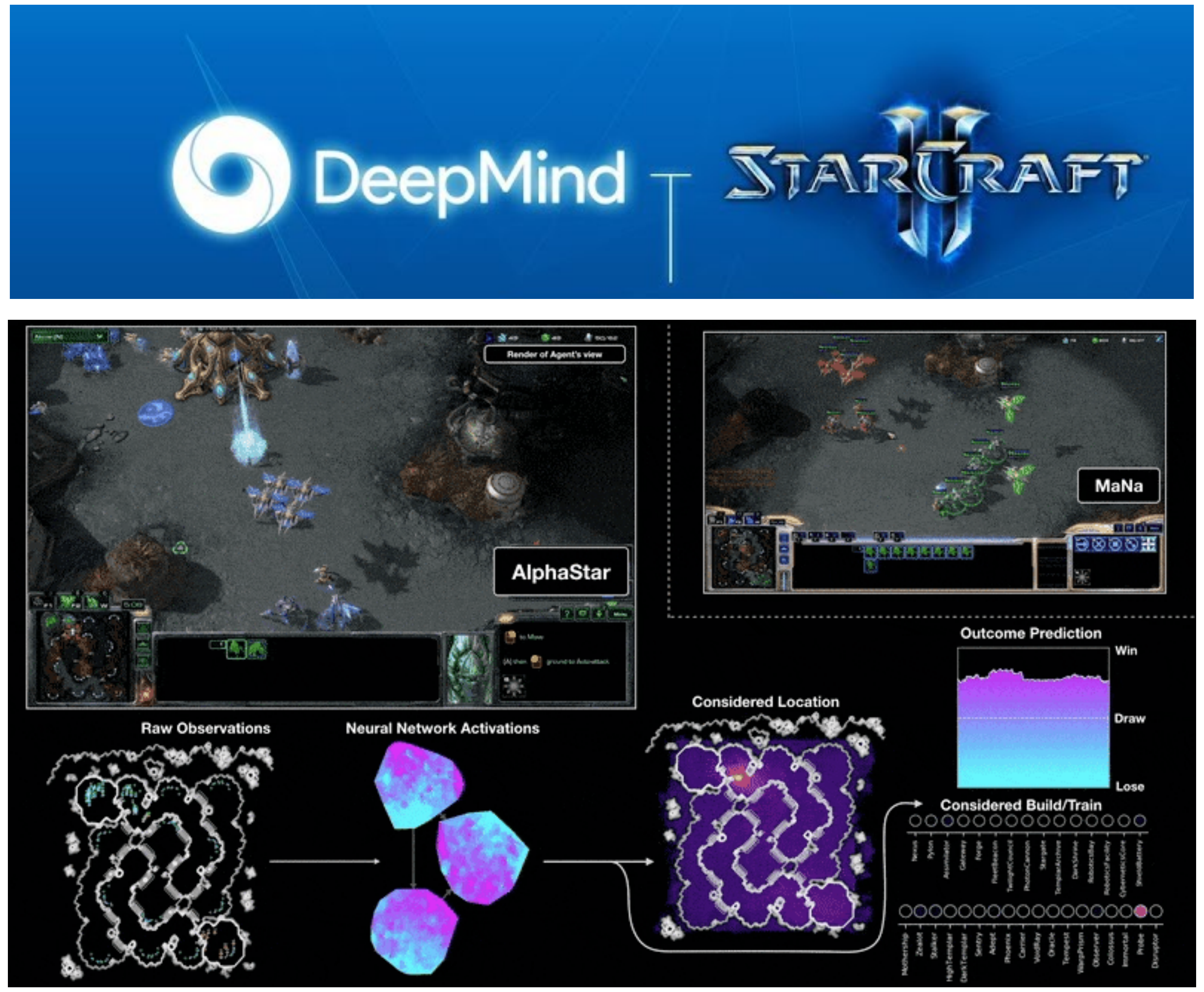
Caltech

Multi-Agent Reinforcement Learning
Theory, Algorithms, and Future Directions

Eric Mazumdar

Computing + Mathematical Sciences and Economics

Reinforcement Learning has been the driver behind many of AI's "successes"



The Guardian

US

My poker face: AI wins multiplayer game for first time

Pluribus wins 12-day session of Texas hold'em against some of the world's best human players

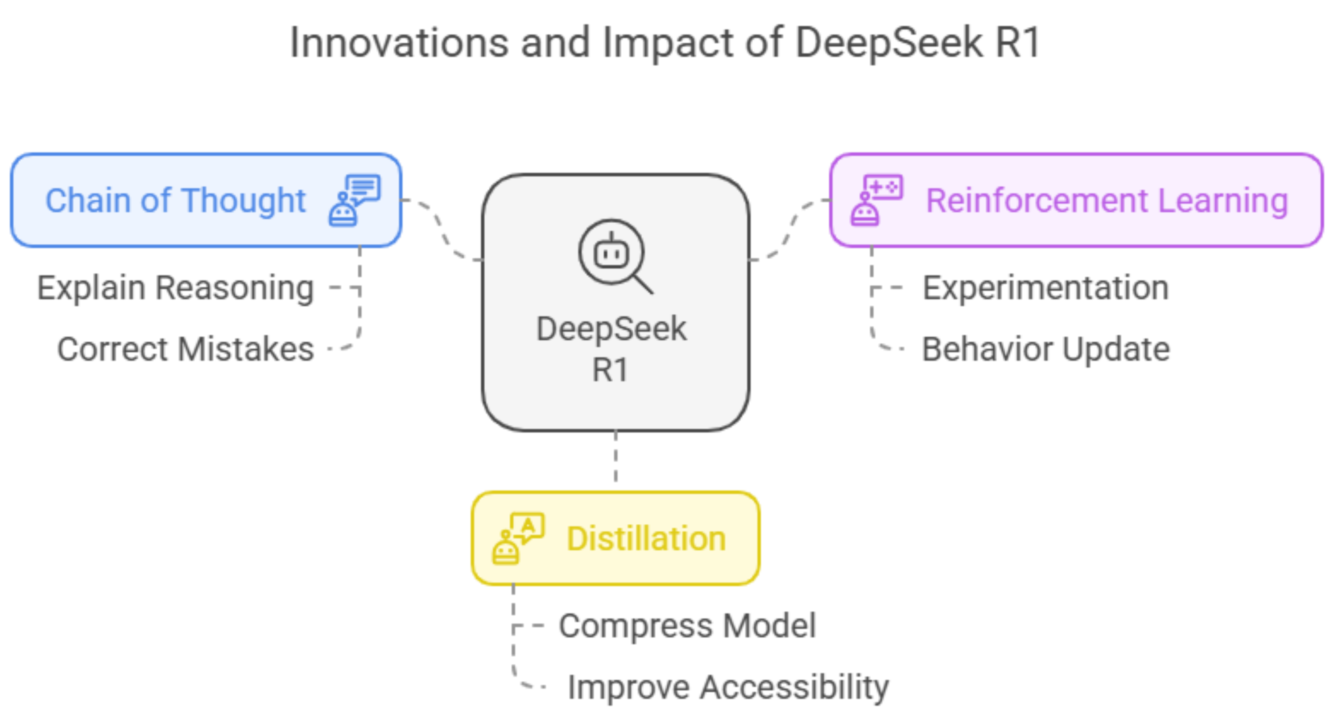
GAMING ENTERTAINMENT TECH

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

deepseek



OpenAI

A practical guide to building agents

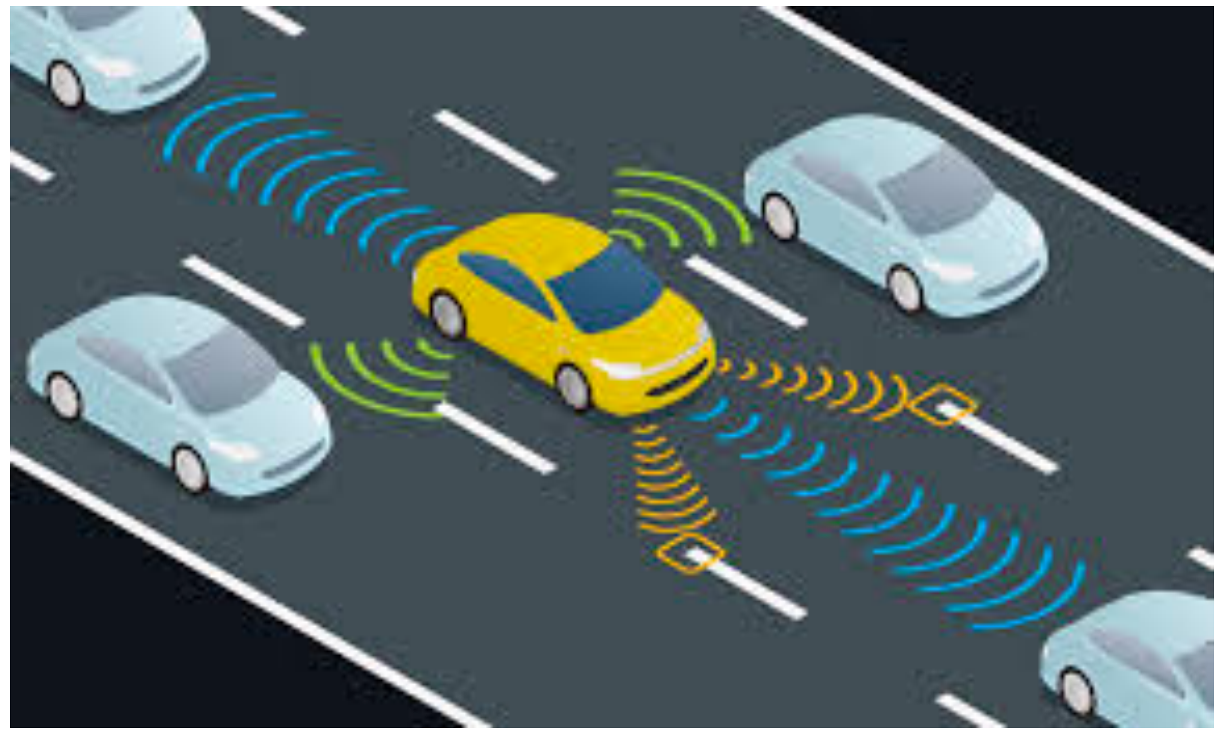
Looking under the surface, many of these are success of *Multi-Agent* Reinforcement Learning

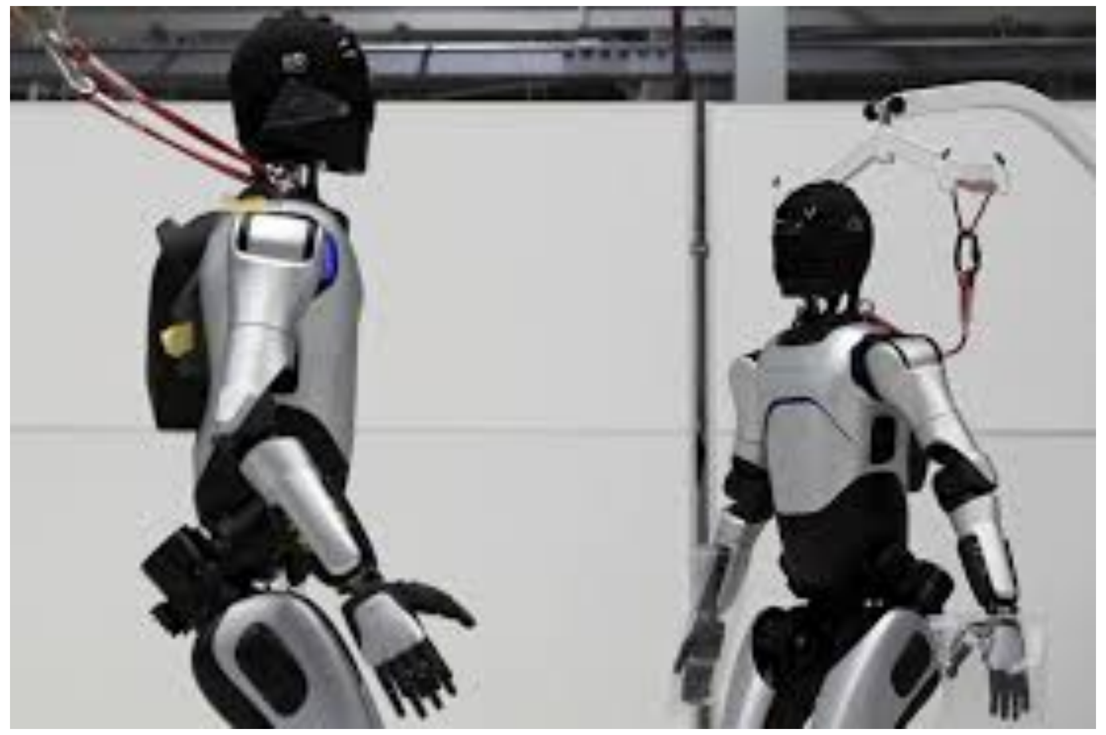

The Guardian

US

My poker face: AI wins multiplayer game for first time

Pluribus wins 12-day session of Texas hold'em against some of the world's best human players





Looking under the surface, many of these are success of *Multi-Agent* Reinforcement Learning

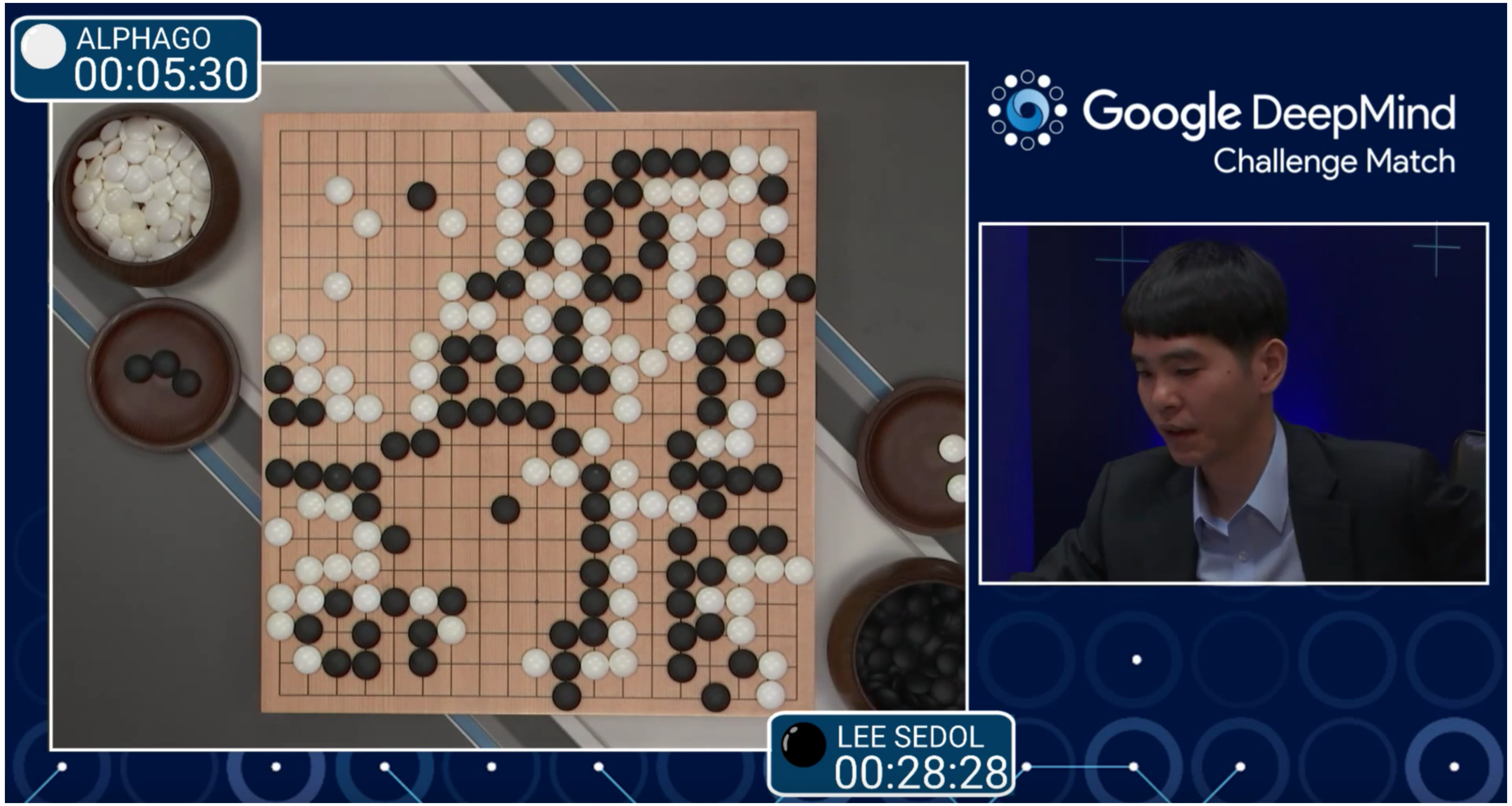
...but multi-agent RL is not very well understood

How can we better understand what constitutes a good Multi-Agent learning algorithm?

Beating the best human player?

March 2016:

Deepmind's AlphaGo beats the human champion 4-1.



How can we better understand what constitutes a good Multi-Agent learning algorithm?

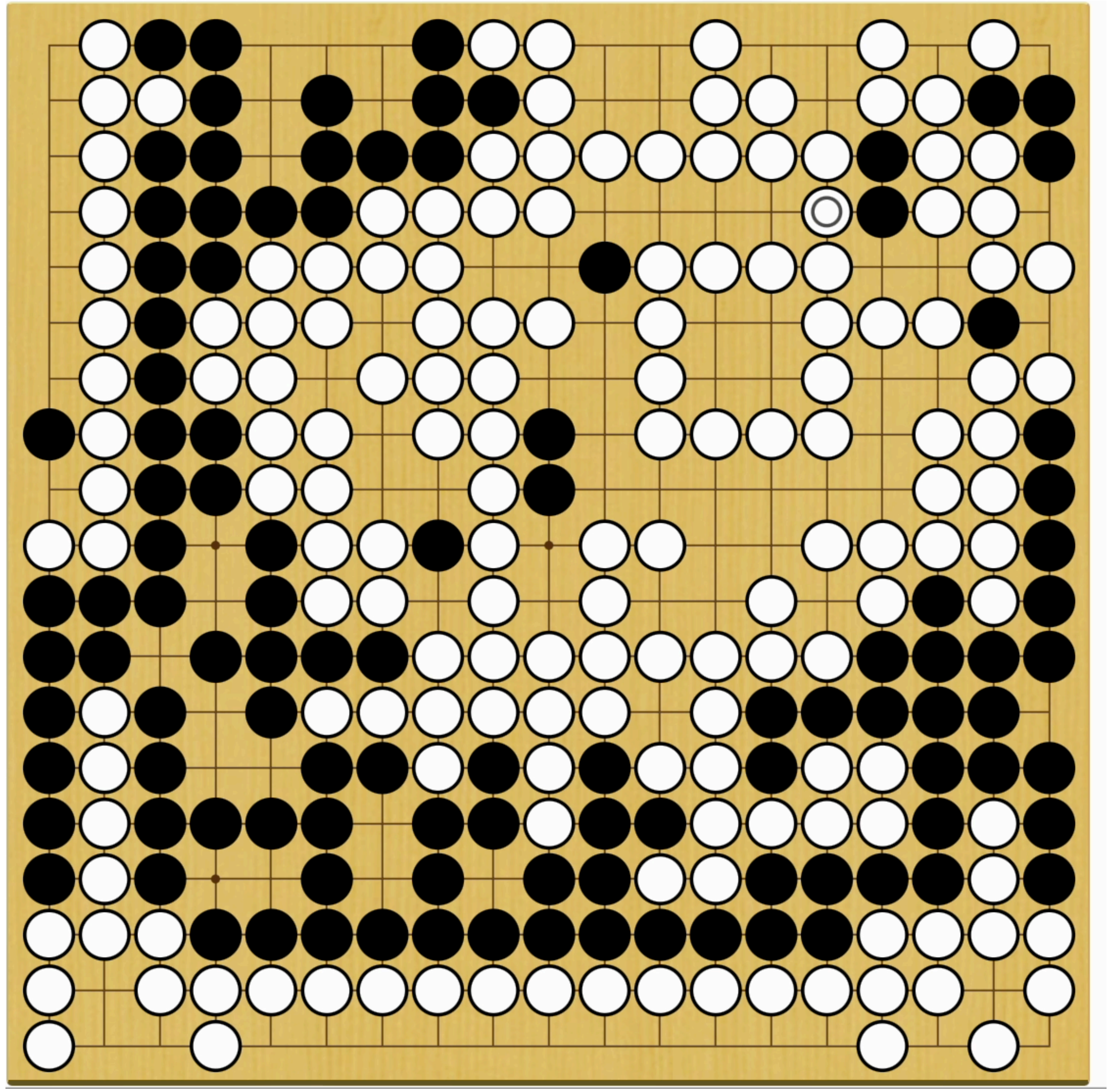
Consistently beating all players?

Adversarial Policies Beat Superhuman Go AIs

Tony Wang* Adam Gleave* Tom Tseng Nora Belrose Kellin Pelrine
Joseph Miller Michael D Dennis Yawen Duan Viktor Pogrebniak
Sergey Levine Stuart Russell

2023

Researchers show that the current best Go bot can be consistently beaten by simple strategies that can be used by amateur players.

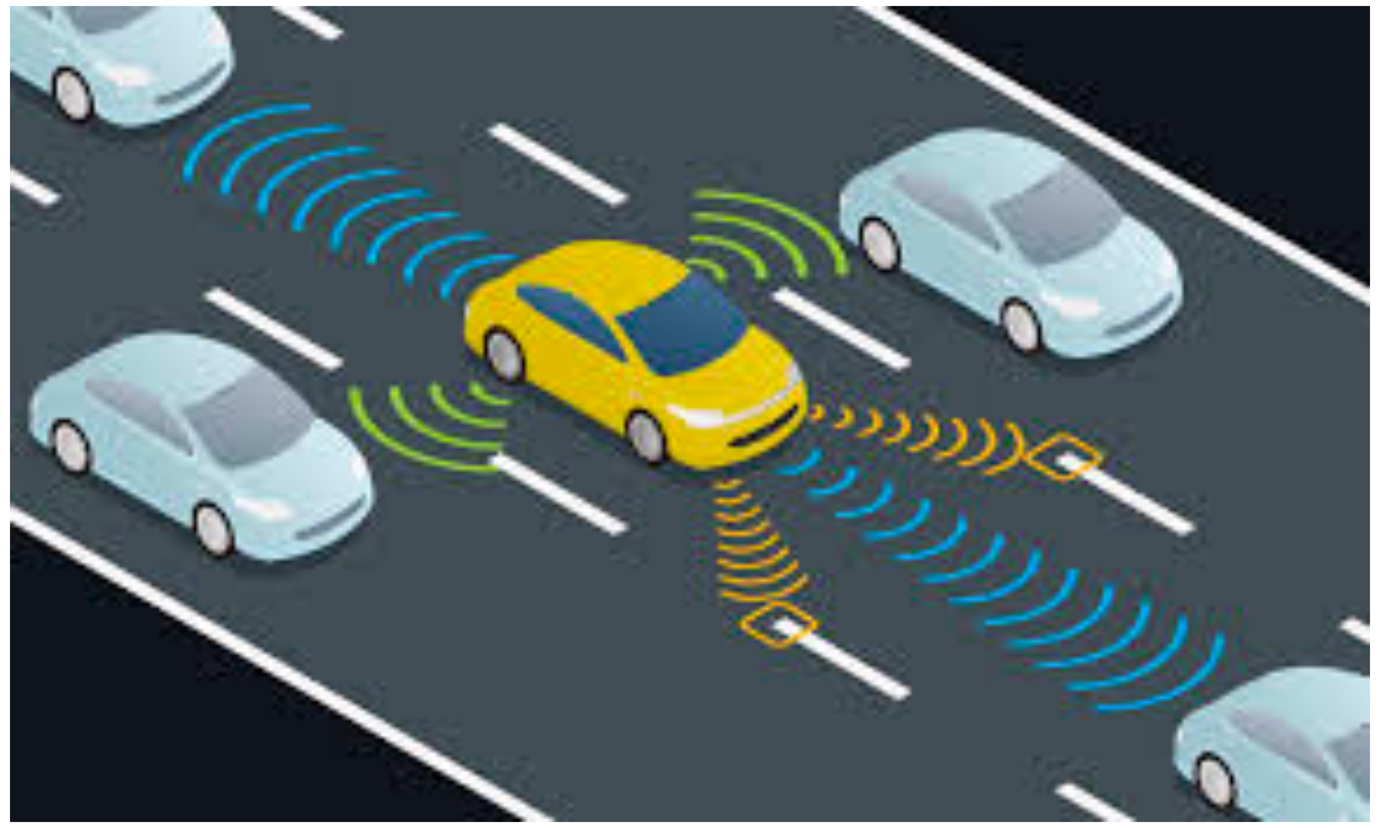


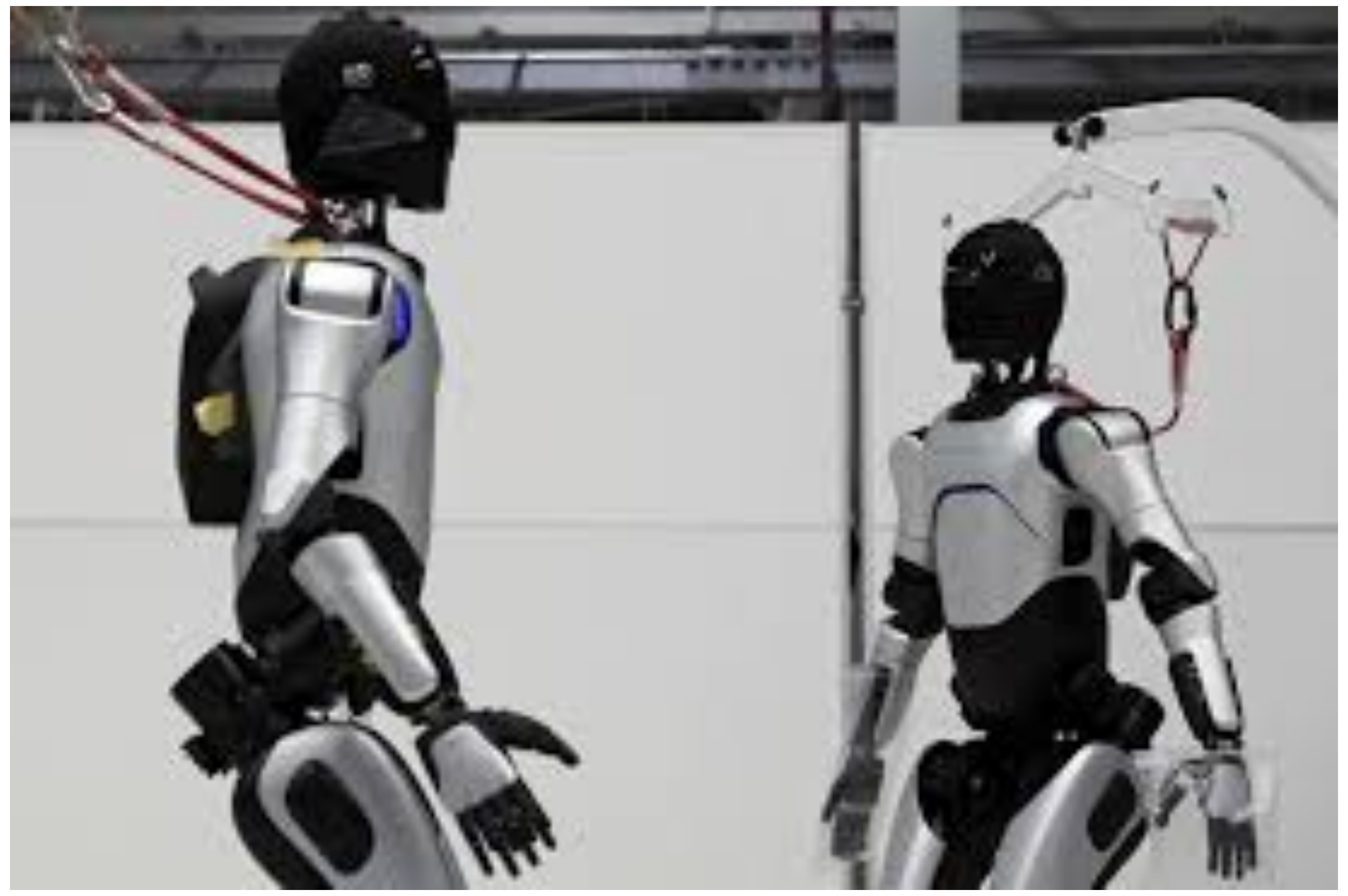
Efficiency is crucial

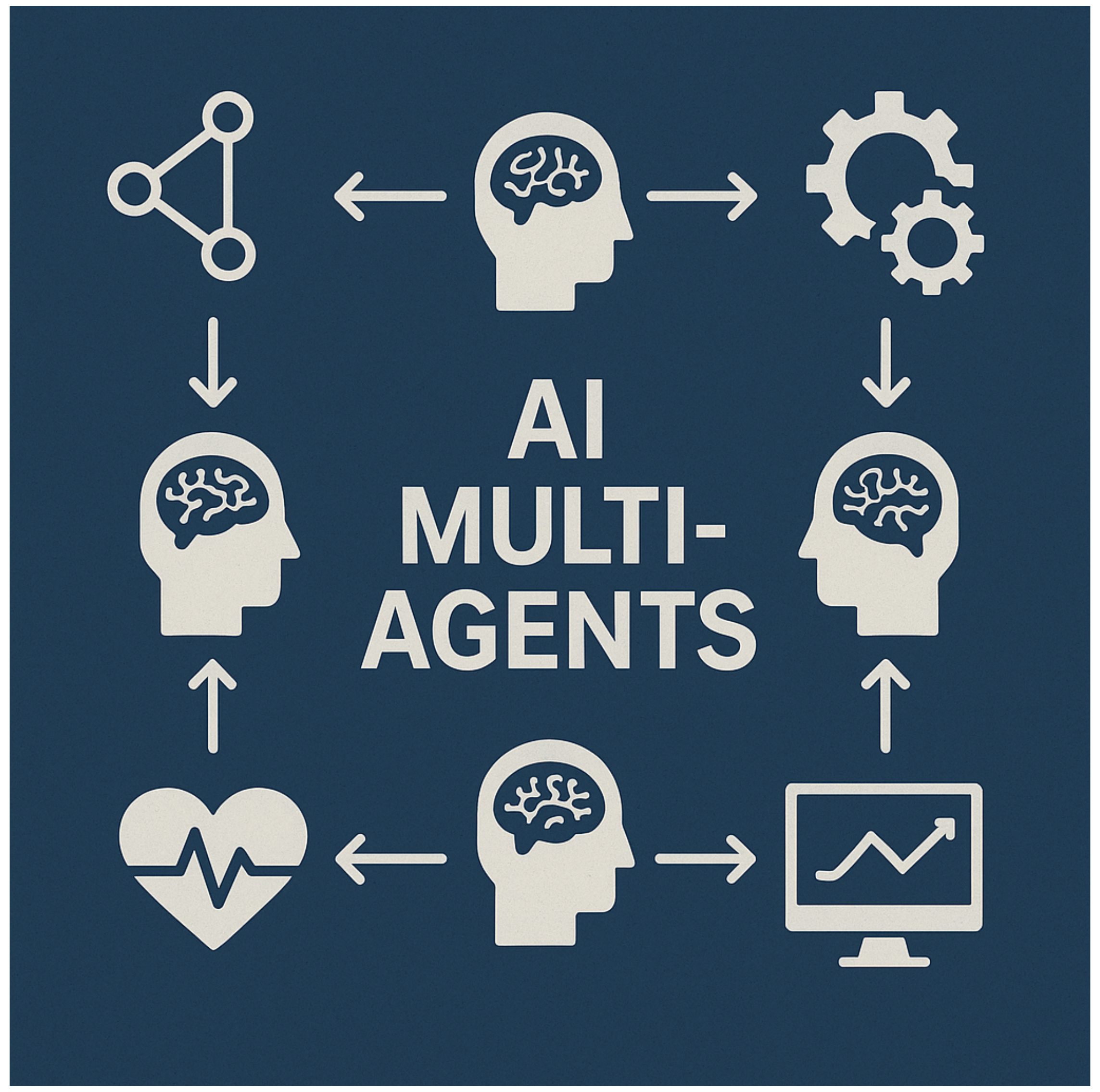
$>10^7$ games of Go
>1 month of training time on dedicated servers

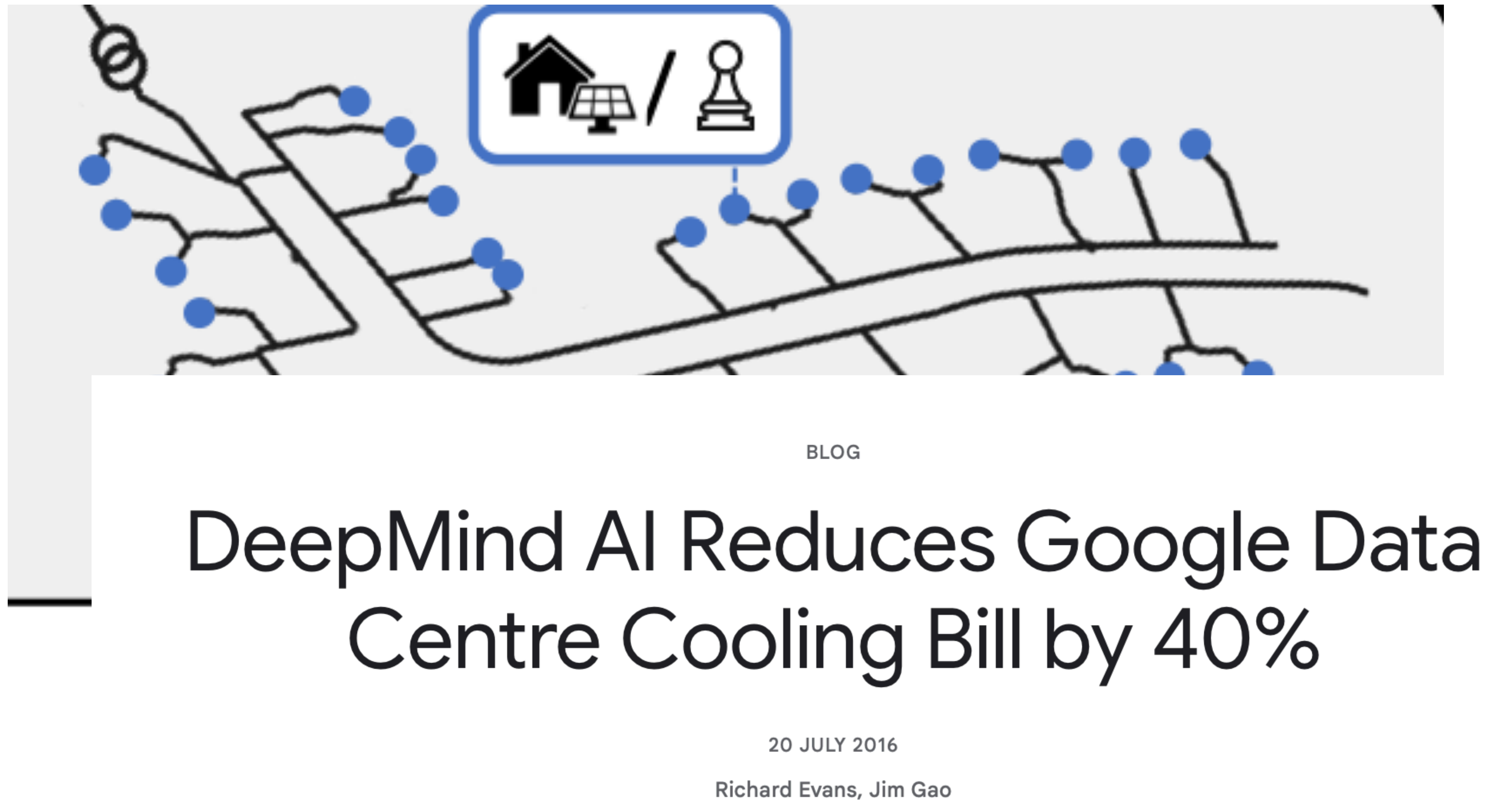
200 years of real-time StarCraft games
>1 month of training time on dedicated servers

RL algorithms are increasingly deployed in real-world systems









Can we establish a principled foundation for Multi-Agent Reinforcement Learning?

What makes these problems hard?

What *simple algorithmic principles* should we build on?

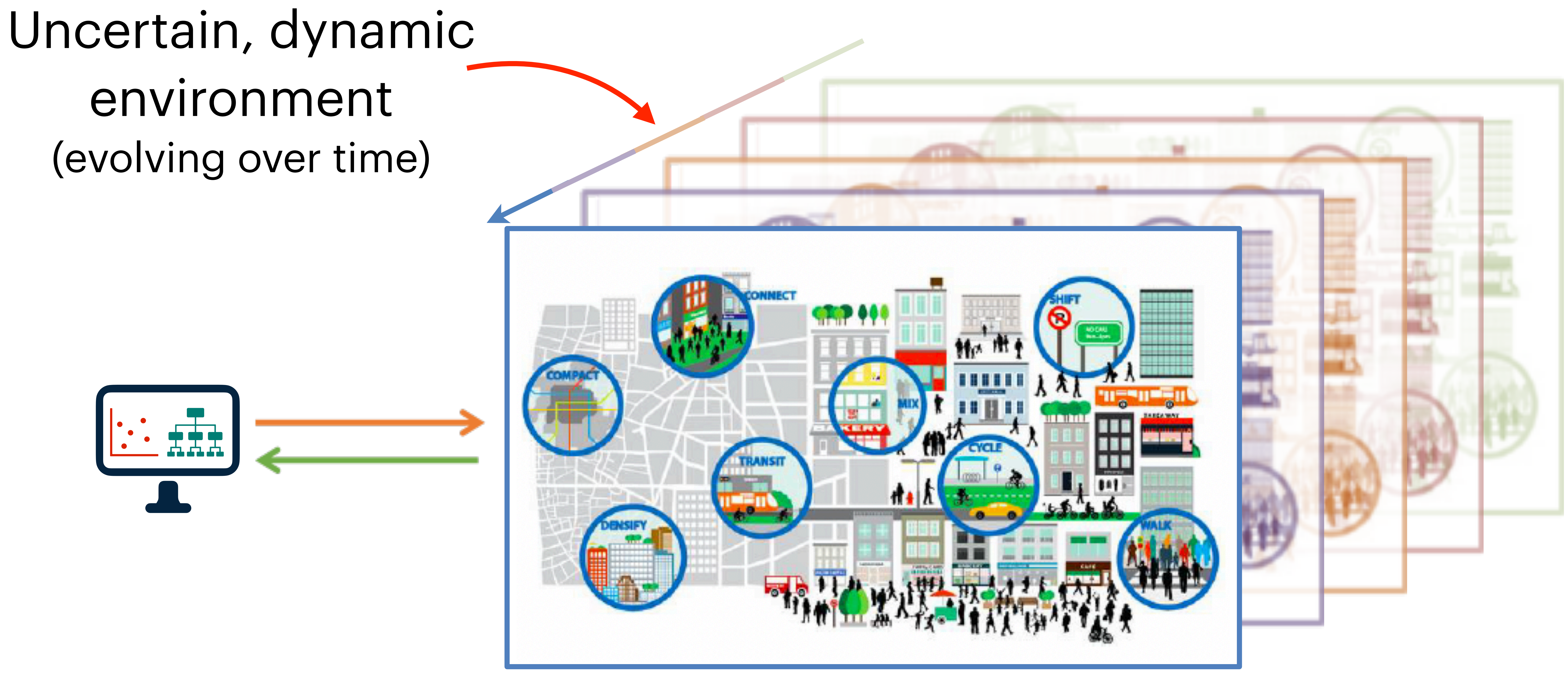
What are *fundamental limits* and how can we achieve them?

Machine Learning

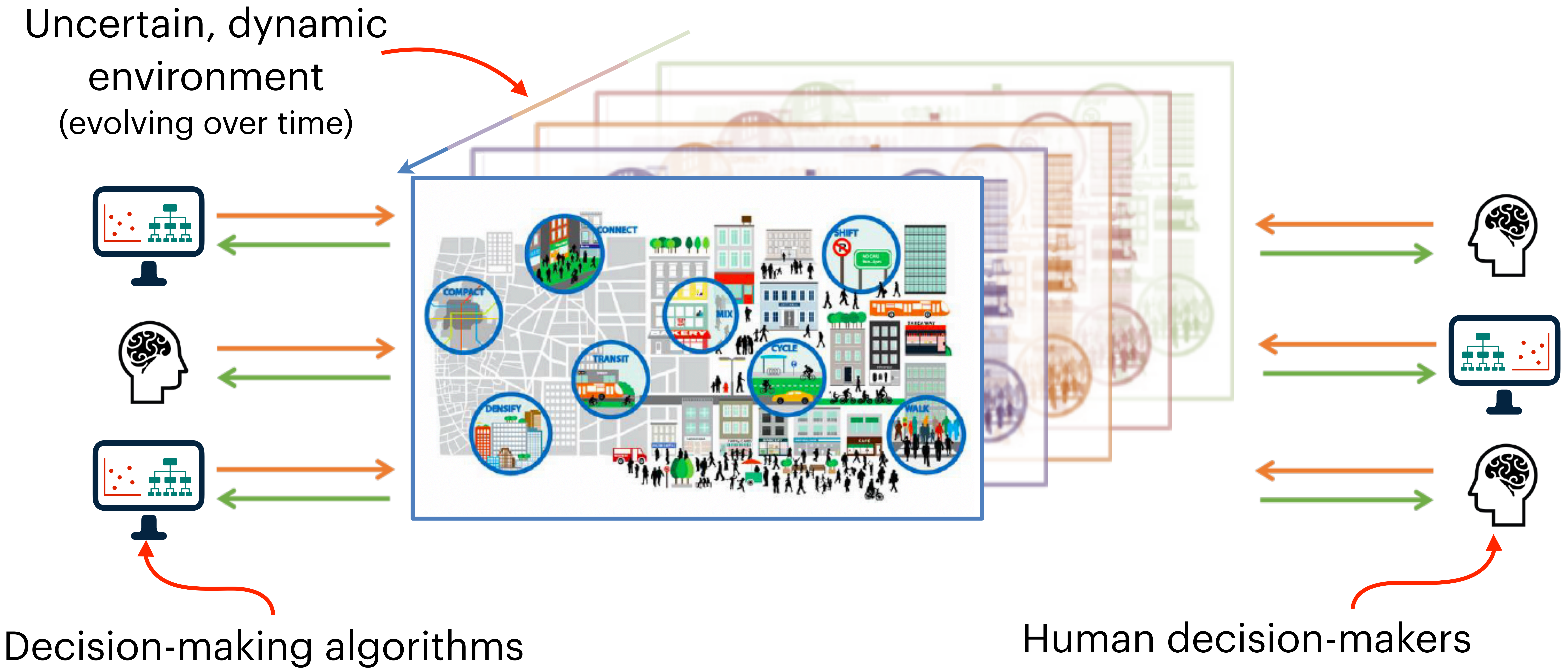
Static or stationary environment

Decision-making algorithm

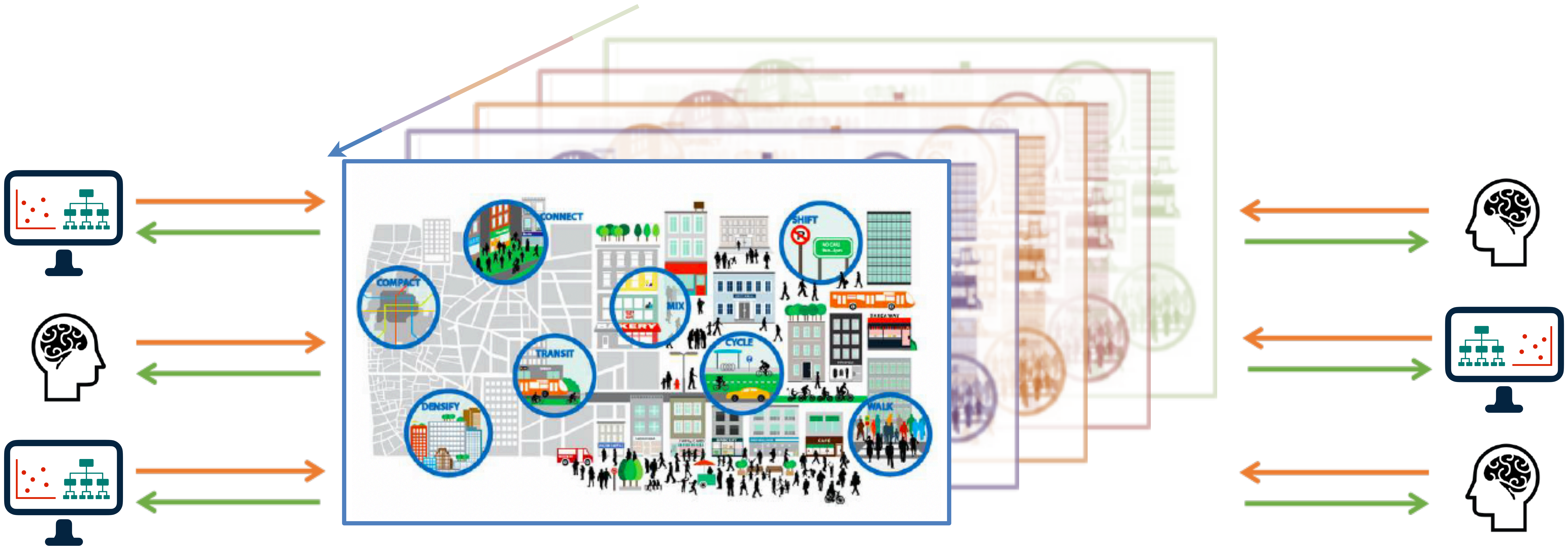
Reinforcement Learning



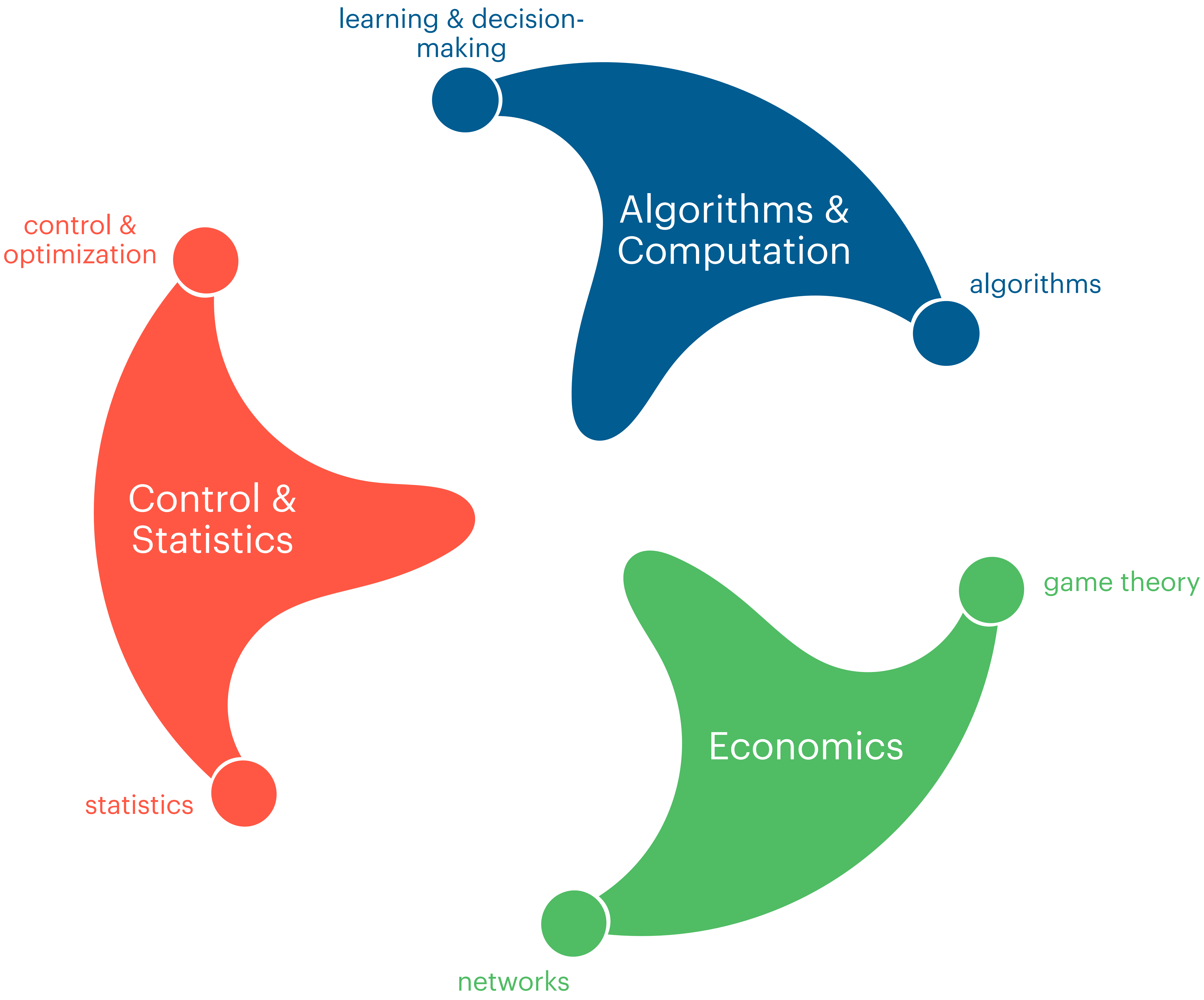
Real-World Systems are Inherently Multi-Agent

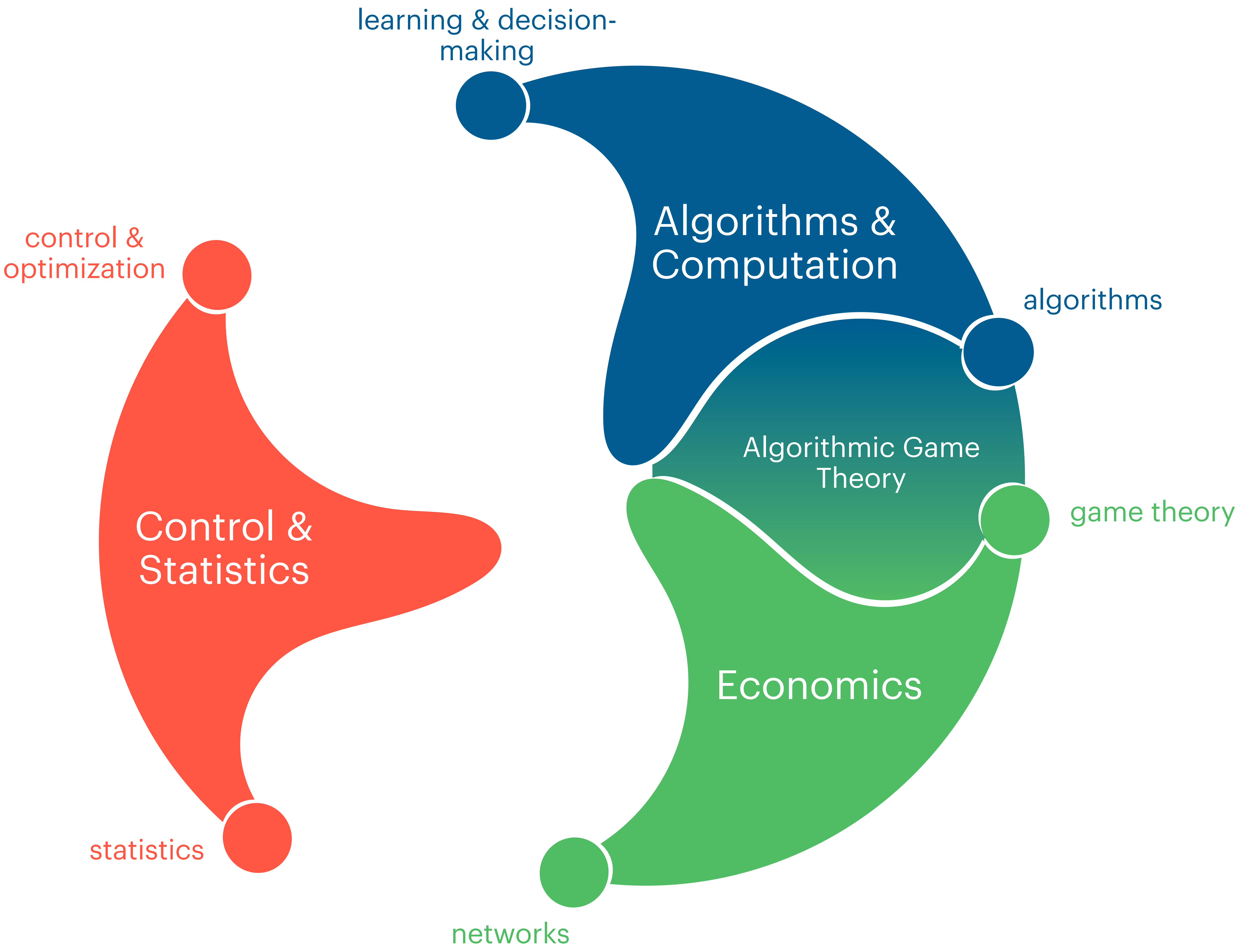


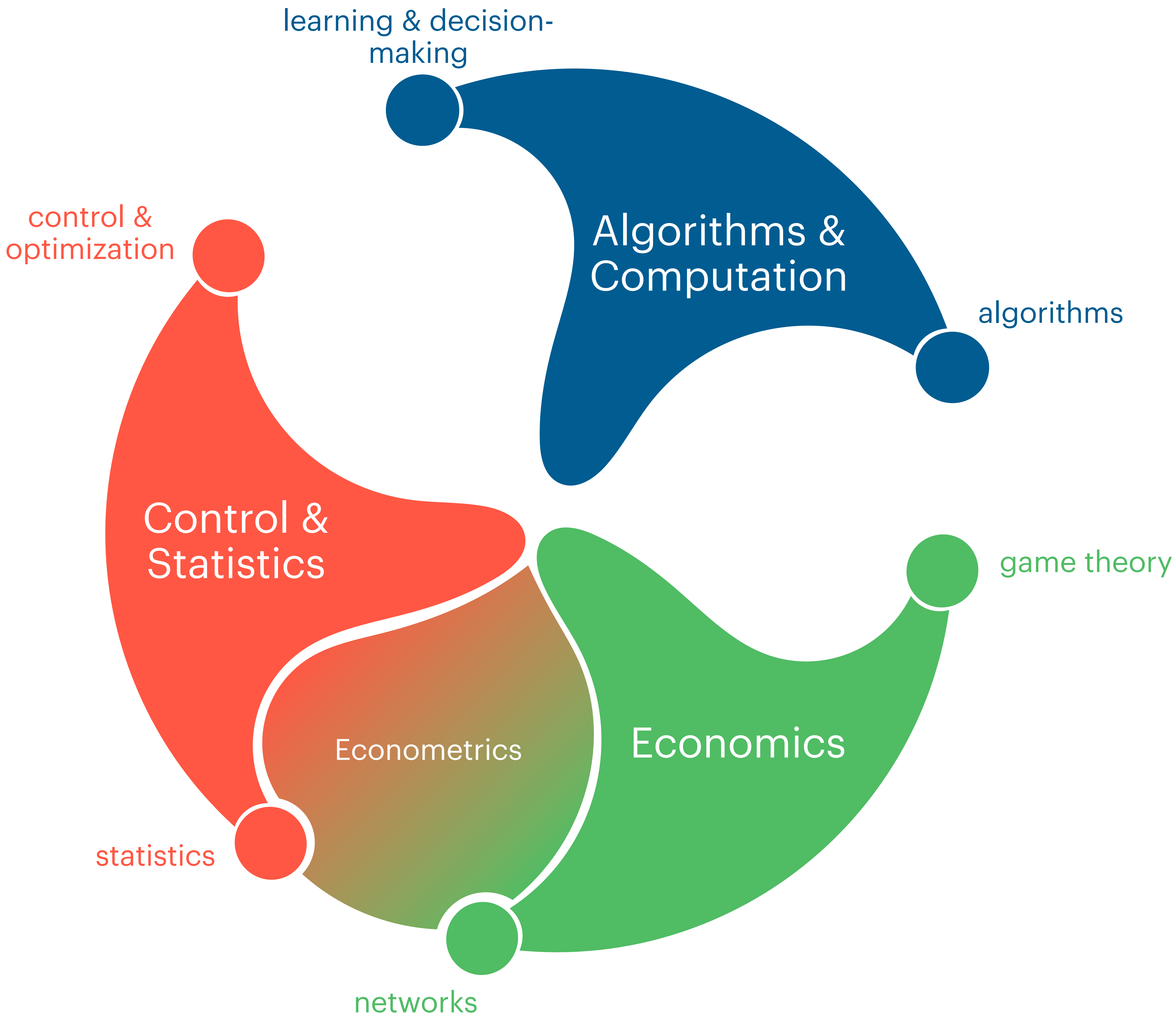
Challenges: Strategic interactions vastly complicate the task of learning

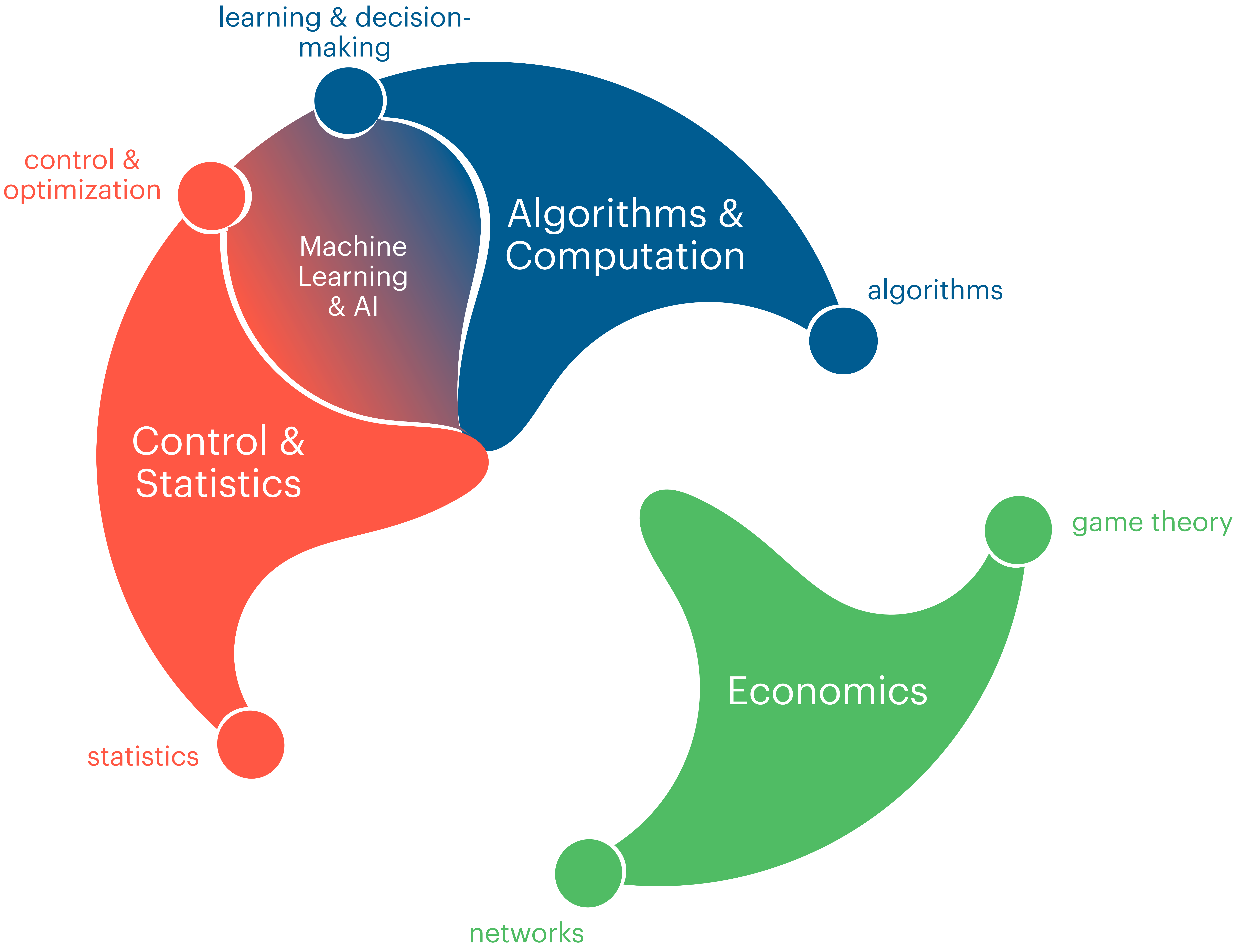


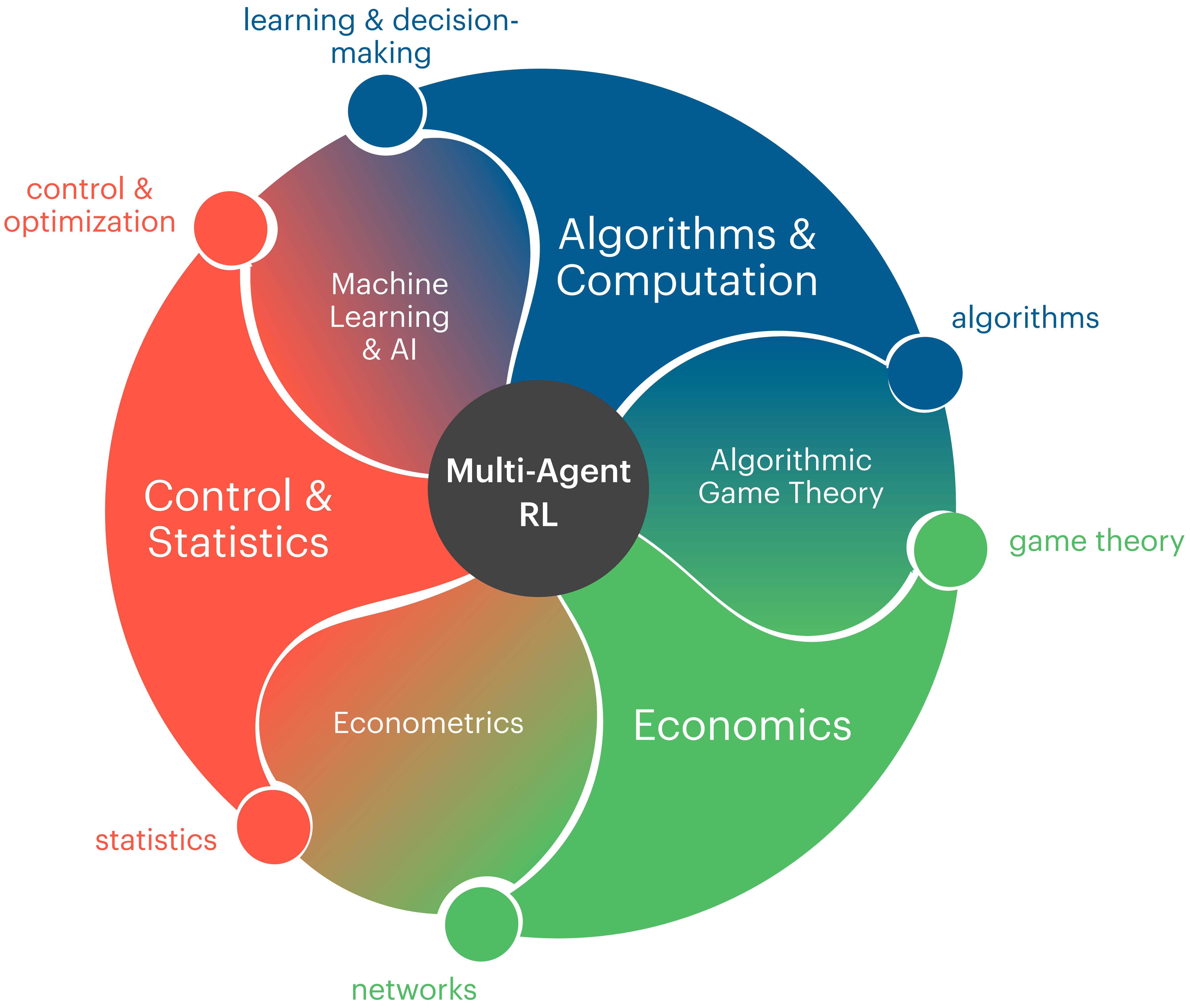
Opportunities: Require a careful rethinking of algorithm design.


learning & decision-making
Algorithms & Computation
algorithms
control & optimization
Control & Statistics
statistics
game theory
Economics
networks

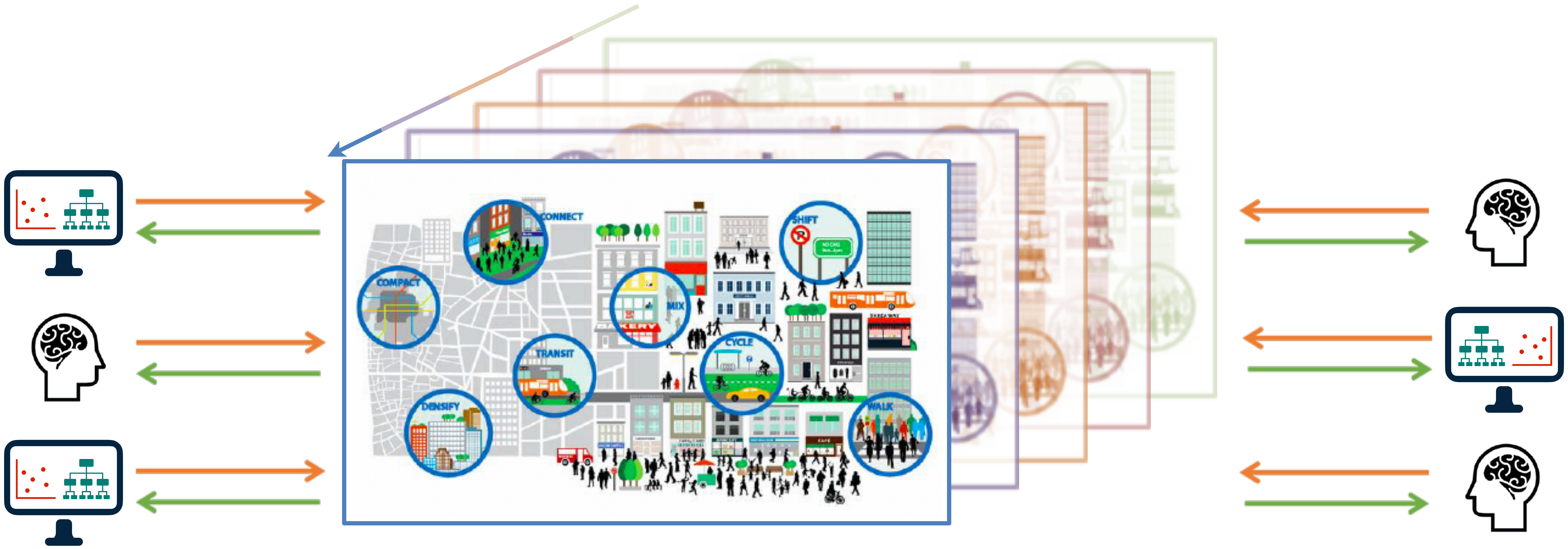

learning & decision-
making
Algorithms &
Computation
algorithms
control &
optimization
Algorithmic Game
Theory
game theory
Control &
Statistics
Economics
statistics
networks


learning & decision-making
Algorithms & Computation
algorithms
control & optimization
Control & Statistics
game theory
Econometrics
Economics
statistics
networks


learning & decision-making
control & optimization
Machine Learning & AI
Algorithms & Computation
algorithms
Control & Statistics
game theory
Economics
statistics
networks


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

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

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
Structured non-convex optimization	Structured (?) *equilibrium computation*
Stationary environment	Coupling between agents introduce *non-stationarities* in learning

Makes proving convergence of algorithms particularly difficult

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.

Main Question:

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

We will focus on theoretical foundations.

Disclaimer:

This talk will hardly be an exhaustive overview of MARL.

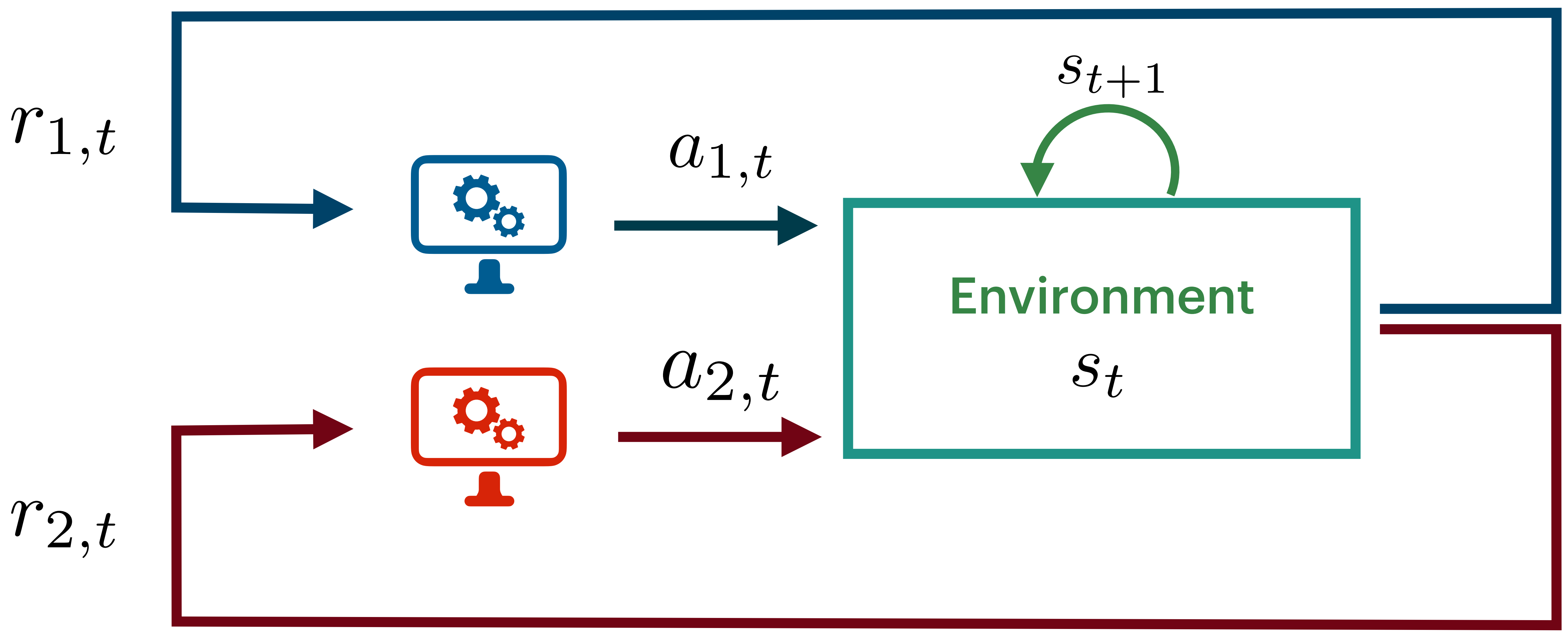
I hope to give you intuition for why it is difficult, what general principles are used, and potential new research directions.

To that end I have selected a set of results that I hope make the point.

Markov Games

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

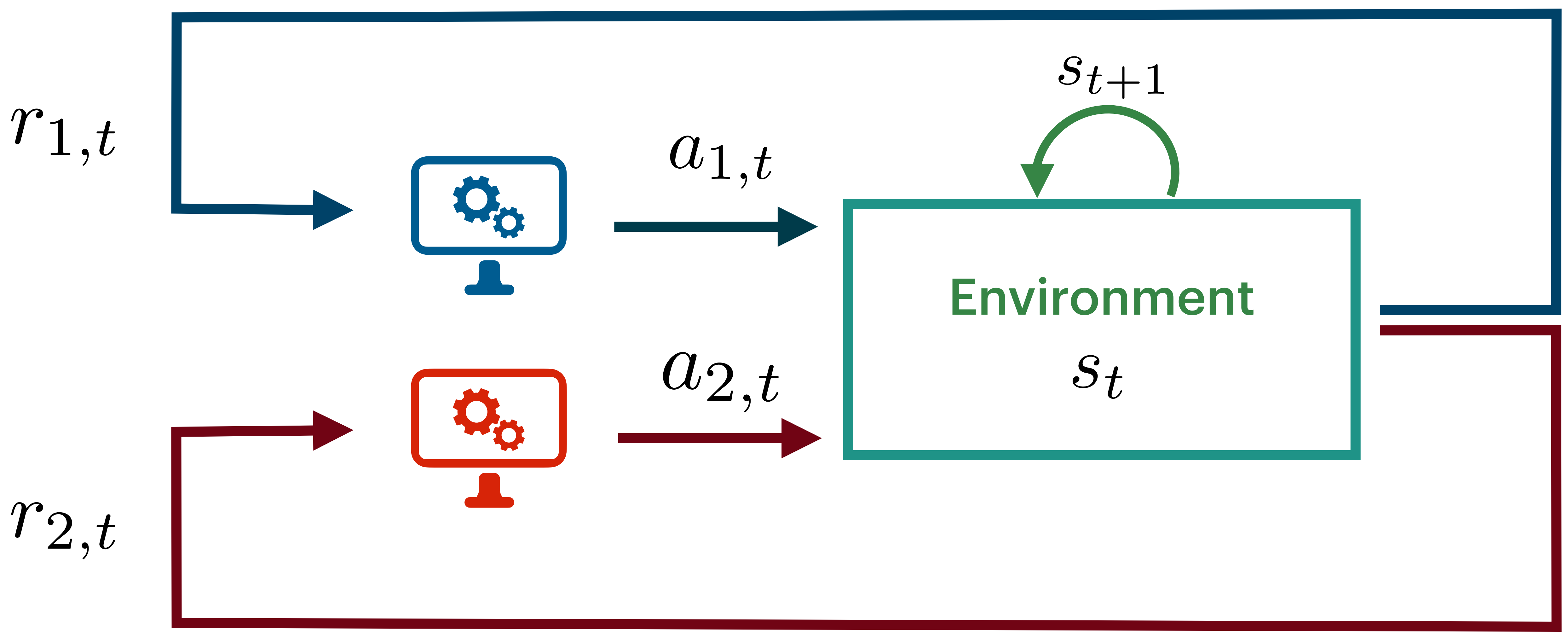
- Action Spaces: $\mathcal{A}_1, ..., \mathcal{A}_n, \quad \mathcal{A} = \prod_{i=1}^{n} \mathcal{A}_i$
- State Spaces: $\mathcal{S}$
- Dynamics: $P(s' \mid s, a_1, ..., a_n)$
- Reward functions: $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
- Horizon: H or ∞
- Initial state distribution: ρ_0



Markov Games

Interaction Protocol:

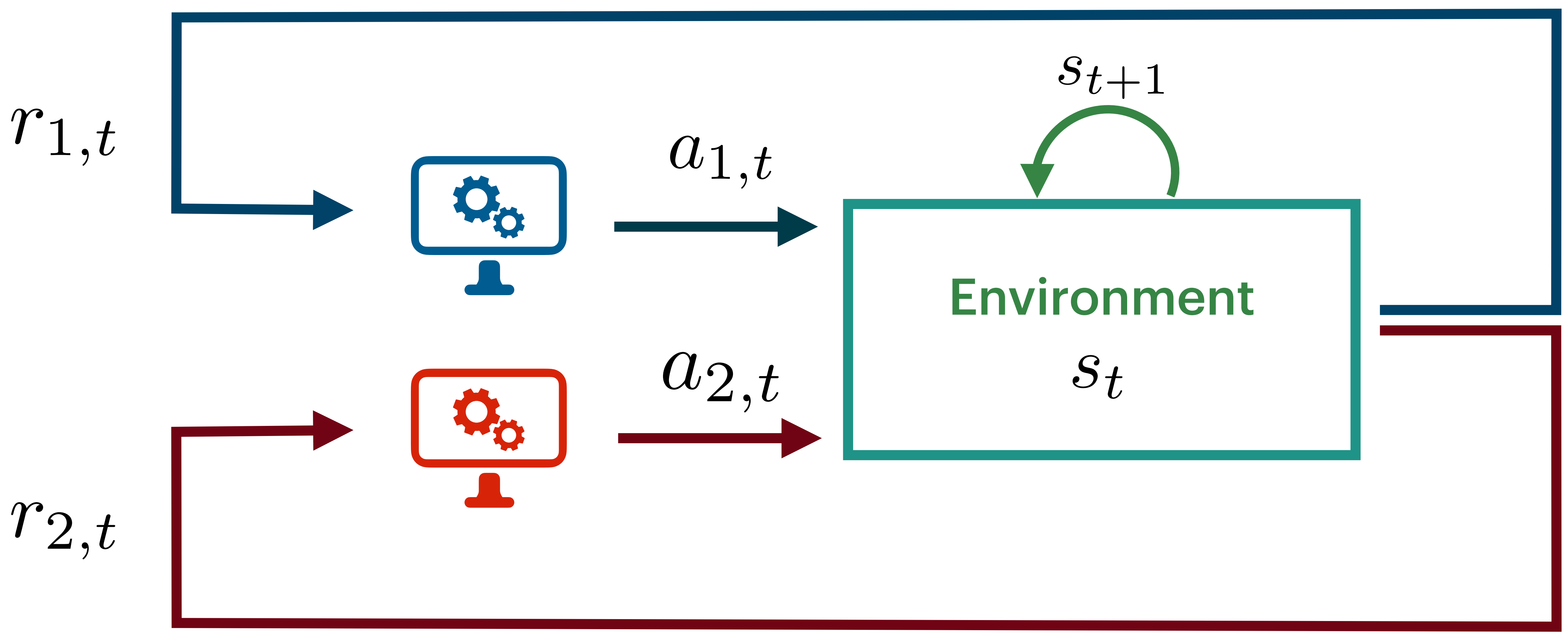
- Environment samples initial state: $s_0 \sim \rho_0$
- For step t=0,1,2,...
    - Each agent plays an action $a_{i,t}$ *simultaneously* $a_t = (a_{1,t}, \ldots a_{n,t})$
    - Agents receive their immediate reward: $r_{i,t} = R_i(s_t, a_t)$
    - Environment transitions to the next state: $s_{t+1}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



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:

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

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 i depends on the policy of agent i as well as the policies of all other agents π_{-i}

Infinite Horizon:

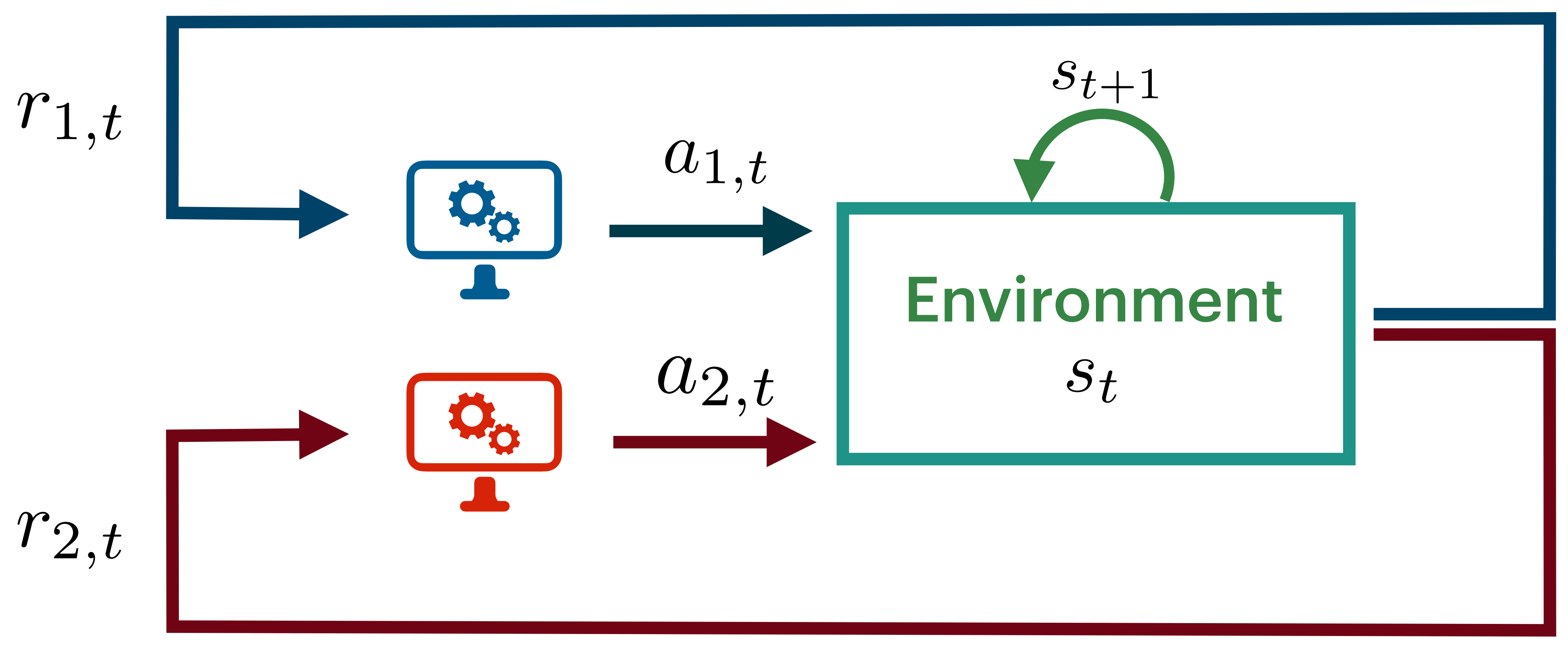
$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_{i,t}\right]$$

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

Recap: Markov Games Setup

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

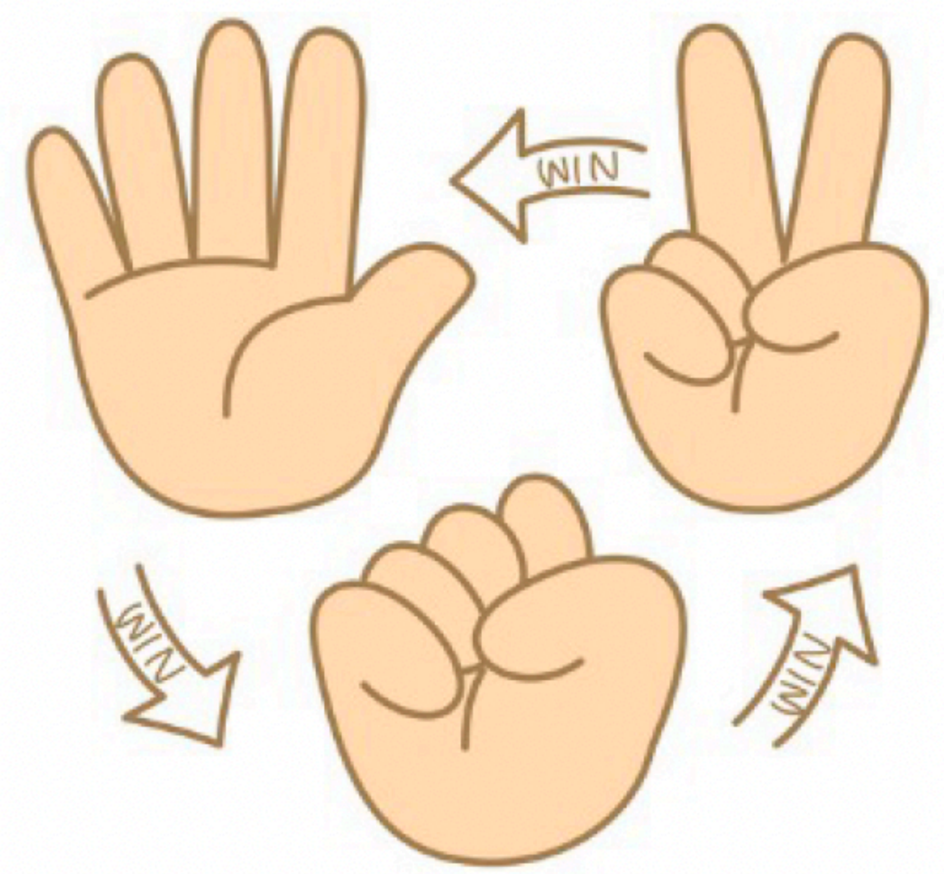
$$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{\pi, P, \rho_0}\left[\sum_{t=0}^{\infty} \gamma^t r_{i,t}\right]$$



Special Cases:

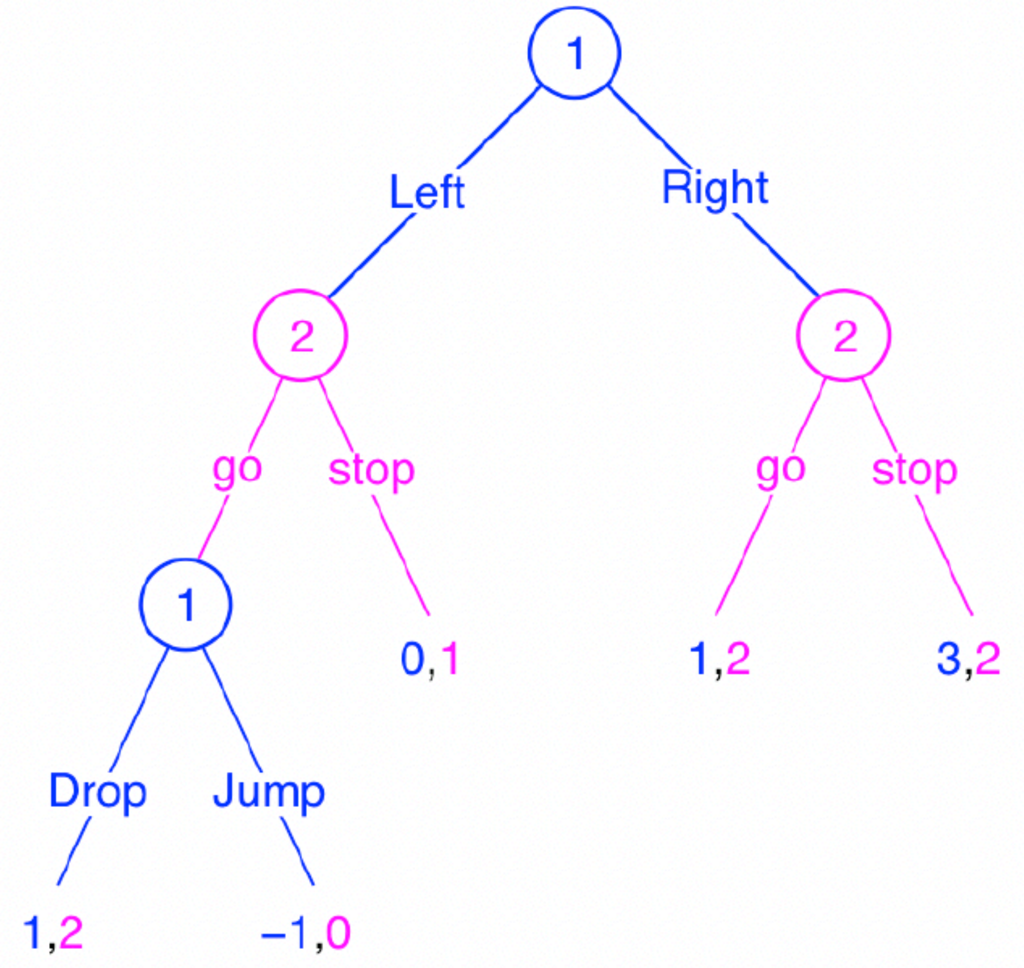
- Single-agent RL
- Two-player Zero-sum $(R_1 = -R_2)$
- Cooperative $(R_i = R_j \quad \forall i, j)$

Expressivity of Markov Games



Generalizes Classic Game Theoretic Paradigms:

- Normal-form games (no state transitions/single state).
  - e.g., repeated prisoner's dilemma, rock-paper-scissors,...
- Extensive-form games (tree structured state transition).
  - e.g., poker, Go, card games



Outcomes

What are good outcomes for Markov Games?

- In a single agent RL problem, the goal is to maximize reward: $\max_{\pi \in \Pi} U(\pi)$

- In games, each player would like to maximize their own utility but their objectives may not be aligned.
  e.g., Zero-sum games, an *increase* in my utility is a *decrease* in my opponent's.

  - Find a policy that best exploits my opponent's policy:

$$BR(\pi_{-i}) = \arg\max_{\pi_i \in \Pi_i} U(\pi_i, \pi_{-i})$$

Best-response to π_{-i}

Outcomes

What are good outcomes for Markov Games?

- In a single agent RL problem, the goal is to maximize reward: $\max_{\pi \in \Pi} U(\pi)$

- In games, each player would like to maximize their own utility but their objectives may not be aligned.
  e.g., Zero-sum games, an *increase* in my utility is a *decrease* in my opponent's.

    - Find a policy that best exploits my opponent's policy:

$$BR(\pi_{-i}) = \arg\max_{\pi_i \in \Pi_i} U(\pi_i, \pi_{-i})$$

Good against a fixed strategy, but may not be good if my opponents adapt!

A good solution for games should be an equilibrium: No player should deviate under "rational"-play

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 space of non-stationary Markov policies for *finite horizon* games.
  - In space of *stationary* Markov policies for *infinite horizon* games.

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: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Pi_i$

- **In zero-sum games $(R_1 = -R_2)$ the Nash equilibrium is the min-max solution satisfying**

$$\min_{\pi_2 \in \Pi_2} \max_{\pi_1 \in \Pi_1} U(\pi_1, \pi_2) = U(\pi_1^*, \pi_2^*) = \max_{\pi_1 \in \Pi_1} \min_{\pi_2 \in \Pi_2} U(\pi_1, \pi_2)$$

Analog to Von Neumann's minimax theorem (though it is proved by Shapley via dynamic programming since U is not convex in π_2 or concave in π_1)

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: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Pi_i$

- **Unfortunately computing a Nash equilibrium even in simple 2-player normal-form games is computationally hard.**

Thm [Daskalakis & Papadimitriou 2009]:

Computing a Nash equilibrium of a 2-player normal-form game is in PPAD

Class of problems that are generally considered to be intractable (like NP-hard) - computing a Brouwer fixed point

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: $U_i(\pi_i^*, \pi_{-i}^*) \geq U_i(\pi_i, \pi_{-i}^*) \quad \forall \pi_i \in \Pi_i$

Thm [Daskalakis & Papadimitriou 2009]:

Computing a Nash equilibrium of a 2-player normal-form game is in PPAD

- **Are there other equilibrium concepts more amenable to learning?**

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

Normal-form & concave games

To begin understanding learning in Markov games, we focus on normal-form and concave games:

Normal-form game: $U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{a\sim\pi}[R_i(a)] \quad ; \pi_i \in \Delta_i$

Policy space simplifies to the $|\mathcal{A}_i|$- dimensional simplex.

Thm [Nash 1950]:

Nash equilibria always exist in mixed strategies in normal-form games.

Normal-form & concave games

To begin understanding learning in Markov games, we focus on normal-form and concave games:

Normal-form game: $$U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{a \sim \pi}[R_i(a)] \quad ; \pi_i \in \Delta_i$$

In two-player games, this simplifies to a simple ***matrix game:*** $U_i(\pi_i, \pi_{-i}) = \pi_i^T R_i \pi_{-i}$

$|\mathcal{A}_i| \times |\mathcal{A}_{-i}|$ dimensional matrix.

Normal-form & concave games

This can be generalized to a general class of *concave games:*

$U_i(\pi_i, \pi_{-i})$ is concave in $\boldsymbol{\pi}_i$ for all fixed $\boldsymbol{\pi}_{-i}$

Thm [Rosen 1965]:

Nash equilibria always exist in concave games over compact & convex strategy spaces.

Learning in concave games

Consider players in ***concave games:*** $U_i(\pi_i, \pi_{-i})$ is concave in $\boldsymbol{\pi}_i$ for all fixed $\boldsymbol{\pi}_{-i}$

Assume we are in the full information regime, where players know their utility, observe their opponents' full policy, and seek to adapt ***online*** to their opponents' strategies:

Learning in concave games

Consider players in ***concave games:*** $U_i(\pi_i, \pi_{-i})$ is concave in $\boldsymbol{\pi}_i$ for all fixed $\boldsymbol{\pi}_{-i}$

Assume we are in the full information regime, where players know their utility, observe their opponents' full policy, and seek to adapt ***online*** to their opponents' strategies:

Interaction Protocol:

- Each agent chooses an initial action, $\boldsymbol{\pi}_{i,0}$
- For step t=0,1,2,...
  - Each agent plays an action $\boldsymbol{\pi}_{i,t}$ *simultaneously*
  - Agents receive their immediate utility $U_i(\pi_{i,t}, \pi_{-i,t})$ and observe opponent's policy $\pi_{-i,t}$
  - Agents update their policy given their observation: $\pi_{i,t+1} = g_i(\pi_{i,t}, \pi_{-i,t})$

What do good algorithms look like?

Algorithms for learning in games

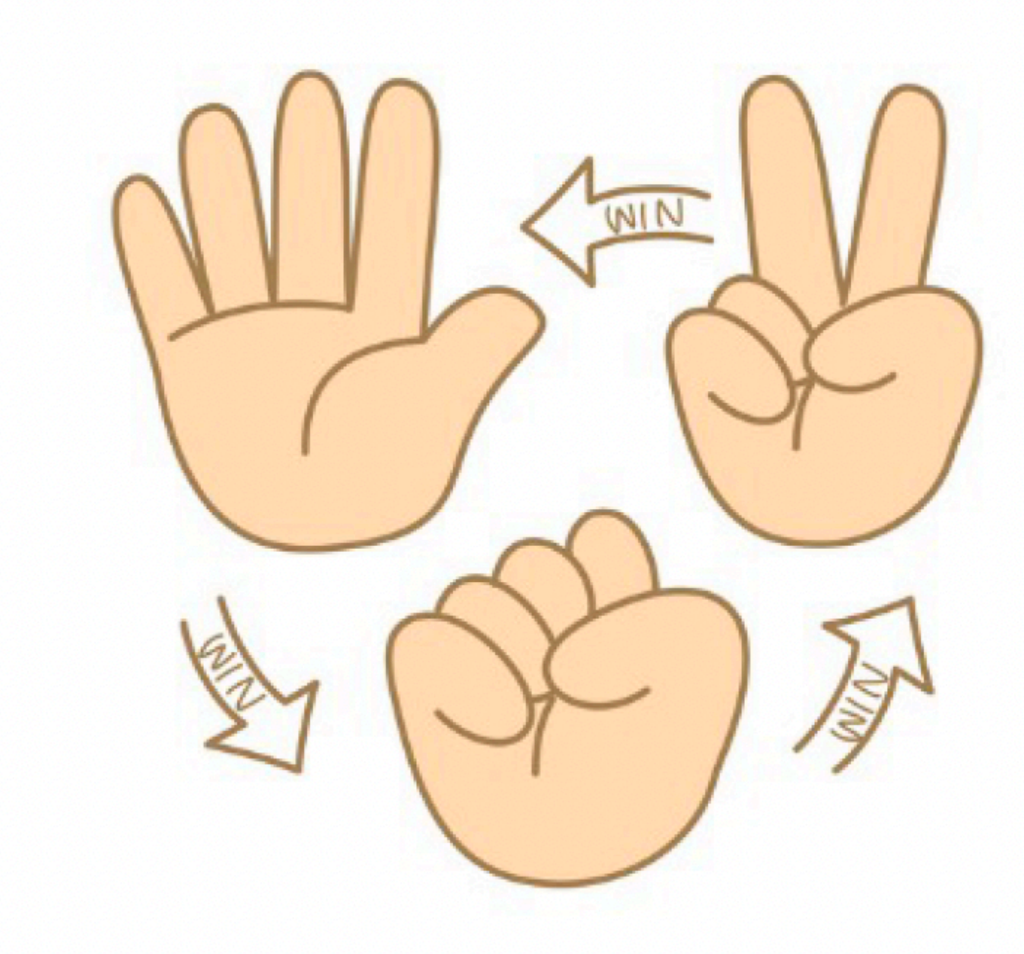
The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Best-Response Dynamics

- Each agent chooses an initial action, $\pi_{i,0}$
- For step t=0,1,2,...

$$\pi_{i,t+1} = BR(\pi_{-i,t})$$



Does not converge to Nash even in rock-paper-scissors!
(Too greedy)

Algorithms for learning in games

The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Fictitious-Play [Brown, 1949]

- Each agent initializes their belief over their opponents' strategy $\hat{\pi}_{-i}$.
- For step t=0,1,2,...
  - Play $a_{i,t} \sim \pi_{i,t+1} = BR(\hat{\pi}_{-i,t})$
  - Observe $a_{-i,t}$
  - Update belief $\hat{\pi}_{-i,t+1} = \hat{\pi}_{-i,t} + \frac{1}{t+1}(e(a_{-i,t}) - \hat{\pi}_{-i,t})$

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

Algorithms for learning in games

The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Fictitious-Play [Brown, 1949]

- Each agent initializes their belief over their opponents' strategy $\hat{\pi}_{-i}$.
- For step t=0,1,2,...
  - Play $a_{i,t} \sim \pi_{i,t+1} = BR(\hat{\pi}_{-i,t})$
  - Observe $a_{-i,t}$
  - Update belief $\hat{\pi}_{-i,t+1} = \hat{\pi}_{-i,t} + \frac{1}{t+1}(e(a_{-i,t}) - \hat{\pi}_{-i,t})$

Robinson [1951]
Fictitious-Play asymptotically converges to Nash eq. in zero-sum games.

Algorithms for learning in games

The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Fictitious-Play [Brown, 1949]

- Each agent initializes their belief over their opponents' strategy $\hat{\pi}_{-i}$.
- For step t=0,1,2,...
  - Play $a_{i,t} \sim \pi_{i,t+1} = BR(\hat{\pi}_{-i,t})$
  - Observe $a_{-i,t}$
  - Update belief $\hat{\pi}_{-i,t+1} = \hat{\pi}_{-i,t} + \frac{1}{t+1}(e(a_{-i,t}) - \hat{\pi}_{-i,t})$

Robinson [1951]

Fictitious-Play asymptotically converges to Nash eq. in zero-sum games.

- In his proof, convergence to an epsilon-approximate Nash equilibrium took at most $1/\epsilon^{\Omega(A)}$ iterations

Algorithms for learning in games

The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Fictitious-Play [Brown, 1949]

- Each agent initializes their belief over their opponents' strategy $\hat{\pi}_{-i}$.
- For step t=0,1,2,...
  - Play $a_{i,t} \sim \pi_{i,t+1} = BR(\hat{\pi}_{-i,t})$
  - Observe $a_{-i,t}$
  - Update belief $\hat{\pi}_{-i,t+1} = \hat{\pi}_{-i,t} + \frac{1}{t+1}(e(a_{-i,t}) - \hat{\pi}_{-i,t})$

Robinson [1951]

Fictitious-Play asymptotically converges to Nash eq. in zero-sum games.

- In his proof, convergence to an epsilon-approximate Nash equilibrium took at most $1/\epsilon^{\Omega(A)}$ iterations

- Karlin [1959] conjectured that it actually converged in $O(1/\epsilon^2)$ iterations
- Daskalakis & Pan [2014] *refuted* this conjecture under worst case tie-breaking, showing a $1/\epsilon^{\Omega(A)}$ rate is unavoidable

Algorithms for learning in games

The simplest algorithms for learning in games come from economics and rely on *best-response oracles*

$$BR(\pi_{-i}) = \arg\max_{\pi} \; U_i(\pi, \pi_{-i})$$

Fictitious-Play [Brown, 1949]

- Each agent initializes their belief over their opponents' strategy $\hat{\pi}_{-i}$.
- For step t=0,1,2,...
  - Play $a_{i,t} \sim \pi_{i,t+1} = BR(\hat{\pi}_{-i,t})$
  - Observe $a_{-i,t}$
  - Update belief $\hat{\pi}_{-i,t+1} = \hat{\pi}_{-i,t} + \frac{1}{t+1}(e(a_{-i,t}) - \hat{\pi}_{-i,t})$

However, fictitious-play has no convergence guarantees in general non-zero-sum games.

What are good algorithms for learning in concave games?

What properties may we want for algorithms in games?

- Independent learning
  agents should not know anything about their opponents utility
- Individually Rationalizable
  agents should be “rational” (e.g., take advantage of naive opponents)
- Convergent
  convergence to Nash

What are good algorithms for learning in concave games?

What properties may we want for algorithms in games?

- Independent learning
  agents should not know anything about their opponents utility
- Individually Rationalizable
  agents should be "rational" (e.g., take advantage of naive opponents)
- Convergent
  convergence to Nash

We've already seen that this is too much to hope for in general!
Uncoupled dynamics cannot always converge to Nash [Hart & Mas-Colell 2003]

What are good algorithms for learning in concave games?

What properties may we want for algorithms in games?

- Independent learning

  agents should not know anything about their opponents utility

- Individually Rationalizable

  agents should be “rational” (e.g., take advantage of naive opponents)

- No-regret

  Algorithm should compete with the best fixed action in hindsight:

$$\max_{\pi' \in \Delta_i} \frac{1}{T} \left(\sum_{t=1}^{T} U_i(\pi', \pi_{-i,t}) - U_i(\pi_{i,t}, \pi_{-i,t}) \right) \leq o(1)$$

What are good algorithms for learning in concave games?

What properties may we want for algorithms in games?

- Independent learning
  agents should not know anything about their opponents utility
- Individually Rationalizable
  agents should be "rational" (e.g., take advantage of naive opponents)
- No-regret
  Algorithm should compete with the best fixed action in hindsight:

$$\max_{\pi' \in \Delta_i} \frac{1}{T} \left(\sum_{t=1}^{T} U_i(\pi', \pi_{-i,t}) - U_i(\pi_{i,t}, \pi_{-i,t}) \right) \leq o(1)$$

Many well known algorithms turn out to be no-regret:

e.g., online gradient-play, multiplicative weights, mirror descent, smoothed fictitious-play

Note: fictitious-play and best-response dynamics ***are not no-regret***

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Definition: Coarse Correlated Equilibrium [Aumann 1974]

A joint distribution $\sigma \in \Delta_{\mathcal{A}}$ is a coarse correlated equilibrium (CCE) if, for all i:

$$\mathbb{E}_{\pi \sim \sigma}[U_i(\pi)] \geq \mathbb{E}_{\pi_{-i} \sim \sigma}[U_i(\pi_i', \pi_{-i})] \quad \forall \pi_i' \in \Pi_i$$

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Thm: No-regret algorithms converge to CCE

Suppose all players in a game use no-regret algorithms to choose their policies at each time. The the average sequence of play converges to a CCE.

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Thm: No-regret algorithms converge to CCE

Suppose all players in a game use no-regret algorithms to choose their policies at each time. The the average sequence of play converges to a CCE.

Proof: Almost by definition.

Consider the sequence of correlated joint strategies, $\sigma_T = \mathrm{Uniform}\left(\{\pi_t\}_{t=1}^{T}\right)$

Via no-regret: $$\max_{\pi' \in \Delta_i} \mathbb{E}_{\sigma_T}[U_i(\pi', \pi_{-i,t})] - \mathbb{E}_{\sigma_T}[U_i(\pi_t)] = \max_{\pi' \in \Delta_i} \frac{1}{T}\left(\sum_{t=1}^{T} U_i(\pi', \pi_{-i,t}) - U_i(\pi_{i,t}, \pi_{-i,t})\right) \leq o(1)$$

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Thm: No-regret algorithms converge to CCE

Suppose all players in a game use no-regret algorithms to choose their policies at each time. The the average sequence of play converges to a CCE.

Proof: Almost by definition.

Consider the sequence of correlated joint strategies, $\sigma_T = \text{Uniform}\left(\{\pi_t\}_{t=1}^T\right)$

Via no-regret: $$\max_{\pi' \in \Delta_i} \mathbb{E}_{\sigma_T}[U_i(\pi', \pi_{-i,t})] - \mathbb{E}_{\sigma_T}[U_i(\pi_t)] = \max_{\pi' \in \Delta_i} \frac{1}{T}\left(\sum_{t=1}^{T} U_i(\pi', \pi_{-i,t}) - U_i(\pi_{i,t}, \pi_{-i,t})\right) \leq o(1)$$

Taking limits: $$\max_{\pi' \in \Delta_i} \mathbb{E}_{\sigma_T}[U_i(\pi', \pi_{-i,t})] \leq \mathbb{E}_{\sigma_T}[U_i(\pi_t)]$$ Definition of CCE!

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Definition:* *Coarse Correlated Equilibrium [Aumann 1974]

A joint distribution $\sigma \in \Delta_{\mathcal{A}}$ is a coarse correlated equilibrium (CCE) if, for all i:

$$\mathbb{E}_{\pi \sim \sigma}[U_i(\pi)] \geq \mathbb{E}_{\pi_{-i} \sim \sigma}[U_i(\pi'_i, \pi_{-i})] \quad \forall \pi'_i \in \Pi_i$$

CCE have several desirable properties:

- Always exist (generalization of Nash).
- Set of CCE is convex.
- CCE can be found in normal-form games via linear programming or no-regret learning.

No-Regret and Coarse Correlated Eq.

No-regret algorithms have convergence guarantees to another form of game theoretic equilibrium:

Definition: Coarse Correlated Equilibrium [Aumann 1974]

A joint distribution $\sigma \in \Delta_{\mathcal{A}}$ is a coarse correlated equilibrium (CCE) if, for all i:

$$\mathbb{E}_{\pi\sim\sigma}[U_i(\pi)] \geq \mathbb{E}_{\pi_{-i}\sim\sigma}[U_i(\pi'_i, \pi_{-i})] \quad \forall \pi'_i \in \Pi_i$$

CCE have several desirable properties:

- Always exist (generalization of Nash).
- Set of CCE is convex.
- CCE can be found in normal-form games via linear programming or no-regret learning.

...and some less desirable ones:

- Can have support on dominated strategies [Viossat & Zapechelnyuk (2013)] - no rational agent would implement!
- Can be no-regret but never converge
- Requires coordination to implement.

Dynamics of no-regret algorithms

Consider a very simple instantiation of a no-regret algorithm in rock-paper scissors:

2-players

$$R_1 = R_2 = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

Players optimize over softmax policies using gradient descent:

$$\pi_{i,t} = \text{softmax}(w_{i,t}, \beta_i)$$

$$w_{i,t+1} = w_{i,t} + \eta R_i \pi_{-i,t}$$

Dynamics of no-regret algorithms

Consider a very simple instantiation of a no-regret algorithm in rock-paper scissors:

2-players

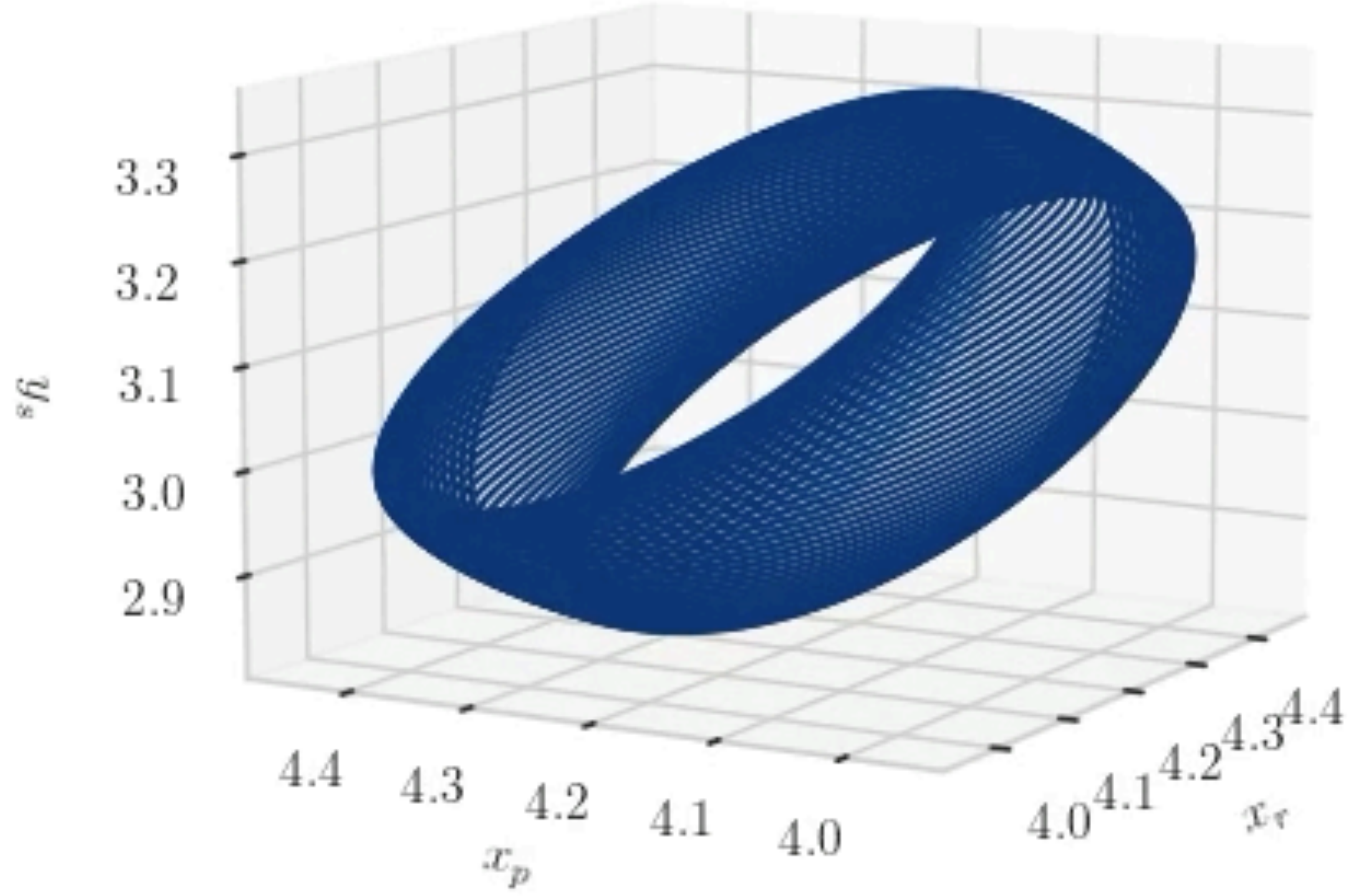
$$R_1 = R_2 = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

Players optimize over softmax policies using gradient descent:

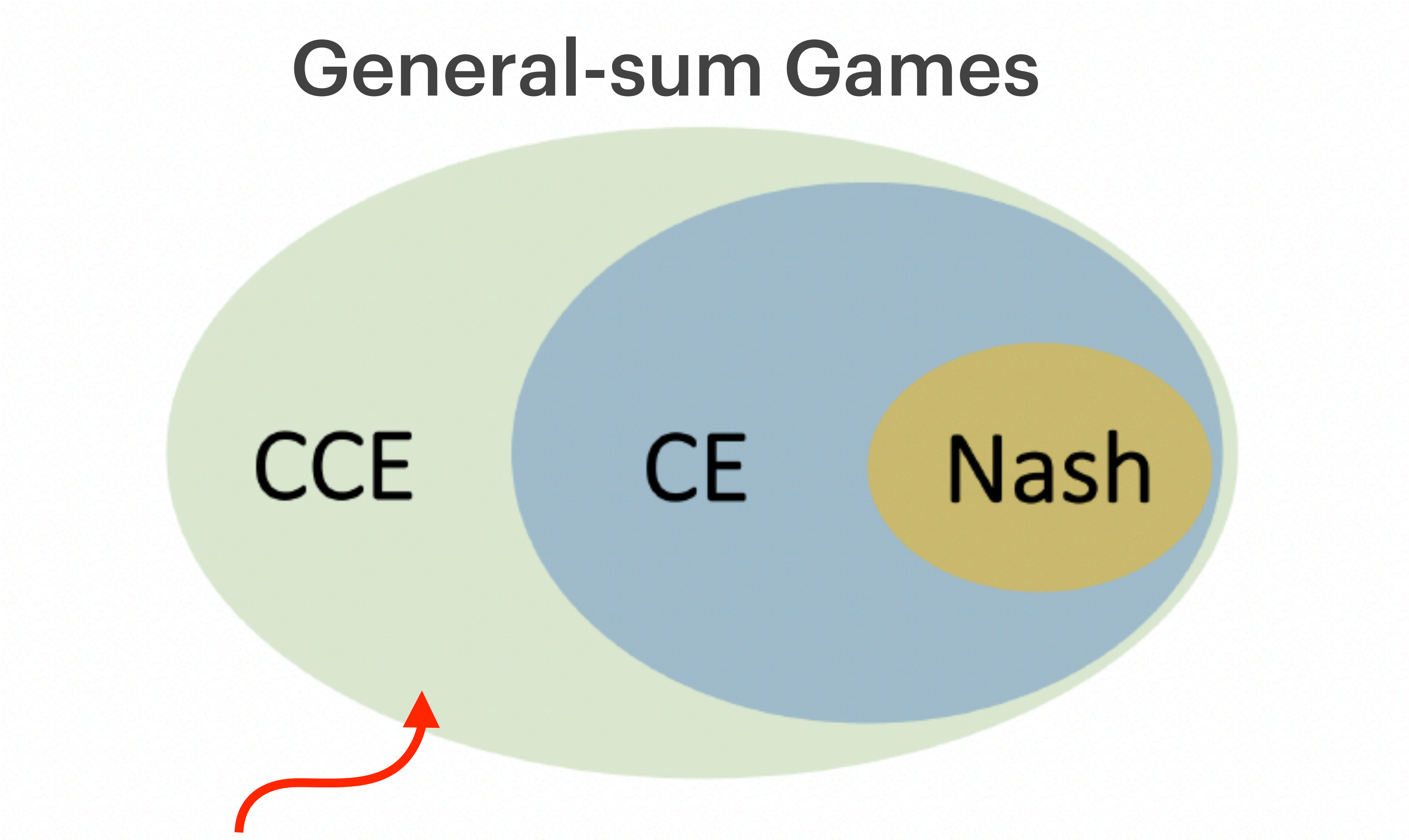
$$\pi_{i,t} = \text{softmax}(w_{i,t}, \beta_i)$$

$$w_{i,t+1} = w_{i,t} + \eta R_i \pi_{-i,t}$$

Players using gradient-play in Rock-Paper Scissors exhibit chaos

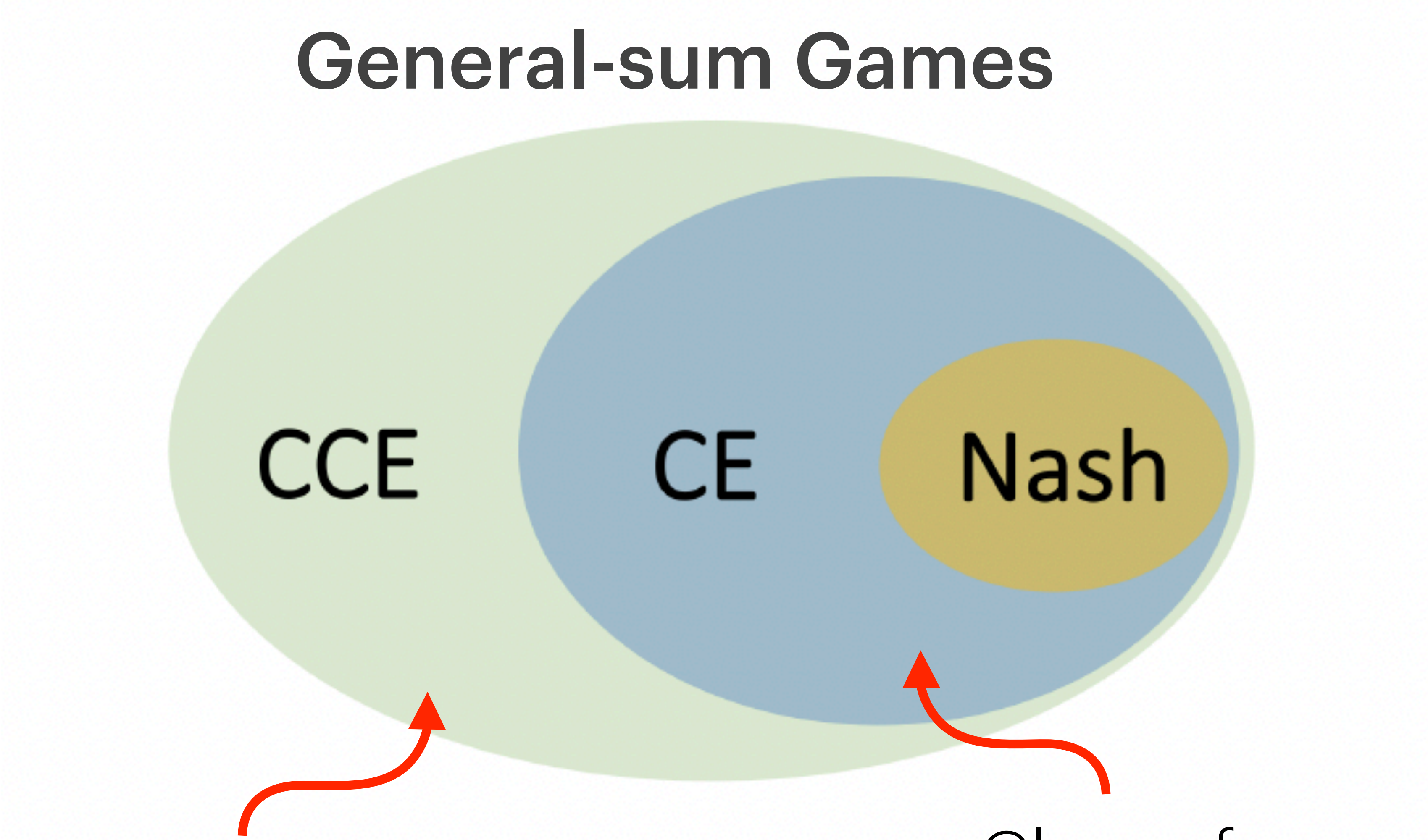


Recap: No-regret learning algorithms



Class of CCE
Can be computed via
no-regret learning

Recap: No-regret learning algorithms

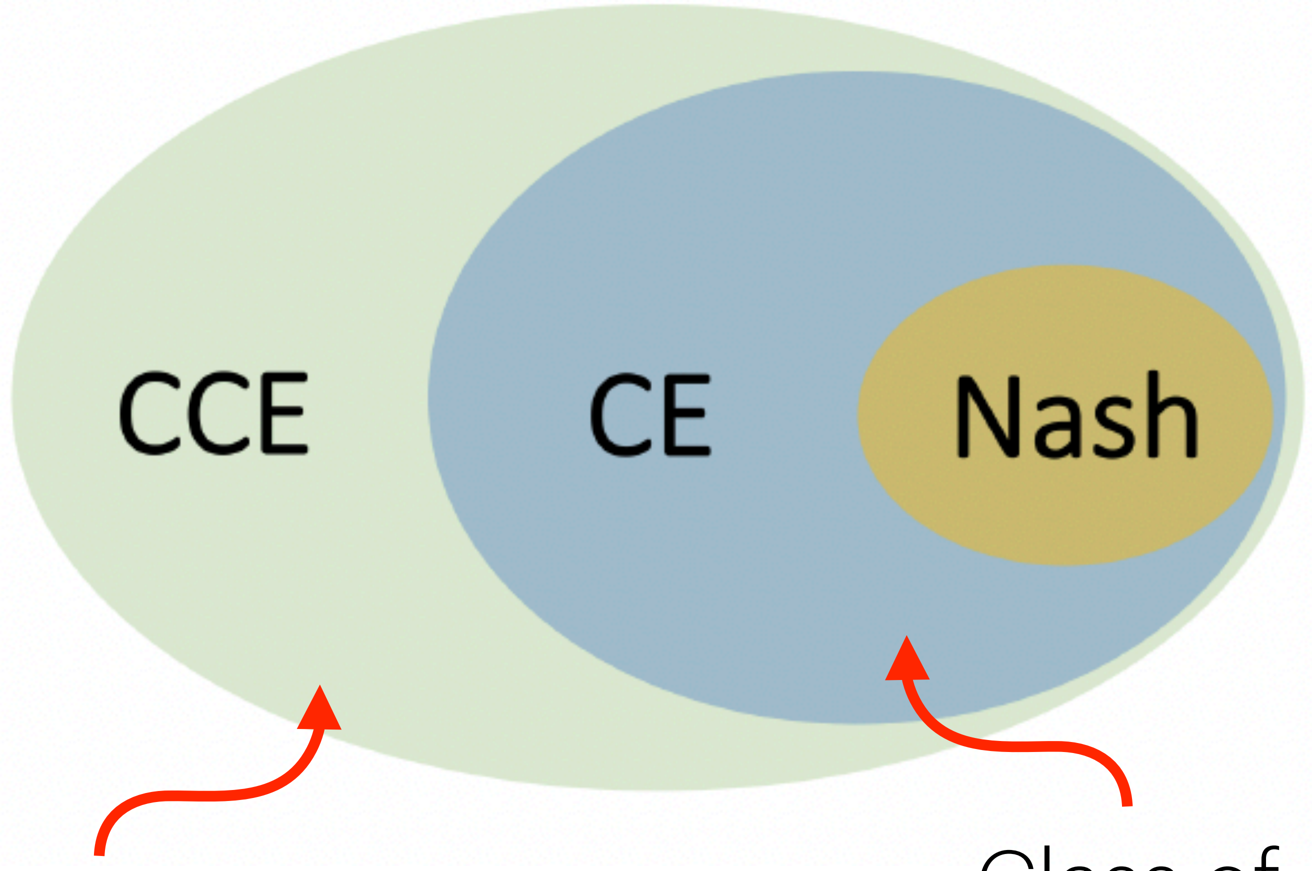


Class of CCE
Can be computed via
no-regret learning

Class of correlated equilibria
(inherit many properties of CCE)
can be computed via ***no-swap-regret***
algorithms

Recap: No-regret learning algorithms

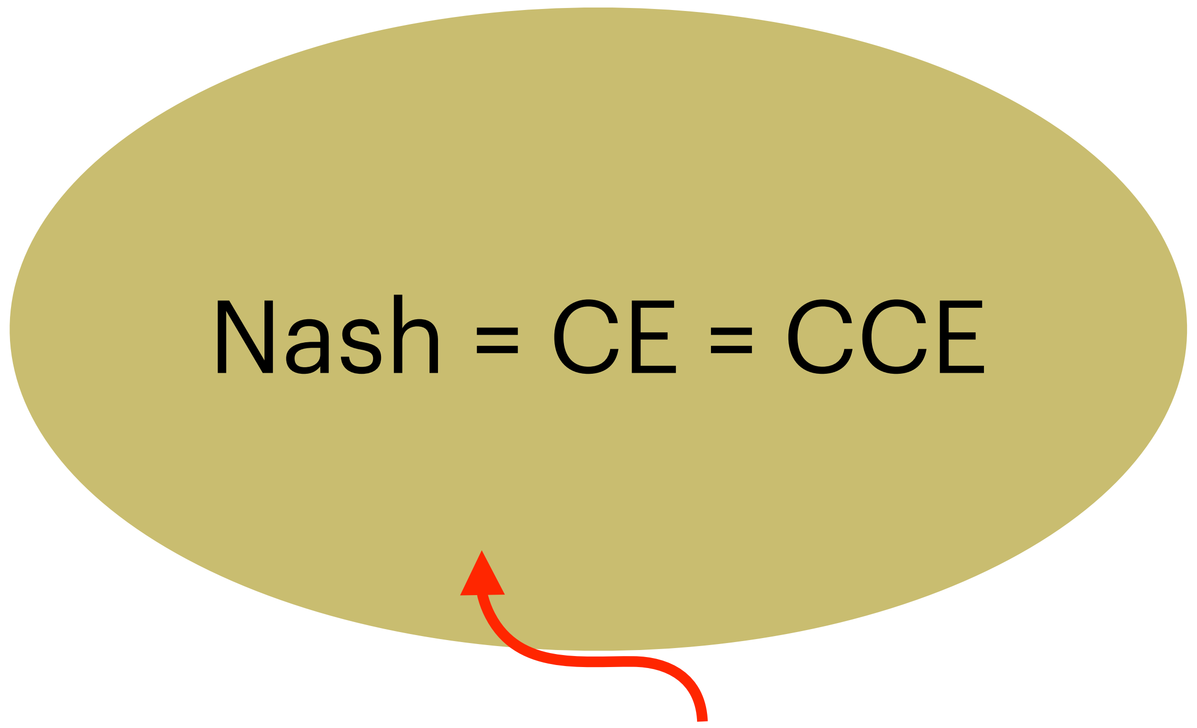
General-sum Games



Class of CCE
Can be computed via
no-regret learning

Class of correlated equilibria
(inherit many properties of CCE)
can be computed via ***no-swap-regret***
algorithms

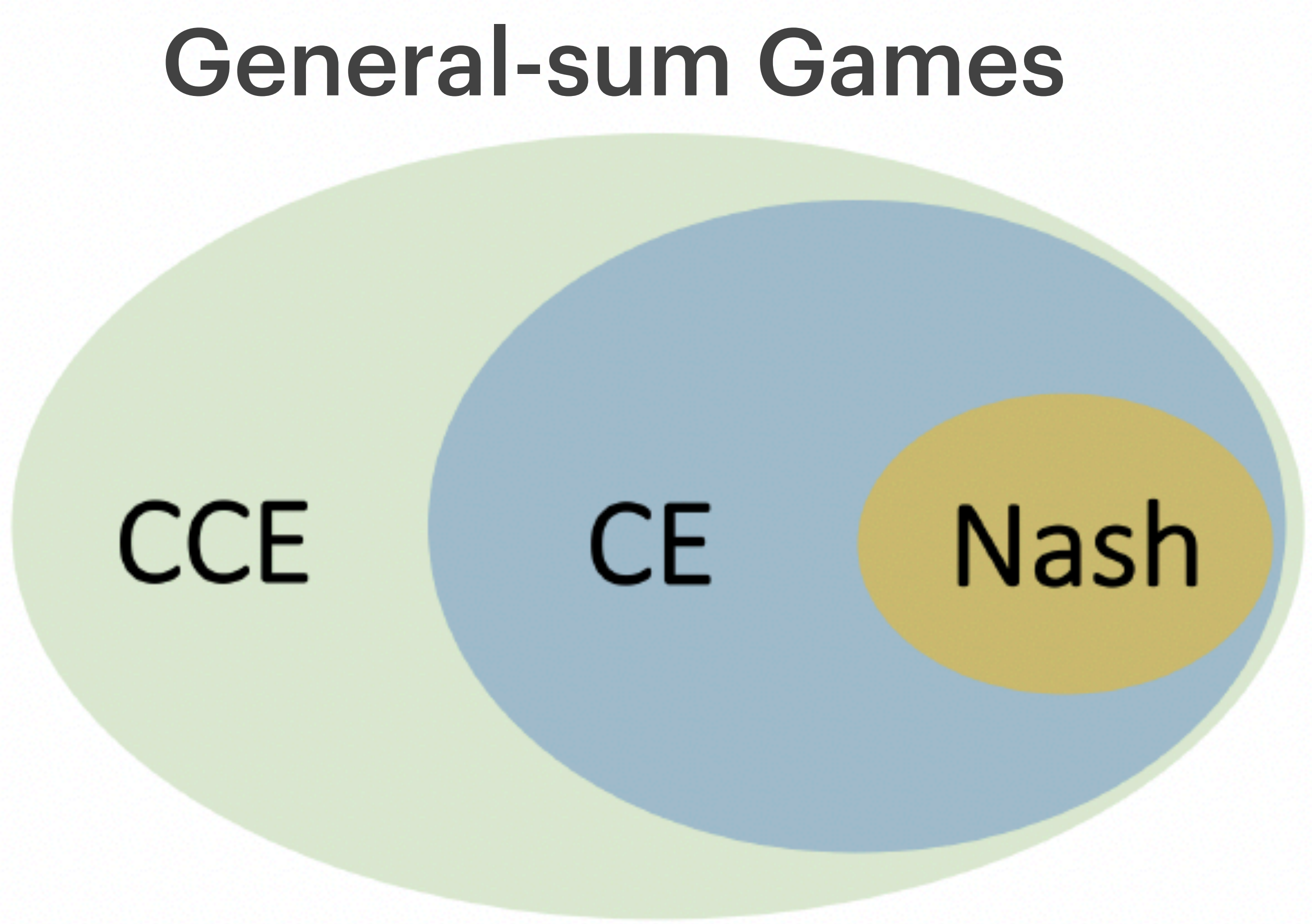
Zero-sum games

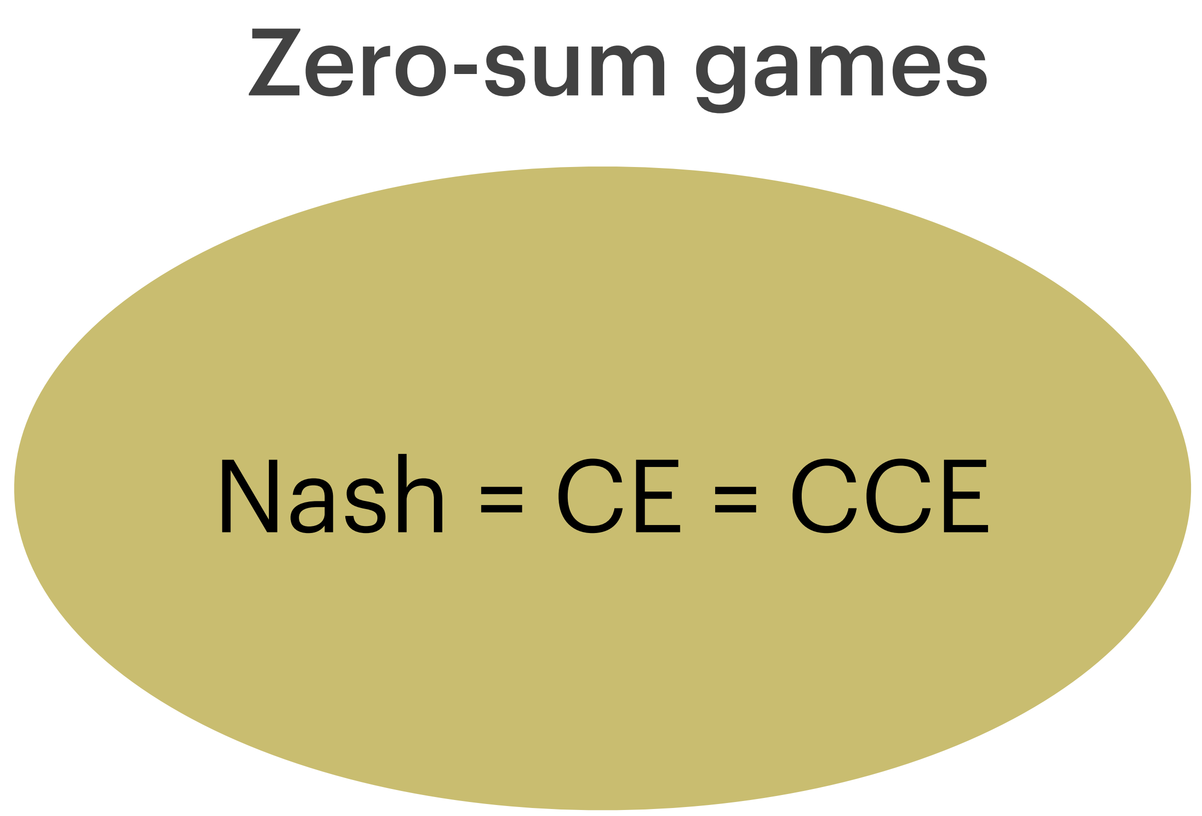


"Equilibrium Collapse"
No-regret algorithms find Nash (min-max) eq. in zero-sum matrix games.

(Similar phenomenon occurs in other "strictly" competitive games e.g., network zero-sum games)

Recap: No-regret learning algorithms





Designing and analyzing no-regret learning algorithms is still an active research area:

Optimal rates of convergence
[Daskalakis et al. 2021, Cai & Zheng 2023, Farina et al. 2023]

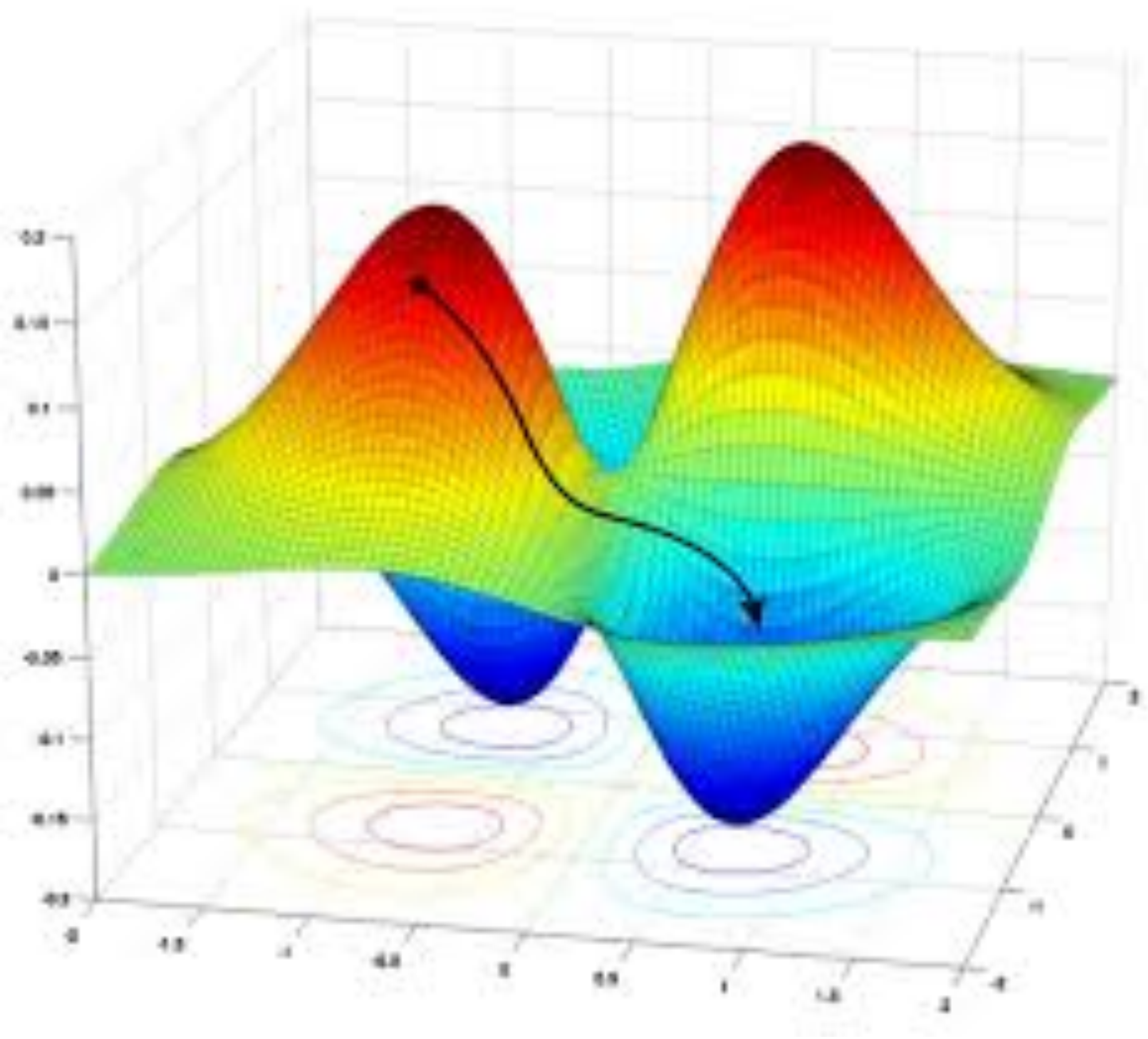
Swap-regret
[Arunachaleswaran, et al. 2025, Fishelson et al. 2025]

Characterizing subset CCE that are computed by no-regret algorithms
[Anagnostides et al. 2022]

Time-varying games, pure exploration, ... many different variations of the problem...

Equilibrium computation through the lens of optimization

Another approach to the problem of equilibrium computation is through the lens of optimization.



Equilibrium computation through the lens of optimization

Viewed through this lens, computing/learning a Nash equilibrium is equivalent to solving a ***variational inequality problem.***

Definition: Variational definition of Nash equilibrium of a concave game

A joint policy π^* is a Nash equilibrium if

$$\langle F(\pi^*), \pi^* - \pi \rangle \geq 0 \quad \forall \pi \in \prod_{i=1}^{n} \Delta_{\mathcal{A}_i} \qquad F(\pi) = \begin{bmatrix} \nabla_1 U_1(\pi_1, \pi_{-1}) \\ \vdots \\ \nabla_n U_n(\pi_n, \pi_{-n}) \end{bmatrix}$$

This is a simple generalization of the max of a concave function over a compact set.

$$\langle \nabla f(x^*), x^* - x \rangle \geq 0 \;\; \forall x \in \mathcal{X} \qquad \text{concave } f$$

Equilibrium computation through the lens of optimization

Viewed through this lens, computing/learning a Nash equilibrium is equivalent to solving a ***variational inequality problem.***

Definition: Variational definition of Nash equilibrium of a concave game

A joint policy π^* is a Nash equilibrium if

$$\langle F(\pi^*), \pi^* - \pi \rangle \geq 0 \quad \forall \pi \in \prod_{i=1}^{n} \Delta_{\mathcal{A}_i} \qquad F(\pi) = \begin{bmatrix} \nabla_1 U_1(\pi_1, \pi_{-1}) \\ \vdots \\ \nabla_n U_n(\pi_n, \pi_{-n}) \end{bmatrix}$$

If $F(\pi)$ has useful structure, e.g., monotonicity, then we can apply tools from the literature on solving VIPs to compute/learn Nash.

Monotone Variational Inequalities

(Projected) Gradient-Play

- Each agent initializes policy at random.
- For step t=0,1,2,...
    - Play $\pi_{i,t}$
    - Observe $\nabla_i U_i(\pi_t)$
    - Update policy $\pi_{i,t+1} = \mathcal{P}_{\Pi_i}\left(\pi_{i,t} + \eta \nabla_i U_i(\pi_t)\right)$

Equivalently, analyze joint dynamics:

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$$

Joint dynamics are not gradient descent on a function!

Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $F(\pi)$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Convergence to Nash in a ***last iterate sense.***

$$\|\pi_t - \pi^*\|^2 \leq (1 - \eta\alpha)^t \|\pi_0 - \pi^*\|^2$$

Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Convergence to Nash in a ***last iterate sense***

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and $\boldsymbol{\Pi}$ is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

Convergence to Nash in an ***ergodic sense***

Convergence in Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $F(\pi)$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Thm:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

F is monotone in all zero-sum normal form and convex-concave games.

Monotone F is a restriction on concave games.

Convergence in Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $F(\pi)$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Thm:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

F is monotone in all zero-sum normal form and convex-concave games

$$\min_{\pi_1} \max_{\pi_2} \; \pi_1^T R \pi_2 \implies F(\pi) = A\pi \quad \text{where: } A = \begin{bmatrix} 0 & R \\ -R^T & 0 \end{bmatrix} = -A^T$$

Convergence in Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $F(\pi)$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Thm:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

F is monotone in all zero-sum normal form and convex-concave games

$$\min_{\pi_1} \max_{\pi_2} \; \pi_1^T R \pi_2 \implies F(\pi) = A\pi \quad \text{where: } A = \begin{bmatrix} 0 & R \\ -R^T & 0 \end{bmatrix} = -A^T$$

$$\implies \langle F(\pi) - F(\pi'), \pi - \pi' \rangle = 0 \quad \forall \pi, \pi'$$

Convergence in Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

F is monotone in all zero-sum normal form and convex-concave games

Simple decentralized gradient ascent-descent computes Nash in these games!

Convergence in Monotone Variational Inequalities

Joint dynamics of gradient-play

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$$

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., strongly monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq -\alpha \|\pi - \pi'\|^2 \quad \forall \pi, \pi'$$

Then $\pi_t \to \pi^*$ under projected gradient-play.

Thm:

If $\boldsymbol{F(\pi)}$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and $\boldsymbol{\Pi}$ is convex and compact, then under projected gradient-play:

$$\frac{1}{T}\sum_{t=0}^{T} \pi_t \to \pi^*$$

F is monotone in all zero-sum normal form and convex-concave games

Simple decentralized gradient ascent-descent computes Nash in these games!

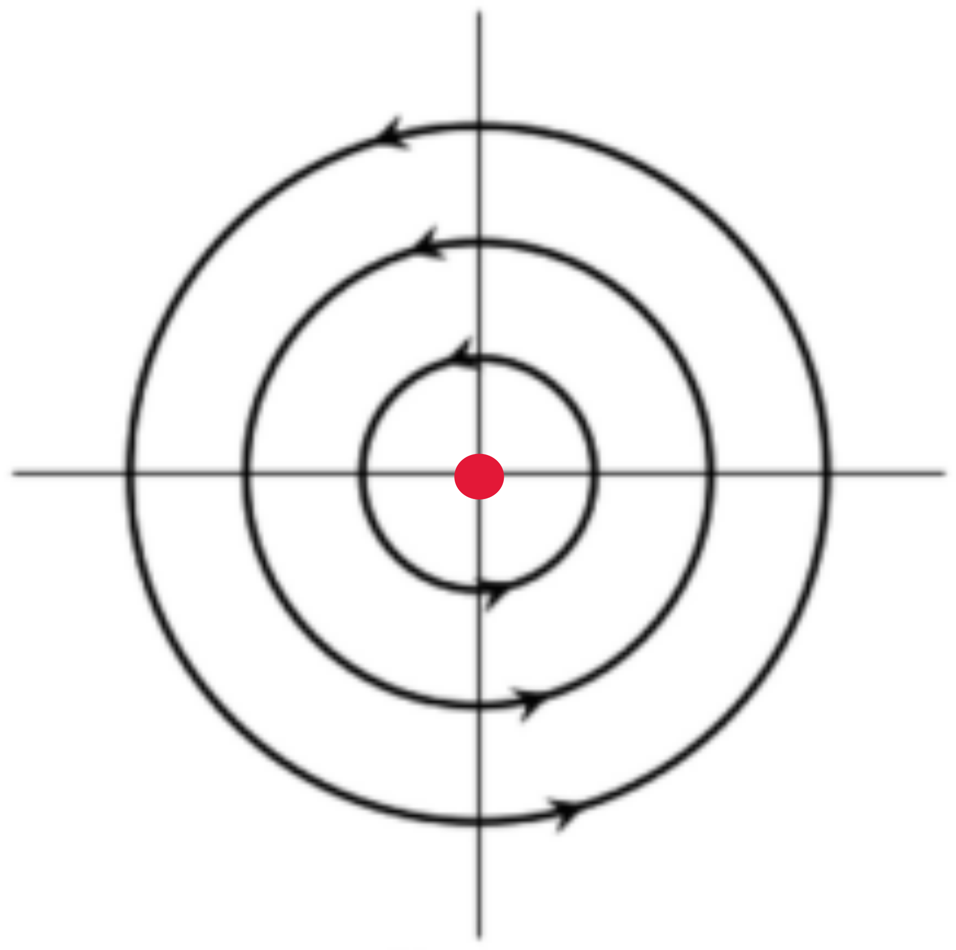
But we can do even better!

Gradient-play in Monotone Variational Inequalities

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

$$\dot{\pi} = F(\pi)$$

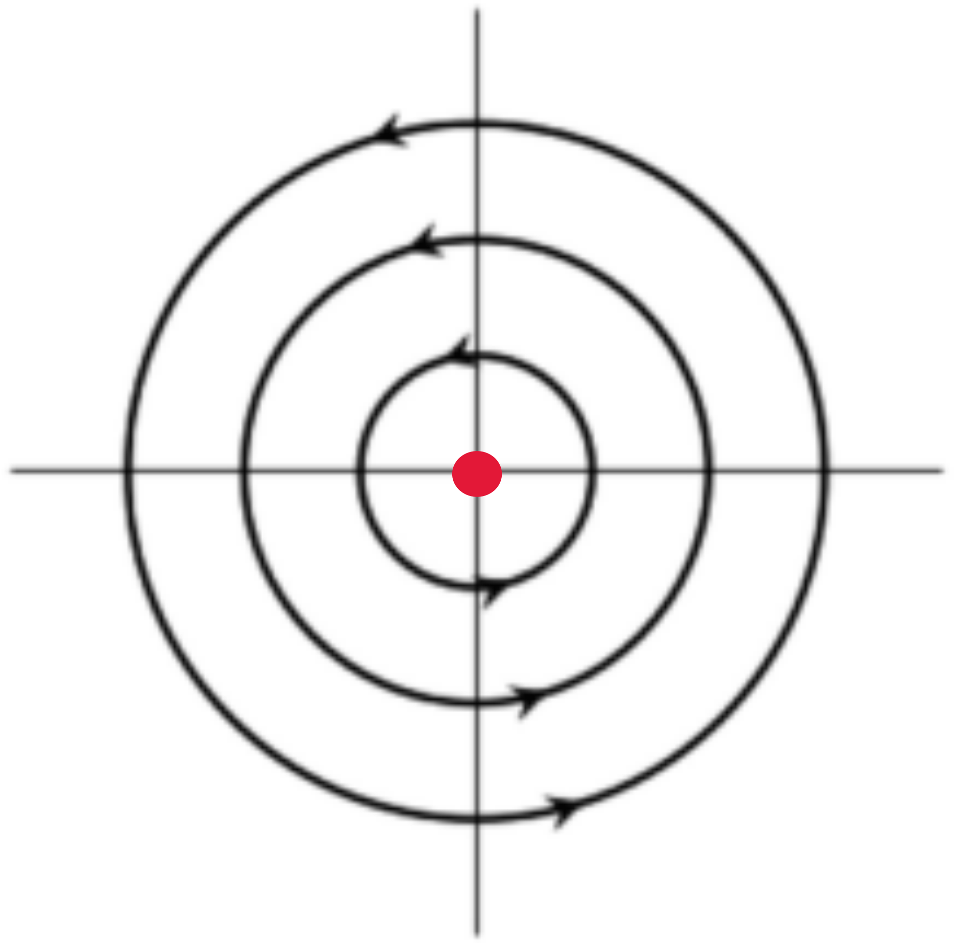


Gradient-play in Monotone Variational Inequalities

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

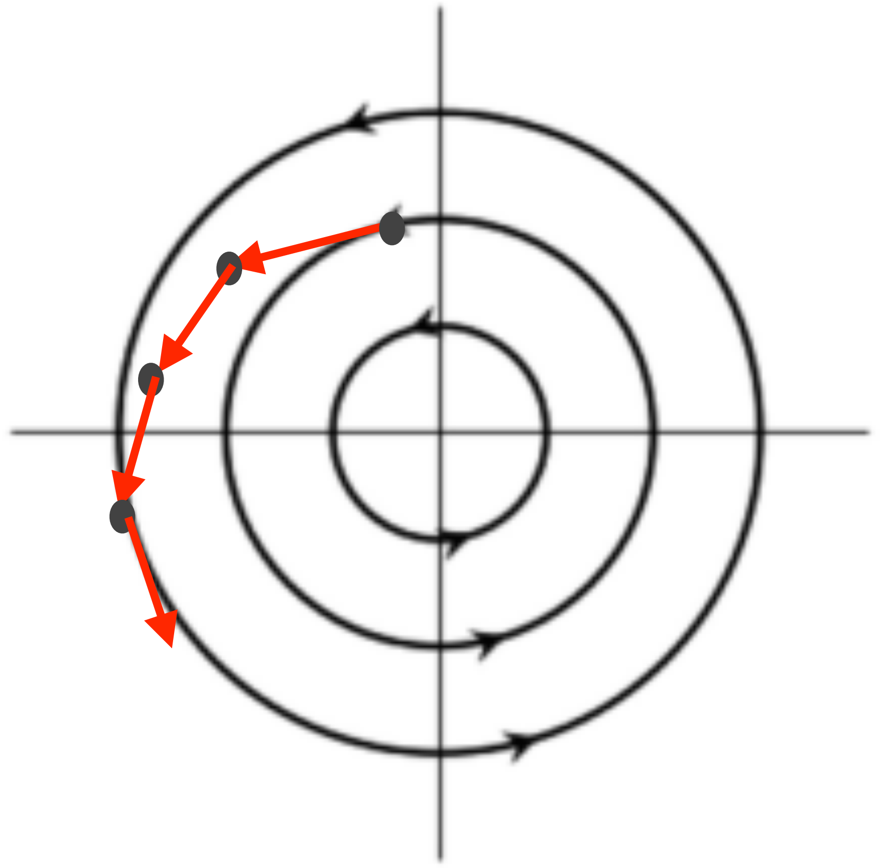
$$\dot{\pi} = F(\pi)$$



Gradient-play

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$$

Gradient-play is the forward Euler discretization of the limiting continuous-time dynamics

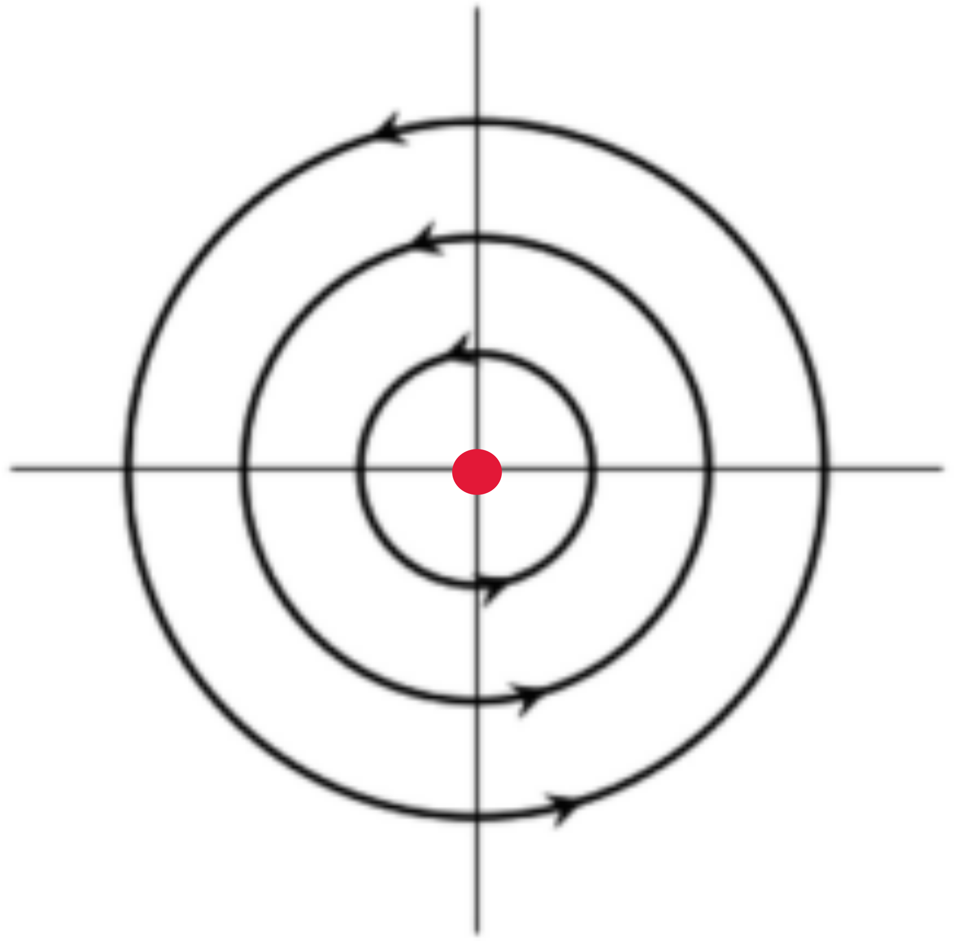


Gradient-play in Monotone Variational Inequalities

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

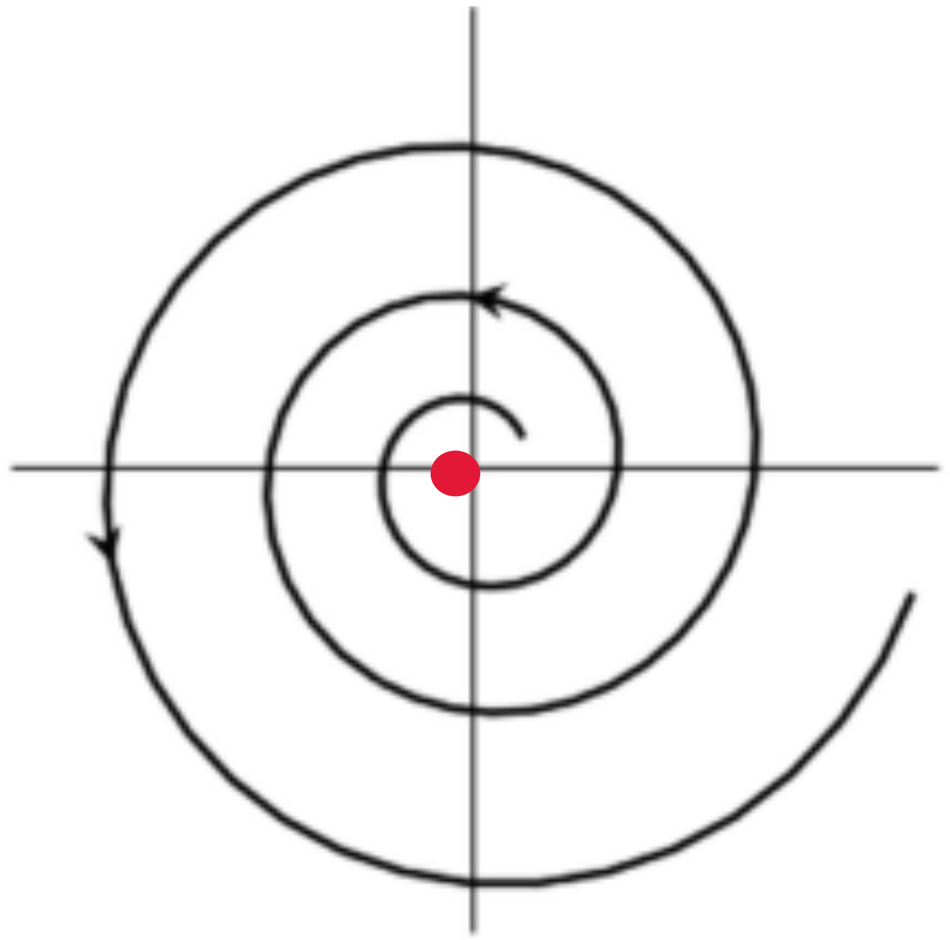
$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

$$\dot{\pi} = F(\pi)$$



Gradient-play

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$$



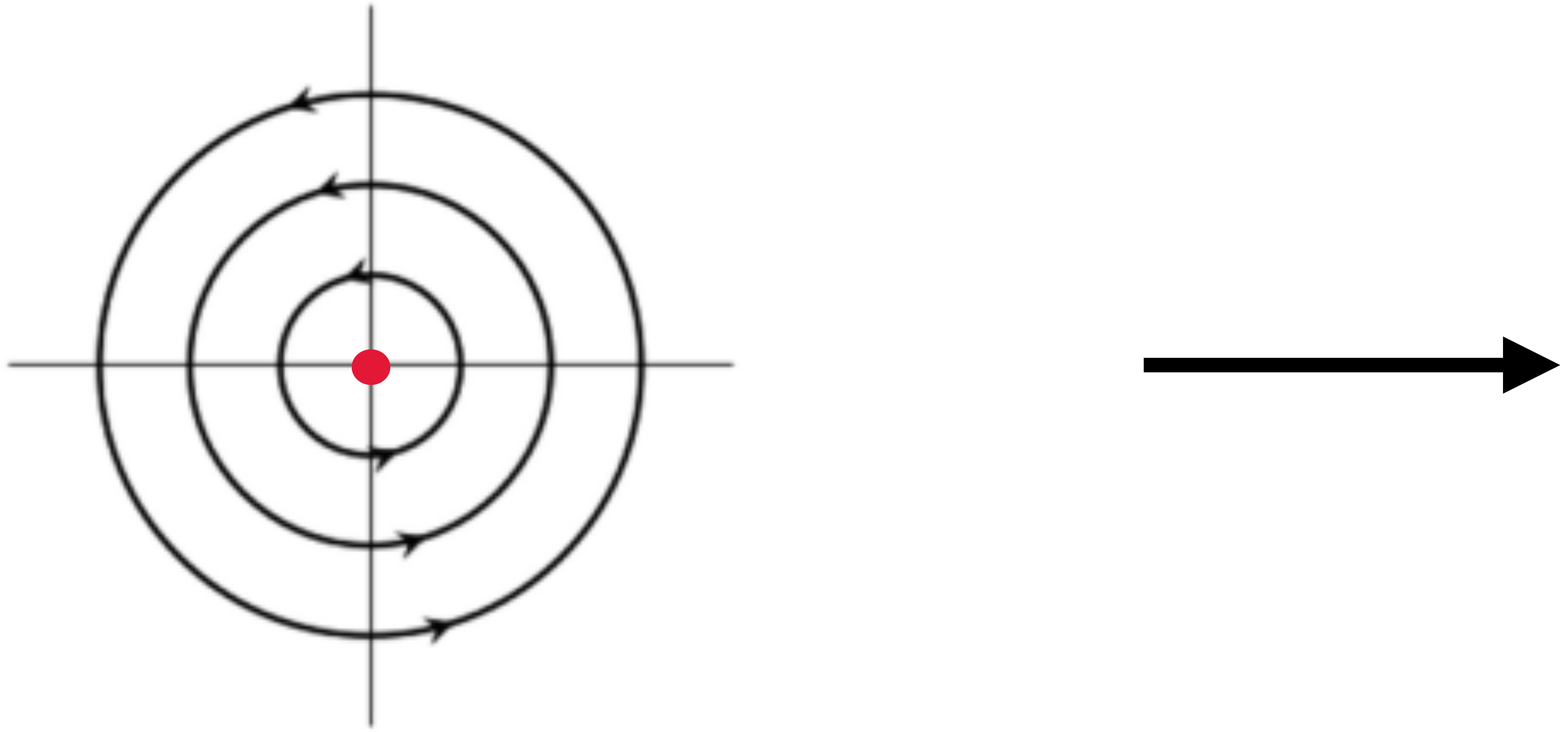
Bad discretization causes divergence!
(compact convex set allows for ergodic convergence)

Gradient-play in Monotone Variational Inequalities

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

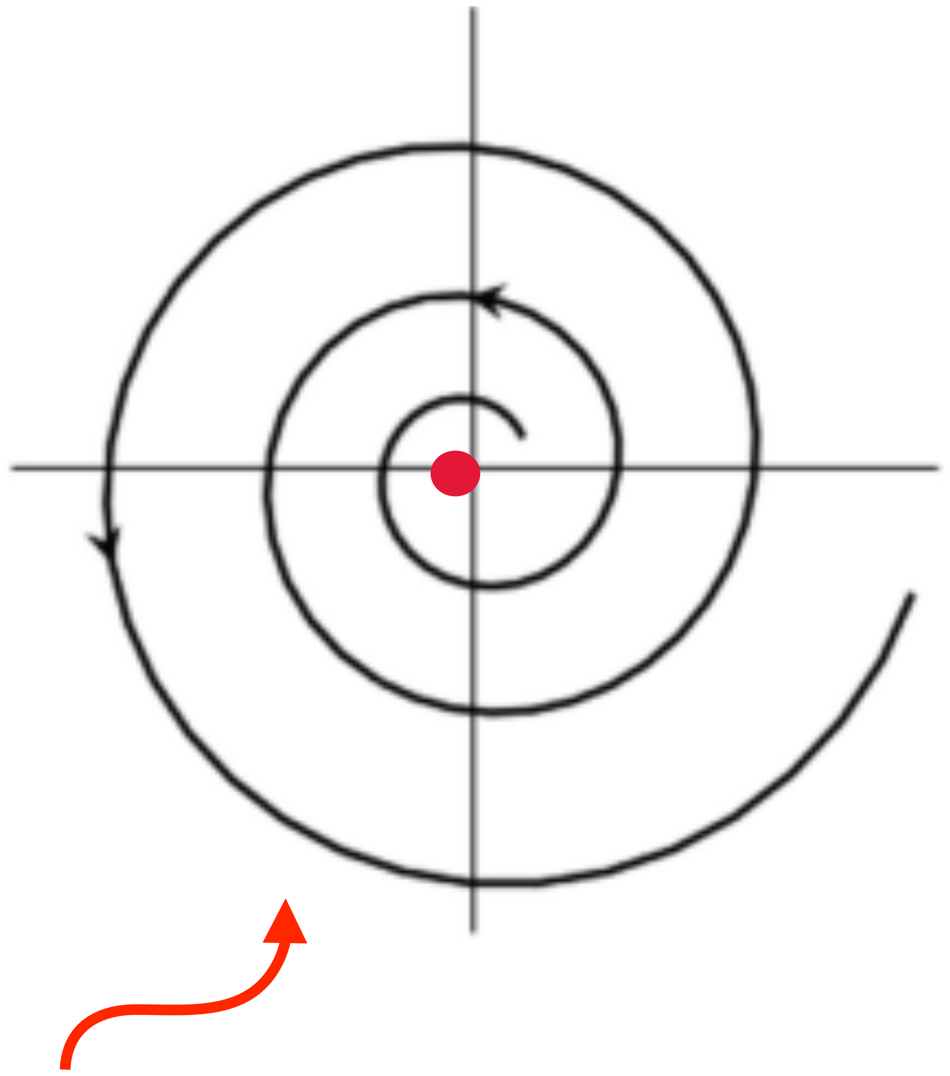
$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

$$\dot{\pi} = F(\pi)$$



Gradient-play

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$$



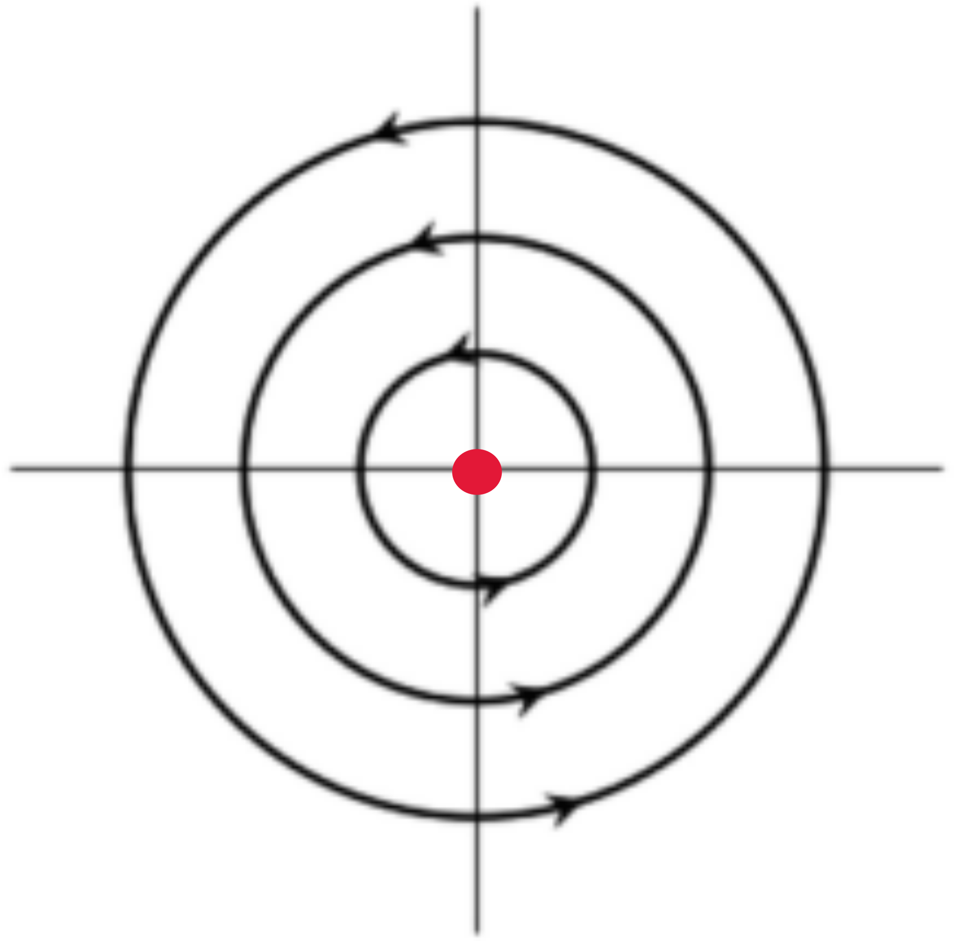
Can fix this behavior by doing an implicit discretization!

Last-iterate convergence via the Proximal Point Algorithm

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

$$\dot{\pi} = F(\pi)$$

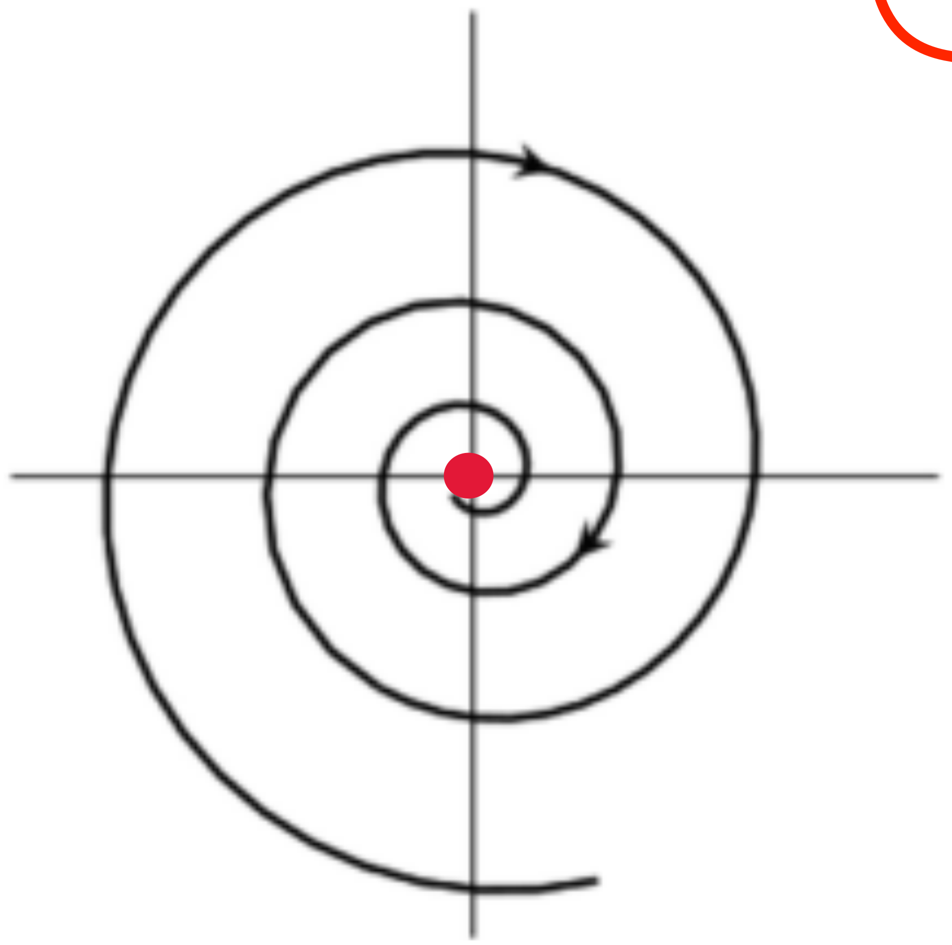


Proximal-Point Algorithm

[Martinet 1970], [Rockefellar 1976]

$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_{t+1})\right)$$

Implicit definition



Last-iterate convergence via the Proximal Point Algorithm

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

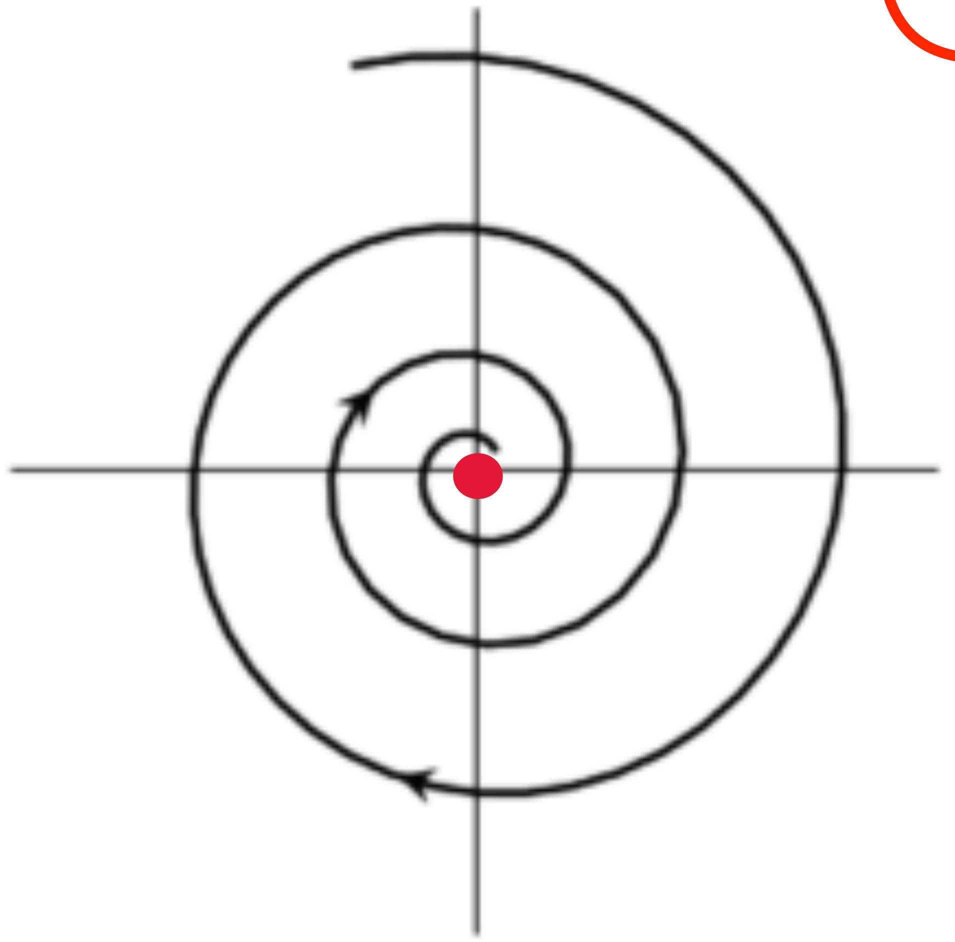
$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

Proximal-Point Algorithm

[Martinet 1970], [Rockefellar 1976]

$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_{t+1})\right)$$

Implicit definition



Thm:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then the proximal point algorithm guarantees:

$$\pi_t \to \pi^*$$

In convex-concave zero-sum games:

$$\max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2} U(\pi_{1,t}, \pi_2) = O\left(\frac{1}{\sqrt{t}}\right)$$

Convergence to Nash in an ***last-iterate sense!***

(This is the “Nash-Gap”, a measure of distance from Nash)

Last-iterate convergence via the Proximal Point Algorithm

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

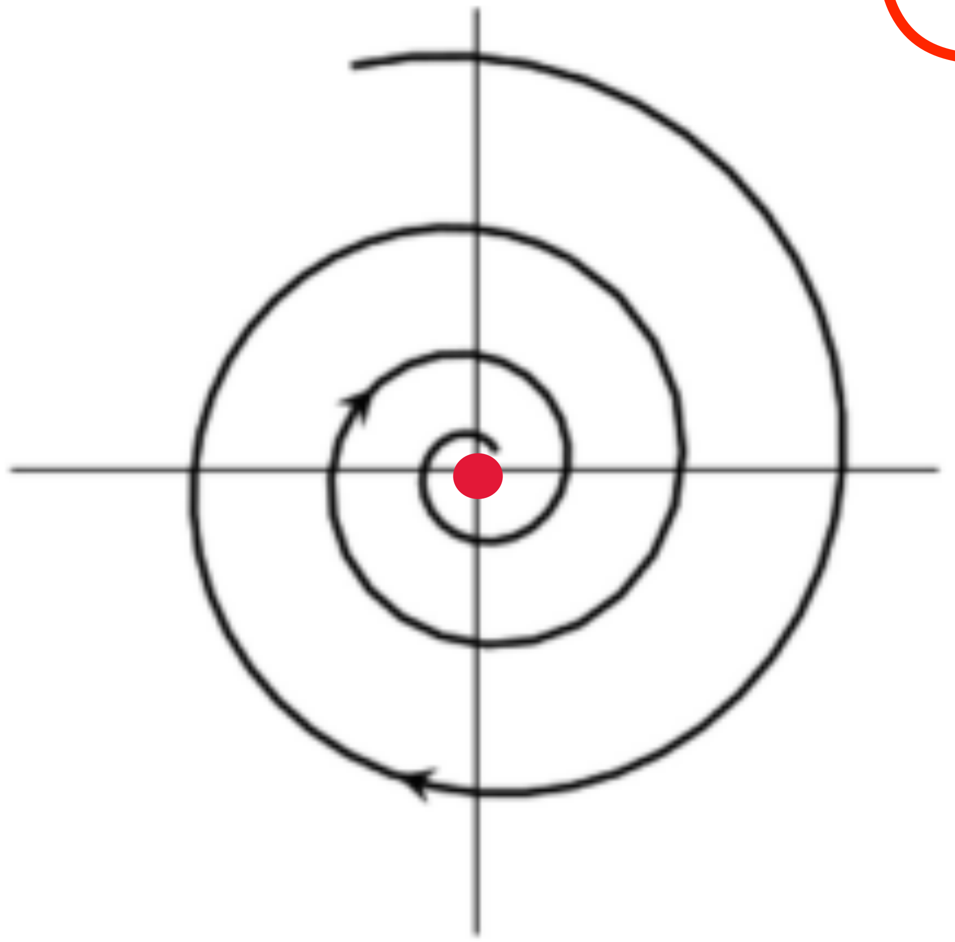
$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

Proximal-Point Algorithm

[Martinet 1970], [Rockefellar 1976]

$$\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_{t+1})\right)$$

Implicit definition



Not an implementable algorithm!

Thm:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then the proximal point algorithm guarantees:

$$\pi_t \rightarrow \pi^*$$

In convex-concave zero-sum games:

$$\max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2} U(\pi_{1,t}, \pi_2) = O\left(\frac{1}{\sqrt{t}}\right)$$

Implementable Algorithms for Last-Iterate Convergence

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

Extra-gradient
[Korpelevich 1976]

$$\nu_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\pi_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_{t+1})\right)$$

Optimistic Gradient
[Popov 1980]

$$\nu_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_{t+1})\right)$$

Implementable Algorithms for Last-Iterate Convergence

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

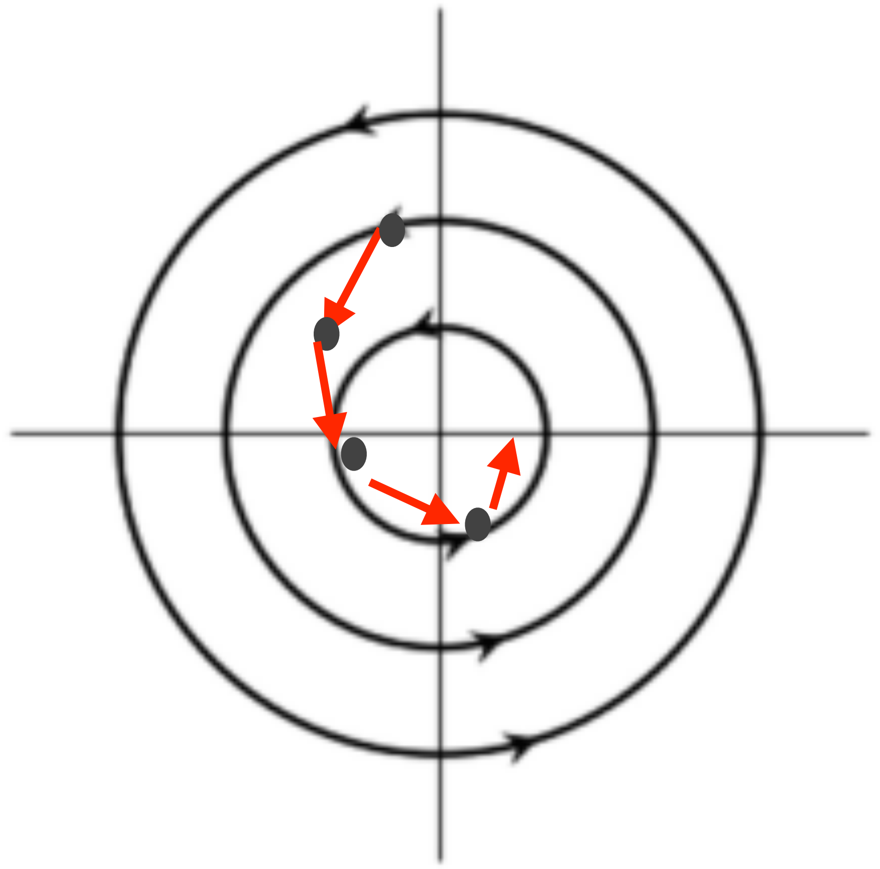
Extra-gradient
[Korpelevich 1976]

$$\nu_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_{t+1})\right)$$

Optimistic Gradient
[Popov 1980]

$$\nu_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_{t+1})\right)$$

Both of these algorithms can be viewed as approximations to the proximal point method [Mokhtari et al. 2019]



Both algorithms are examples of ***independent learning:***

- players do not need any information about their opponents' objective to implement.

Implementable Algorithms for Last-Iterate Convergence

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

Extra-gradient

[Korpelevich 1976]

$$\nu_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\pi_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_{t+1})\right)$$

Optimistic Gradient

[Popov 1980]

$$\nu_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi \left(\pi_t + \eta F(\nu_{t+1})\right)$$

Thm [Gorbonuv et al. 2022, Cai et al 2022]:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then the optimistic gradient and extra-gradient algorithms guarantee that:

$$\pi_t \to \pi^*$$

In convex-concave zero-sum games:

$$\max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2} U(\pi_{1,t}, \pi_2) = O\left(\frac{1}{\sqrt{t}}\right)$$

Implementable Algorithms for Last-Iterate Convergence

Consider the continuous-time dynamics of gradient play in zero-sum Matrix games

$$F(\pi) = A\pi \quad \text{where: } A = -A^T$$

Extra-gradient
[Korpelevich 1976]

$$\nu_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_{t+1})\right)$$

Optimistic Gradient
[Popov 1980]

$$\nu_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_t)\right)$$
$$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\nu_{t+1})\right)$$

Thm [Gorbonuv et al. 2022, Cai et al 2022]:

If $F(\pi)$ is e.g., monotone:

$$\langle F(\pi) - F(\pi'), \pi - \pi' \rangle \leq 0 \quad \forall \pi, \pi'$$

and Π is convex and compact, then the optimistic gradient and extra-gradient algorithms guarantee that:

$$\pi_t \to \pi^*$$

In convex-concave zero-sum games:

$$\max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2} U(\pi_{1,t}, \pi_2) = O\left(\frac{1}{\sqrt{t}}\right)$$

Interestingly, the optimistic gradient algorithm is no-regret but the extra-gradient algorithm is not!

Recap: Equilibrium Computation as Optimization

Equilibrium computation can be cast as solving variational inequality problems.

- In sub-classes of games (e.g., zero-sum games, potential games) this viewpoint allows us to leverage ***new classes of algorithms with strong guarantees of convergence***

  Extra-gradient and Optimistic gradient methods are:

  1. independent learning algorithms
  2. Have last-iterate convergence to Nash in monotone variational inequalities

Recap: Equilibrium Computation as Optimization

Equilibrium computation can be cast as solving variational inequality problems.

- In sub-classes of games (e.g., zero-sum games, potential games) this viewpoint allows us to leverage ***new classes of algorithms with strong guarantees of convergence***

  Extra-gradient and Optimistic gradient methods are:

  1. independent learning algorithms
  2. Have last-iterate convergence to Nash in monotone variational inequalities

- This is a very active area of research:
  - Beyond monotone variational inequalities
    [Cai et al 2022], [Gorbonov et al. 2023], [Alacaoglu et al. 2025],…
  - Accelerated convergence
    [Huang et al. 2021], [Cai et al. 2022],…
  - Convergence in stochastic-gradient regime
    [Gorbonov et al. 2022], [Beznosikov et al. 2023], [Chen & Mazumdar et al. 2024], [Zhang et al. 2025],…

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**
   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

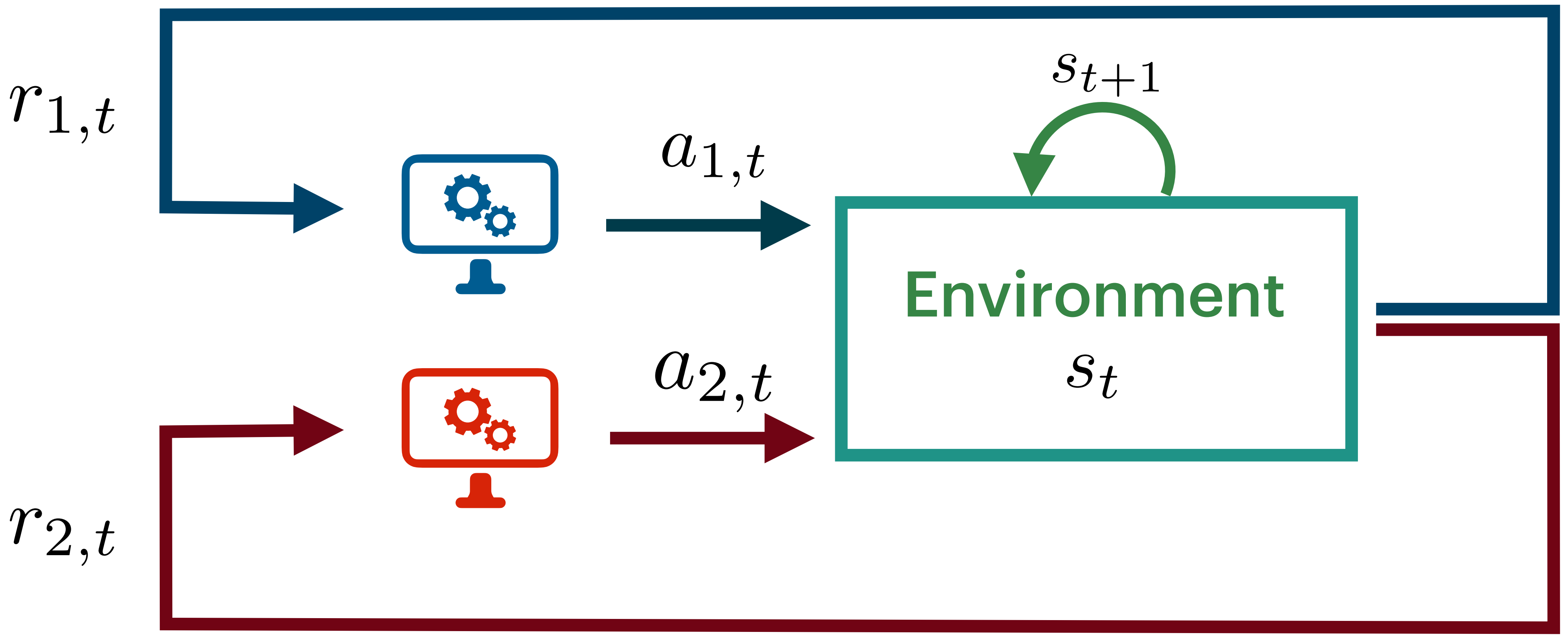
1. **Normal-form & concave games: equilibrium computation and learning in games**

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

Recall: 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: $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
- Horizon: H or ∞
- Initial state distribution: ρ_0



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

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
   (Including optimistic gradient methods)

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

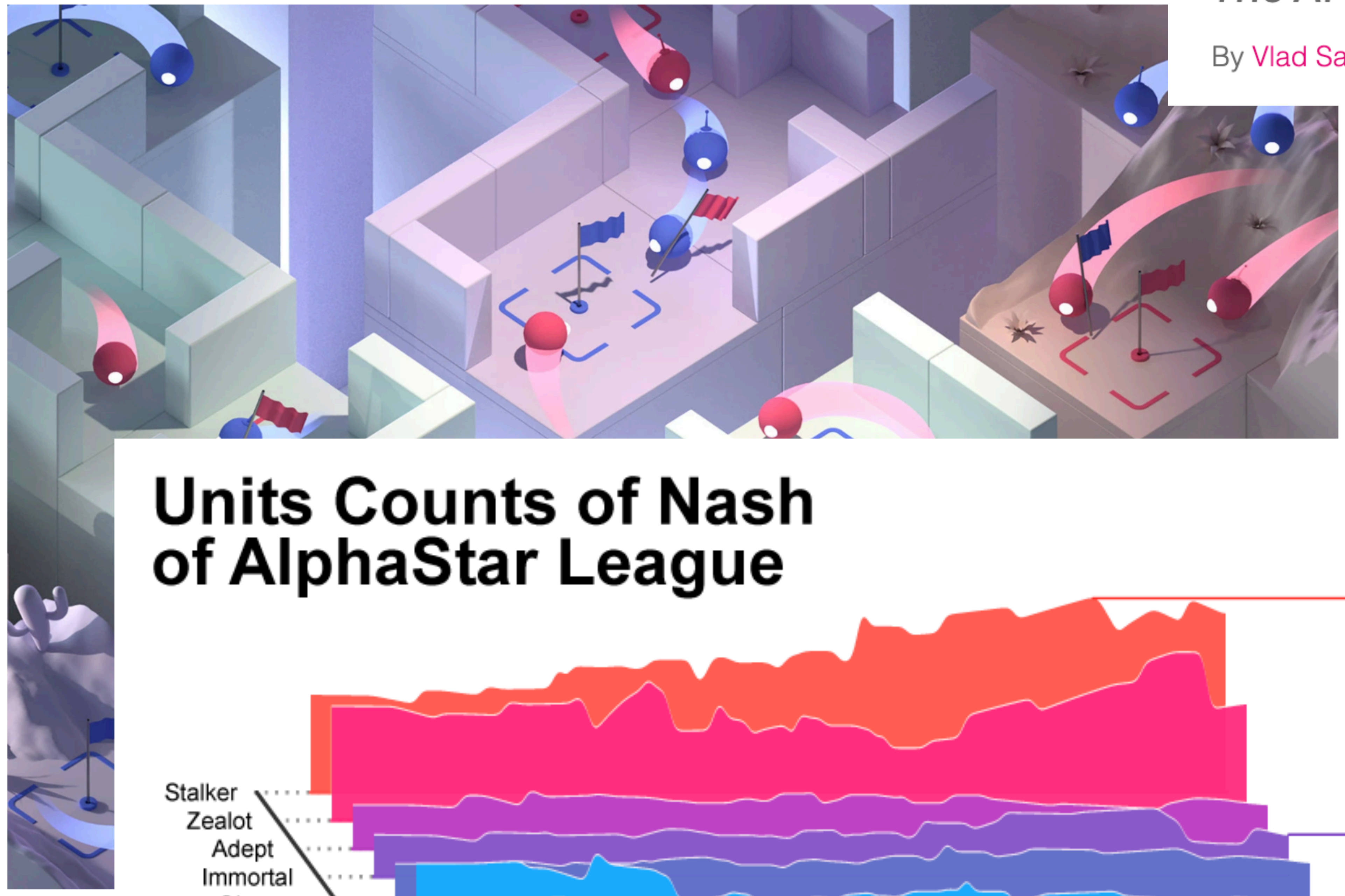
- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
   (Including optimistic gradient methods)

2. *No-regret Learning in Markov Games*

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?



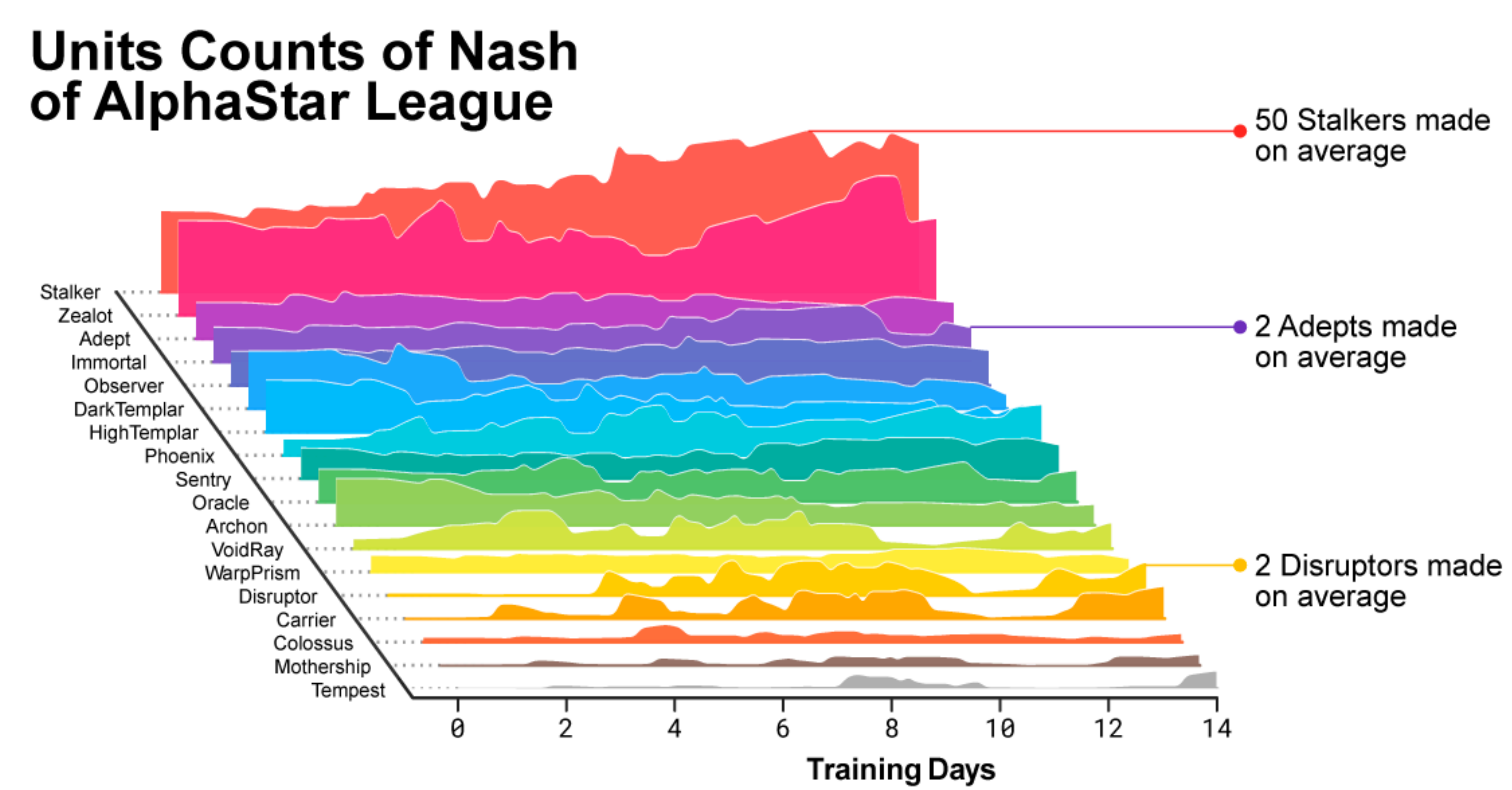
DeepMin

GAMING ENTERTAINMENT TECH

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

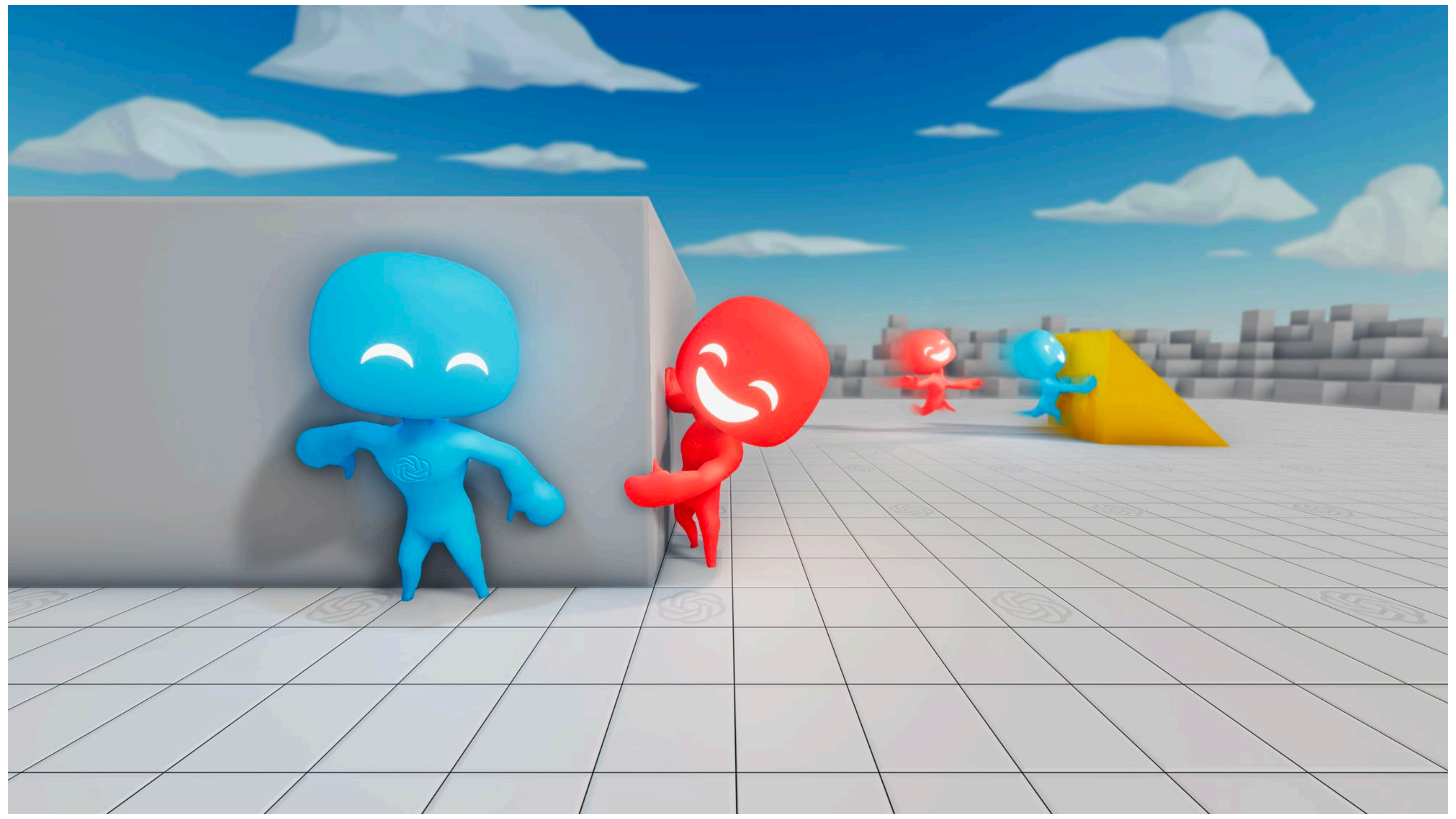
The AI won 7,215 matches against humans, losing only 42 in the process

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



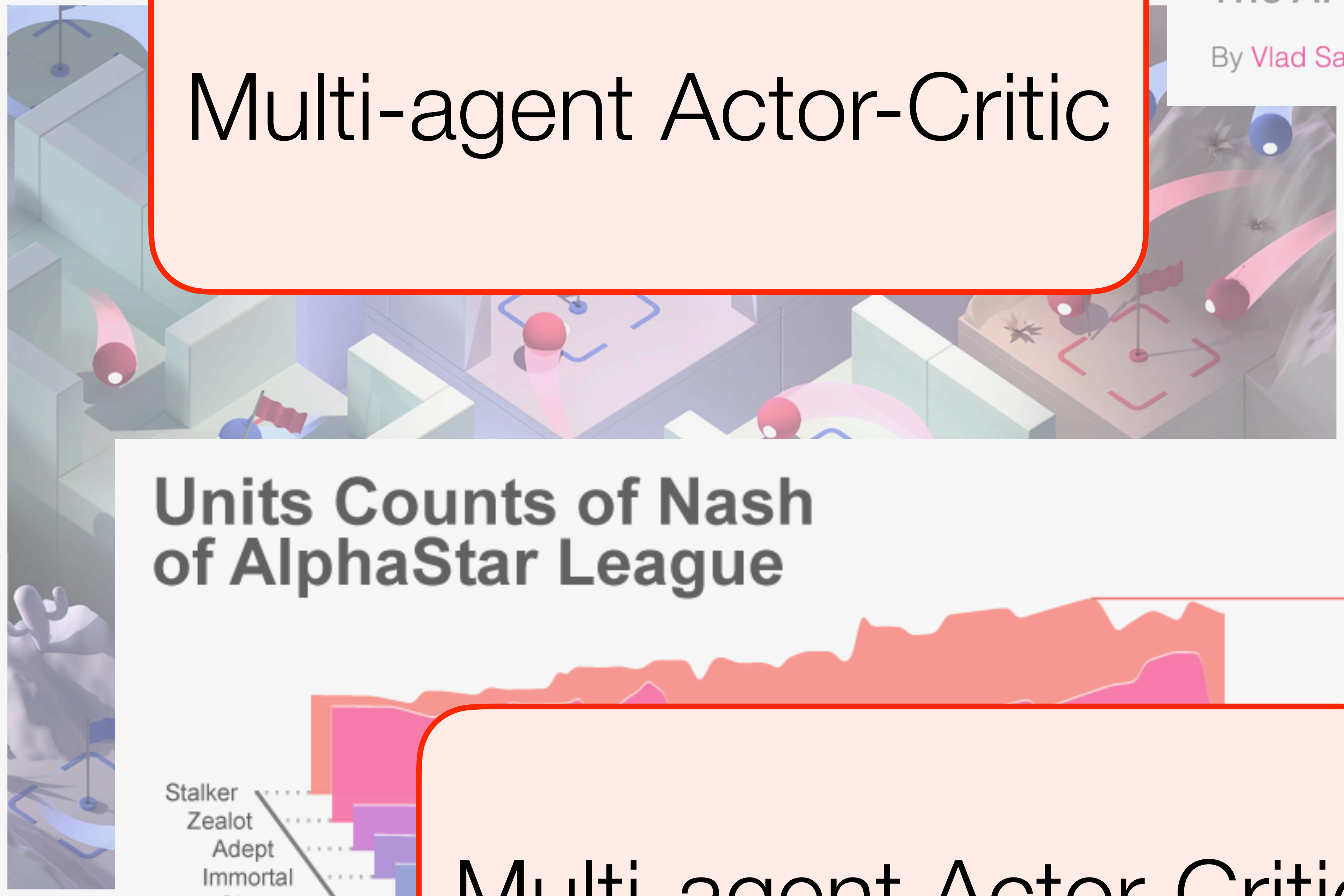
JUNE 8, 2017 • 5 MINUTE READ

Learning to Cooperate, Compete, and Communicate

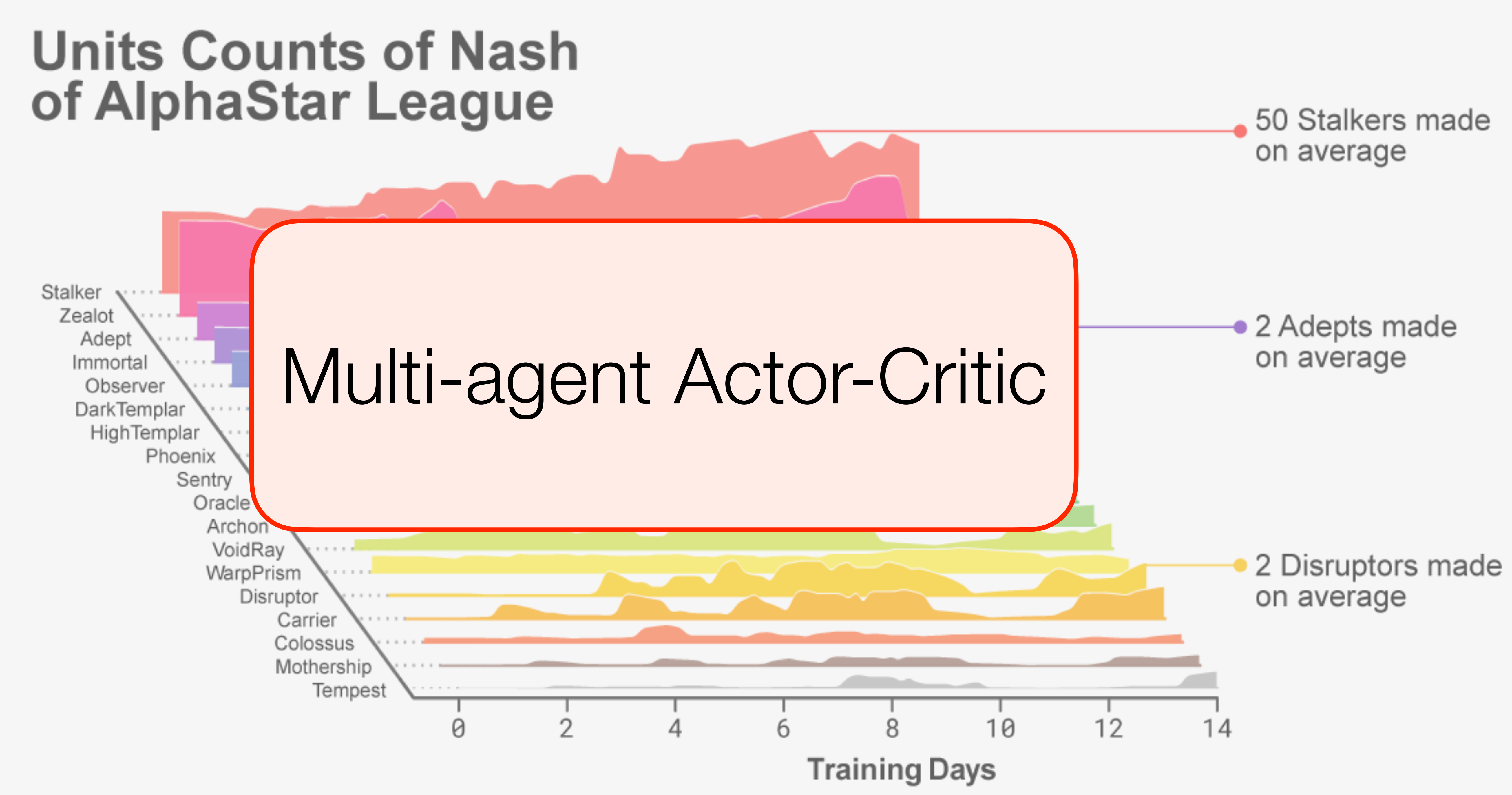


DeepMind Can Now Beat Us at Multiplayer Games, Too

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



DeepMind



GAMING ENTERTAINMENT TECH

Feeble humans … OpenAI's Dota 2 gods

Multi-agent PPO
(Proximal Policy Optimization)

The AI won 7,215 matches against…

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

JUNE 8, 2017 • 5 MINUTE READ

Learning to Cooperate, Compete, and Communicate

Multi-agent PPO
(Proximal Policy Optimization)

Multi-agent Actor-Critic

All of these are constructing estimates of the "policy gradient"

Multi-agent Actor-Critic

Multi-agent DDPG
(Deep Deterministic Policy Gradient)

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

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

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

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

Assumption: All players optimize over ***stationary Markov policies***

- Optimization problem reduces to ***single-agent RL problem*** for fixed $\boldsymbol{\pi_{-i}}$ with dynamics:

$$\hat{P}(s' \mid s, a_i) = \sum_{a_{-i} \in \mathcal{A}_{-i}} \pi_{-i}(a_{-i}|s) P(s' \mid s, a_i, a_{-i}) \qquad \hat{R}(s, a_i) = \sum_{a_{-i} \in \mathcal{A}_{-i}} \pi_{-i}(a_{-i}|s) R(s, a_i, a_{-i})$$

Not true without Assumption!

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

Assumption 1: All players optimize over ***stationary Markov policies***

- Agent i's ***policy gradient*** is simply their policy gradient in this single-agent problem. Treat opponents as part of env.

$$\nabla_{\pi(s)} U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s \sim d^\pi} \left[Q_i^\pi(s, \cdot) \right]$$

$$d^\pi = (1 - \gamma_i) \sum_{t=0}^{\infty} \gamma_i^t Pr(s_t = s | \pi)$$

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

Discounted state visitation distribution

Marginalized Q value for agent i

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

Assumption 1: All players optimize over ***stationary Markov policies***

- Agent i's ***policy gradient*** is simply their policy gradient in this single-agent problem. Treat opponents as part of env.

$$\nabla_{\pi(s)} U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s \sim d^\pi} \left[Q_i^\pi(s, \cdot) \right]$$

Many tricks for estimating this via samples:
Rollout policy + construct estimate

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

Assumption 1: All players optimize over ***stationary Markov policies***

- Agent i's ***policy gradient*** is simply their policy gradient in this single-agent problem. Treat opponents as part of env.

$$\nabla_{\pi(s)} U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s \sim d^\pi} \left[Q_i^\pi(s, \cdot) \right]$$

Many tricks for estimating this via samples:
Rollout policy + construct estimate

Gaurav will go into much more detail on policy gradient algorithms
for single agent RL tomorrow!

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

Assumption 1: All players optimize over ***stationary Markov policies***

- Agent i's ***policy gradient*** is simply their policy gradient in this single-agent problem. Treat opponents as part of env.

$$\nabla_{\pi(s)} U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s\sim d^\pi}\left[Q_i^\pi(s,\cdot)\right]$$

Independent-Policy Gradients

- Each agent initializes policy at random.
- For step t=0,1,2,...
  - Fix policy, collect rollouts, estimate $\nabla_i U_i(\pi_t)$
  - Update policy $\pi_{i,t+1} = \mathcal{P}_{\Pi_i}\left(\pi_{i,t} + \eta \nabla_i U_i(\pi_t)\right)$

For now assume full information/perfect estimation

Policy Gradients in Markov Games

Let's look at ***full information*** Policy Gradient Algorithms in ***Infinite*** horizon Markov games:

Assumption 1: All players optimize over ***stationary Markov policies***

- Agent i's ***policy gradient*** is simply their policy gradient in this single-agent problem. Treat opponents as part of env.

$$\nabla_{\pi(s)} U_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s \sim d^{\pi}} \left[Q_i^{\pi}(s, \cdot) \right]$$

Independent-Policy Gradients

- Each agent initializes policy at random.
- For step t=0,1,2,...
  - Observe $\nabla_i U_i(\pi_t)$
  - Update policy $\pi_{i,t+1} = \mathcal{P}_{\Pi_i} \left(\pi_{i,t} + \eta \nabla_i U_i(\pi_t) \right)$

What can we say about the convergence of this algorithm from the lens of optimization?

Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$

Definition: *Variational definition of Nash equilibrium of a Markov game*

A stationary Markov Nash equilibrium π^*, must satisfy:

$$\langle F(\pi^*), \pi^* - \pi \rangle \geq 0 \quad \forall \pi \in \prod_{i=1}^{n} \Pi_i \qquad F(\pi) = \begin{bmatrix} \nabla_1 U_1(\pi_1, \pi_{-1}) \\ \vdots \\ \nabla_n U_n(\pi_n, \pi_{-n}) \end{bmatrix}$$

Immediate conclusions:

- Since reinforcement learning is non-convex, $F(\pi)$ is non-monotone.

Previous results on the convergence of gradient-play, proximal point, optimistic gradients do not apply!

Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

Definition: *Variational definition of Nash equilibrium of a Markov game*

A stationary Markov Nash equilibrium π^*, must satisfy:

$$\langle F(\pi^*), \pi^* - \pi \rangle \geq 0 \quad \forall \pi \in \prod_{i=1}^{n} \Pi_i \qquad\qquad F(\pi) = \begin{bmatrix} \nabla_1 U_1(\pi_1, \pi_{-1}) \\ \vdots \\ \nabla_n U_n(\pi_n, \pi_{-n}) \end{bmatrix}$$

Immediate conclusions:

- Since reinforcement learning is non-convex, $\boldsymbol{F(\pi)}$ is non-monotone.
- All stationary points of the joint policy gradient dynamics are Nash.

No spurious fixed points!
If policy gradients converge, they converge to Nash

Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

Definition: Variational definition of Nash equilibrium of a Markov game

A stationary Markov Nash equilibrium π^*, must satisfy:

$$\langle F(\pi^*), \pi^* - \pi \rangle \geq 0 \quad \forall \pi \in \prod_{i=1}^{n} \Pi_i \qquad F(\pi) = \begin{bmatrix} \nabla_1 U_1(\pi_1, \pi_{-1}) \\ \vdots \\ \nabla_n U_n(\pi_n, \pi_{-n}) \end{bmatrix}$$

Immediate conclusions:

- Since reinforcement learning is non-convex, $F(\pi)$ is non-monotone.
- All stationary points of the joint policy gradient dynamics are Nash.

Impossible to give global convergence in general…

Non-convergence of Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

Proposition [Mazumdar et al. 2020]:

Policy Gradients have no —even local— guarantees of convergence in general-sum games.

Nash equilibria in general-sum Markov games can be ***strictly unstable for the continuous-time dynamics.***

- Policy gradient algorithms would almost surely avoid the Nash under a random initialization.

Non-convergence of Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

Proposition [Mazumdar et al. 2020]:

Policy Gradients have no —even local— guarantees of convergence in general-sum games.

Nash equilibria in general-sum Markov games can be ***strictly unstable for the continuous-time dynamics.***

- Policy gradient algorithms would almost surely avoid the Nash under a random initialization.

Proof Sketch:

We consider the continuous-time dynamics in a neighborhood of an interior Nash equilibrium and look at the linearization

$$\dot{\pi} = F(\pi)$$

Note: these are the limiting dynamics of proximal point, extra gradient, and optimistic gradient algorithms

Non-convergence of Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

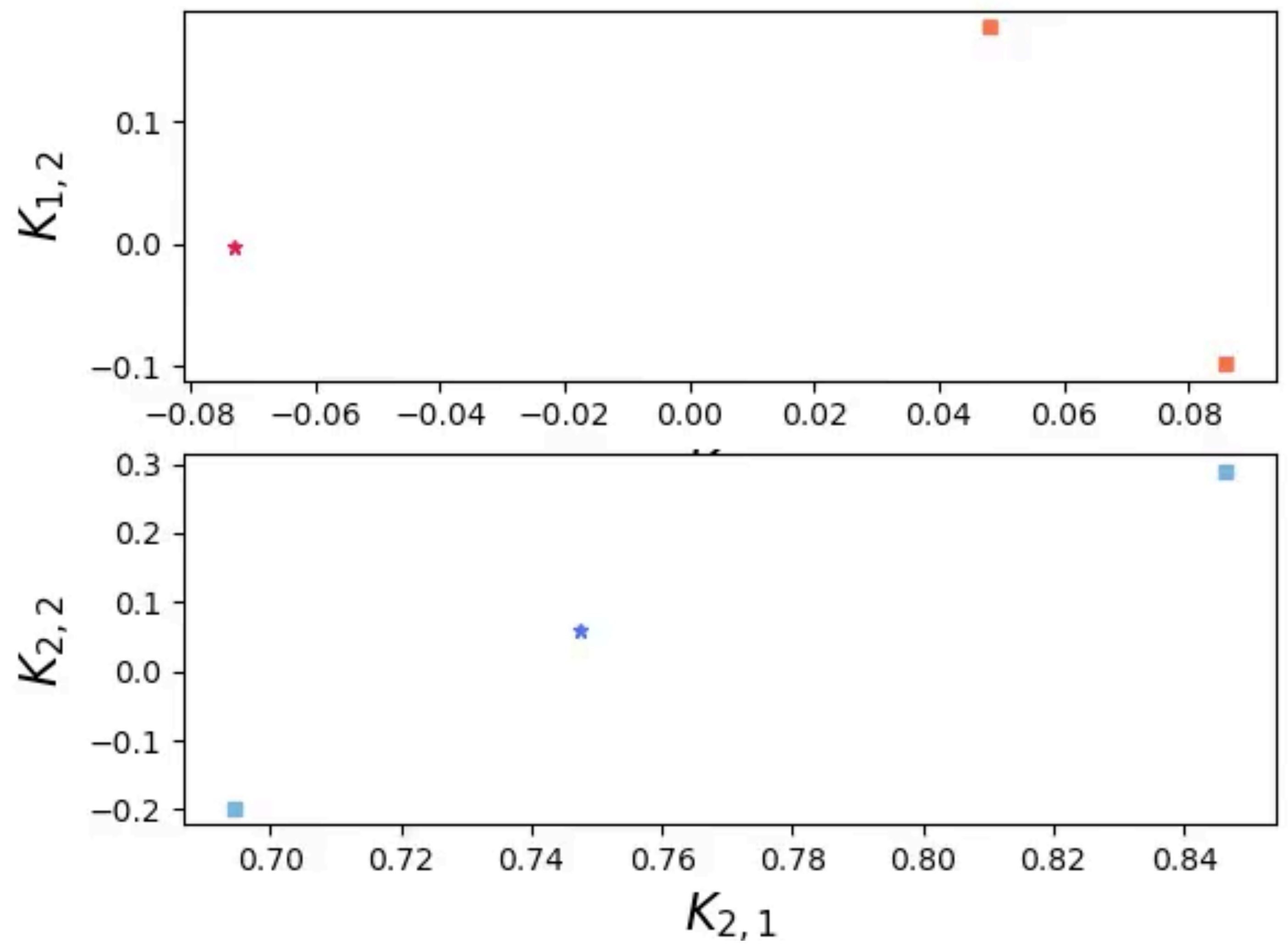
Proposition [Mazumdar et al. 2020]:

Policy Gradients have no —even local— guarantees of convergence in general-sum games.

Nash equilibria in general-sum Markov games can be ***strictly unstable for the continuous-time dynamics.***

- Policy gradient algorithms would almost surely avoid the Nash under a random initialization.

Players diverge from Nash and converge to limit cycles



Non-convergence of Policy Gradients in Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

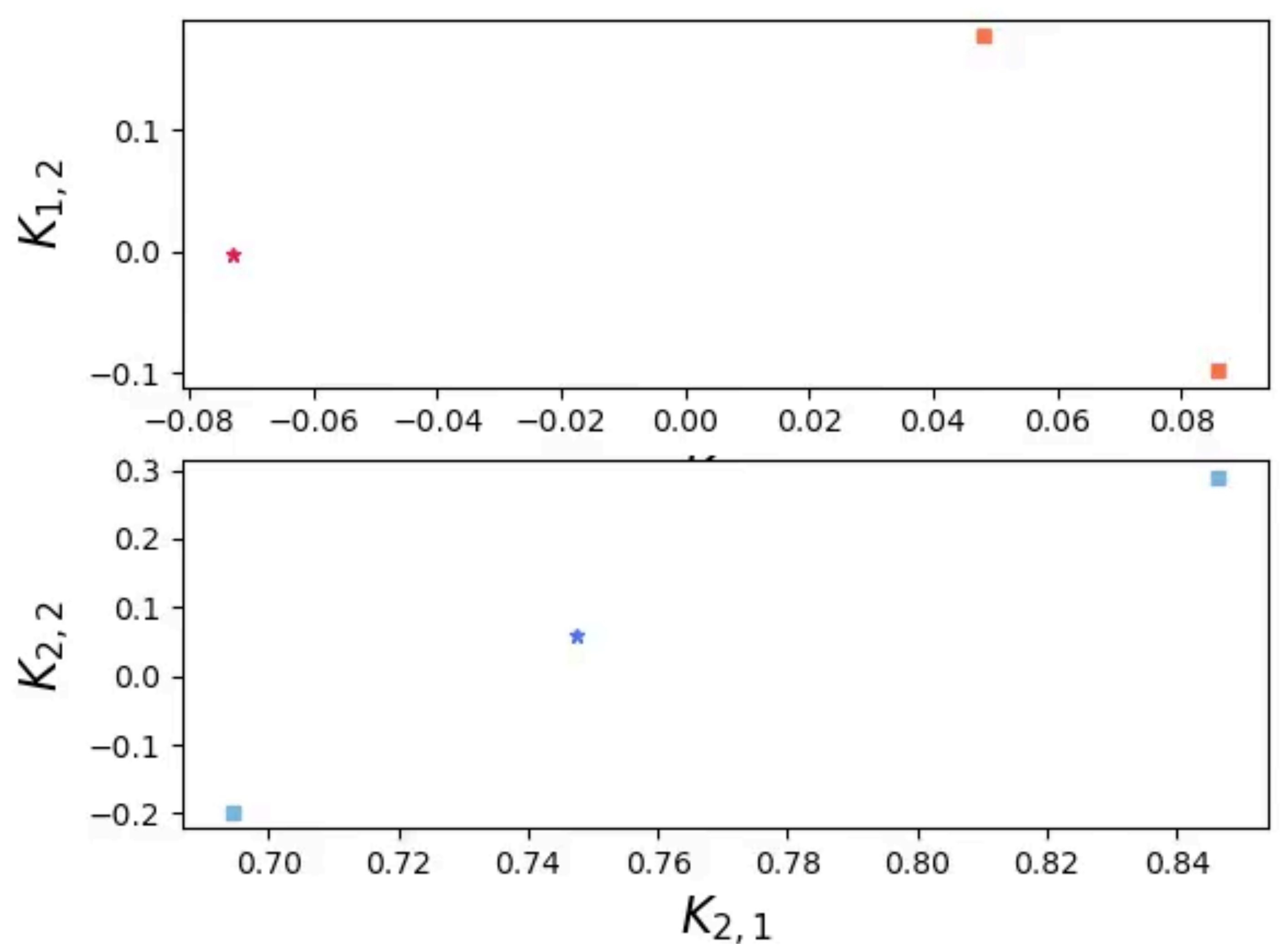
Proposition [Mazumdar et al. 2020]:

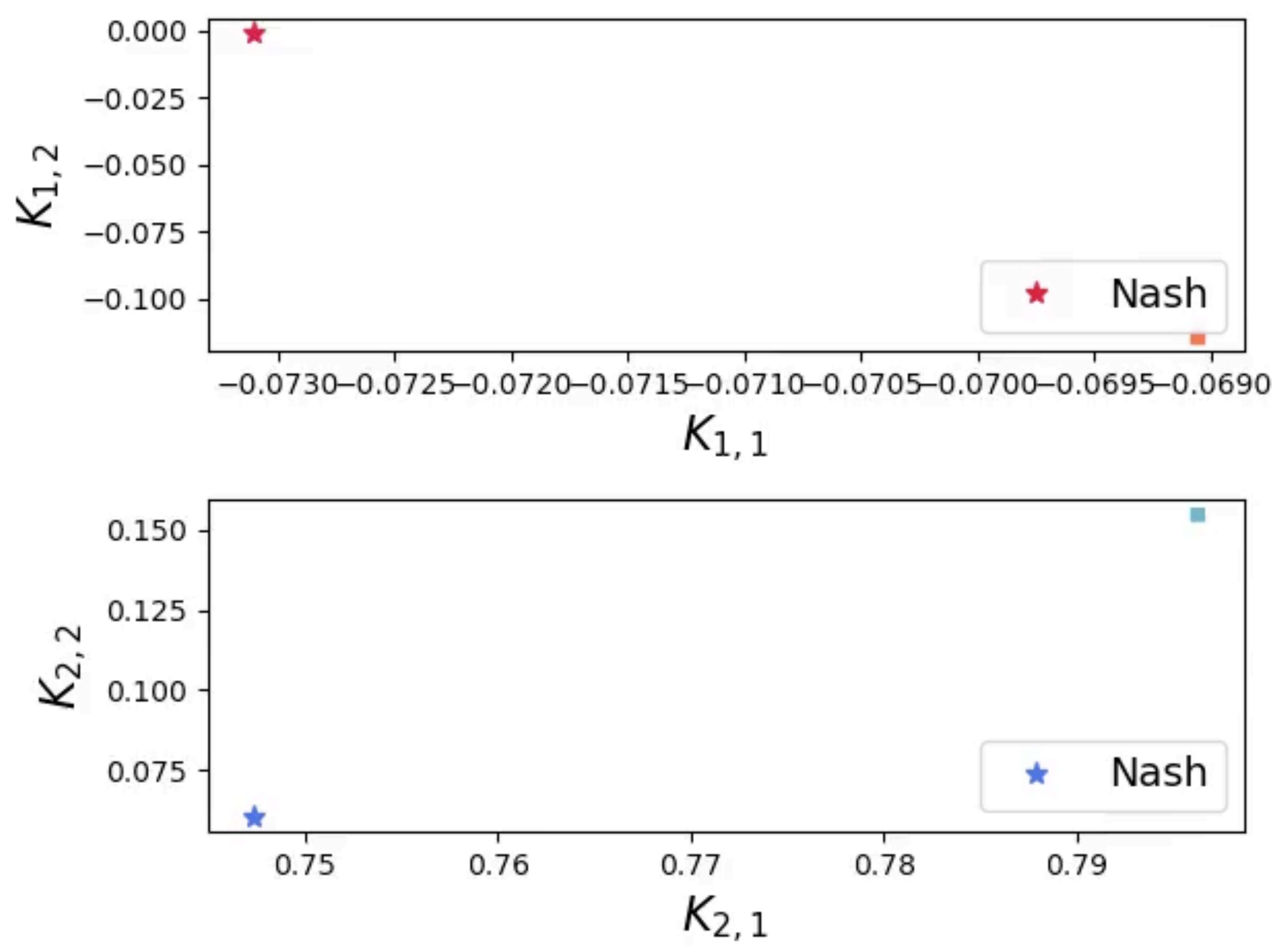
Policy Gradients have no —even local— guarantees of convergence in general-sum games.

Nash equilibria in general-sum Markov games can be ***strictly unstable for the continuous-time dynamics.***

- Policy gradient algorithms would almost surely avoid the Nash under a random initialization.

Players diverge from Nash and converge to limit cycles





Average sequence of play does not converge to Nash

Policy gradient in Zero-sum Markov Games

Independent-Policy Gradients: $$\pi_{t+1} = \mathcal{P}_\Pi\left(\pi_t + \eta F(\pi_t)\right)$$

In zero-sum Markov games we can give a more positive result:

Policy gradient in Zero-sum Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

In zero-sum Markov games we can give a more positive result:

Proposition [Mazumdar et al. 2020]:

All Nash equilibria are locally stable in zero-sum games

- Proximal point and similar algorithms would always converge Nash when initialized close enough.

Local stability means only `bad' discretization can cause divergence.

Policy gradient in Zero-sum Markov Games

Independent-Policy Gradients: $\pi_{t+1} = \mathcal{P}_{\Pi}\left(\pi_t + \eta F(\pi_t)\right)$

In zero-sum Markov games we can give a more positive result:

Proposition [Mazumdar et al. 2020]:

All Nash equilibria are locally stable in zero-sum games

- Proximal point and similar algorithms would always converge Nash when initialized close enough.

Proof Sketch:

Analyze limiting continuous-time dynamics: $\dot{\pi} = F(\pi)$

We show the local linearization around Nash in zero-sum games is alway negative (semi)-definite, which implies local stability.

Policy gradient in Zero-sum Markov Games

More recently:

[Daskalakis et al., 2020]

*Thm: **Time-scale separated** independent policy gradients converge in zero-sum Markov games.*

Consider the independent policy gradient algorithm with $\eta_2 << \eta_1$:

$$\pi_{1,t+1} = \mathcal{P}_{\Pi_1}\left(\pi_{1,t} + \eta_1 \nabla_1 U(\pi_{1,t}, \pi_{2,t})\right)$$
$$\pi_{2,t+1} = \mathcal{P}_{\Pi_2}\left(\pi_{2,t} - \eta_2 \nabla_2 U(\pi_{1,t}, \pi_{2,t})\right)$$

Policy gradient in Zero-sum Markov Games

More recently:

[Daskalakis et al., 2020]

*Thm: **Time-scale separated** independent policy gradients converge in zero-sum Markov games.*

Consider the independent policy gradient algorithm with $\eta_2 << \eta_1$:

$$\pi_{1,t+1} = \mathcal{P}_{\Pi_1}\left(\pi_{1,t} + \eta_1 \nabla_1 U(\pi_{1,t}, \pi_{2,t})\right)$$
$$\pi_{2,t+1} = \mathcal{P}_{\Pi_2}\left(\pi_{2,t} - \eta_2 \nabla_2 U(\pi_{1,t}, \pi_{2,t})\right)$$

Then:

$$\frac{1}{T}\sum_{t=1}^{T} \max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2}\max_{\pi_1} U(\pi_1, \pi_2) \to 0$$

Paper has a polynomial rate of convergence of $O\left(T^{-1/10.5}\right)$

Policy gradient in Zero-sum Markov Games

More recently:

[Daskalakis et al., 2020]

*Thm: **Time-scale separated** independent policy gradients converge in zero-sum Markov games.*

Consider the independent policy gradient algorithm with $\eta_2 << \eta_1$:

$$\pi_{1,t+1} = \mathcal{P}_{\Pi_1}\left(\pi_{1,t} + \eta_1 \nabla_1 U(\pi_{1,t}, \pi_{2,t})\right)$$
$$\pi_{2,t+1} = \mathcal{P}_{\Pi_2}\left(\pi_{2,t} - \eta_2 \nabla_2 U(\pi_{1,t}, \pi_{2,t})\right)$$

Then:

$$\frac{1}{T}\sum_{t=1}^{T} \max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2}\max_{\pi_1} U(\pi_1, \pi_2) \to 0$$

Proof Sketch:

Relies on recent work on ***non-convex-non-concave*** min-max optimization.

Timescale separation allows one to overcome the non-monotonicity of the gradient mapping.

- Fast timescale guarantees that: $\pi_{1,t} \to BR(\pi_{2,t})$
- Convergence of fast timescale + Danskin's theorem guarantees that: $\nabla g(\pi_2) = \nabla max_{\pi_1} U(\pi_1, \pi_2) = \nabla_2 U(\pi_1, \pi_2)|_{\pi_1 = BR(\pi_2)}$

Policy gradient in Zero-sum Markov Games

More recently:

[Daskalakis et al., 2020]

*Thm: **Time-scale separated** independent policy gradients converge in zero-sum Markov games.*

Consider the independent policy gradient algorithm with $\eta_2 << \eta_1$:

$$\pi_{1,t+1} = \mathcal{P}_{\Pi_1}\left(\pi_{1,t} + \eta_1 \nabla_1 U(\pi_{1,t}, \pi_{2,t})\right)$$
$$\pi_{2,t+1} = \mathcal{P}_{\Pi_2}\left(\pi_{2,t} - \eta_2 \nabla_2 U(\pi_{1,t}, \pi_{2,t})\right)$$

Then: $$\frac{1}{T}\sum_{t=1}^{T} \max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2}\max_{\pi_1} U(\pi_1, \pi_2) \to 0$$

- "Independent" policy gradients but not symmetric: requires timescale separation

Policy gradient in Zero-sum Markov Games

More recently:

[Daskalakis et al., 2020]

*Thm: **Time-scale separated** independent policy gradients converge in zero-sum Markov games.*

Consider the independent policy gradient algorithm with $\eta_2 << \eta_1$:

$$\pi_{1,t+1} = \mathcal{P}_{\Pi_1}\left(\pi_{1,t} + \eta_1 \nabla_1 U(\pi_{1,t}, \pi_{2,t})\right)$$
$$\pi_{2,t+1} = \mathcal{P}_{\Pi_2}\left(\pi_{2,t} - \eta_2 \nabla_2 U(\pi_{1,t}, \pi_{2,t})\right)$$

Then: $$\frac{1}{T}\sum_{t=1}^{T} \max_{\pi_1} U(\pi_1, \pi_{2,t}) - \min_{\pi_2} \max_{\pi_1} U(\pi_1, \pi_2) \to 0$$

[Zeng et al. 2022]

Follow-up work has made the rates faster ($O\left(T^{-1/3}\right)$) by using decaying entropy regularization, though staying with the two-timescale structure.

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
2. *No-regret Learning in Markov Games*

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
   - No strong convergence guarantees in general
   - Fast algorithms for zero-sum games by exploiting timescale separation.

2. *No-regret Learning in Markov Games*

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
    - No strong convergence guarantees in general
    - Fast algorithms for zero-sum games by exploiting timescale separation.

2. *No-regret Learning in Markov Games*

No-regret Learning in Markov Games

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

Markov Games do not allow efficient no-regret learning: Two hardness results

1. ***Computational hardness of No-Regret*** [Radanovic et al., 2019, Bai et al., 2020]

No-regret learning in finite horizon Markov Games would imply a polynomial time algorithm for solving parity with noise ***(which is conjectured to be hard).***

No-regret Learning in Markov Games

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

Markov Games do not allow efficient no-regret learning: Two hardness results

1. ***Computational hardness of No-Regret*** [Radanovic et al., 2019, Bai et al., 2020]

   No-regret learning in finite horizon Markov Games would imply a polynomial time algorithm for solving parity with noise ***(which is conjectured to be hard).***

2. ***Statistical hardness of No-Regret*** [Liu et al., 2020]

   No-regret learning in Markov Games is at least as hard as learning the best Markov policy in ***partially-observable MDPs***.

No-regret Learning in Markov Games

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

Markov Games do not allow efficient no-regret learning: Two hardness results

1. ***Computational hardness of No-Regret*** [Radanovic et al., 2019, Bai et al., 2020]

No-regret learning in finite horizon Markov Games would imply a polynomial time algorithm for solving parity with noise ***(which is conjectured to be hard).***

2. ***Statistical hardness of No-Regret*** [Liu et al., 2020]

No-regret learning in Markov Games is at least as hard as learning the best Markov policy in ***partially-observable MDPs***.

However, as we will see, we can still efficiently learn and compute non-stationary Markov CCE.

No-regret Learning in Markov Games

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

No-regret Learning in Markov Games

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

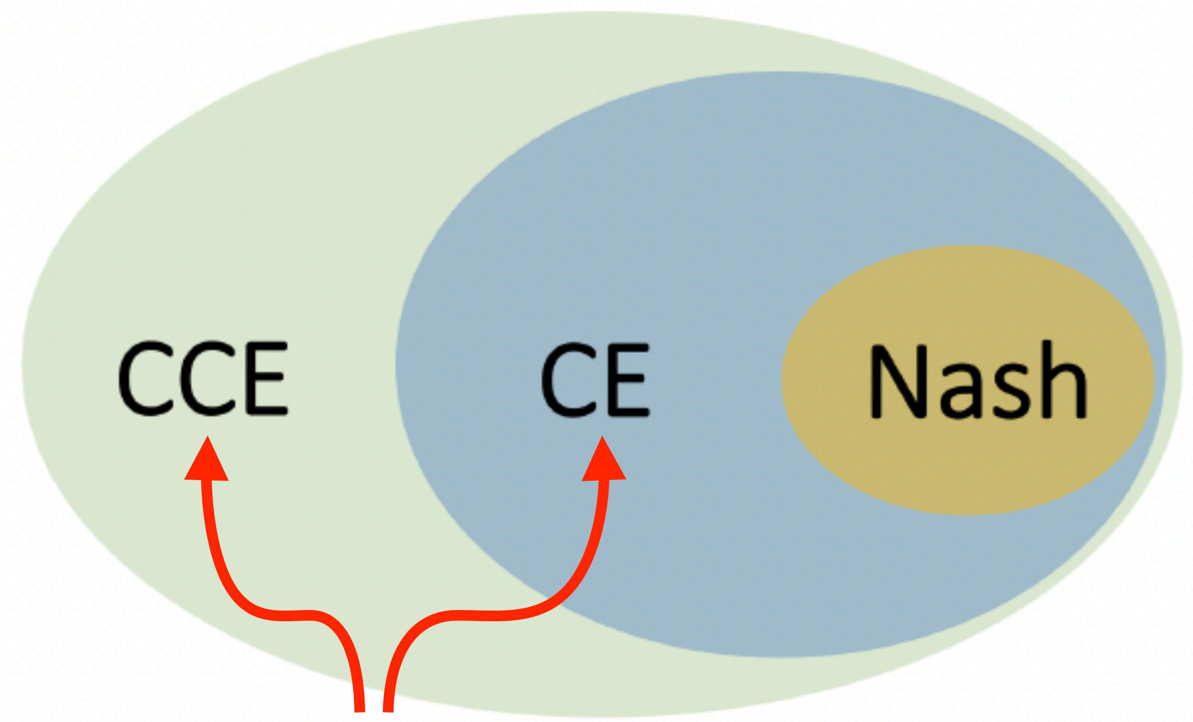
Computing stationary CCE in Infinite Horizon Markov Games is hard

1. ***Computing a stationary CCE is PPAD-hard*** [Daskalakis et al., 2022]

The problem of computing a stationary CCE in infinite-horizon Markov games is as hard as computing a Nash!

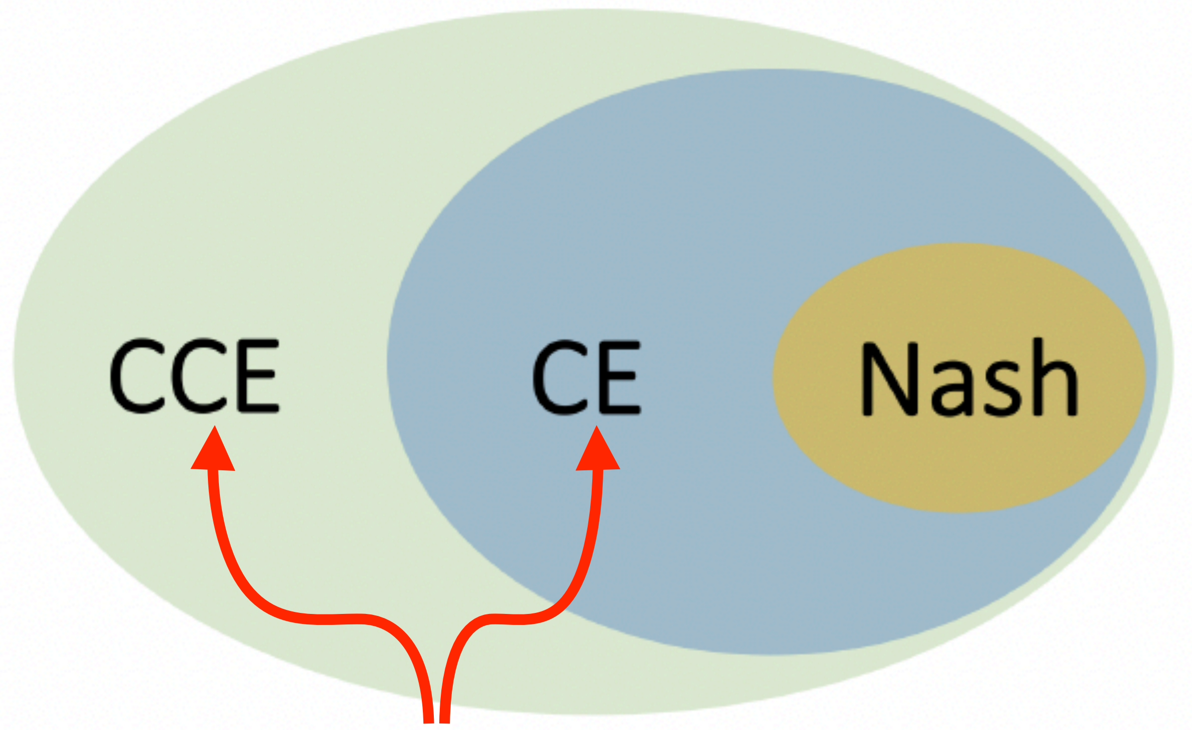
Recap: No-regret Learning in Markov Games

General-sum Normal-form games



Can be computed in polynomial time via
no-regret/no-swap-regret learning

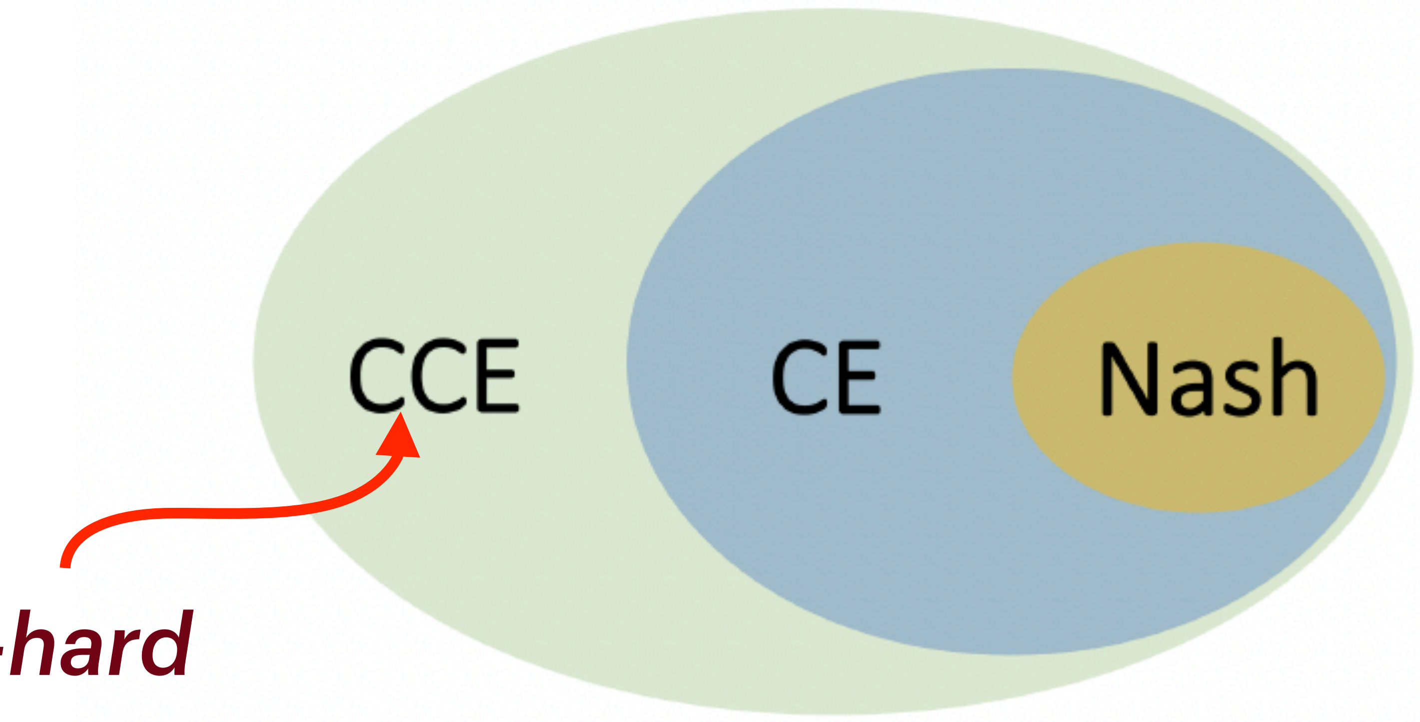
General-sum finite-horizon Markov games



Can be computed in polynomial time ***but not via no-regret/no-swap-regret learning***

Possible in extensive-form games by lifting

General-sum infinite-horizon Markov games



PPAD-hard

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
    - No strong convergence guarantees in general
    - Fast algorithms for zero-sum games by exploiting timescale separation.

2. *No-regret Learning in Markov Games*
    - Impossible in general, still should be able to compute non-stationary CCE though.

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We'll look at two classes of algorithms:

1. *Individual Policy Gradient Algorithms in Markov Games*
   - No strong convergence guarantees in general
   - Fast algorithms for zero-sum games by exploiting timescale separation.

2. *No-regret Learning in Markov Games*
   - Impossible in general, still should be able to compute non-stationary CCE though.

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We ***cannot*** blindly use no-regret learning algorithms or variational inequality methods for Multi-agent RL.

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We ***cannot*** blindly use no-regret learning algorithms or variational inequality methods for Multi-agent RL.
- Need more specialized approaches that better use the ***underlying structure.***
  - No-regret algorithms and variational inequality methods will be useful building blocks however!

From Normal-Form to Markov Games

Can we use algorithms that we have seen for normal-form games for
multi-agent reinforcement learning?

- We ***cannot*** blindly use no-regret learning algorithms or variational inequality methods for Multi-agent RL.
- Need more specialized approaches that better use the ***underlying structure.***
    - No-regret algorithms and variational inequality methods will be useful building blocks however!

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

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

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