)$$

$$\min_{\pi_2 \in \Delta_2} \mathbb{E}_{\pi_2, p_2}[R_2(a)] - \frac{1}{\tau_2} D(p_2, \pi_1) + \epsilon_2 \nu(\pi_2)$$

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

Proof sketch:

- ***This four player game is convex and nonlinear but zero-sum.***

$$\min_{\pi_1 \in \Delta_1} \mathbb{E}_{\pi_1, p_1}[R_1(a)] - \frac{1}{\tau_1} D(p_1, \pi_2) + \epsilon_1 \nu(\pi_1)$$

$$\min_{p_1 \in \Delta_2} -\mathbb{E}_{\pi_1, p_1}[R_1(a)] + \frac{1}{\tau_1} D(p_1, \pi_2) - \epsilon_2 \nu(\pi_2)$$

$$\min_{p_2 \in \Delta_1} -\mathbb{E}_{\pi_2, p_2}[R_2(a)] + \frac{1}{\tau_2} D(p_2, \pi_1) - \epsilon_1 \nu(\pi_1)$$

$$\min_{\pi_2 \in \Delta_2} \mathbb{E}_{\pi_2, p_2}[R_2(a)] - \frac{1}{\tau_2} D(p_2, \pi_1) + \epsilon_2 \nu(\pi_2)$$

Using convexity and concavity in opponents' strategies one can show that CCE coincide with Nash for all parameters in the range

$$\epsilon_1 \tau_1 \geq 1/\epsilon_2 \tau_2$$

independent of R_1 and R_2!

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

Theorem: Computational Tractability of RQE (2-player)

Suppose. $\epsilon_1 \tau_1 \geq 1/\epsilon_2 \tau_2$ then a resulting RQE can be computed using no-regret learning on a related 4-player convex game.

- Class is independent of reward structure (i.e., applies to all games)
- Arises due to the *combination* of risk aversion and bounded rationality.

Similar results hold for n-player games and via dynamic programming in finite-horizon Markov games.

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Computationally Tractability of RQE in Matrix Games

Theorem: Computational Tractability of RQE (2-player)

Suppose. $\epsilon_1 \tau_1 \geq 1/\epsilon_2 \tau_2$ then a resulting RQE can be computed using no-regret learning on a related 4-player convex game.

Does this class of RQE capture meaningful solutions?
Or is this too restrictive an assumption?

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Expressivity of Computationally Tractable RQE

- We look at experimental data on human-play in 13 games from [1] and [2].

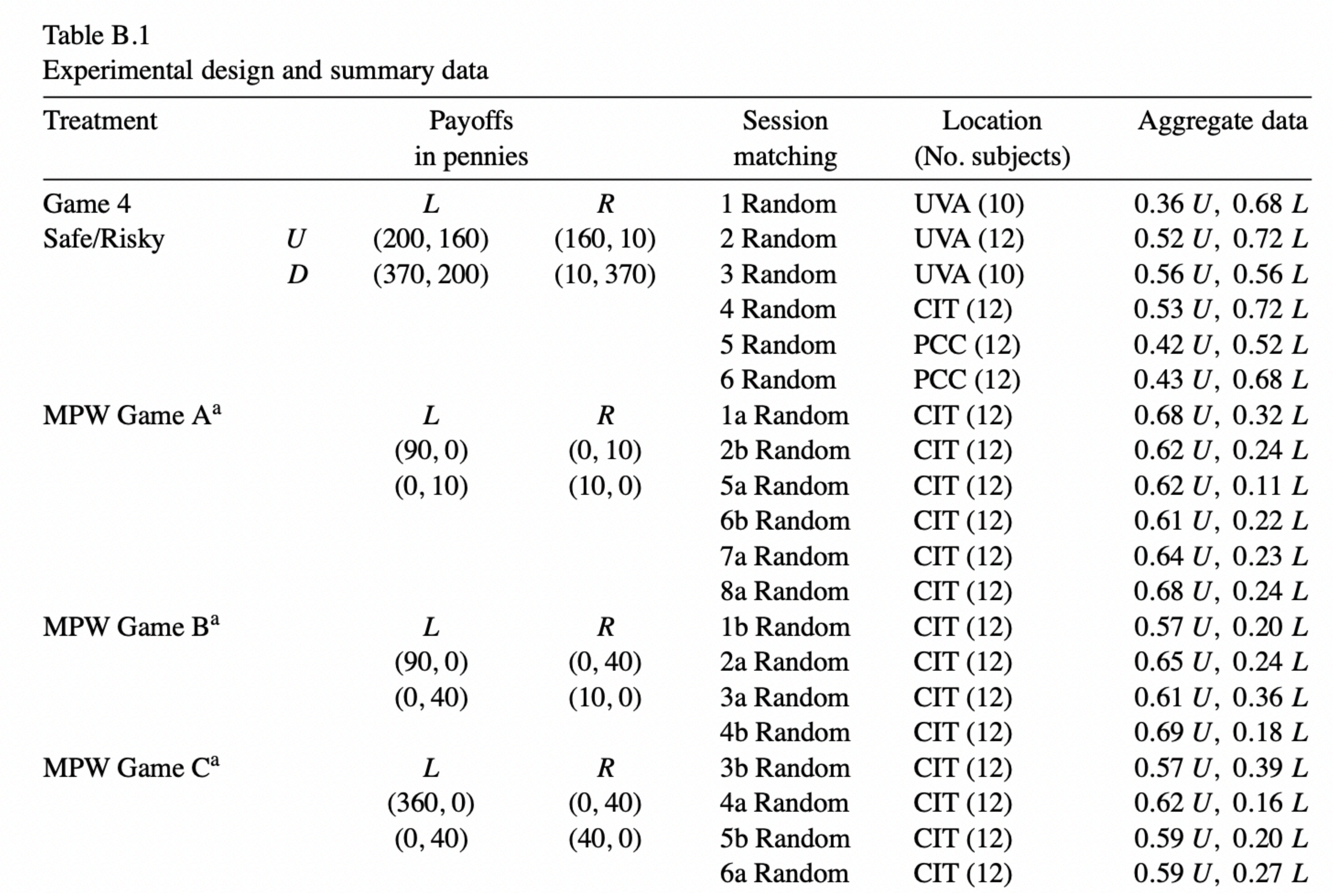
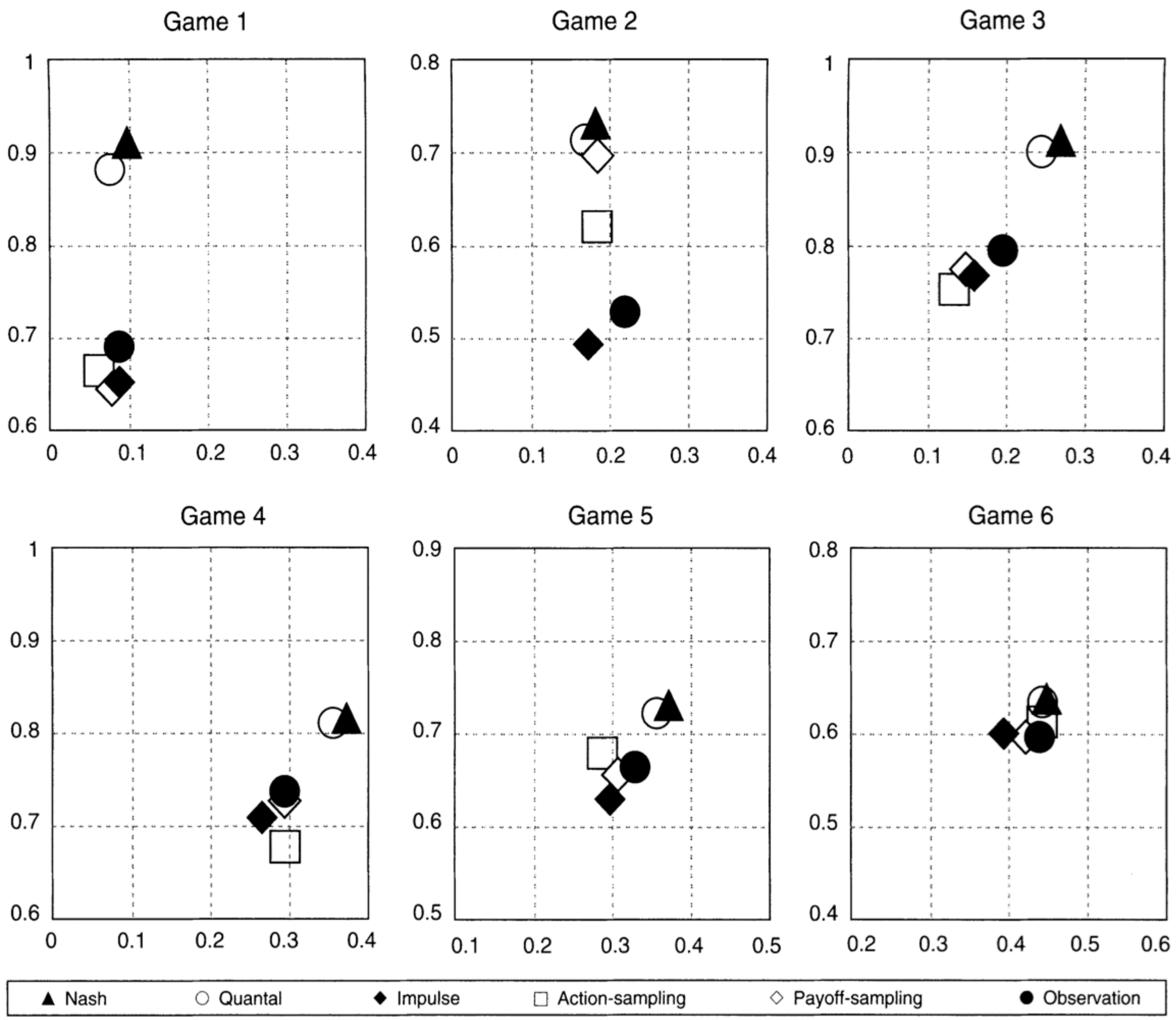


Table B.1
Experimental design and summary data

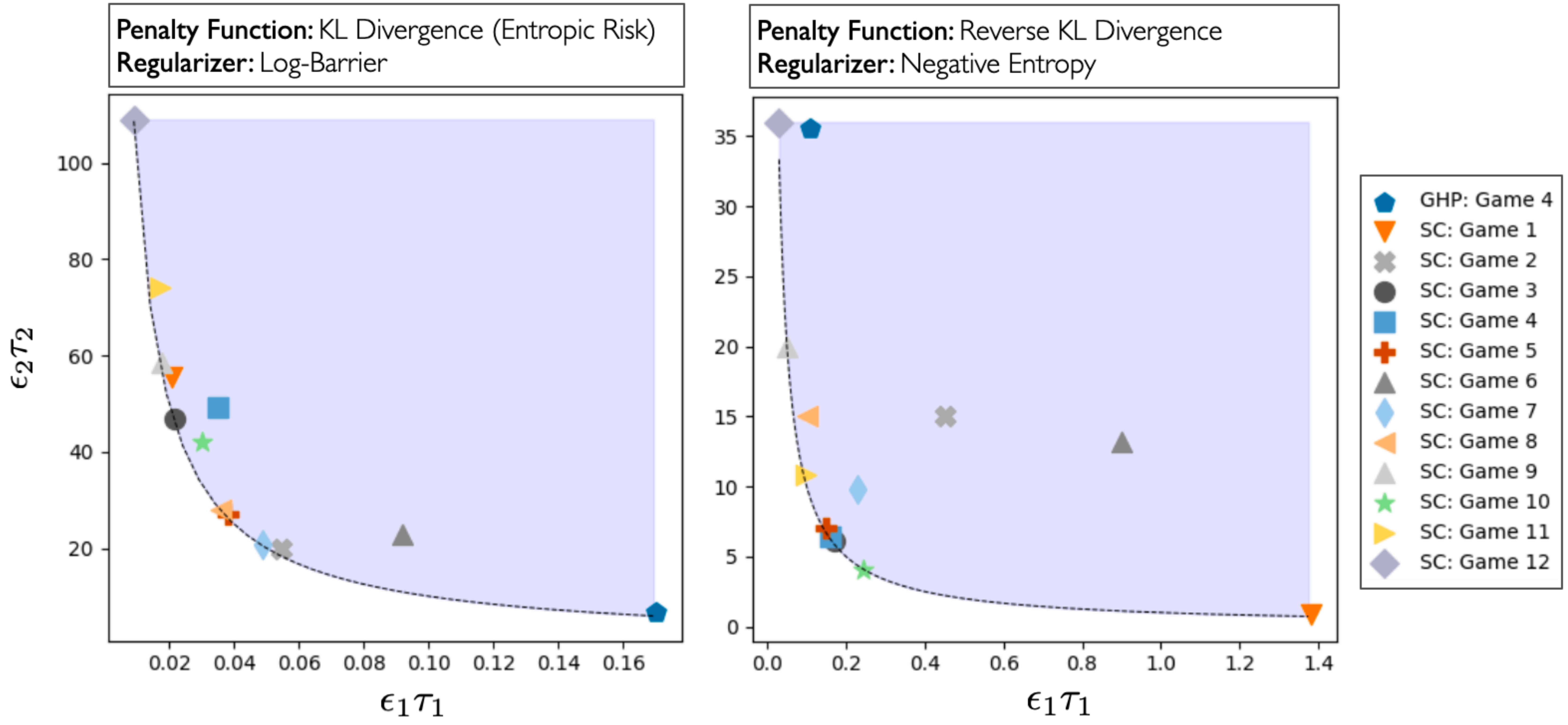
Treatment		Payoffs in pennies		Session matching	Location (No. subjects)	Aggregate data
Game 4		*L*	*R*	1 Random	UVA (10)	0.36 *U*, 0.68 *L*
Safe/Risky	*U*	(200, 160)	(160, 10)	2 Random	UVA (12)	0.52 *U*, 0.72 *L*
	D	(370, 200)	(10, 370)	3 Random	UVA (10)	0.56 *U*, 0.56 *L*
				4 Random	CIT (12)	0.53 *U*, 0.72 *L*
				5 Random	PCC (12)	0.42 *U*, 0.52 *L*
				6 Random	PCC (12)	0.43 *U*, 0.68 *L*
MPW Game A[a]		*L*	*R*	1a Random	CIT (12)	0.68 *U*, 0.32 *L*
		(90, 0)	(0, 10)	2b Random	CIT (12)	0.62 *U*, 0.24 *L*
		(0, 10)	(10, 0)	5a Random	CIT (12)	0.62 *U*, 0.11 *L*
				6b Random	CIT (12)	0.61 *U*, 0.22 *L*
				7a Random	CIT (12)	0.64 *U*, 0.23 *L*
				8a Random	CIT (12)	0.68 *U*, 0.24 *L*
MPW Game B[a]		*L*	*R*	1b Random	CIT (12)	0.57 *U*, 0.20 *L*
		(90, 0)	(0, 40)	2a Random	CIT (12)	0.65 *U*, 0.24 *L*
		(0, 40)	(10, 0)	3a Random	CIT (12)	0.61 *U*, 0.36 *L*
				4b Random	CIT (12)	0.69 *U*, 0.18 *L*
MPW Game C[a]		*L*	*R*	3b Random	CIT (12)	0.57 *U*, 0.39 *L*
		(360, 0)	(0, 40)	4a Random	CIT (12)	0.62 *U*, 0.16 *L*
		(0, 40)	(40, 0)	5b Random	CIT (12)	0.59 *U*, 0.20 *L*
				6a Random	CIT (12)	0.59 *U*, 0.27 *L*



[1] Risk averse behavior in generalized matching pennies games Goeree, Holt, Palfrey, Games and Economic Behavior, 2003

[2] Stationary concepts for experimental 2x2-games, Selten & Chmura, American Economic Review, 2008

Expressivity of Computationally Tractable RQE



- For each game, we show that there exists an RQE can recreate the peoples' aggregate play (to within 3 decimal places).

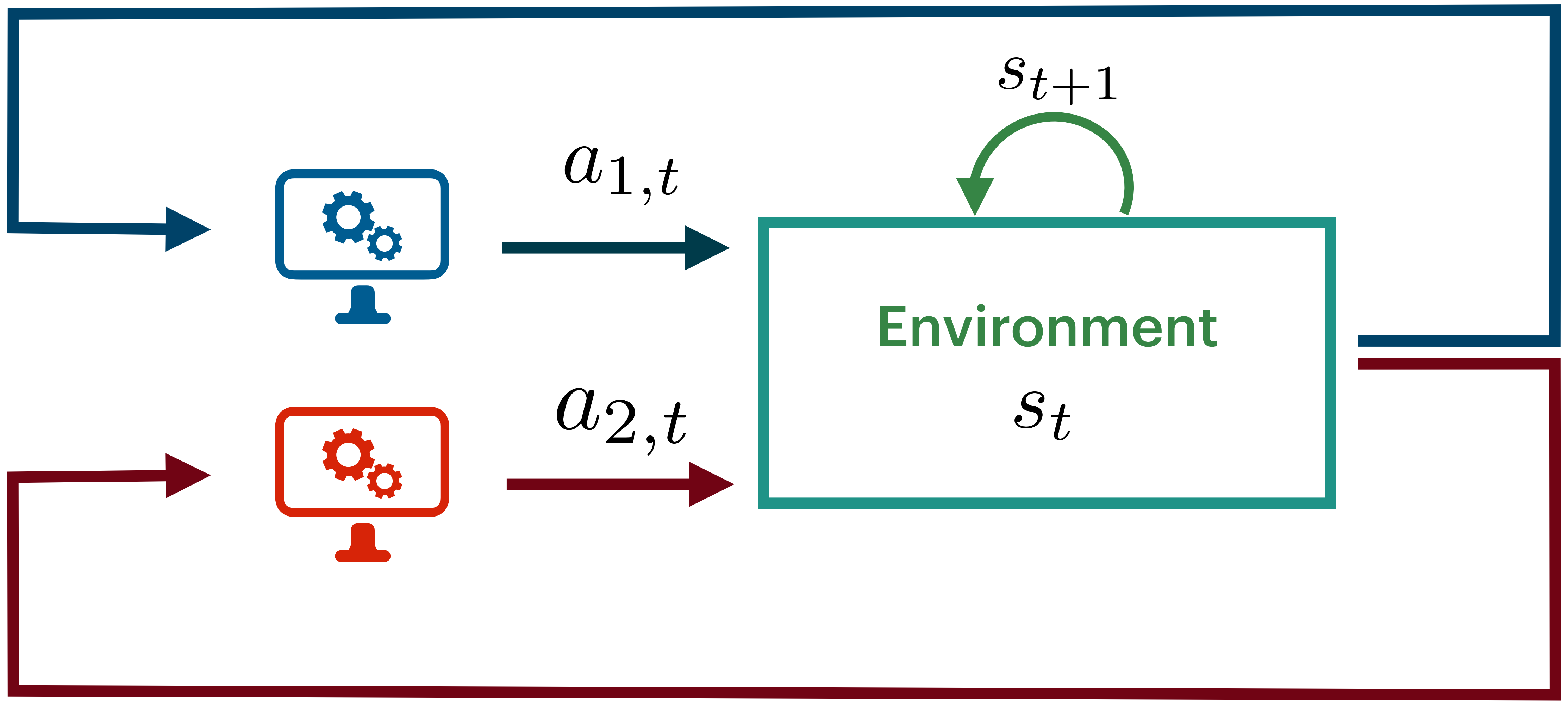
Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

Finite-Horizon Risk-Averse Markov Games

- Finite Action Spaces: $\mathcal{A}_1, \mathcal{A}_2$
- Finite State Spaces: $\mathcal{S}$
- Finite Horizon: $\boldsymbol{H}$
- Dynamics: $P\left(s \,|\, s, a_1, a_2\right)$
- Players are also risk averse to the stochastic transitions (i.e., environmental uncertainties)

$$U_1(\pi_1, \pi_2) = \mathbb{E}_{\pi_1, \pi_2, P}\left[\sum_{t=0}^{H} R_{1,t}\right]$$

$$U_2(\pi_1, \pi_2) = \mathbb{E}_{\pi_1, \pi_2, P}\left[\sum_{t=0}^{H} R_{2,t}\right]$$

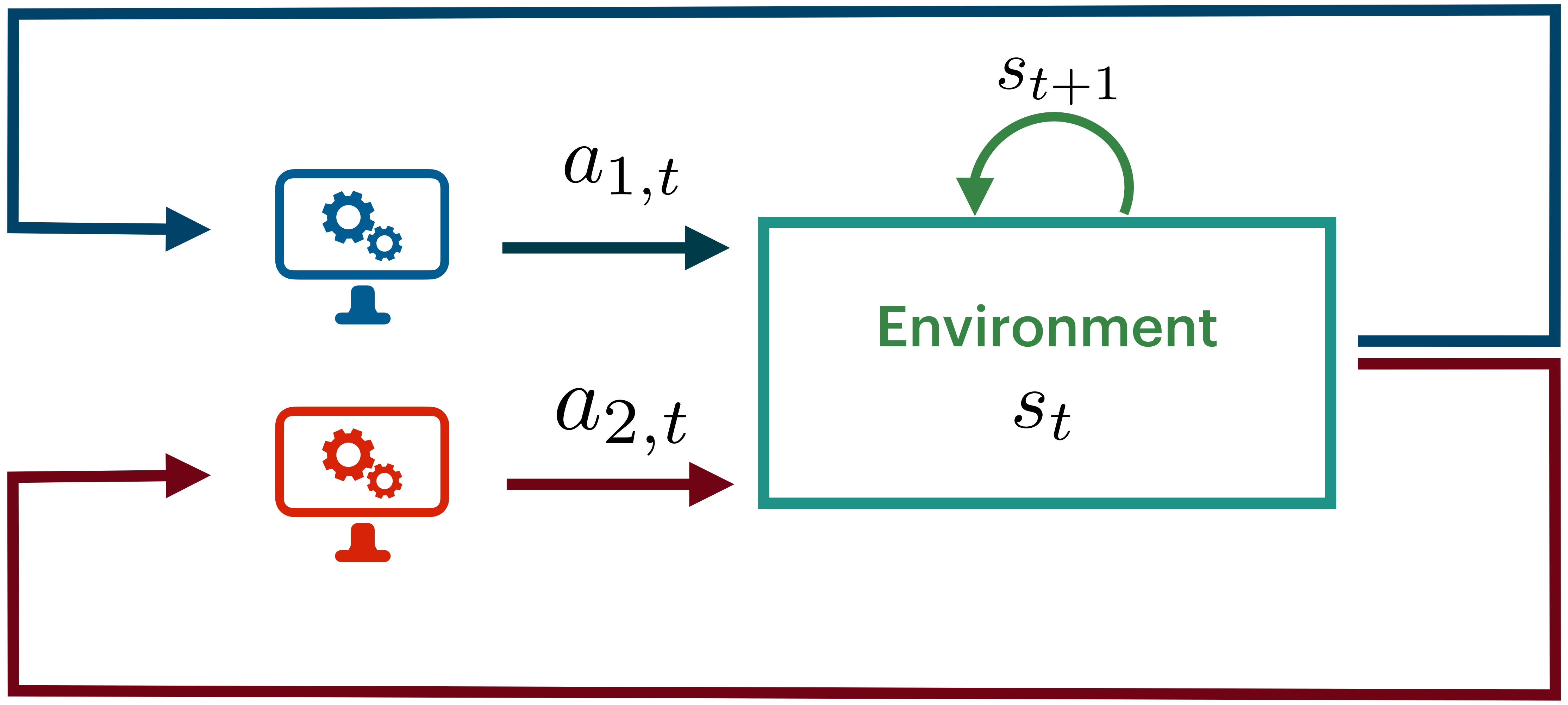


Finite-Horizon Risk-Averse Markov Games

- Finite Action Spaces: $\mathcal{A}_1, \mathcal{A}_2$
- Finite State Spaces: $\mathcal{S}$
- Finite Horizon: ***H***
- Dynamics: $P\left(s \,|\, s, a_1, a_2\right)$
- Players are also risk averse to the stochastic transitions (i.e., environmental uncertainties)

$$U_1(\pi_1, \pi_2) = \mathbb{E}_{\pi_1,\pi_2,P}\left[\sum_{t=0}^{H} R_{1,t}\right]$$

$$U_2(\pi_1, \pi_2) = \mathbb{E}_{\pi_1,\pi_2,P}\left[\sum_{t=0}^{H} R_{2,t}\right]$$



Assume access to a ***generative model***

(i.e., can collect collect i.i.d samples of transitions and rewards to estimate R_i and P)

Approximating RQE in Finite-Horizon Markov Games

Theorem: Approximating (non-stationary) Markov RQE in Markov Games

Suppose $\epsilon_i \geq \sum_{j \neq i} 1/\tau^*_{j,i}$ then a $\delta-$RQE can be computed in $poly(S, H, \delta^{-1}) \prod A_i$

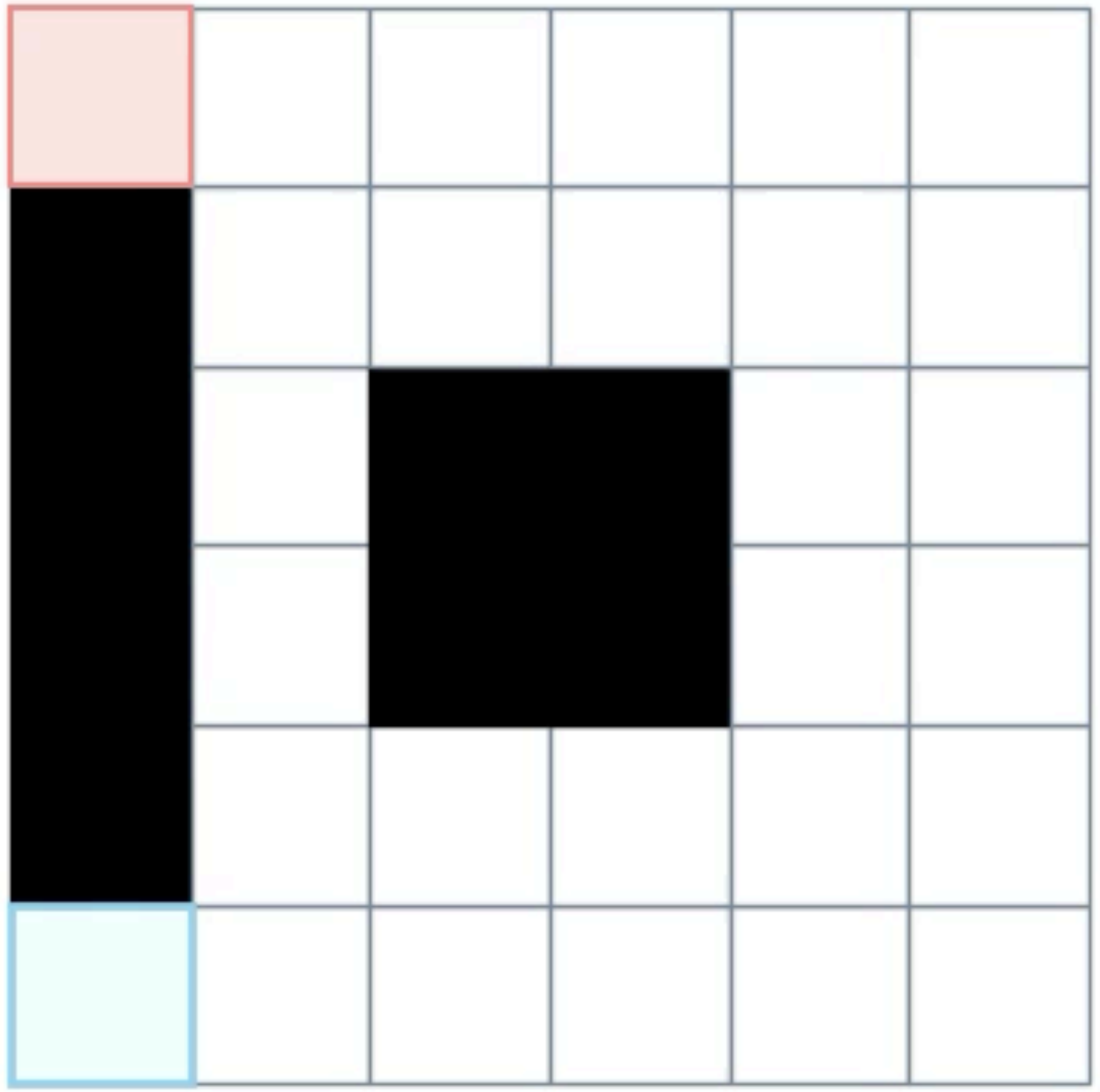
Our algorithm suffers from the curse of multi-agency!

Open Question: Can we overcome this?

May be hard since the intermediate adversaries action spaces are essentially the size of the joint action space, which is $\prod A_i$.

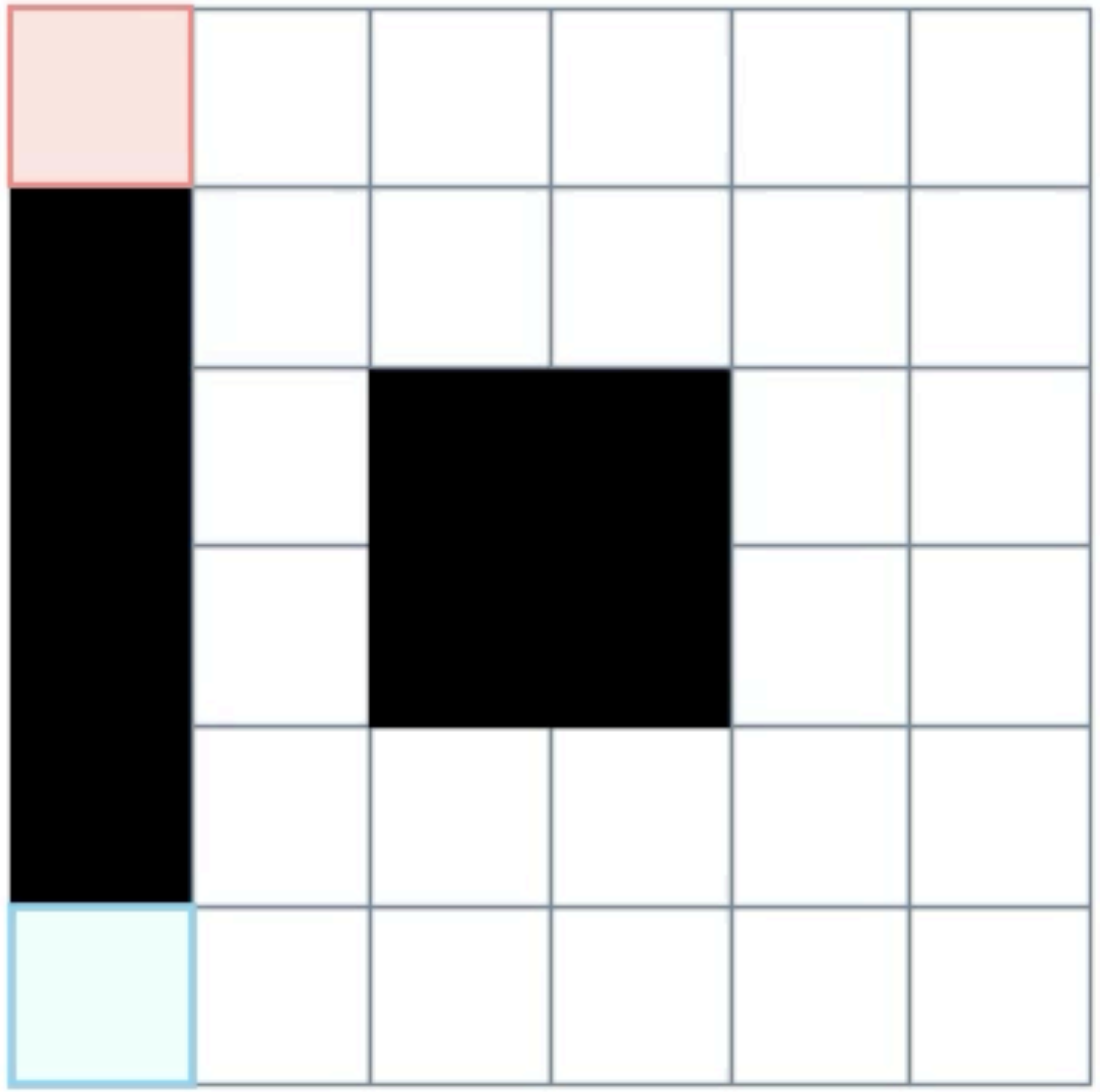
Tractable Equilibrium Computation in Markov Games through Risk Aversion, Eric Mazumdar, Kishan Panaganti, Laixi Shi, (under submission)

Extensions to Finite-Horizon MARL



Red is risk-averse: $\tau_1 = 0.01$

Blue is more risk-averse: $\tau_2 = 0.02$



Red is risk-averse: $\tau_1 = 0.01$

Blue is less risk-averse: $\tau_2 = 0.005$

Tractable Multi-Agent Reinforcement Learning through Behavioral Economics, Eric Mazumdar, Kishan Panaganti, Laixi Shi, ICLR 2025 (Oral)

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**

   i. Policy-gradient algorithms in games

   ii. Value-based algorithms

3. **Further directions**

   i. The role of function approximation

   ii. Scalable algorithms for zero-sum games

   iii. New equilibrium concepts

A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   i. Policy-gradient algorithms in games
   ii. Value-based algorithms

3. **Further directions**
   i. The role of function approximation
   ii. Scalable algorithms for zero-sum games
   iii. New equilibrium concepts
      - Takeaway:
         - Equilibria rooted in behavioral economics may be more amenable to learning than Nash and have more desirable properties than CCE.

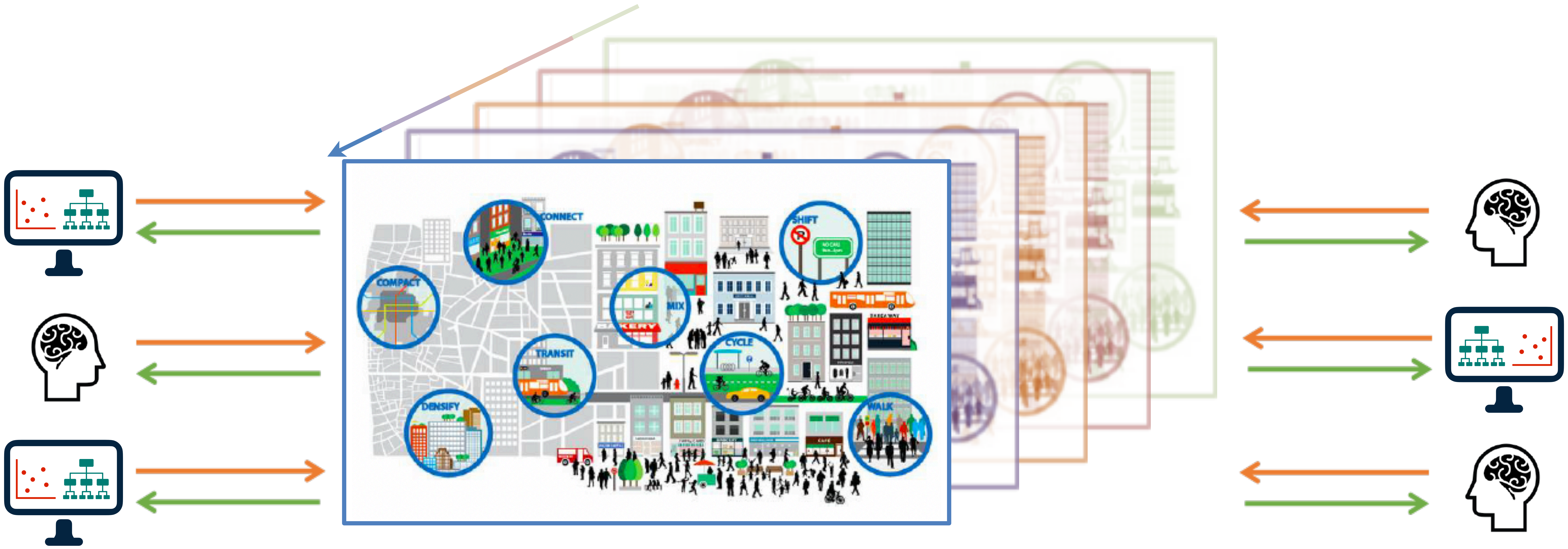
A Road Map

1. **Normal-form & concave games: equilibrium computation and learning in games**

2. **Algorithmic structures in Multi-Agent Reinforcement Learning**
   1. Policy-gradient algorithms in games
   2. Value-based algorithms

3. **Further directions**
   1. The role of function approximation
   2. Scalable algorithms for zero-sum games
   3. New equilibrium concepts
      - Takeaway:
        - ***Current Work:*** Showing it gives rise to a computationally feasible eq. In infinite-horizon Markov Games (CDC 2025), and a set of MARL learning algorithms.

Challenges: Strategic interactions vastly complicate the task of learning



Opportunities: Require a careful rethinking of algorithm design.

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Reinforcement Learning

Multi-Agent Reinforcement Learning

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Reinforcement Learning

Structured non-convex optimization

Multi-Agent Reinforcement Learning

Structured (?) *equilibrium computation*

This is fundamentally hard in general, but notions like non stationary CCE are feasible to learn

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Reinforcement Learning

Structured non-convex optimization

Stationary environment

Multi-Agent Reinforcement Learning

Structured (?) *equilibrium computation*

Coupling between agents introduce *non-stationarities* in learning

Makes proving convergence of algorithms particularly difficult, though ***timescale separation*** is a useful principle in MARL.

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Reinforcement Learning

Structured non-convex optimization

Stationary environment

Role of function approximation is clear

Larger, more expressive function classes have the potential to yield better performance (Modulo optimization/data)

Multi-Agent Reinforcement Learning

Structured (?) *equilibrium computation*

Coupling between agents introduce *non-stationarities* in learning

Choosing a function class is *non-trivial*

Larger, more expressive function classes can yield worse solutions!

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Opportunities: Require a careful rethinking of algorithm design.

Though it is less well understood, we can build on foundations from game theory and reinforcement learning to explore and design new algorithmic principles.

Challenges: Strategic interactions vastly complicate the task of learning

As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Opportunities: Require a careful rethinking of algorithm design.

Though it is less well understood, we can build on foundations from game theory and reinforcement learning to explore and design new algorithmic principles.

- Convergence of algorithms
  - e.g., stochastic approximation ideas [Sayin et al. 2023], [Chen et al 2024], no -regret learning [Farina et al. 2024], [Cai et al. 2024],...
- New equilibrium concepts
  - e.g., behavioral Econ ideas [Mazumdar et al 2024], robust Eq. [Lanzetti et al. 2025], smoothed analysis [Daskalakis, et al 2023],...
- New formulations of Markov games
  - e.g., Convex Markov games [Gemp et al. 2024, 2025], Stackelberg Markov Games [Gerstgrasser et al. 2022],...
- Better algorithms with function approximation for zero-sum games.
  - e.g., surveys [Wong el at. 2022], [Gemp et al. 2022], stabilizing actor critic algorithms [Foerester et al. 2017, 2018], PPO in games [Yu et al. 2022],...

Challenges: Strategic interactions vastly complicate the task of learning

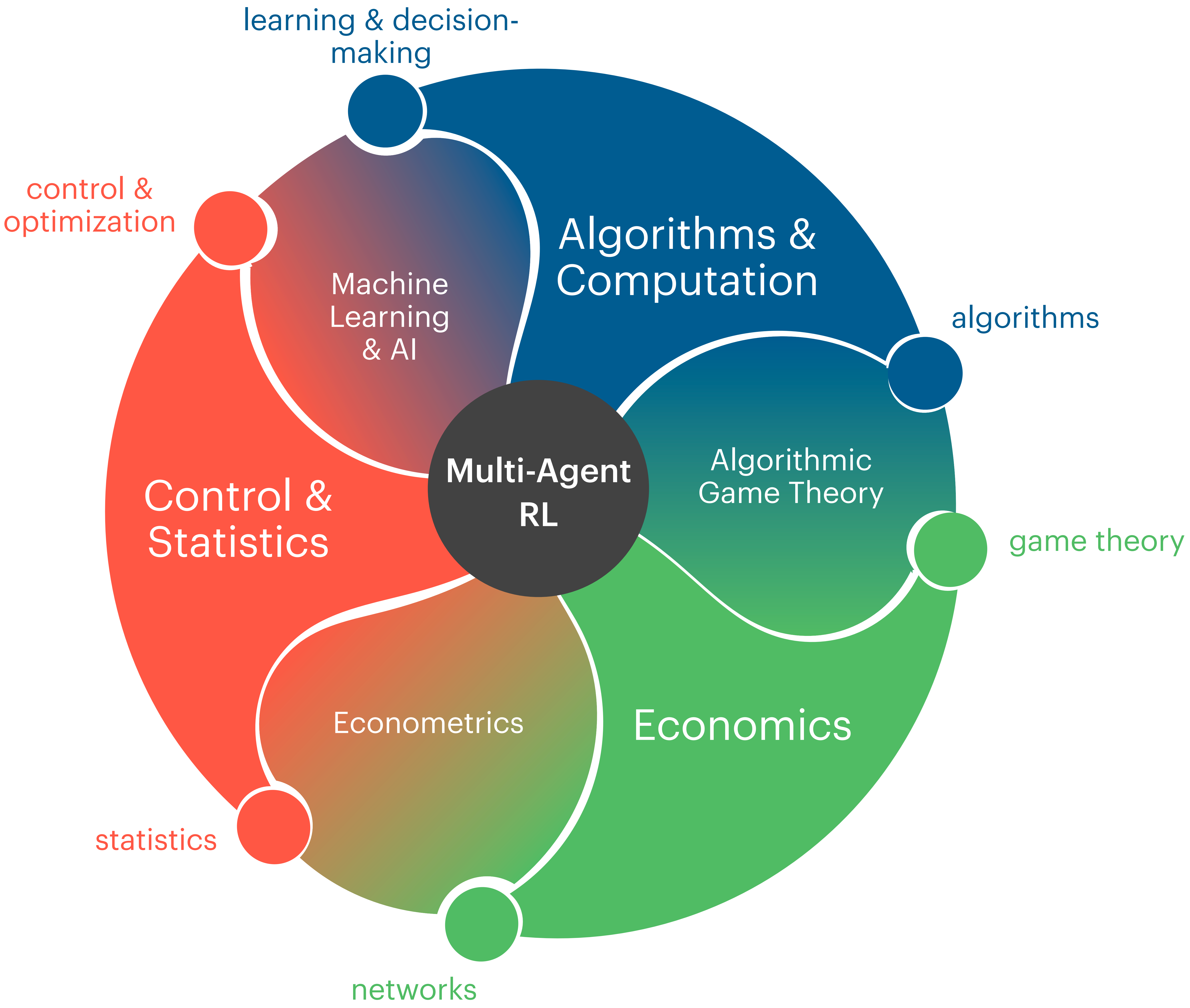
As we will see, strategic interactions can break our intuition on the behavior of learning algorithms and give rise to new challenges for algorithm design.

Opportunities: Require a careful rethinking of algorithm design.

Though it is less well understood, we can build on foundations from game theory and reinforcement learning to explore and design new algorithmic principles.

Many other directions:

- robustness in Markov games
- sim-to-real gaps
- incomplete information games (e.g., partially observed Markov games)
- continuous action/state spaces
- empirical evaluation of algorithms
- training LLM agents for multi-agent problems
- exploring other game theoretic strategies
- ....


learning & decision-making
control & optimization
Machine Learning & AI
Algorithms & Computation
algorithms
Multi-Agent RL
Algorithmic Game Theory
game theory
Control & Statistics
Econometrics
Economics
statistics
networks

Other Useful Resources:

Simons Institute Bootcamps on Learning in games

Great talks like Chi Jin (Princeton) overview of MARL, Costis Daskalakis (MIT) overview of Eq. Computation,…
https://simons.berkeley.edu/workshops/learning-games-boot-camp

Multi-agent reinforcement learning: A selective overview of theories and algorithms

Kaiqing Zhang, Zhuoran Yang, Tamer Başar
https://arxiv.org/pdf/1911.10635

Predictions, Learning, & Games

Nicolò Cesa-Bianchi & Gabor Lugosi
Book on learning in games