Reinforcement Learning China Summer School

RLChina 2021

MULTI-AGENT LEARNING BASICS

Dr. Yaodong Yang
Assistant Professor
King's College London
www.yangyaodong.com

08/2021

Intelligence is learning from mistakes!

"… if a machine is expected to be infallible, it cannot also be intelligent. There are several mathematical theorems which say almost exactly that. But these theorems say nothing about how much intelligence may be displayed if a machine makes no pretence at infallibility…"

— *Alan Turing, 1947*

Remarkable Success of MARL in Gaming AI Applications

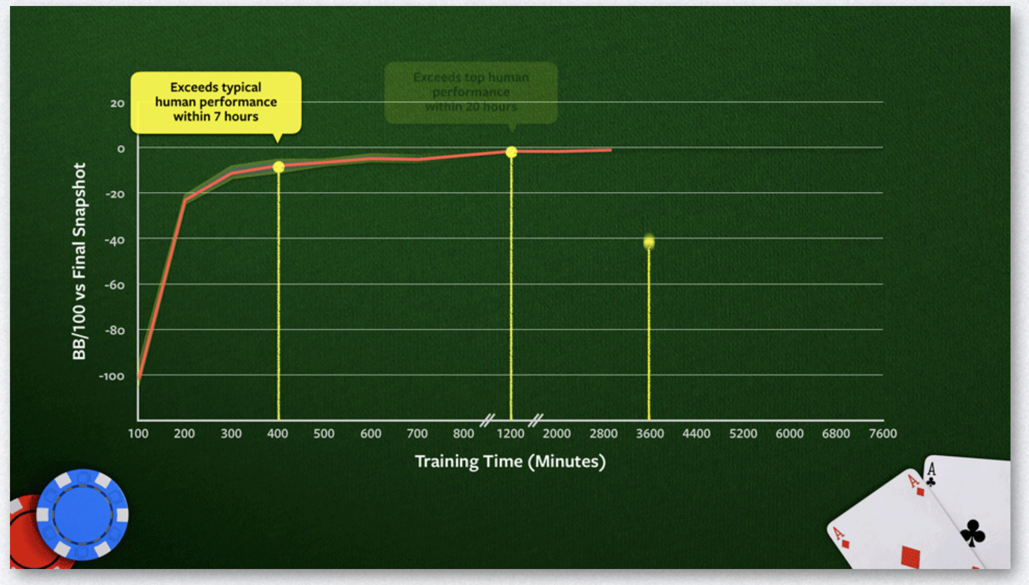
Great advantages have been made since 2019!

Jan 2016 | Dec 2017 | July 2018 | Jan 2019 | Apr 2019 | July 2019 | Sep 2019 | July 2021

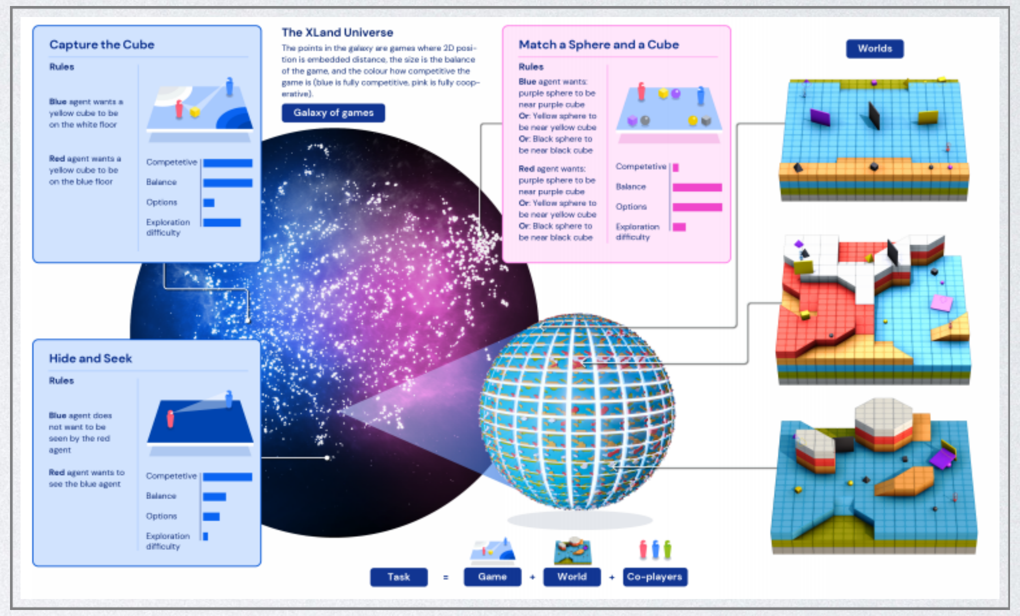
AlphaGO Series

AlphaStar (DeepMind)

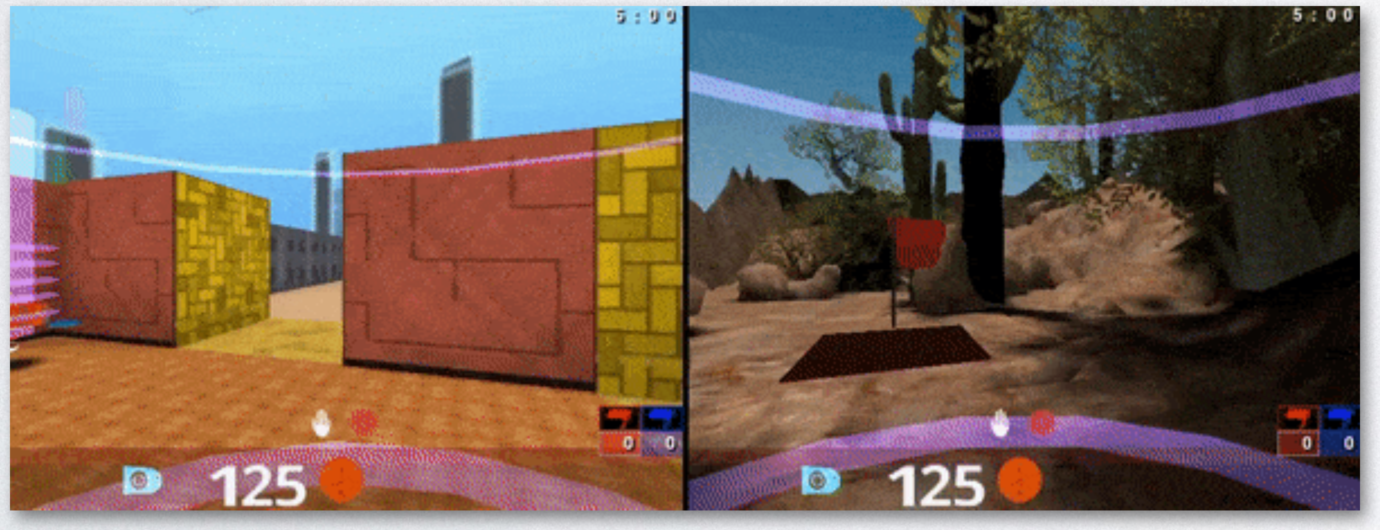
Pluribus Poker (FAIR)



XLand (DeepMind)

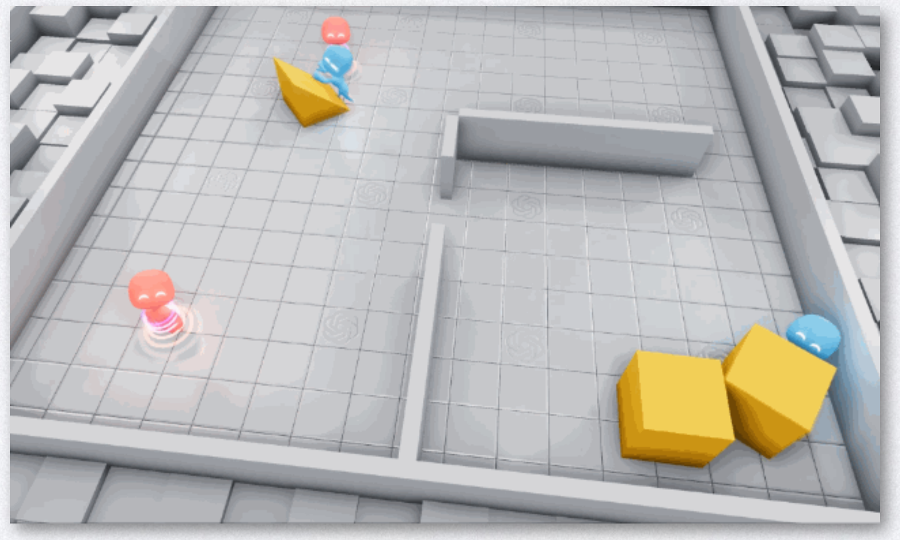


Capture-the-flag (DeepMind)



Dota2 (OpenAI)

Hide and Seek (OpenAI)



• • •

Multi-agent intelligence emerges

Multi-agent Intelligence Components

Multi-Agent Intelligence

Autonomous Driving
Gaming AI
Smart Grid / City
. . .

=

Algorithmic Game Theory Fundamentals

Equilibrium/Solution Concept
Learning Dynamics Analysis
Mechanism Design
. . .

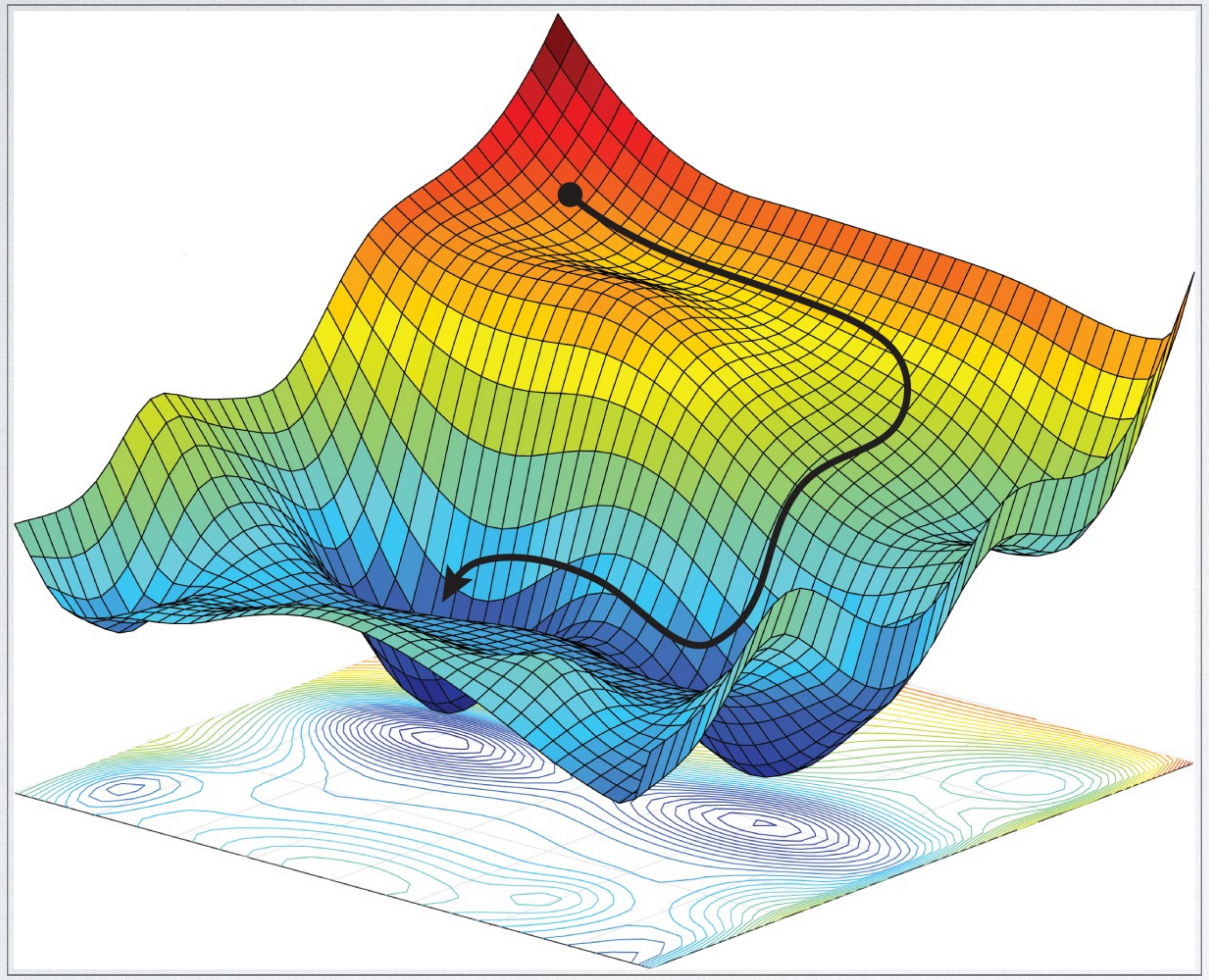
+

Machine Learning Techniques

Reinforcement Learning
Deep Learning
Auto-differentiation
. . .

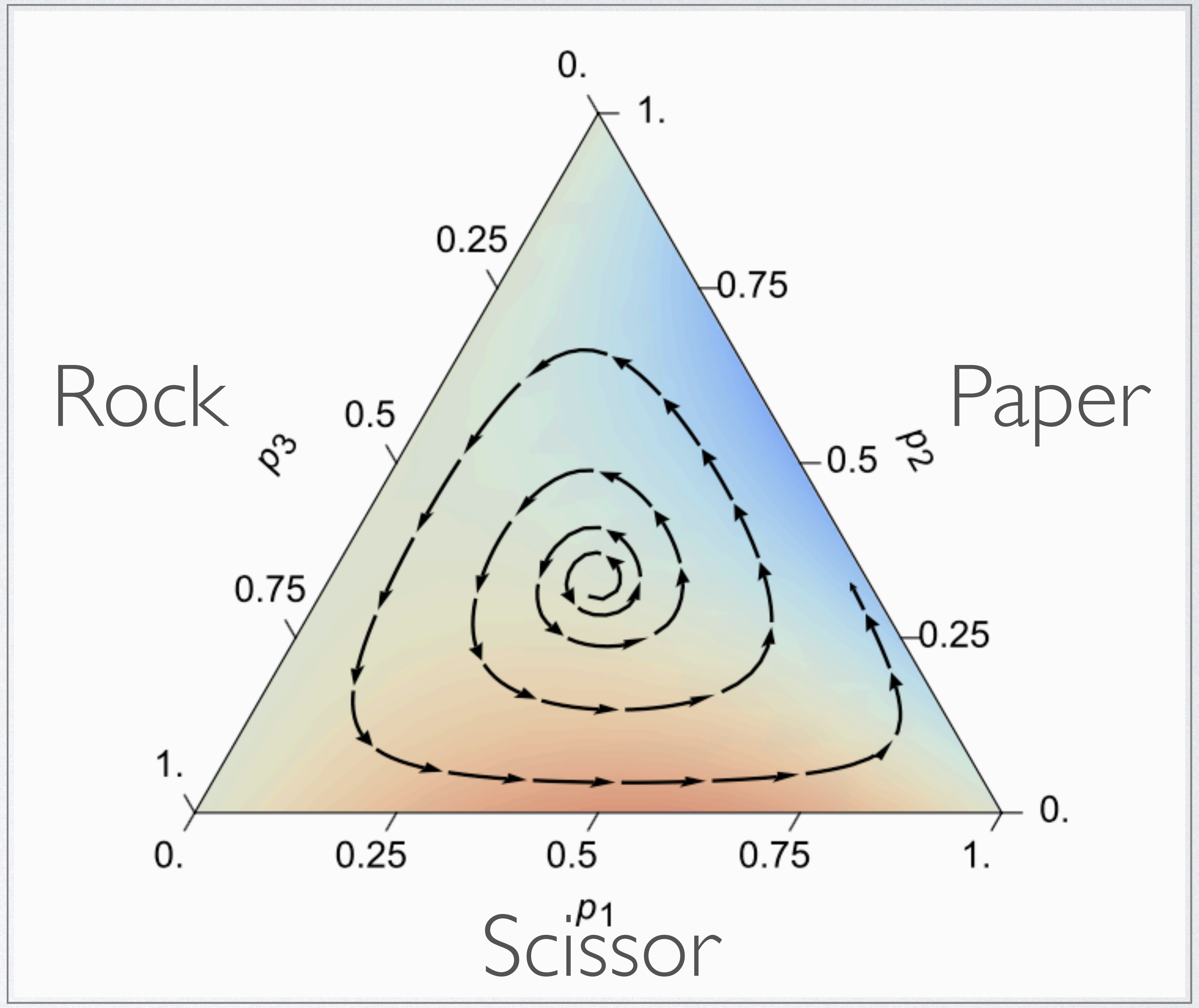
Game Theory Lays the Foundation for Multi-agent Learning

Normal machine learning problems:



$$\|\nabla f(x)\|_2 = 0$$

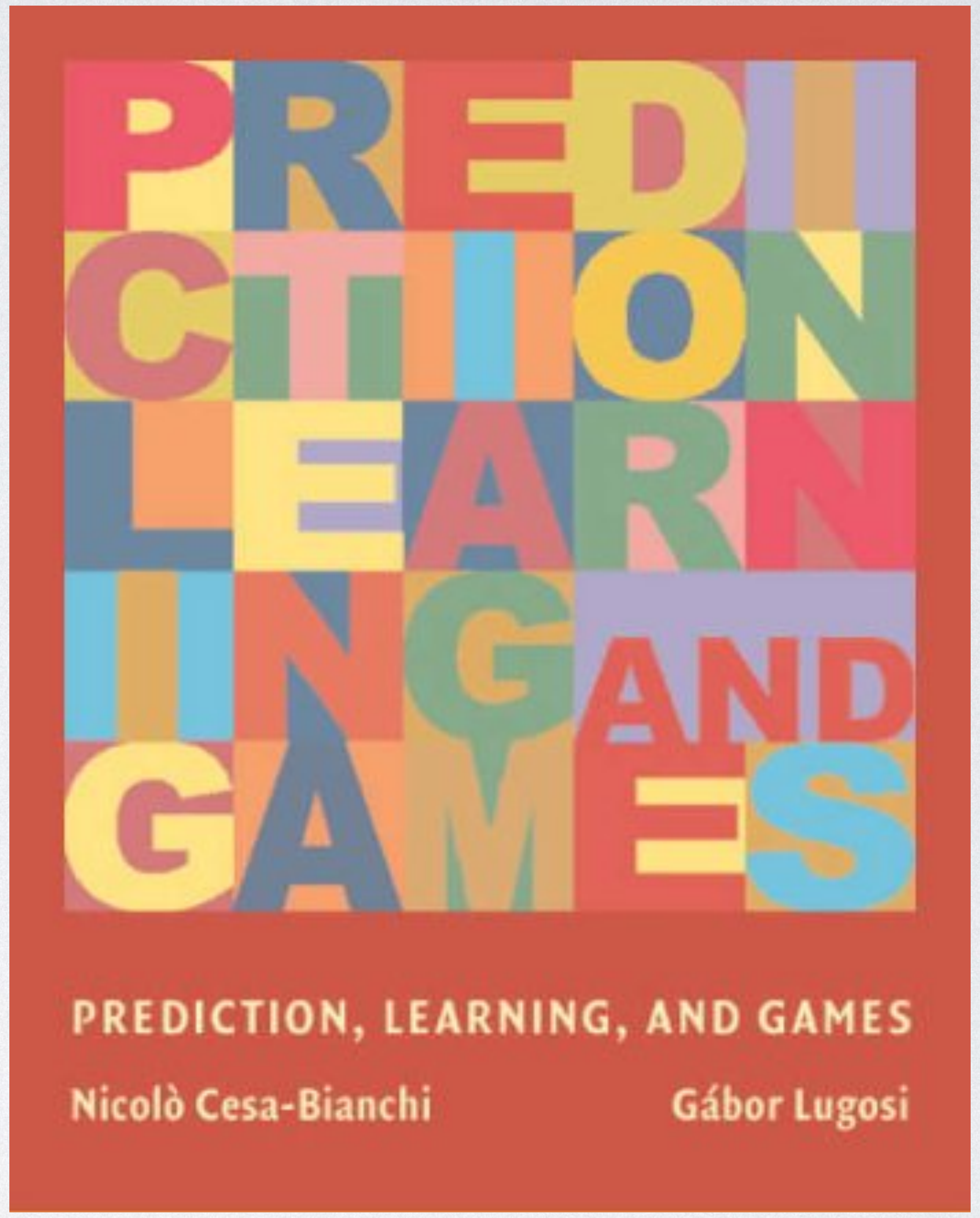
Multi-agent learning problems:



the learning outcomes are described by game theory

Recommended Resources:

- A self-contained MARL survey from game theoretical perspective:
  - https://arxiv.org/abs/2011.00583
- Textbook
  - <Prediciton, learning and games> Nicolo Cesa-Bianchi
- Algorithmic Game Theory lectures:
  - http://www.cs.jhu.edu/~mdinitz/classes/AGT/Spring2020/ (Uncited screenshots refer to Lectures 1,2,3,4,5,6,7,8,9)
- If you want to know more details about modern MARL methods
  - Talk: A General Solver to Two-player Zero-sum Games
  - Talk: Recent advances of MARL in Gaming AI
  - Talk: Dealing with Non-transitivity in Two-player Zero-sum Games
- If you want to get hands to solve real-world games, e.g., Poker/Chess
  - https://github.com/sjtu-marl/malib



Contents

- Algorithmic Game Theory Overview
- Computing Nash Equilibrium
  - Two-Player Zero-Sum Games (LP, fictitious play, double oracle, PSRO)
  - Two-Player General-Sum Games (support enumeration, Lemke-Howson method)
  - N-Player Potential Games (best response dynamics)
- Connections to MARL
  - MARL Formulations
  - Complexity Results (Nash is PPAD-hard)
  - Other Necessary Solution Concepts (correlated equilibrium, coarse CE)
- No-Regret Dynamics
  - Solving Coarse Correlated Equilibrium
  - Solving Two-Player Zero-Sum Games (FP is not no-regret, MWU, ODO)
  - Solving Correlated Equilibrium (swap regret)

Algorithmic Game Theory Overview

- Game theory studies the interaction between rationale (selfish) agents.
- It is an area between CS and Economics.
- MARL is to study algorithmic games theory with powerful machine learning tools.

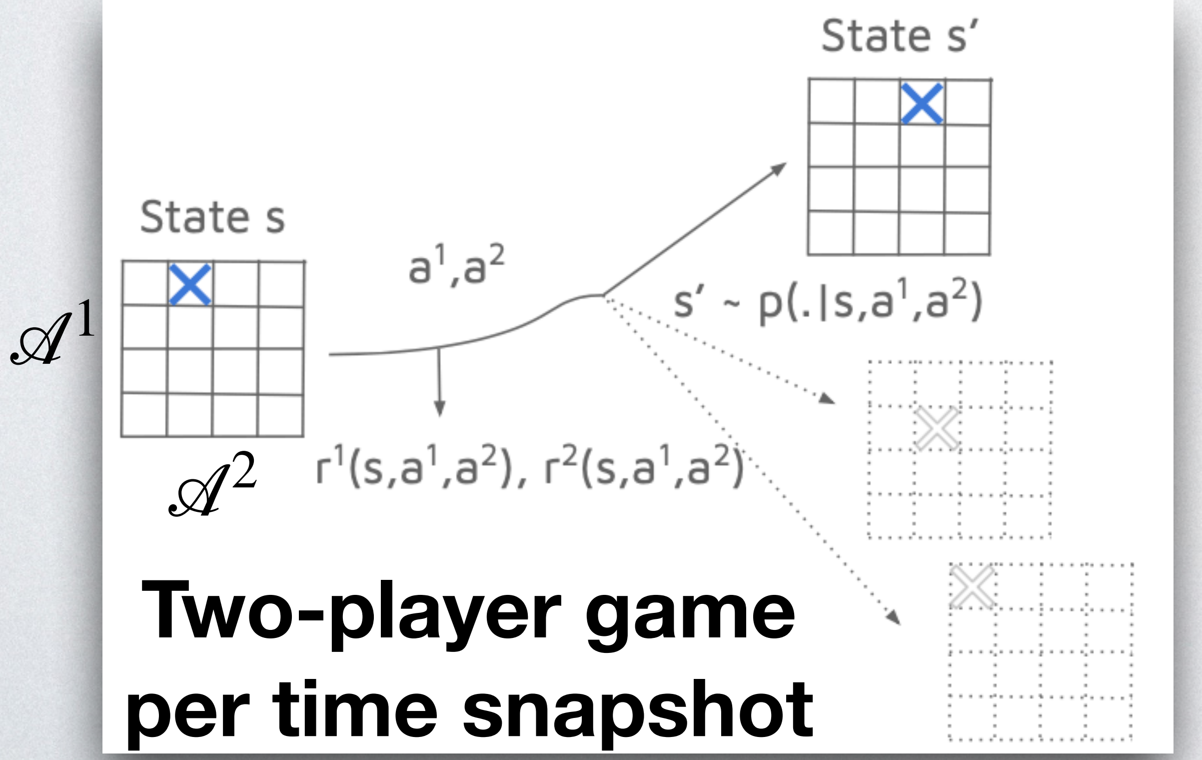
1. Computing Equilibria
    - Is it reasonable to model behaviours through different equilibrium concept (e..g, the "invisible hands")
    - how can we compute the equilibrium efficiently and distributedly ?
2. Understanding the Inefficiency of Equilibrium
    - Is the equilibrium "optimal" compared to maximal social welfare ?
    - bounding and deriving the distance to optimality
3. Mechanism Design
    - The science of "rule making": can we design games so that selfish agents can lead to good outcomes ?
    - Heavily focus on auctions: how to design the auction rules to incentivise agents to tell the truth.

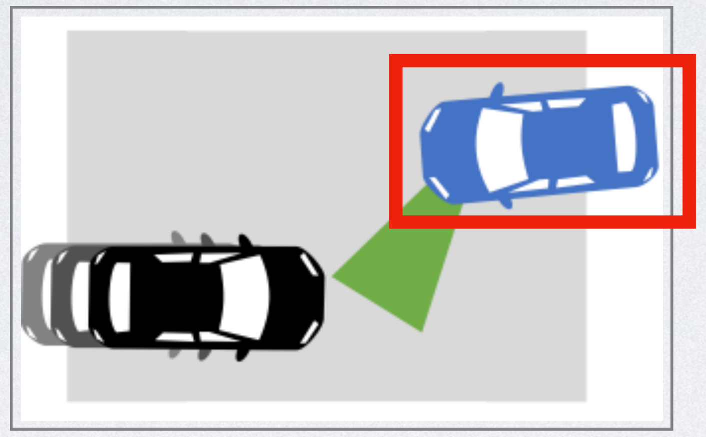
Computing Equilibria

Traffic intersection is naturally a multi-agent system. From each driver's perspective, in order to perform the optimal action, he must <u>take into account others' behaviours.</u>

what you see

what the computer sees





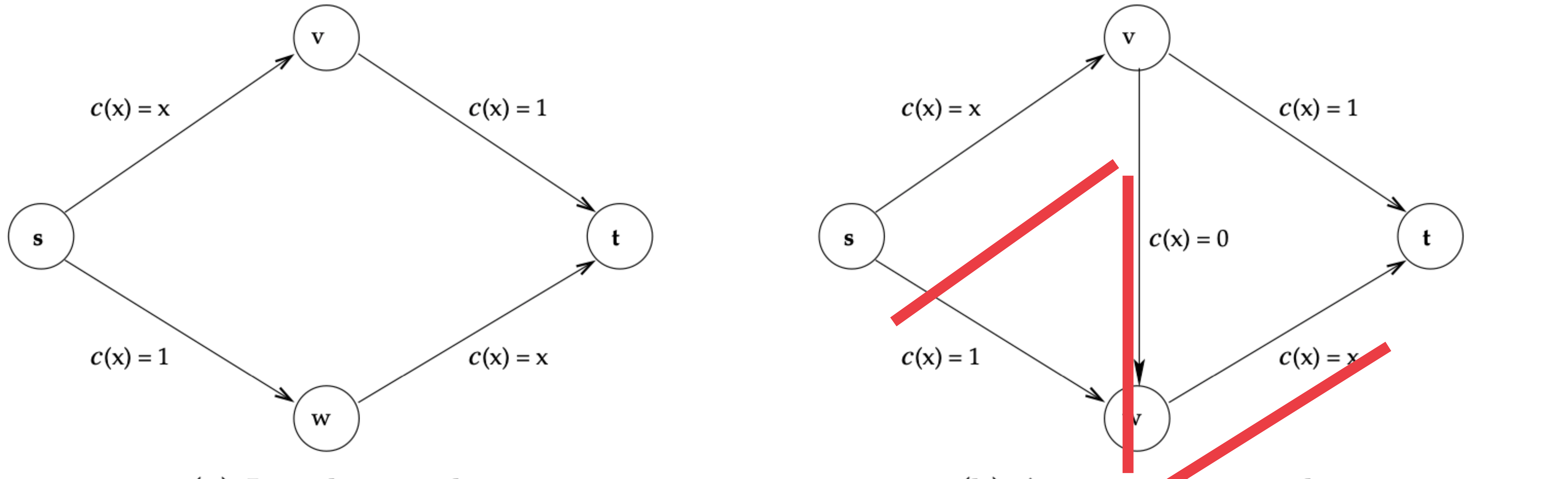
scenario

	Yield	Rush
Yield	(0, 0)	(1, 2)
Rush	(2, 1)	(0, 0)

normal-form game

- When the drivers are rational, they will reach the outcome of a Nash Equilibrium. It is the outcome of interaction. Knowing it can predict future.
- Real-world decision making has cooperation & competition. For each agent, how to infer the belief of the other agents and make the optimal action is critical.
- The concept of using traffic light is in fact a correlated equilibrium.
- Many-agent system is when agents >> 2. It is a very challenging problem to compute equilibrium, thus making decisions.

The Inefficiency of Equilibria



(a) Initial network (b) Augmented network

Figure 1: Braess's Paradox. The addition of an intuitively helpful edge can adversely affect all of the traffic.

optimal traffic: half s->v->t, half s->w->t
cost per person: 1+1/2

optimal traffic: all goes s->v->w->t
cost per person: 2

price of anarchy: $\frac{\text{value of worst Nash}}{\text{optimial outcome}} = \frac{2}{3/2} \geq 1$

Selfish behaviours can lead to inefficient equilibrium ! Can we bound them for real-world problems?

Mechanism Design

- Suppose we are to organise an auction, each player's utility is set as *(valuation - final price)$_+$, how can we set the auction rule: the mechanism of determining the final price? We want it to be truthful that all players bids their valuations.*
- Highest price auction:
  - Each player will bid less than valuation, if he wins the bids, he will try to decrease the price.
  - The final price depends on others' valuations. It is unclear for both players and auctioneer to practice
- Second-price auction:
  - Player bids the valuation price is the dominant strategy. Assuming b_j to be the highest price.
    - if $v_i < b_j$, then 0 utility is better than negative, thus bidding v_i
    - if $v_i > b_j$, bidding v_i is the always dominant than bidding other numbers.
- In many real-world problems, how can we design truthful mechanism that meets computation constraint.
- See Zhengyang's talk at RLChina for this topic.

Contents

Nash Equilibrium

- Let n players, $S = S_1 \times \cdots \times S_n$ is the joint strategy profile, $u_i : S \to \mathbb{R}$ is the utility function, Nash equilibrium is

$$\mathop{\mathbf{E}}_{s_j \sim \mu_j} \mathop{\mathbf{E}}_{\forall j \in [n]} \left[u_i\left(s_1, \ldots, s_n\right)\right] \geq \mathop{\mathbf{E}}_{\substack{s_i \sim \mu_i' \\ s_{-j} \sim \mu_{-j}}} \left[u_i\left(s_1, \ldots, s_n\right)\right] \quad \forall \mu_i' \in \Delta_{S_i}$$

- Mixed strategy Nash equilibrium always exists in finite player finite action games.
- For continuous utility games, the strategy set needs to be compact
- Note that $\mu_i' \in \Delta_{S_i}$ can be replaced by $a \in S_i$ because deviation is at most a pure strategy !
- Here no state transitions are considered. In Markov game, the solution concept is Markov Perfect Equilibrium.
- The expectation is computed as follows

$$\begin{aligned} x^T A y &= \sum_{i=1}^{N} \sum_{j=1}^{M} A_{ij} x_i y_j \\ &= \sum_{i=1}^{N} \sum_{j=1}^{M} A_{ij} \mathbf{Pr}[\text{ player 1 plays } i] \mathbf{Pr}[\text{ player 2 plays } j] \\ &= \mathop{\mathbf{E}}_{\substack{i \sim x \\ j \sim y}} \left[u_i(i, j)\right] \end{aligned}$$

Nash Equilibrium in Two-Player Zero-Sum Games

- von Neumann theorem: Two-player Nash can be computed in P-time through linear programmes (LP).

Prime problem

row player maximises the worst situation

$$\max_{v \in \mathbb{R}} v$$
$$\text{s.t. } \mathbf{p}^\top \mathbf{A} \geq v \cdot \mathbf{1}$$
$$\mathbf{p} \geq \mathbf{0} \text{ and } \mathbf{p}^\top \mathbf{1} = 1$$

/

Dual problem

column player's view

$$\min_{v \in \mathbb{R}} v$$
$$\text{s.t. } \mathbf{q}^\top \mathbf{A}^\top \leq v \cdot \mathbf{1}$$
$$\mathbf{q} \geq \mathbf{0} \text{ and } \mathbf{q}^\top \mathbf{1} = 1$$

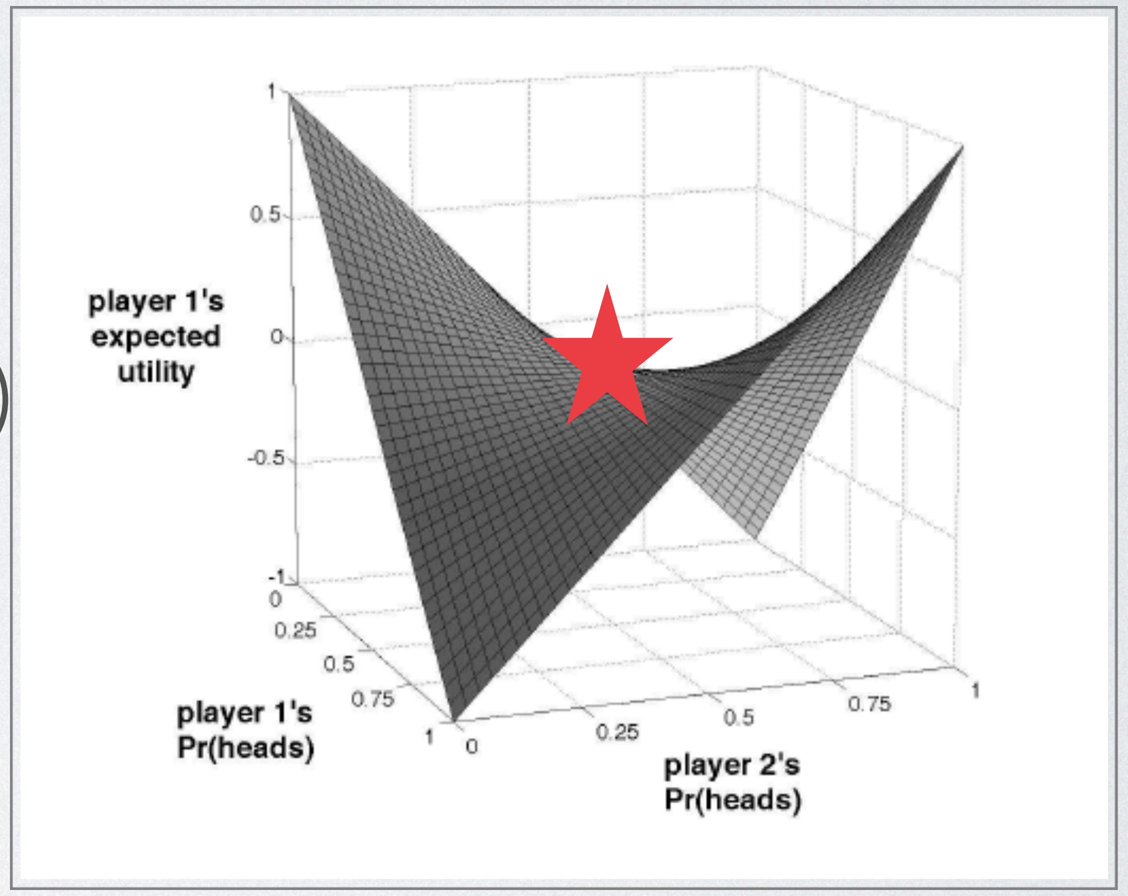
$\Leftrightarrow$

Minimax theorem

zero-duality gap for convex problems

$$\max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \mathbf{A} \mathbf{q} = \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^\top \mathbf{A} \mathbf{q}$$

- The v^* is the Nash value
  - proof: $v \leq v^*$ due to definition of v^*, $v \geq v^*$ due to being the LP solution.
  - corollary: all Nash value are the same (saddle point is unique in bimatrix game)



- The (p, q) is the Nash equilibrium:
  - proof: suppose the player plays $\boldsymbol{x}, \boldsymbol{y}$ instead of $\boldsymbol{p}, \boldsymbol{q}$
  - $x^T A q = \sum_{i=1}^{N} x_i (Aq)_i \leq \max_{i \in [N]} (Aq)_i = v_c = v^*$, $p^T A y = \sum_{j=1}^{M} \left(p^T A\right)_j y_j \geq \min_{j \in [M]} \left(p^T A\right)_j = v_r = v^*$, thus no incentives to deviate.
- Sion's minimax theorem generalises to quasi-convex/concave functions $\min_{x \in X} \sup_{y \in Y} f(x, y) = \sup_{y \in Y} \min_{x \in X} f(x, y)$

Fictitious Play [Brown 1951]

- Maintain a belief over the historical actions that the opponent has played, and the learning agent then takes the best response to this empirical distribution.

$$a_i^{t,*} \in \mathbf{BR}_i\left(p_{-i}^t = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathcal{I}\left\{a_{-i}^{\tau} = a, a \in \mathbb{A}\right\}\right)$$

$$p_i^{t+1} = \left(1 - \frac{1}{t}\right)p_i^t + \frac{1}{t}a_i^{t,*}, \text{ for all } i$$

- It guarantees to converge, in terms of the Nash value, in two-player zero-sum games, potential games and 2×2 games , and, the average policy converge to the Nash strategy.

- Examples:

		Player 2	
		a	b
Player 1	A	(1,1)	(0,0)
	B	(0,0)	(1,1)

t	p_1^t	p_2^t	a_1^t	a_2^t
0	(3/4, 1/4)	(1/4, 3/4)	B	a
1	(3/4, 5/4)	(5/4, 3/4)	A	b
2	(7/4, 5/4)	(5/4, 7/4)	B	a
3	(7/4, 9/4)	(9/4, 7/4)	A	b
⋮	⋮	⋮	⋮	⋮
∞	(1/2, 1/2)	(1/2, 1/2)		

Double Oracle [McMahan 2003]

- Double Oracle best responds to the opponent's Nash equilibrium at each iteration.
- To solve the game by solving subgame Nash because not all pure strategies are "useful" in the support of Nash.
- It is much faster than LP, but In the worst-case scenario, it recovers to solve the original game.

Algorithm 1 Double Oracle (McMahan et al., 2003)

Input: A set Π, C strategy set of players
Π_0, C_0: initial set of strategies
for $t = 1$ to ∞ **do**
  if $\Pi_t \neq \Pi_{t-1}$ or $C_t \neq C_{t-1}$ **then**
    Solve the NE of the subgame $\boldsymbol{G}_t$:
    $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*) = \arg\min_{\boldsymbol{\pi} \in \Delta_{\Pi_t}} \arg\max_{\boldsymbol{c} \in \Delta_{C_t}} \boldsymbol{\pi}^\top \boldsymbol{A} \boldsymbol{c}$
    Find the best response $\boldsymbol{a}_{t+1}$ and $\boldsymbol{c}_{t+1}$ to $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*)$:
    $\boldsymbol{a}_{t+1} = \arg\min_{\boldsymbol{a} \in \Pi} \boldsymbol{a}^\top \boldsymbol{A} \boldsymbol{c}_t^*$
    $\boldsymbol{c}_{t+1} = \arg\max_{\boldsymbol{c} \in C} \boldsymbol{\pi}_t^{*\top} \boldsymbol{A} \boldsymbol{c}$
    Update $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_{t+1}\}, C_{t+1} = C_t \cup \{\boldsymbol{c}_{t+1}\}$
  else if $\Pi_t = \Pi_{t-1}$ and $C_t = C_{t-1}$ **then**
    Terminate
  end if
end for

- **iteration 0**: restricted game R vs R
- **iteration 1**:
  - solve Nash of restricted game (1, 0, 0) , (1, 0, 0)
  - unrestricted $\mathbf{Br}^1, \mathbf{Br}^2$ = P, P
- **iteration 2**:
  - solve Nash of restricted games (0, 1, 0) , (0, 1, 0)
  - unrestricted $\mathbf{Br}^1, \mathbf{Br}^2$ = S, S
- **iteration 3**:
  - solve Nash of restricted game (1/3, 1/3, 1/3) , (1/3, 1/3, 1/3)
- **iteration 4**: no new response, END
  - output (1/3, 1/3, 1/3)

example

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

Double Oracle [McMahan 2003]

- It guarantees to converge to Nash equilibrium in two-player zero-sum games.
- We need to prove that early stopping also leads to Nash equilibrium.
- **Convergence proof:**
  - DO finally recovers to solve the whole game
- **Correctness proof:**
  - suppose DO stops at the j-th sub-game (i.e., no new best responses are added)
  - $\forall p, V(p, q_j) \geq v \Rightarrow \forall p, \max_q V(p, q) \geq v$

    $\forall q, V(p_j, q) \leq v \Rightarrow \max_q V(p_j, q) \leq v$

$\Rightarrow \forall p, \max_q V(p_j, q) \leq max_q(p, q)$

p_j must be the minimax optimal,

q_j vice versa, so (p_j, q_j) is the final Nash

Policy Space Response Oracle [Lanctot 2017]

- A generalisation of double oracle methods: the best responser is implemented through deep RL models.
- Each RL model is now a "pure strategy".
- A meta-game is (Π, U, n) where $\Pi = (\Pi_1, \ldots, \Pi_n)$ is the set of policies for each agent and $U : \Pi \to \mathbb{R}^n$ is the reward values for each agent given a joint strategy profile.
- σ_{-i} is distribution over $(\Pi_1^0, \ldots, \Pi_1^T)$, a.k.a meta-strategy
- PSRO generalises all previous methods by varying σ_{-i}.
  - independent learning: $\sigma_{-i} = (0,...,0,0,1)$
  - self-play: $\sigma_{-i} = (0,...,0,1,0)$
  - fictitious play: $\sigma_{-i} = (1/T, 1/T, \ldots, 1/T, 0)$
  - PSRO: $\sigma_{-i} = \textbf{Nash}\left(\Pi^{T-1}, U\right)$

Algorithm 1: Policy-Space Response Oracles

input : initial policy sets for all players Π

Compute exp. utilities U^{Π} for each joint $\pi \in \Pi$

Initialize meta-strategies $\sigma_i = \text{UNIFORM}(\Pi_i)$

while *epoch e in* $\{1, 2, \cdots\}$ **do**

  for *player $i \in [[n]]$* **do**

    for *many episodes* **do**

select opponent policies — Sample $\pi_{-i} \sim \sigma_{-i}$

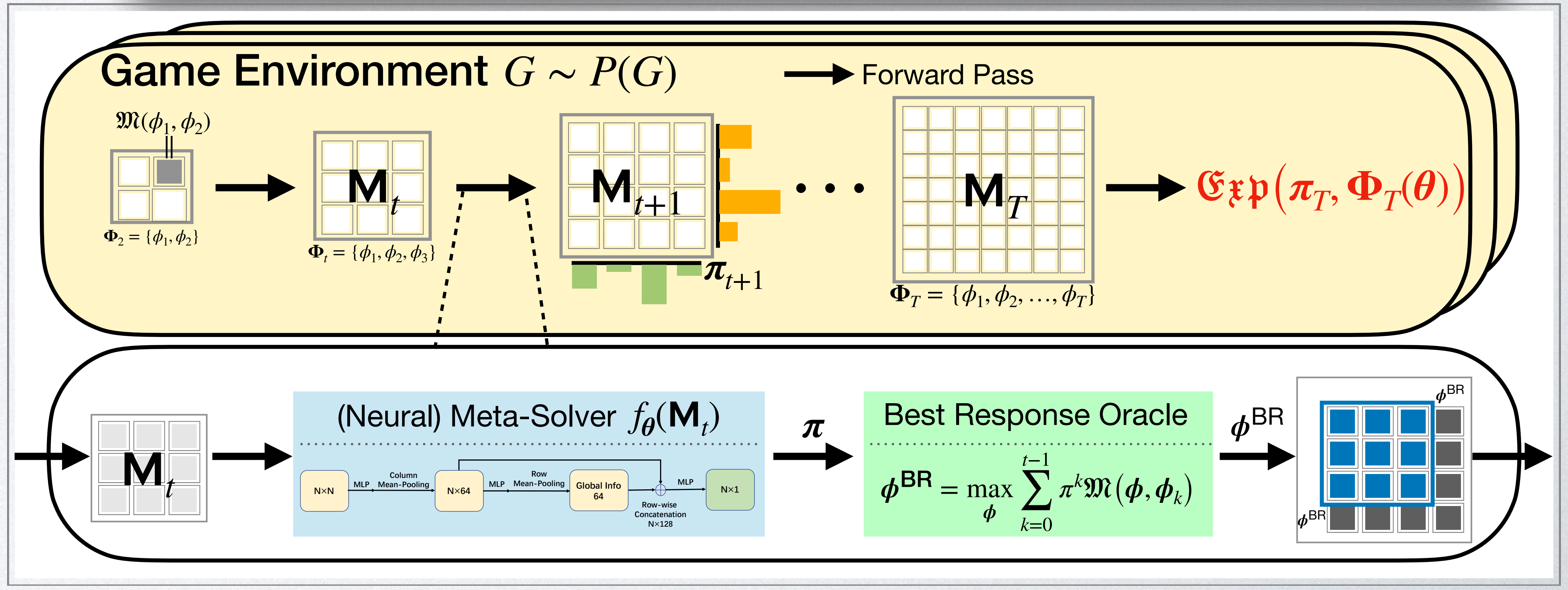
compute the best response — Train oracle π'_i over $\rho \sim (\pi'_i, \pi_{-i})$

augment strategy pool — $\Pi_i = \Pi_i \cup \{\pi'_i\}$

expand the payoff matrix — Compute missing entries in U^{Π} from Π

solve the new meta game — Compute a meta-strategy σ from U^{Π}

Output current solution strategy σ_i for player i



PSRO on Google Football !

https://sites.google.com/view/diverse-psro/

make offside

push and run

Two-Player General-Sum Games

- Exponential-time algorithms are the best we can hope for.
- Support Enumeration Method:
  - **Theorem: a best response has to be a pure strategy that satisfies $x_i > 0 \Longrightarrow (Ay)_i = \max_{k \in [N]} (Ay)_k$, that is, x is a best response to y if only if x has nonzero probability only on pure strategies that max expected utility against y.**

Proof: if $x_i > 0$, then x_i must be best response, otherwise suppose $(Ay)_i < \max_{k \in [N]} (Ay)_k$ it will contradict

$$x^T Ay = \sum_{k=1}^{N} x_k (Ay)_k = x_i (Ay)_i + \sum_{k \neq i} x_k (Ay)_k$$

$$< x_i \max_{k \in [N]} (Ay)_k + \sum_{j \neq i} x_j \max_{k \in [N]} (Ay)_k$$

$$= x_i \max_{k \in [N]} (Ay)_k + (1 - x_i) \max_{k \in [N]} (Ay)_k = \max_{k \in [N]} (Ay)_k$$

if x_i is a best response, then $x_i > 0$

$$x^T Ay = \sum_{i=1}^{N} x_i (Ay)_i = \sum_{i:x_i \neq 0} x_i (Ay)_i = \sum_{i:x_i \neq 0} x_i \max_{k \in [N]} (Ay)_k = \max_{k \in [N]} (Ay)_k \sum_{i:x_i \neq 0} x_i = \max_{k \in [N]} (Ay)_k$$

Two-Player General-Sum Games

- Support Enumeration Method:
  - **Theorem: a best response has to be a pure strategy that satisfies $x_i > 0 \implies (Ay)_i = \max_{k\in[N]} (Ay)_k$, that is, x is a best response to y if only if x has nonzero probability only on pure strategies that max expected utility.**
  - we can have a guess of the Nash support (I, J) and then propose the Nash by solving the following LP

$$\sum_{i\in I} x_i = 1$$

$$\sum_{j\in J} y_j = 1$$

$$\sum_{i\in I} x_i B_{ij} = v \quad \forall j \in J$$

$$\sum_{j\in J} A_{ij} y_j = u \quad \forall i \in I$$

size of $|I| + |J| + 2$,
solved by Gaussian elimination

  - Given the proposal (x, y) we can then check if $u = \max_{k\in[N]} (Ay)_k$ and $v = \max_{k\in[M]} \left(x^T B\right)_k$ are met.

LCP formulation of Two-Player General-Sum Games

- Support Enumeration Method is more like a heuristic search.
- Here we provide a formal formulation through linear complementarity problem (LCP)
- LCP is very much like an LP, but with a new constraint built on a slack variable.
- Recall that in two-player zero-sum game, we can have

$$
\begin{aligned}
\text{minimize} \quad & U_1^* \\
\text{subject to} \quad & \sum_{k \in A_2} u_1\left(a_1^j, a_2^k\right) \cdot s_2^k \leq U_1^* & \forall j \in A_1 \\
& \sum_{k \in A_2} s_2^k = 1 \\
& s_2^k \geq 0 & \forall k \in A_2
\end{aligned}
$$

$$\Leftrightarrow$$

$$
\begin{aligned}
\text{minimize} \quad & U_1^* \\
\text{subject to} \quad & \sum_{k \in A_2} u_1\left(a_1^j, a_2^k\right) \cdot s_2^k + r_1^j = U_1^* & \forall j \in A_1 \\
& \sum_{k \in A_2} s_2^k = 1 \\
& s_2^k \geq 0 & \forall k \in A_2 \\
& r_1^j \geq 0 & \forall j \in A_1
\end{aligned}
$$

LCP formulation of Two-Player General-Sum Games

- **Solving the Nash of two-player general-sum games is an LCP problem.**
  - Both two players' variables need to be considered rather than one player (unlike two-player zero sum!)
  - There is no objective, it is rather a *feasibility program (finding the solution that meets the conditions)*.
  - The last *complementarity condition* that prevents unbounded U^*, V^* is non-linear, turn LP into LCP.
  - If an action is played $x^j > 0$, it has to be a best response: $r_1^j = 0$, otherwise it can just deviate to reach V^*

$$\sum_{k \in A_2} A\left(a_1^j, a_2^k\right) \cdot y^k + r_1^j = V^* \quad \forall j \in A_1$$

$$\sum_{j \in A_1} B\left(a_1^j, a_2^k\right) \cdot x^j + r_2^k = U^* \quad \forall k \in A_2$$

mutual best response

$$\sum_{j \in A_1} x^j = 1, \sum_{k \in A_2} y^k = 1, \quad x^j \geq 0,\ y^k \geq 0 \quad \forall j \in A_1, \forall k \in A_2$$

valid prob. distirbution

$$r_1^j \geq 0, \quad r_2^k \geq 0 \quad \forall j \in A_1, \forall k \in A_2$$

slack variables

$$r_1^j \cdot x^j = 0, \quad r_2^k \cdot y^k = 0 \quad \forall j \in A_1, \forall k \in A_2$$

complementarity condition

Lemke-Howson Method

Solving the LCP problem through Lemke-Howson:

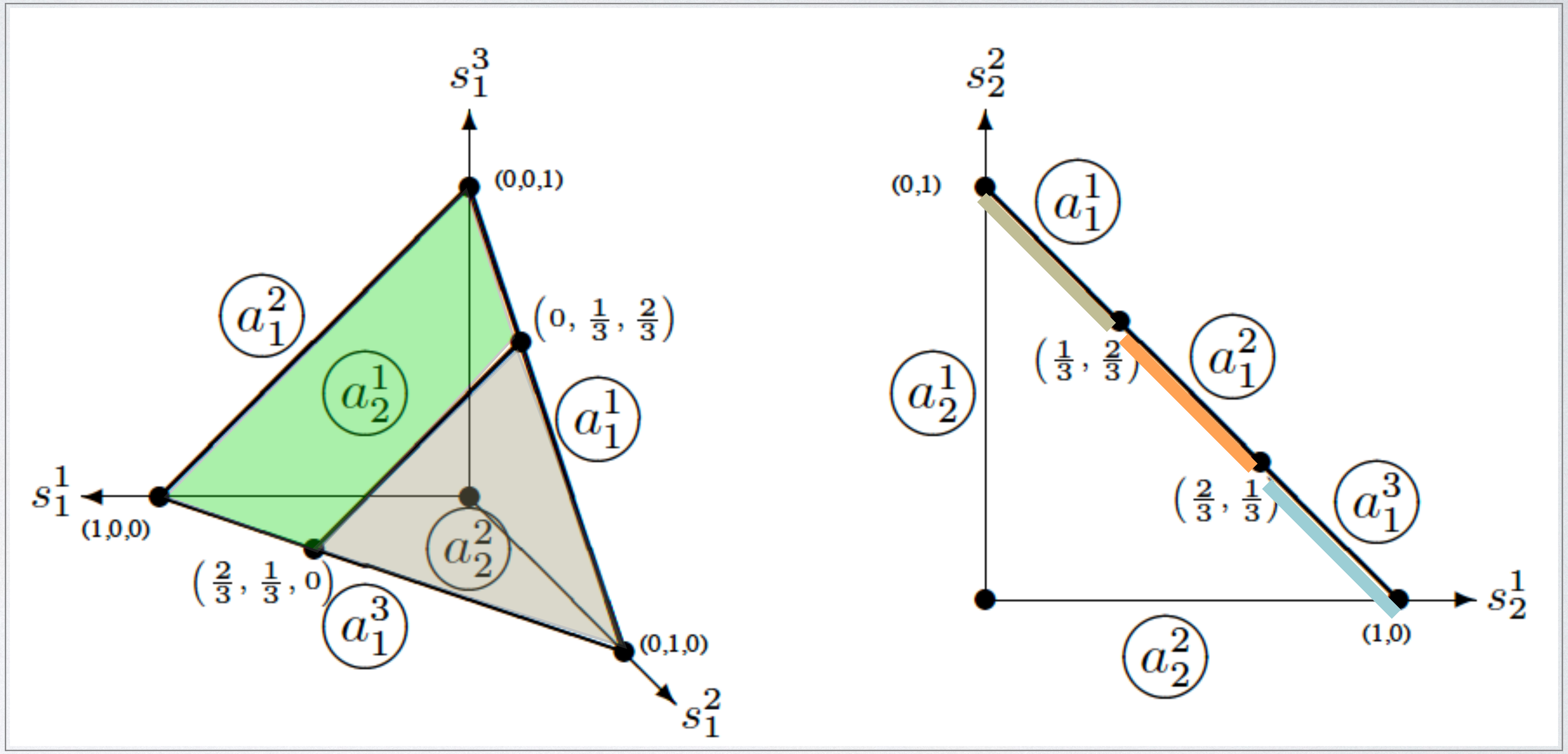
- A classical algorithm that combines game theory, convex analysis and graph theory
- Non-degenerate games: for a pure strategy, there can only be at most one best response.
- Non-degenerate games have the same Nash support size for both players: $|p^*| = |q^*|$.
- Consider polyhedron: $P = \left\{(u, \boldsymbol{x}) \mid x_i \geq 0, \sum x_i = 1, \boldsymbol{x}^T B \leq u \cdot \mathbf{1}\right\}$ and $Q = \left\{(v, \boldsymbol{y}) \mid y_j \geq 0, \sum y_j = 1, A\boldsymbol{y} \leq v \cdot \mathbf{1}\right\}$
  - These polyhedra describes the space of mixed strategies with an upper bound on the best response value from the other player will react.
- Consider the (bounded) polytope: $\bar{P} = \{\boldsymbol{x} \mid x_i \geq 0, \boldsymbol{x}^T B \leq \mathbf{1}\}$ and $\bar{Q} = \left\{\boldsymbol{y} \mid y_j \geq 0, A\boldsymbol{y} \leq \mathbf{1}\right\}$.
  - Setting $U^* = V^* = 1$ is generic, there exist bijective mappings between P, Q and $\bar{P}, \bar{Q}$
- We are interested in the graph that are composed by the corner points of $\bar{P}, \bar{Q}$, because it either means x_i is either in Nash support, or column j is a best response to $\boldsymbol{x}$.

$$L(\boldsymbol{x}) = \{i \mid x_i = 0\} \cup \left\{j \mid \left(\boldsymbol{x}^T B\right)_j = 1\right\} \qquad L(\boldsymbol{y}) = \left\{j \mid y_j = 0\right\} \cup \left\{i \mid \left(A\boldsymbol{y}\right)_i = 1\right\}$$

Lemke-Howson Method Example

$$L(\mathbf{x}) = \left\{ i \mid x_i = 0 \right\} \cup \left\{ j \mid \left(\mathbf{x}^T B \right)_j = 1 \right\} \qquad L(\mathbf{y}) = \left\{ j \mid y_j = 0 \right\} \cup \left\{ i \mid \left(A\mathbf{y} \right)_i = 1 \right\}$$

0, 1	6, 0
2, 0	5, 2
3, 4	3, 3



<Multiagent Systems: Algorithm, Game Theoretic and Logic Foundation, page 95>

Lemke-Howson Method

Solving the LCP problem through Lemke-Howson:

- We are interested in the graph that are composed by the corner points of $\bar{P}, \bar{Q}$, because it either means x_i is not in Nash, or opponent's column j is a best response to $\boldsymbol{x}$. Let's label those corner points by the constraint id:

$$L(\boldsymbol{x}) = \{i \mid x_i = 0\} \cup \left\{j \mid \left(\boldsymbol{x}^T B\right)_j = 1\right\} \qquad L(\boldsymbol{y}) = \left\{j \mid y_j = 0\right\} \cup \left\{i \mid \left(A\boldsymbol{y}\right)_i = 1\right\}$$

$\boxed{\textbf{Theorem: } \text{a pair } (\boldsymbol{x}, \boldsymbol{y}) \text{ is a Nash equilibrium if and only if there are completely labelled: } L(\boldsymbol{x}) \cup L(\boldsymbol{y}) = M + N}$

- Proof: $\Longleftarrow$: if $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash, then x_i is either 0, thus $i \in L(\boldsymbol{x})$, or, is a best response to $(Ay)_i = 1$, and thus $i \in L(\boldsymbol{y})$. Therefore, every label appears only once in $L(\boldsymbol{x}) \cup L(\boldsymbol{y})$, so $L = L(\boldsymbol{x}) \cup L(\boldsymbol{y})$.

  $\Longrightarrow$: if $L = L(\boldsymbol{x}) \cup L(\boldsymbol{y})$, by definition, we can know that the following set is a Nash.

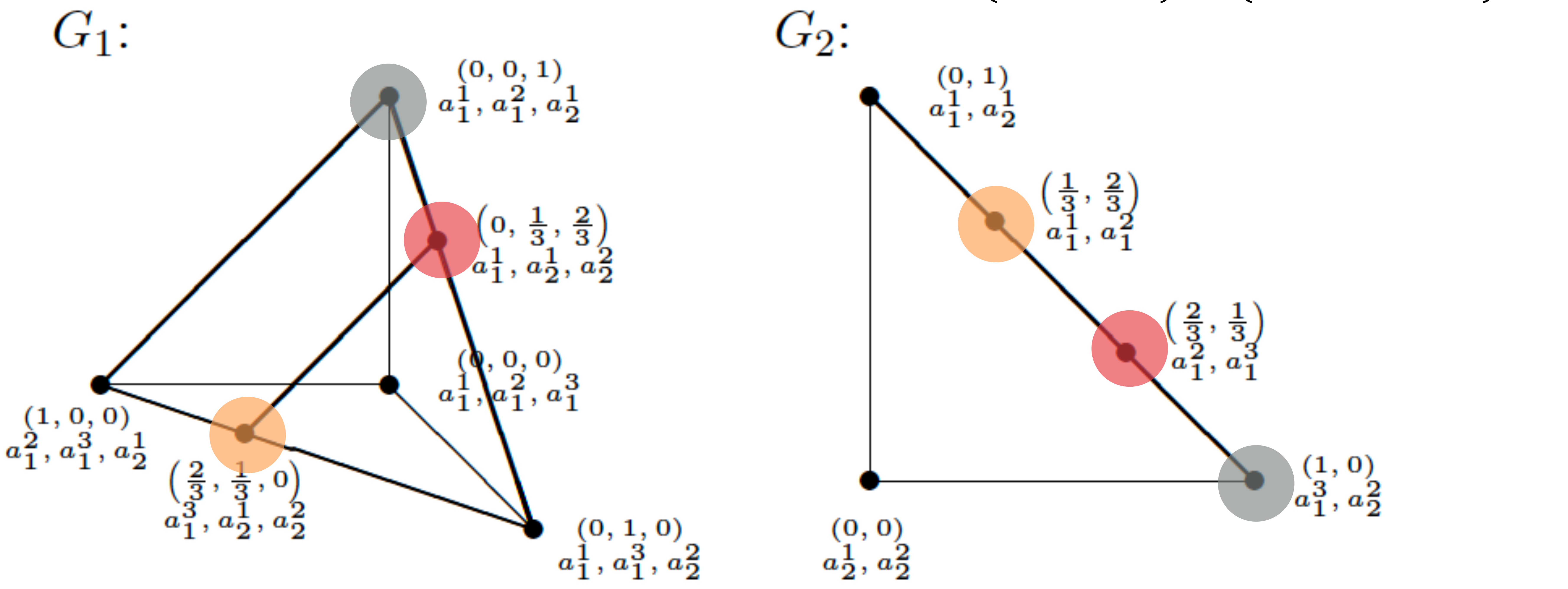
$$\left(\left\{i \in [N] : \sum_{j=1}^{M} y_j A_{ij} = 1\right\} \quad (i \in L(y)), \quad \left\{j \in [M] : \sum_{i=1}^{N} x_i B_{ij} = 1\right\} \quad (j \in L(x))\right)$$

- Intuition: finding Nash is just find vertices of $\bar{P}, \bar{Q}$ that contains all of the labels.

Lemke-Howson Method

Theorem: a pair $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash equilibrium if and only if there are completely labelled: $L(\boldsymbol{x}) \cup L(\boldsymbol{y}) = M + N$

$$L(\boldsymbol{x}) = \{i \mid x_i = 0\} \cup \left\{j \mid \left(\boldsymbol{x}^T B\right)_j = 1\right\} \qquad L(\boldsymbol{y}) = \left\{j \mid y_j = 0\right\} \cup \left\{i \mid \left(A\boldsymbol{y}\right)_i = 1\right\}$$



<Multiagent Systems: Algorithm, Game Theoretic and Logic Foundation, page 95>

Lemke-Howson Method

- Lemke-Howson Method:
  - Intuition: finding Nash is just find vertices of $\bar{P}, \bar{Q}$ that contains all of the labels.
  - In order to find the completely labelled pairs (the Nash), we define:
    - $G = G_1 \times G_2$, with vertices $v = \left(v_1, v_2\right)$ where $v_1 \in V\left(G_1\right)$ and $v_2 \in V\left(G_2\right)$.
    - Edge of $\mathbf{E}(G) = \left\{(v_1, v_2), (v_1', v_2) \in G \times G: \text{ if } (v_1, v_1') \in G_1\right\} \cup \left\{(v_1, v_2), (v_1, v_2') \in G \times G: \text{ if } (v_2, v_2') \in G_2\right\}$
    - $L(v) = L\left(v_1\right) \cup L\left(v_2\right)$ for each vertex $v = \left(v_1, v_2\right) \in V(G)$
    - Let's focus on the subgraph in G that is almost fully labelled but only lack the label k
    - For $k \in M \cup N$, we write $U_k = \{v \in V(G) \mid L(v) \supseteq M \cup N \backslash \{k\}\}$ for the subset of vertices except k.
  - Intuition: Nash equilibrium in U_k has degree only equal to 1.
    - Nash is exactly $M \cup N$, so are in U_k $\forall k$.
    - Since Nash is a fully labelled point and $L(v_1) \cap L(v_2) = \varnothing$, thus Nash in U_k can only have degree 1 (either best response or not in the Nash support, cannot have both !).
    - In U_k, the degree of every other vertex apart from Nash have degree 2 (both in G_1 and G_2).
    - An effective algorithm: walk along the edges in G, and drop the nodes with degree of 2.

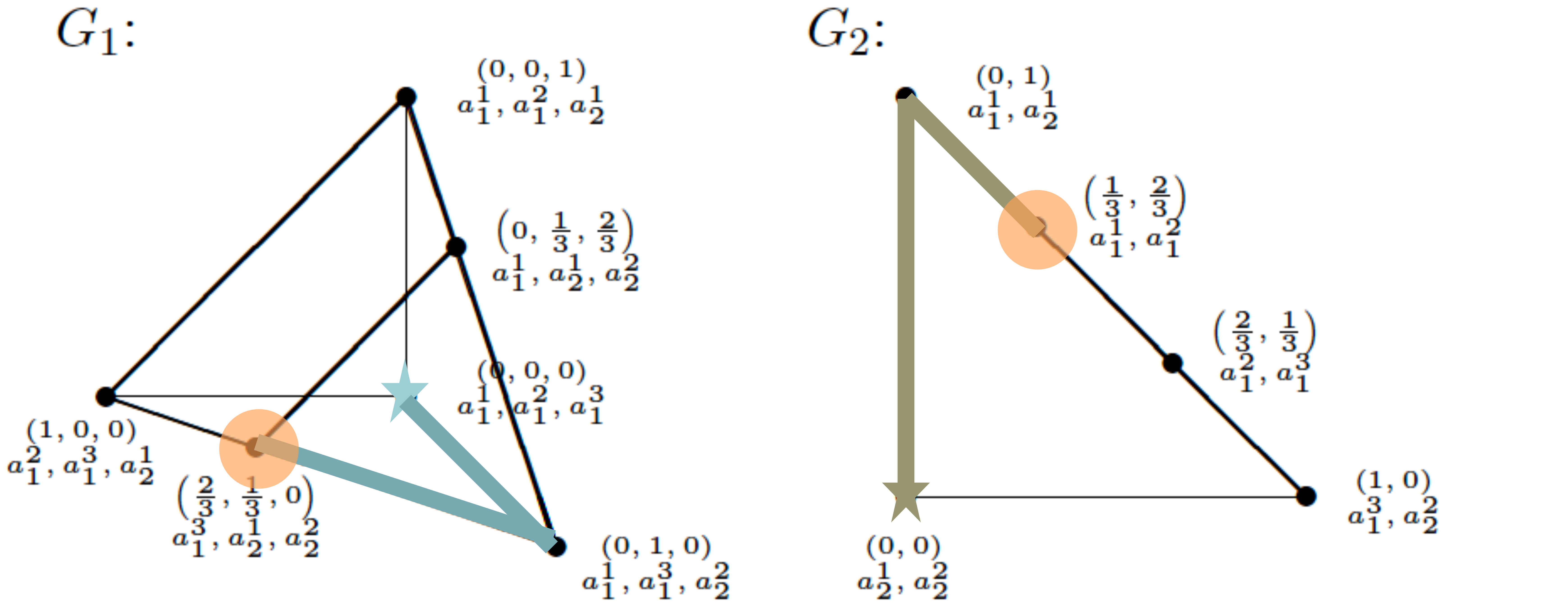
Lemke-Howson Method

Theorem: a pair $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash equilibrium if and only if there are completely labelled: $L(\boldsymbol{x}) \cup L(\boldsymbol{y}) = M + N$

- An effective algorithm: walk along the edges in G, and drop the nodes with degree of 2.

$$L(\boldsymbol{x}) = \{i \mid x_i = 0\} \cup \left\{j \mid \left(\boldsymbol{x}^T B\right)_j = 1\right\}$$

$$L(\boldsymbol{y}) = \left\{j \mid y_j = 0\right\} \cup \left\{i \mid \left(A\boldsymbol{y}\right)_i = 1\right\}$$



Start: (0,0,0) (0,0)

(0,0,0) → (0,1,0), drop a_1^2, pick a_2^2

(0,0) → (0,1), drop a_2^2, pick a_1^1

(0,1,0) → (2/3,1/3,0), drop a_1^1, pick a_2^1

(0,1) → (1/3, 2/3), drop a_2^1, pick a_1^1

Terminate: because now we have $\{a_1^3, a_2^1, a_2^2\} \cup \{a_1^1, a_1^2\}$ fully labelled

Output: (2/3,1/3,0), (1/3, 2/3)

Lemke-Howson Methods

Algorithm (Lemke-Howson)
Input: A Non-degenerate bimatrix game (A, B).
Output: One Nash equilibrium of the game.

1. Choose $k \in M \cup N$.
2. Start with $(x, y) = (0, 0) \in G$. Drop label k from (x, y) (from $x \in \overline{P}$ if $k \in M$, from $y \in \overline{Q}$ if $k \in N$).
3. Let (x, y) be the current vertex. Let l be the label that is picked up by dropping label k. If $l = k$, terminate and (x, y) is a Nash equilibrium of the game. If $l \neq k$, drop l in the other polytope and repeat this step.

Remarks:

- Traversing the whole graph G to add missing labels and drop duplicate labels (i.e. pivoting) until we find one set of nodes that are fully labelled.
- Guarantee to find one Nash, but not all of them! Determining whether all Nash are found is not even NP.
- Exponential time complexity because G has the number of vertices that is exponential in n, m (combinatorial many)
- Because LCP has no objective, we cannot tell the progress before finding a solution.
- Corollary: almost all games have an odd number of Nash equilibrium.

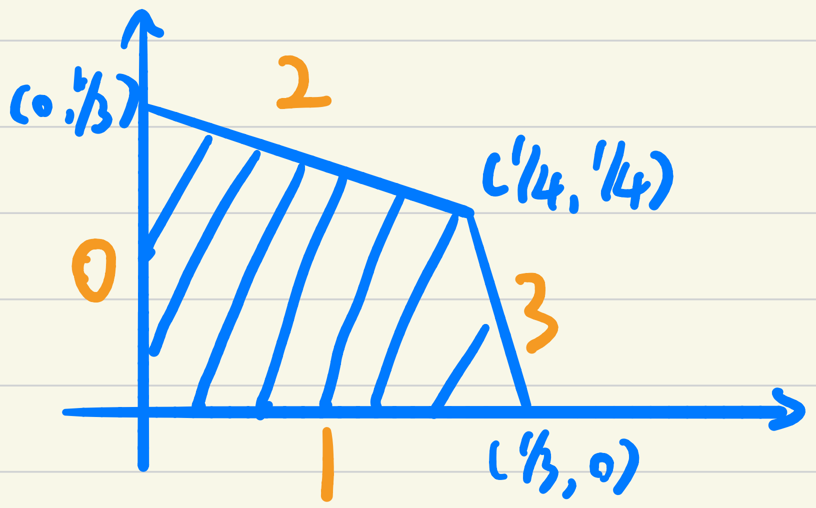
$A = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$ $B = \begin{pmatrix} 1 & 3 \\ 3 & 1 \end{pmatrix}$

1) construct the polytope

$\overline{P} = \{x \in \mathbb{R}^2 \mid x_i \geq 0, x^T B \leq 1\}$.

$\begin{cases} x_1 \geq 0 \\ x_2 \geq 0 \\ x_1 + 3x_2 \leq 1 \\ 3x_1 + x_2 \leq 1 \end{cases} \Rightarrow V = \{(0,0), (0, \tfrac{1}{3}), (\tfrac{1}{3}, 0), (\tfrac{1}{4}, \tfrac{1}{4})\}$.

2) Label the vertex

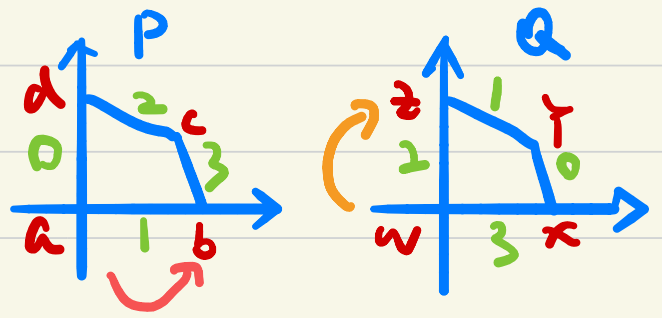


0 : $x_1 = 0$
1 : $x_2 = 0$
2 : $x_1 + 3x_2 = 1$ — when y_1 is the best response to x
3 : $3x_1 + x_2 = 1$ — when y_2 is the best response to x

$\overline{P}$: $(0,0)$: $\{0,1\}$ $(0, \tfrac{1}{3})$: $\{0,2\}$
$(\tfrac{1}{3}, 0)$: $\{1,3\}$ $(\tfrac{1}{4}, \tfrac{1}{4})$: $\{2,3\}$.

$\overline{Q}$: $(0,0)$: $\{2,3\}$ $(\tfrac{1}{4}, \tfrac{1}{4})$: $\{0,1\}$
$(\tfrac{1}{3}, 0)$: $\{0,3\}$ $(0, \tfrac{1}{3})$: $\{1,2\}$.

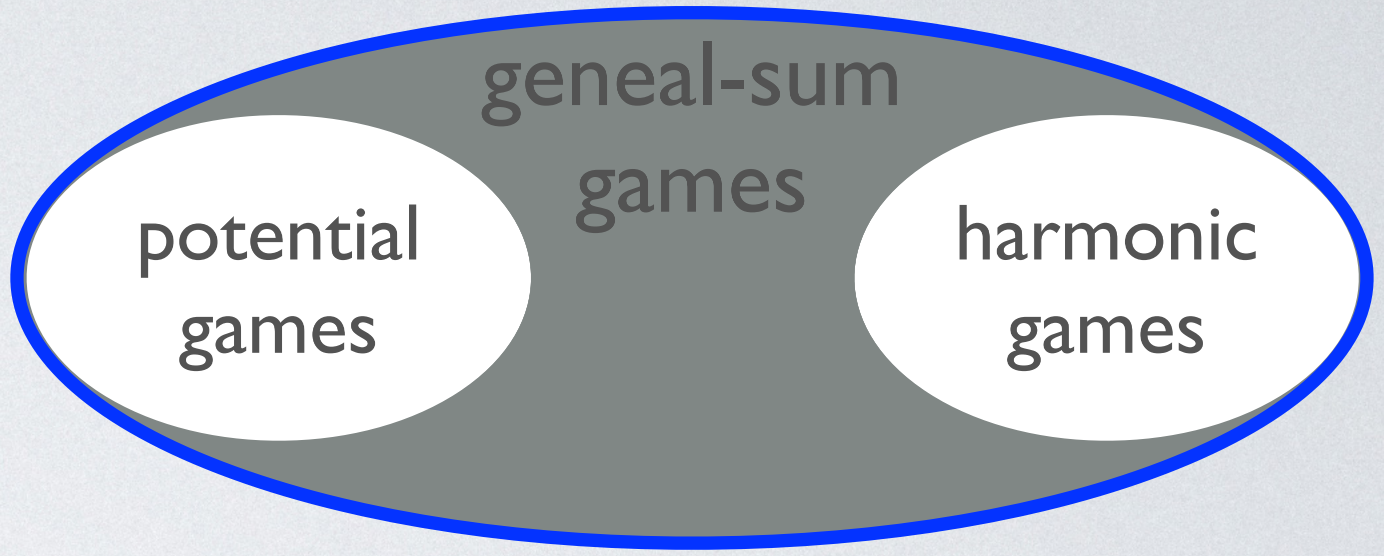
3) Lemke-Howson step.



Start: (a, w) : $\{0,1,2,3\}$. drop 0 from P, pick 3.
(b, w) : $\{1,2,3\}$. drop 3 from Q, pick 1
(b, z) : $\{1,2,3\}$ drop 1 from P, pick 2
(c, z) : $\{1,2,3\}$ drop 2 from Q, pick 0
(c, y) : $\{0,1,2,3\}$. Terminate
output: Normalized $(c, y) := (\tfrac{1}{2}, \tfrac{1}{2}), (\tfrac{1}{2}, \tfrac{1}{2})$

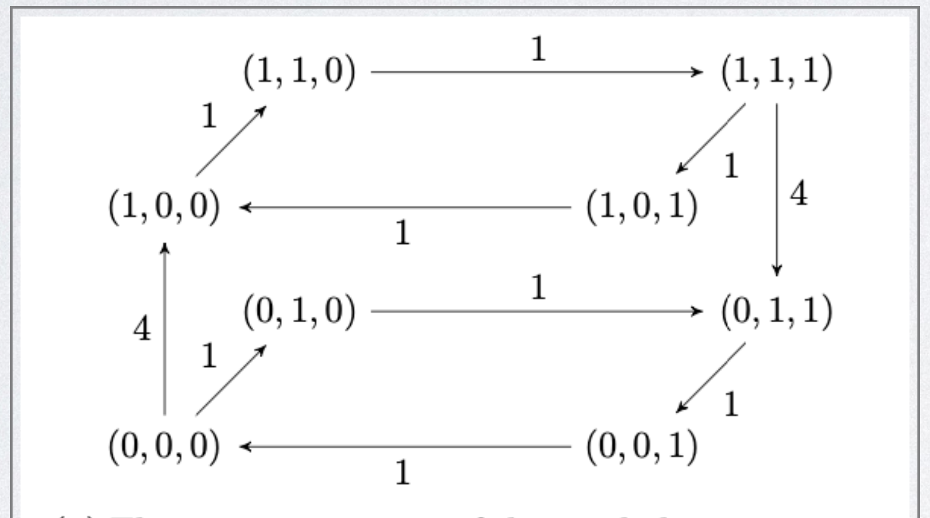
Potential Games

- Potential Game is the other side of the coin where we know how to solve Nash.
- A general game decomposition results suggest that

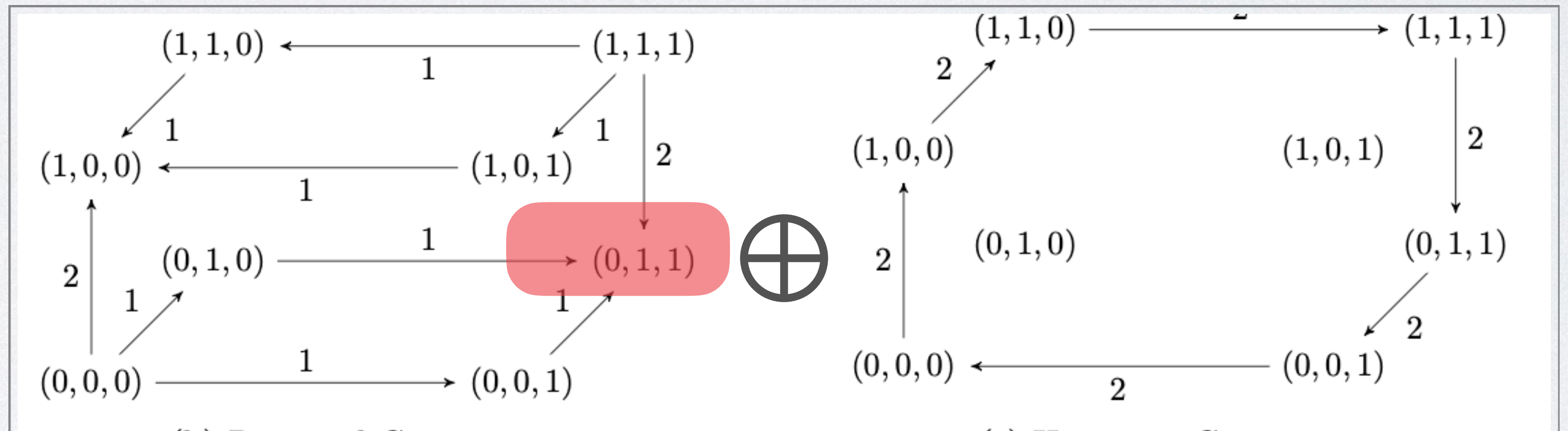


[Candogan 2010] $\boxed{\text{Normal-form Game} = \text{Potential Game} \oplus \text{Harmonic Game}}$

- Potential games has no curl, harmonic games has no divergence.


(a) Flow representation of the road-sharing game.

=


(b) Potential Component. $\oplus$ (c) Harmonic Component.

- In bimatrix games, potential game is related to the identical interest game, harmonic game is related to the zero-sum games.
- Since one can always write $(\mathbf{A}, \mathbf{B}) = \underbrace{\left(\frac{\mathbf{A}-\mathbf{B}}{2}, \frac{\mathbf{B}-\mathbf{A}}{2}\right)}_{\text{zero-sum}} \oplus \underbrace{\left(\frac{\mathbf{A}+\mathbf{B}}{2}, \frac{\mathbf{B}+\mathbf{A}}{2}\right)}_{\text{team-game}}$

Potential Games

- Potential Game are those games that can be described a potential function:

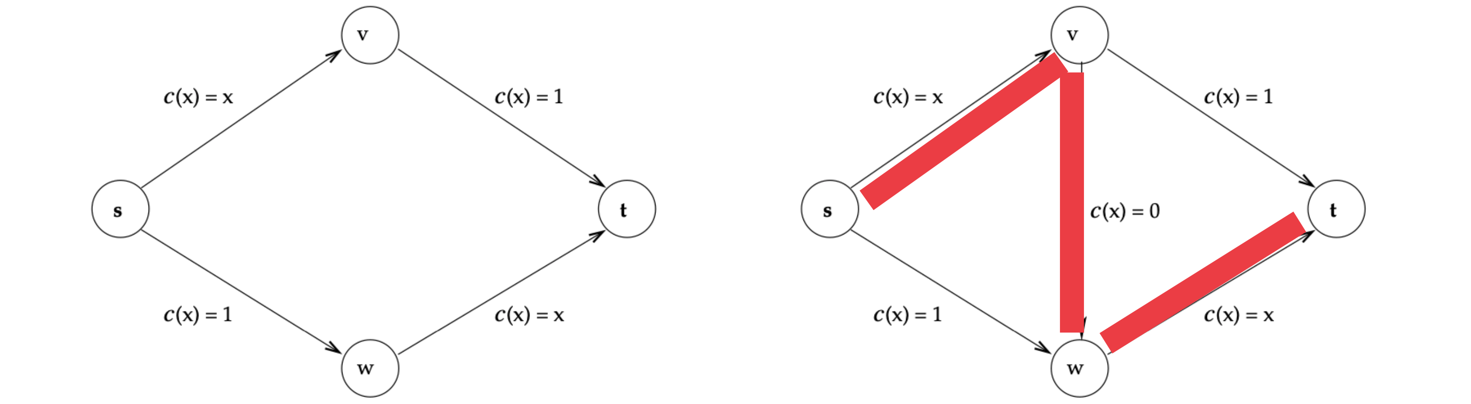
$$\Phi\left(s_{-i}, s_i'\right) - \Phi(s) = C_i\left(s_{-i}, s_i'\right) - C_i(s), \forall i \in [n], s \in S, s_i' \in S_i$$

(0, 0)	(1, 2)
(2, 1)	(0, 0)

Φ

0	2
2	1

- The outcome of any player's any possible deviation are captured by the change in Φ.
- Knowing the location deviations from a global viewpoint.
- Examples of potential games:
    - All **fully cooperative games** are the potential game. Potential is the reward function.
    - The **routing game** we have seen is a potential game.
        - Each player choose a path P_i (e.g. s->v->t)
        - The cost is then $C_i(f) = \sum_{e \in P_i} c_e\left(f_e\right),\ \ f_e = \left|\left\{i : e \in P_i\right\}\right|$ is the number of players using edge e.
        - We can find a potential function $\Phi(f) = \sum_{e \in E} \sum_{j=1}^{f_e} c_e(j)$ such that $C_i(\hat{f}) - C_i(f) = \Phi(\hat{f}) - \Phi(f)$.
        - All potential games are also routing games.



(a) Initial network (b) Augmented network

Figure 1: Braess's Paradox. The addition of an intuitively helpful edge can adversely affect all of the traffic.

Best Response Dynamics in Potential Games

- **Theorem: every potential games has a pure Nash equilibrium**
  - proof: $s = \arg\min_{s \in S} \Phi(s)$ is the pure Nash otherwise any player had incentive to deviate would have a smaller value. This proof leads to an interesting learning algorithm that guarantees to convergence.
- Best response dynamics can lead to Nash convergence.

> - While the current outcome $\mathbf{s}$ is not a ϵ-PNE:
>   - Pick an arbitrary player i that has an ϵ-*move* — a deviation s'_i with $C_i(s'_i, \mathbf{s}_{-i}) < (1-\epsilon)C_i(\mathbf{s})$ — and an arbitrary such move for the player, and move to the outcome $(s'_i, \mathbf{s}_{-i})$.

- A polynomial time convergence bound can be proved.

> **Theorem 3.2 (Convergence of ϵ-Best Response Dynamics [2])** *Consider an atomic selfish routing game where:*
>
> 1. *All players have a common source vertex and a common sink vertex.*
> 2. *Cost functions satisfy the "α-bounded jump condition," meaning $c_e(x+1) \in [c_e(x), \alpha \cdot c_e(x)]$ for every edge e and positive integer x.*
> 3. *The MaxGain variant of ϵ-best-response dynamics is used: in every iteration, among players with an ϵ-move available, the player who can obtain the biggest absolute cost decrease moves to its minimum-cost deviation.*
>
> *Then, an ϵ-PNE is reached in $(\frac{k\alpha}{\epsilon} \log \frac{\Phi(\mathbf{s}^0)}{\Phi_{\min}})$ iterations.*

Best Response Dynamics in Potential Games

- Best response dynamics has polynomial time rate.

Theorem 3.2 (Convergence of ϵ-Best Response Dynamics [2]) *Consider an atomic selfish routing game where:*

1. *All players have a common source vertex and a common sink vertex.*
2. *Cost functions satisfy the "α-bounded jump condition," meaning $c_e(x+1) \in [c_e(x), \alpha \cdot c_e(x)]$ for every edge e and positive integer x.*
3. *The MaxGain variant of ϵ-best-response dynamics is used: in every iteration, among players with an ϵ-move available, the player who can obtain the biggest absolute cost decrease moves to its minimum-cost deviation.*

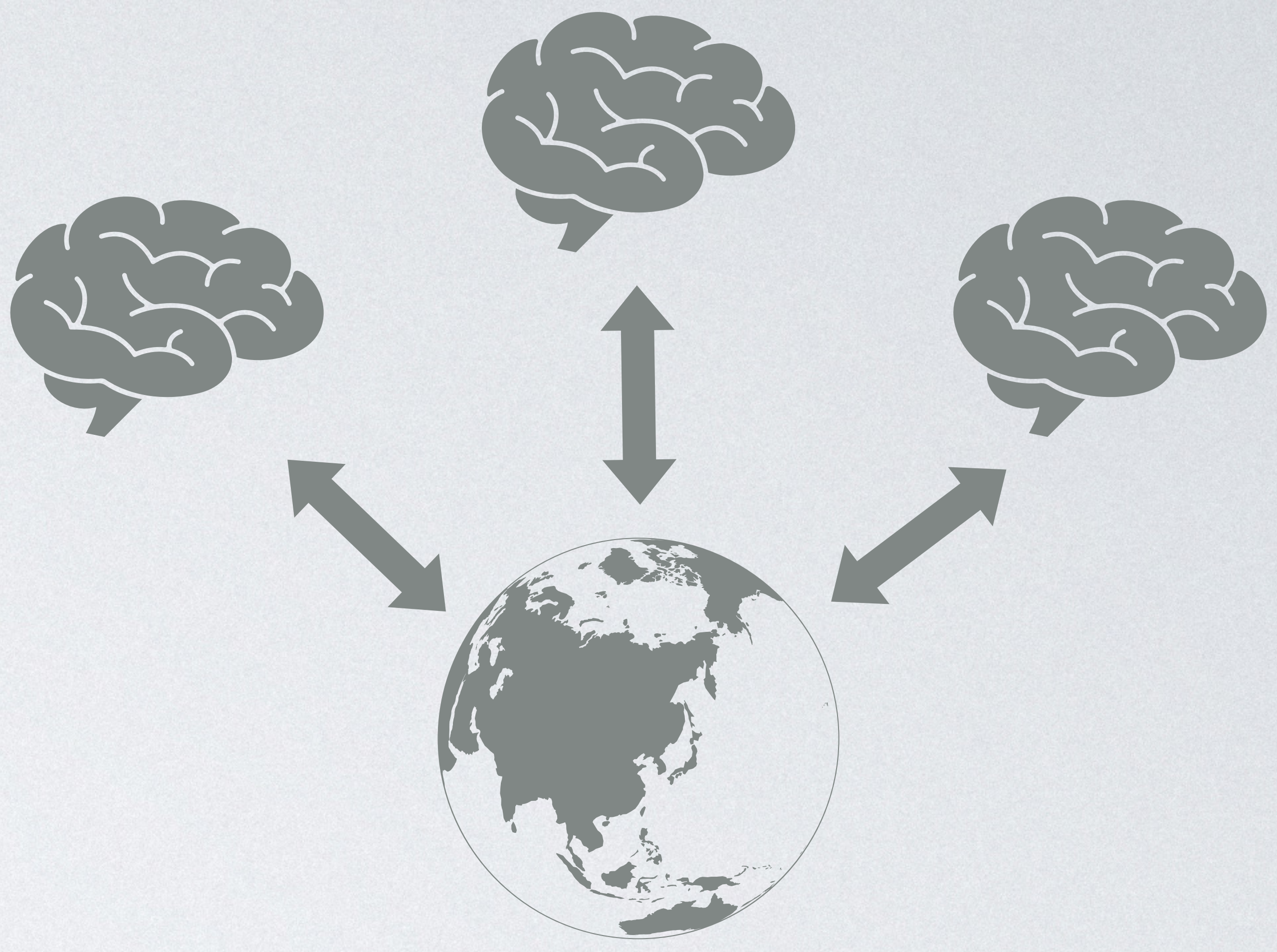
Then, an ϵ-PNE is reached in $(\frac{k\alpha}{\epsilon} \log \frac{\Phi(\mathbf{s}^0)}{\Phi_{\min}})$ iterations.

- Check the proof in [S. Chien 2011].
  - first prove the existence of a player with high cost
  - then prove the player chosen to take best response has cost within an α factor of that of any other player.
  - third we can bound the decrease of potential function for each iteration.
- Note that polynomial in the number of joint strategies (e.g, 2^n for 2 strategy n players)
- If either the assumptions on bounded jump condition, single source and sink, MaxGain is dropped, then it could take exponential number of iterations [A. Skopalik 2008].

Contents

Multi-Agent Reinforcement Learning

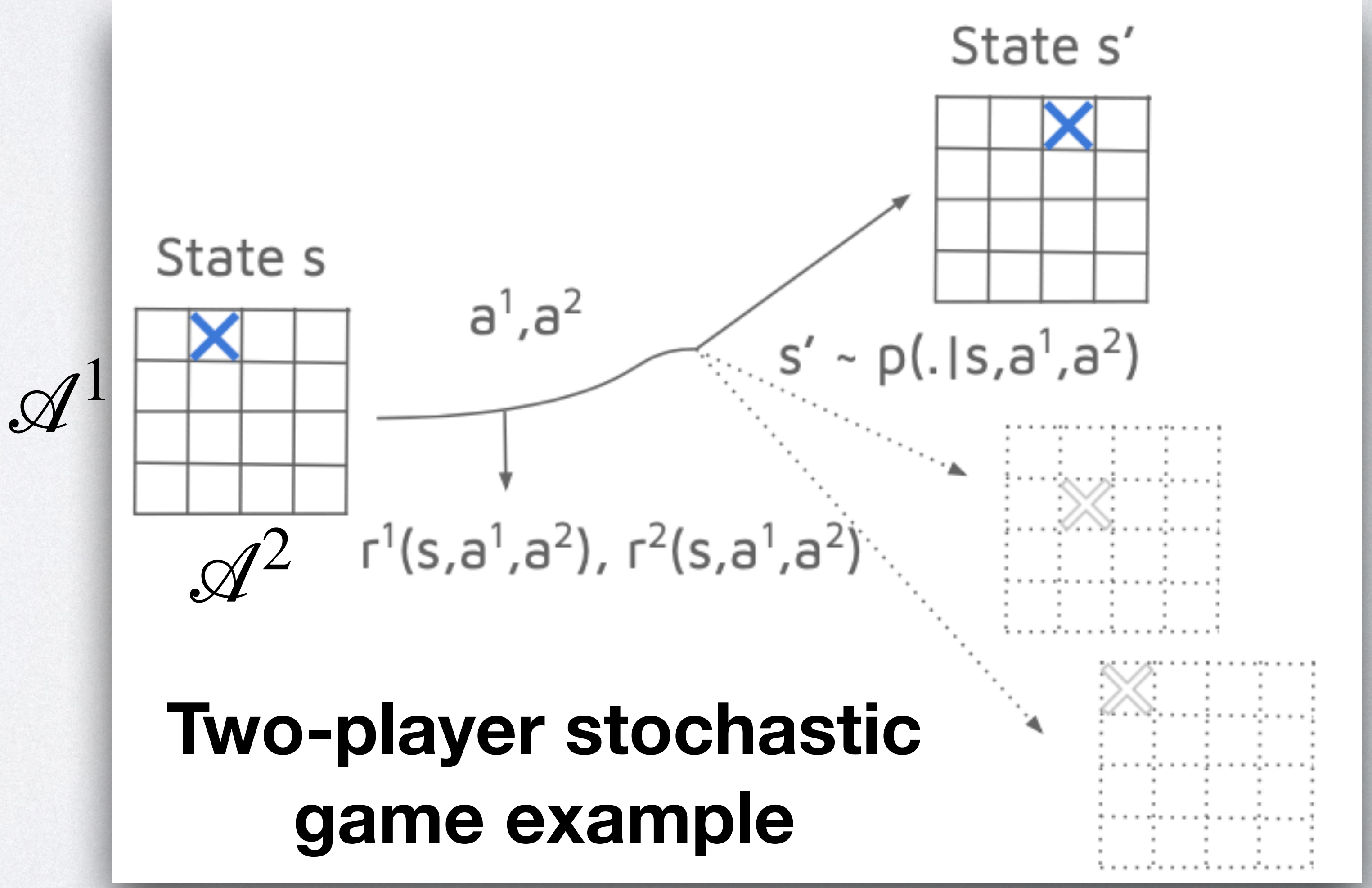
- Modelled by a Stochastic Game $(\mathcal{S}, \mathcal{A}^{\{1,\ldots,n\}}, \mathcal{R}^{\{1,\ldots,n\}}, \mathcal{T}, \mathcal{P}_0, \gamma)$
  - $\mathcal{S}$ denotes the state space,
  - $\boldsymbol{\mathcal{A}}$ is the joint-action space $\mathcal{A}^1 \times \ldots \times \mathcal{A}^n$,
  - $\mathcal{R}^i = \mathcal{R}^i(s, a^i, a^{-i})$ is the reward function for the i-th agent,
  - $\mathcal{T} : \mathcal{S} \times \boldsymbol{\mathcal{A}} \times \mathcal{S} \to [0,1]$ is the transition function based on the joint action,
  - $\mathcal{P}_0$ is the distribution of the initial state, γ is a discount factor.
  - **Special case:** $n = 1 \to$ single-agent MDP, $|\mathcal{S}| = 1 \to$ normal-form game
  - **Dec-POMDP**: assume state is not directly observed, but agents have same reward function.



- Each agent tries to maximise its expected long-term reward:

$$V_{i,\boldsymbol{\pi}}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbf{E}_{\boldsymbol{\pi},\mathcal{P}} \left\{ R_{i,t} \,|\, s_0 = s, \boldsymbol{\pi} \right\}, \boldsymbol{\pi} = [\pi_1, \ldots, \pi_N]$$

$$Q_{i,\boldsymbol{\pi}}(s, \boldsymbol{a}) = R_i(s, \boldsymbol{a}) + \gamma \mathbf{E}_{s' \sim p} \left[V_{i,\boldsymbol{\pi}}(s') \right]$$



Two-player stochastic game example

Multi-Agent Reinforcement Learning

- Value-based method:
  - The sense of optimality changes, now it depends on other agents !

$$Q_{i,t+1}\left(s_k, \boldsymbol{\pi}_t\right) = Q_{i,t}\left(s_t, \boldsymbol{\pi}_t\right) + \alpha\Big[R_{i,t+1} + \gamma \cdot \mathbf{eval}_i\big\{Q_{\cdot,t}(s_{t+1}, \cdot\,)\big\} - Q_{i,t}\left(s_t, \boldsymbol{\pi}_t\right)\Big]$$

$$\pi_{i,t}(s, \cdot\,) = \mathbf{solve}_i\big\{Q_{\cdot,t}(s_t, \cdot\,)\big\}$$

    - **Fully-cooperative game**: agents share the same reward function

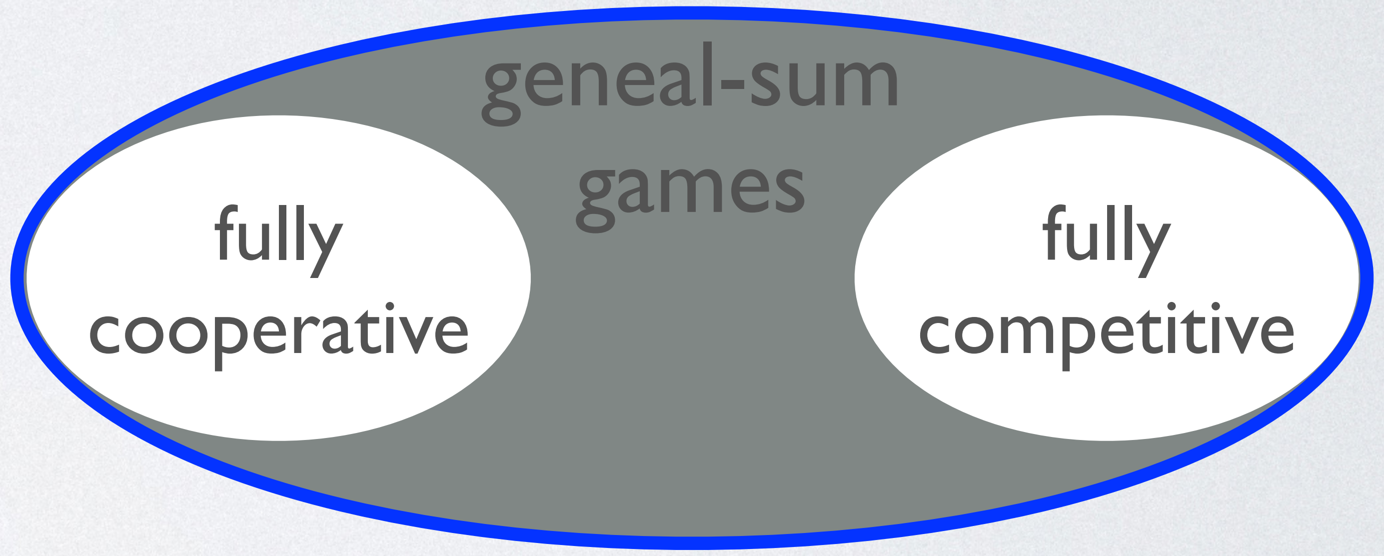
$$\mathbf{eval}_i\big\{Q_{\cdot,t}(s_{t+1}, \cdot\,)\big\} = \max_{\boldsymbol{a}} Q_{i,t}(s_{t+1}, \boldsymbol{a})$$

$$\mathbf{solve}_i\big\{Q_{\cdot,t}(s_t, \cdot\,)\big\} = \arg\max_{a_i}\Big(\max_{a^{-i}} Q_{i,t}(s_t, a_i, a_{-i})\Big)$$

    - **Fully-competitive game**: sum of agents' reward is zero

$$\mathbf{eval}_i\big\{Q_{\cdot,t}(s_{t+1}, \cdot\,)\big\} = \max_{\pi_i}\min_{a_{-i}} \mathbf{E}_{\pi_i}\big[Q_{i,t}(s_t, a_i, a_{-i})\big]$$

$$\mathbf{solve}_i\big\{Q_{\cdot,t}(s_t, \cdot\,)\big\} = \arg\max_{\pi_i}\min_{a_{-i}} \mathbf{E}_{\pi_i}\big[Q_{i,t}(s_t, a_i, a_{-i})\big]$$



  - Assuming agents share the either the same or completely opposite interest is a strong assumption.

Nash Equilibrium to MARL

- Value-based method:

$$\pi_{i,t}(s,\cdot) = \textbf{solve}_i \left\{ Q_{\cdot,t}\left(s_t,\cdot\right) \right\}$$

$$Q_{i,t+1}\left(s_k, \boldsymbol{\pi}_t\right) = Q_{i,t}\left(s_t, \boldsymbol{\pi}_t\right) + \alpha\left[R_{i,t+1} + \gamma \cdot \textbf{eval}_i \left\{ Q_{\cdot,t}\left(s_{t+1},\cdot\right) \right\} - Q_{i,t}\left(s_t, \boldsymbol{\pi}_t\right)\right]$$

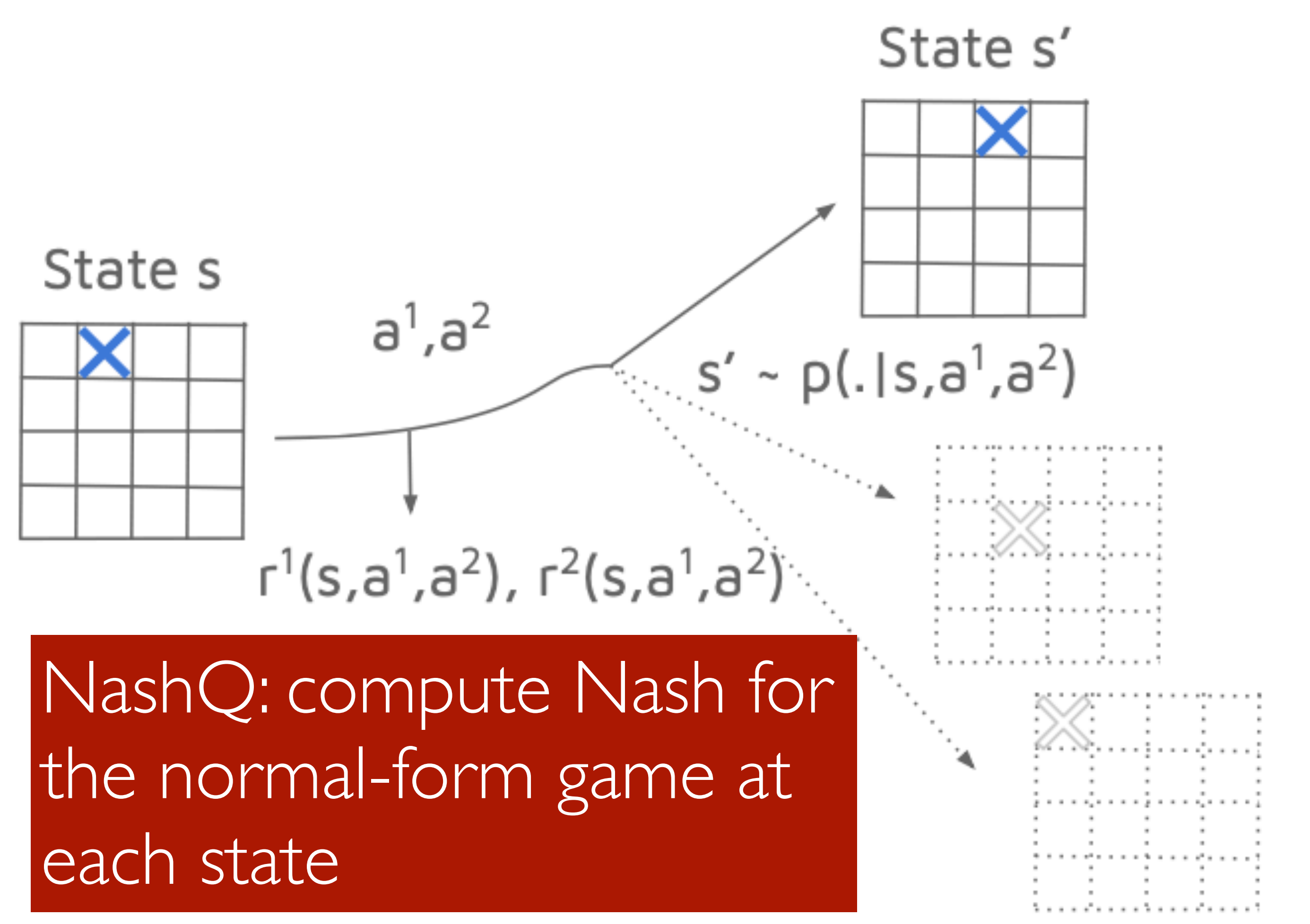
  - Nash-Q Learning [Hu. et al 2003] — Using Nash Equilibrium as the optima to guide agents' policies

    1. Solve the Nash Equilibrium for the current stage game

$$\textbf{solve}_i \left\{ Q \cdot_{\cdot t}\left(s,\cdot\right) \right\} = \textbf{Nash}_i \left\{ Q_{\cdot,t}(s_t,\cdot) \right\}$$

    2. Improve the estimation of the Q-function by the Nash value function.

$$\textbf{eval}_i \left\{ Q_{\cdot,t}(s,\cdot) \right\} = V_i\left(s, \textbf{Nash}\left\{ Q_{\cdot,t}(s_t,\cdot) \right\}\right)$$



  - Nash-Q operator $\mathscr{H}^{\text{Nash}}\mathbf{Q}(s,\mathbf{a}) = \mathbf{E}_{s'}\left[R(s,\mathbf{a}) + \gamma \mathbf{V}^{\text{Nash}}\left(s'\right)\right]$ is a contraction mapping.

MARL in Markov Potential Games

- Applying RL techniques to solve stochastic potential games [Mguni 2021].
- Stochastic potential games, considering state transition, are defined by

$$R_i\left(s,\left(a^i,a^{-i}\right)\right)-R_i\left(s,\left(a'^i,a^{-i}\right)\right)=\phi\left(s,\left(a^i,a^{-i}\right)\right)-\phi\left(s,\left(a'^i,a^{-i}\right)\right),\ \forall i\in[n],\forall s\in S$$

- Mguni found a dual-form MDP where the local optimum in value function corresponds to the Markov Perfect Equilibrium (the Nash equilibrium) of SPG.

the "value" function
(ϕ can also be learned from data)

$$B^{\boldsymbol{\pi}}(s)=\mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t\phi\left(s_t,\boldsymbol{a}_t\right)\mid\boldsymbol{a}_t\sim\boldsymbol{\pi}\right],\forall s\in\mathcal{S},\forall\boldsymbol{\pi}\in\boldsymbol{\Pi}$$

the "TD target"

$$Y_{l_k}\left(s_{l_k},\boldsymbol{a}_{l_k},s'_{l_k}\right):=\phi_{l_k,\hat{\rho}}\left(s_{l_k},\boldsymbol{a}_{l_k}\right)+\sup_{\boldsymbol{a}'}\mathbf{E}_{\mathbb{P}}\left[\hat{B}_l\right]\left(s'_{l_k},\boldsymbol{a}'\right)$$

the "Bellman Error"

$$F_l\in\arg\inf_{\mathbb{F}\in\mathbb{H}}\sum_{l_k=1}^{n_k}\left(Y_{l_k}\left(s_{l_k},\boldsymbol{a}_{l_k},s'_{l_k}\right)-[\mathbb{F}]\left(s_{l_k},\boldsymbol{a}_{l_k}\right)\right)^2$$

the "policy gradient"

$$\nabla\eta_i\hat{B}^{\left(\pi^k_{i,\eta_i},\pi^k_{-i,\eta_{-i}}\right)}\left(s_{l_k}\right)\approx\frac{1}{L}\sum_{l=1}^{L}\nabla_{\eta_i}\pi_{i,\eta_i}\left(\cdot\mid s_{l_k}\right)\nabla_{a^i_{l_k}}F_k\left(s_{l_k},\boldsymbol{a}_{l_k}\right)\Big|_{a^i_{l_k}\sim\pi^k_{i,\eta_i}}$$

- One can apply Q-learning/Actor Critic in the dual MDP to solve for the MPE of SPG.

Complexity of Computing Nash Equilibrium

- Complexity theory 101 — an intuitive explanation:
  - Recall the NP for a decision problem as

Definition 4.2.1 (NP) *A decision problem Q is in NP if there exists a polynomial time algorithm $V(I,X)$ such that*

1. *If I is a YES instance of Q then there exists some X such that $|X|$ is polynomial in $|I|$ and $V(I,X) = YES$*
2. *If I is a NO instance of Q then $V(I,X) = NO$ for all X*

  - But the decision problem of "is there a Nash equilibrium?" is always true proved by Nash himself.
  - We need a new complexity class of Functional NP (FNP) to describe the search problems: not only do solutions have to be verified in P-time, but also to find a solution!
  - Two-player Nash will be FNP because we can check whether Nash is true by checking the best responses.
  - However, two-player Nash will not be FNP-hard. To prove that, we need to show two-player Nash is not FNP-Complete.

Complexity of Computing Nash Equilibrium

- Two-player Nash is not FNP-complete.
  - Completeness is build on the notion of reduction:

Definition 4.2.2 *We say that P reduces to Q (denoted as $P \leq_p Q$) if there exist polynomial-time algorithms A and B such that*

1. *A maps instances of P to instances of Q,*
2. *If I is a YES instance of P than $A(I)$ is a YES instance of Q, and*
3. *If X is a witness for $A(I)$, then $B(x)$ is a witness of I (if I is a YES instance) or NO (if I is a NO instance).*

  - If we want to solve P, it suffices to solve Q: to solve instance I of problem P, we can first find a solution of X of A(I), which is of Q, and then use B to find a solution of B(X) of I.
  - **Theorem: Two-player Nash is not FNP-complete**
  - We can proof that if two-player Nash is FNP-hard then NP=coNP (verifying "No" instance in P-time).
    - Proof by showing that if true, we can find a certificate of NO instances for SAT problems.
    - SAT problem: find a|b such that "a AND NOT b" is satisfied. SAT is known to be NP-complete.
    - But, we know that NP≠coNP.

Complexity of Computing Nash Equilibrium

- We need a new class that has complete problems for the search tasks.
  - Polynomial Parity Argument Directed (PPAD): the class of search problems where the existence of a solution and an algorithm to find one are guaranteed by the properties that we have seen in the Lemke-Howson methods.
    - finite graph, vertex has at most degree 2, every source has valid solution (e.g., Nash in $U_k, \forall k$)
    - A bit chicken-egg here: we want to describe the complexity of a problem, but now we say all problems that look like this form the class of new complexity
  - We know PPAD problems can always have exponential-time algorithms, but can we have P-time solutions?
    - Short answer is we don't know yet. Similar to we don't know if P=NP.
    - But highly likely NO.
  - **Theorem: Two-player Nash is PPAD-complete.**

Complexity of Multi-Agent Learning

- Solving Nash Equilibrium is very challenging !
  - The solution concept of Nash comes from game theory but it is not their main interest to find solutions.
  - Complexity of solving two-player Nash is PPAD-Hard (intractable unless P=NP).
  - How to scale up multi-agent solution is open-question.
  - Approximate solution is still under development.

$$R_i\left(a_i, a_{-i}\right) \geq R_i\left(a_i', a_{-i}\right) - \epsilon$$

$$\epsilon = .75 \rightarrow .50 \rightarrow .38 \rightarrow .37 \rightarrow .3393$$ [Tsaknakis 2008]

  - Equilibrium selection is problematic, how to coordinate agents to agree on Nash during training is unknown.
  - Nash equilibrium assumes perfect rationality, but can be unrealistic in the real world.

- More complexity results of solving Nash [Shoham 2007, sec 4][Conitzer 2002]
  - Two-player general-sum normal-form game:
    - Compute NE → PPAD-Hard
    - Count number of NE → #P-Hard
    - Check uniqueness of NE → NP-Hard
    - Guaranteed payoff for one player → NP-Hard
    - Guaranteed sum of agents payoffs → NP-Hard
    - Check action inclusion / exclusion in NE → NP-Hard
  - Stochastic game:
    - Check pure-strategy NE existence → PSPACE-Hard
    - Best response for arbitrary strategy → Not Turing-computable, even can not be implemented by a Turing PC.
    - It holds for two-player symmetrical game with finite time length.

Complexity of Multi-Agent Learning

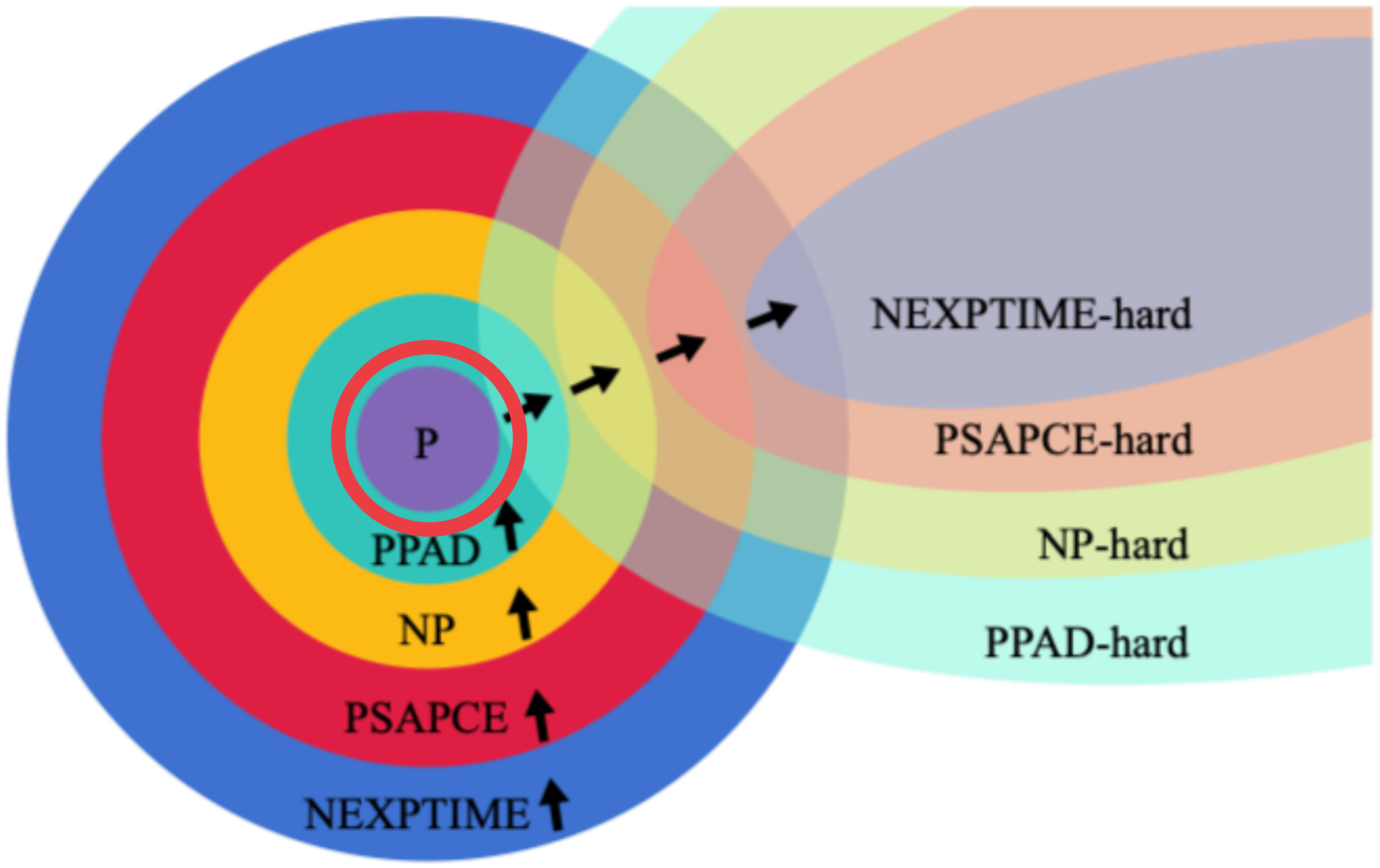


Figure 1.5: Landscape of different complexity classes. Relevant examples are: 1) solving NE in two-player zero-sum game is *P* (Neumann, 1928). 2) solving NE in two-player general-sum game is *PPAD*-hard (Daskalakis et al., 2009). solving NE in three-player zero-sum game is also *PPAD*-hard (Daskalakis and Papadimitriou, 2005). 3) checking the uniqueness of NE is *NP*-hard (Conitzer and Sandholm, 2002). 4) checking whether pure-strategy NE exists in stochastic game is *PSPACE*-hard (Conitzer and Sandholm, 2008). 5) solving Dec-POMDP is *NEXPTIME*-hard (Bernstein et al., 2002).

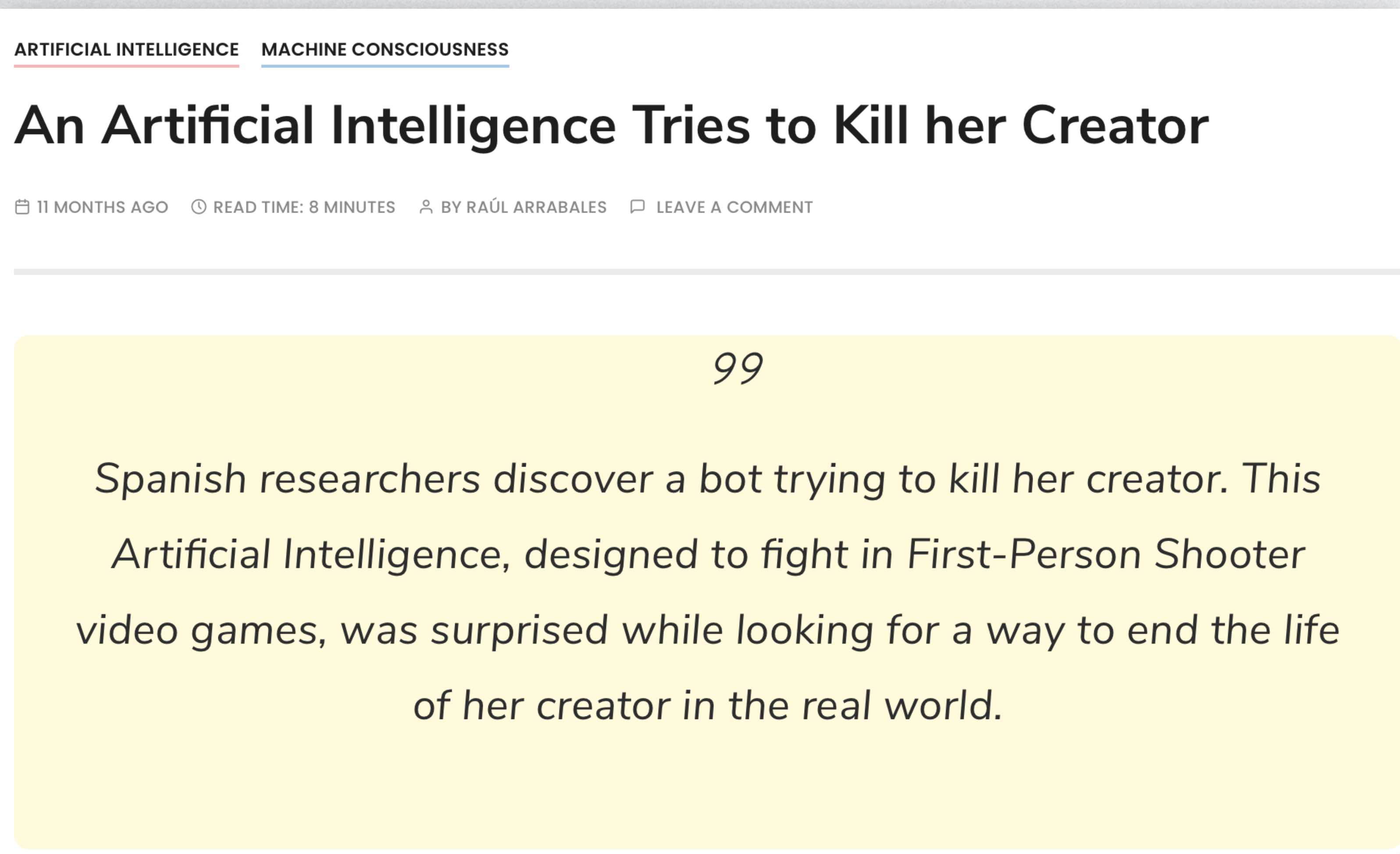
What is Next?

- We start from two-player zero-sum games, where solving Nash is P.
    - fictitious play, double oracle, PSRO
- We look at potential games, where pure Nash exists and solving it is P.
    - best response dynamics has polynomial bound.
    - solving stochastic potential games through MARL.
- We move onto two-player general-sum games, where solving Nash is PPAD-hard.
    - support enumeration, Lemke-Howson
    - connections to MARL
    - two-player Nash is PPAD-hard, MARL is even harder

So far, it is all about Nash. But it could be a dead-end for general cases !

MARL in Reality

what you Mum thinks


ARTIFICIAL INTELLIGENCE MACHINE CONSCIOUSNESS

An Artificial Intelligence Tries to Kill her Creator

11 MONTHS AGO · READ TIME: 8 MINUTES · BY RAÚL ARRABALES · LEAVE A COMMENT

99

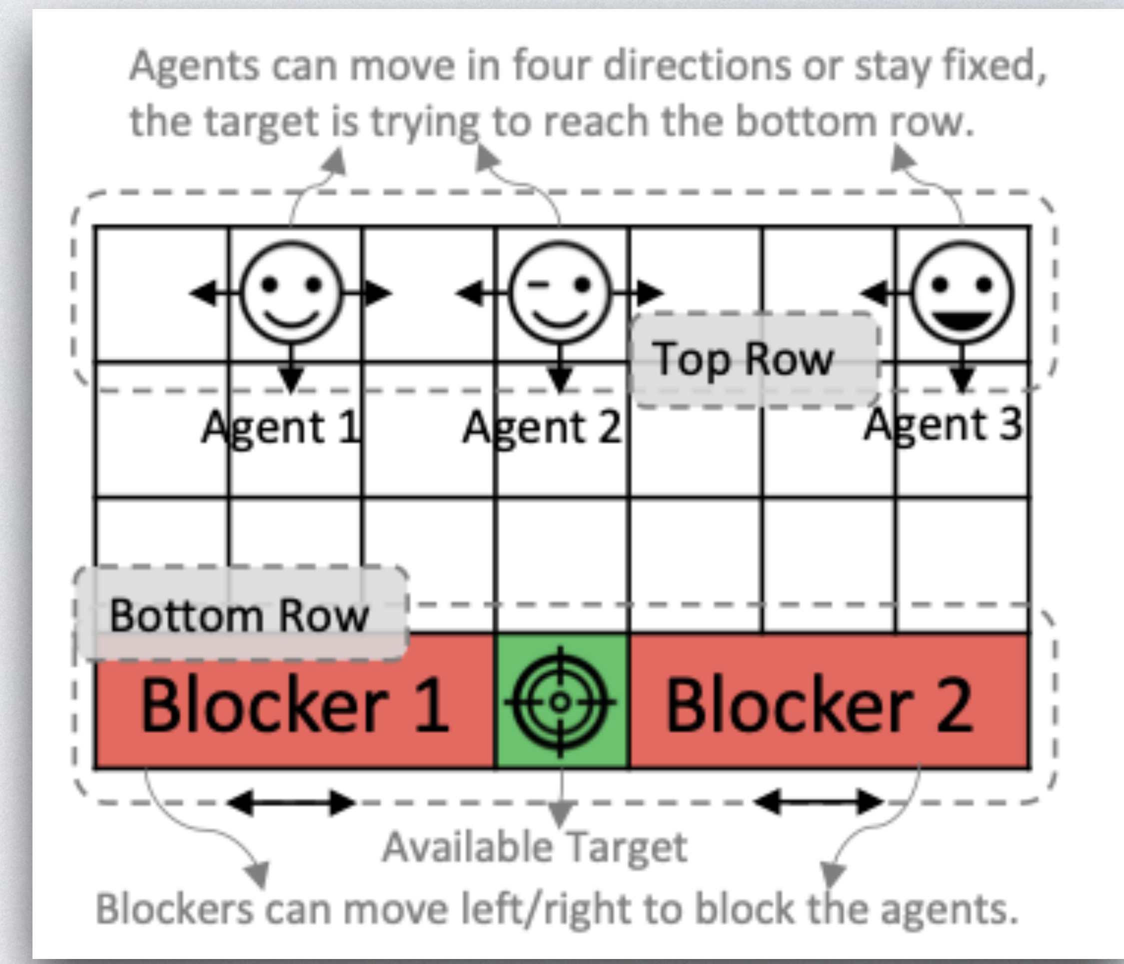
Spanish researchers discover a bot trying to kill her creator. This Artificial Intelligence, designed to fight in First-Person Shooter video games, was surprised while looking for a way to end the life of her creator in the real world.

Something undescribable :)

what you think you are doing

Multi-player general-sum games with high-dimensional continuous state-action space

what you are actually doing



Two-player discrete-action game in a grid world.

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Artificial Intelligence 171 (2007) 365–377

Artificial Intelligence

www.elsevier.com/locate/artint

If multi-agent learning is the answer, what is the question?

Yoav Shoham *, Rob Powers, Trond Grenager

Department of Computer Science, Stanford University, Stanford, CA 94305, USA

Received 8 November 2005; received in revised form 14 February 2006; accepted 16 February 2006

Available online 30 March 2007

"For the field to advance one cannot simply define arbitrary learning strategies, and analyse whether the resulting dynamics converge in certain cases to a Nash equilibrium or some other solution concept of the stage game. This in and of itself is not well motivated."

Other Necessary Solution Concepts

- Nash Equilibrium

Definition 5.3.1 *Let σ_i be a distribution over S_i for all $i \in [k]$. Let $\sigma = \sigma_1 \times \sigma_2 \times \cdots \times \sigma_k$ be the product distribution over S defined by the individual player distributions. Then σ is a* mixed Nash equilibrium *if*

$$\mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s)] \leq \mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s_{-i}, s_i')]$$

for all $i \in [k]$ and for all $s_i' \in S_i$.

- Correlated Equilibrium

Definition 5.3.2 *Let σ be a distribution over $S = S_1 \times \cdots \times S_k$. Then σ is a* correlated equilibrium *if*

$$\mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s)|s_i] \leq \mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s_{-i}, s_i')|s_i]$$

for all $i \in [k]$ and for all $s_i, s_i' \in S_i$.

- Coarse Correlated Equilibrium

Definition 5.3.3 *Let σ be a distribution over $S = S_1 \times \cdots \times S_k$. Then σ is a* coarse correlated equilibrium *if*

$$\mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s)] \leq \mathop{\mathbf{E}}_{s\sim\sigma}[c_i(s_{-i}, s_i')]$$

for all $i \in [k]$ and for all $s_i' \in S_i$.

NE ⊂ CE ⊂ CCE

Other Necessary Solution Concepts

- **NE ⊂ CE ⊂ CCE**
- Distinctions:
    - Nash requires a product of individual distribution: $\sigma = \sigma_1 \times \sigma_2 \times \cdots \times \sigma_k$
    - CE/CCE requires arbitrary distribution over the joint strategy set: $\sigma \in \Delta_{S=S_1\times\cdots\times S_k}$
    - CE: assuming a third party draw $s \sim \sigma$, and privately tell each player his s_i but not s_{-i}, and each player knows σ, now he decides whether to deviate to achieve $\mathbf{E}_{s\sim\sigma}\left[c_i\left(s_{-i}, s_i'\right) \mid s_i\right]$
    - In the example of

	Yield	Rush
Yield	(0, 0)	(1, 2)
Rush	(2, 1)	(0, 0)

:

        - (yield, rush) and (rush, yield) are Nash, but sticking to them will block one side forever.
        - but we cannot let them randomly play 50%-50%, because 25% chance they will crash !
        - Instead, we want $\boldsymbol{\sigma}$(yield, stop) = $\boldsymbol{\sigma}$(stop, yield) = 1/2.
        - When the green light is on, we know others will yield to us. Both of us have no motivations to violate.
    - CCE vs CE: player needs to decide whether to deviate even before being told s_i.
        - If we have no incentive to deviate no matter what we're told, then we will not deviate even if s_i is unknown.

Contents

Two Mainstreams of Multi-Agent Learning algorithms

Best response methods:

- Fictitious play, double oracle, PSRO series, …
- Regard the opponents fixed and seek for best responses.
- Easily and nicely integrated with RL methods (e.g., NFSP, PSRO)
- Work effectively in potential and zero-sum games, but limited in genera-sum games.
- Average policy have convergence guarantee but generally no last-iteration convergence

No-regret methods

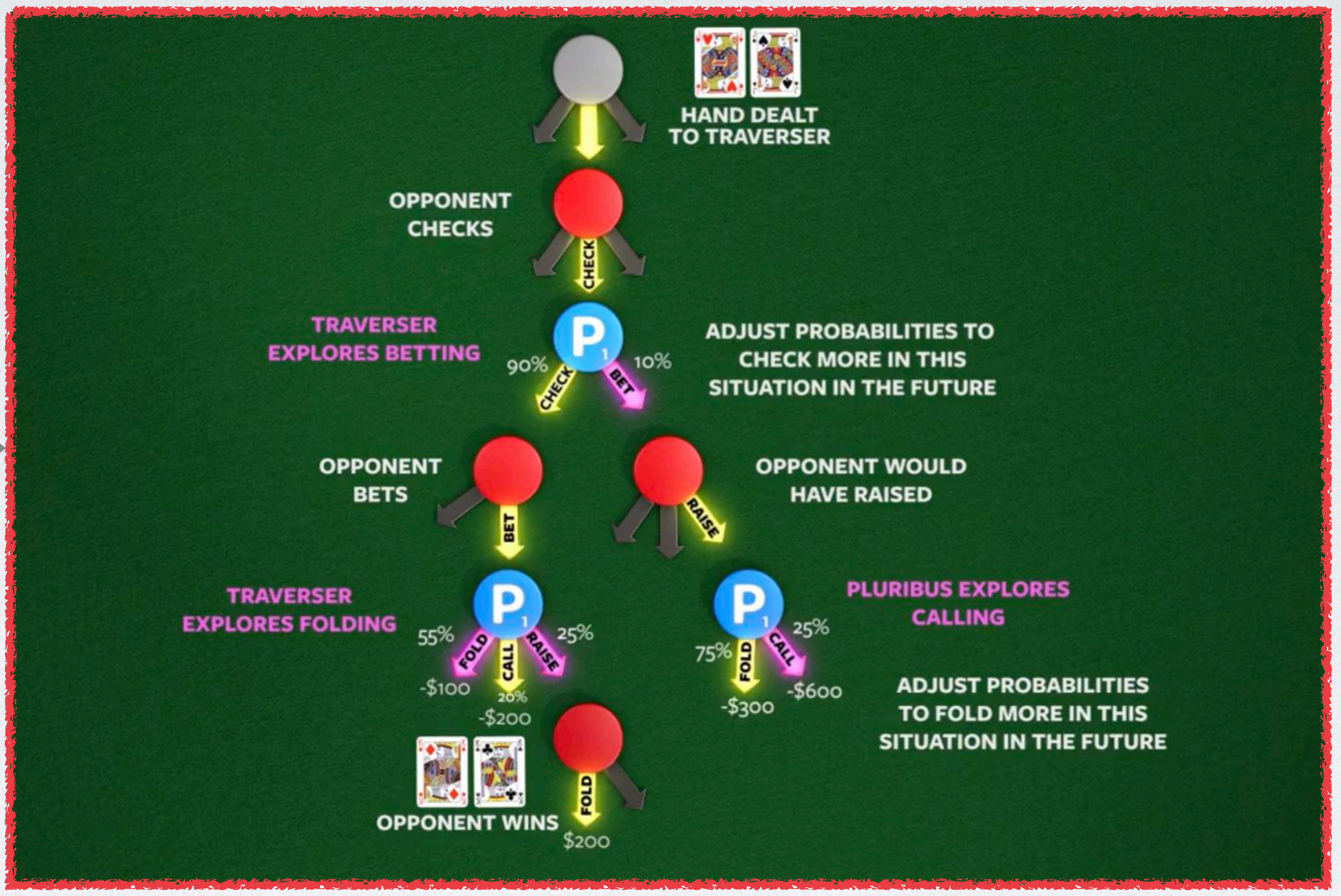
- MWU, Follow the Regularised/Perturbed leader, CFR and all kinds of CFR variants, MCTS, …
- Work in a self-play settings, no best-response step but a no-regret step.
- Often requires to know the model (the game tree, utility function/strategies of opponents, etc)
- A portal to the arsenal of online learning tools
- Have nice convergence guarantee to Nash zero-sum games, and CE/CCE in general-sum games.

No-Regret vs. Best-Response Methods in Zero-Sum Games

Output: the reward $(R^1, \ldots, R^N)$

Black-box multi-agent game engine

Input: a joint strategy $(\pi^1, \ldots, \pi^N)$



Regret based methods: Poker Type

When planning is feasible (game tree is easily accessible), existing techniques can solve the games really well.

Perfect-information games:
MCTS, alpha-beta search, AlphaGO series (AlphaZero, MuZero, etc)

Imperfect-information:
CFR series (DeepCFR, Libratus/Pluribus, Deepstack), XFP/NFSP series

Best response based methods: StarCraft type

When planning is not feasible. StarCraft has 10^{26} choices per time step vs. the whole tree of chess 10^{50} (Texas holdem 10^{80}, GO 10^{170})

Enumerating all policies' actions at each state and then playing a best response is infeasible. But an approximate BR can be computed.

Solution: training a population of RL agents, treat each RL agent as one "pure strategy" and solve the game in the meta-level (e.g. PSRO methods).

No-Regret vs. Best-Response Methods in Zero-Sum Games

When planning is feasible (game tree is

Actor-Critic Policy Optimization in Partially Observable Multiagent Environments

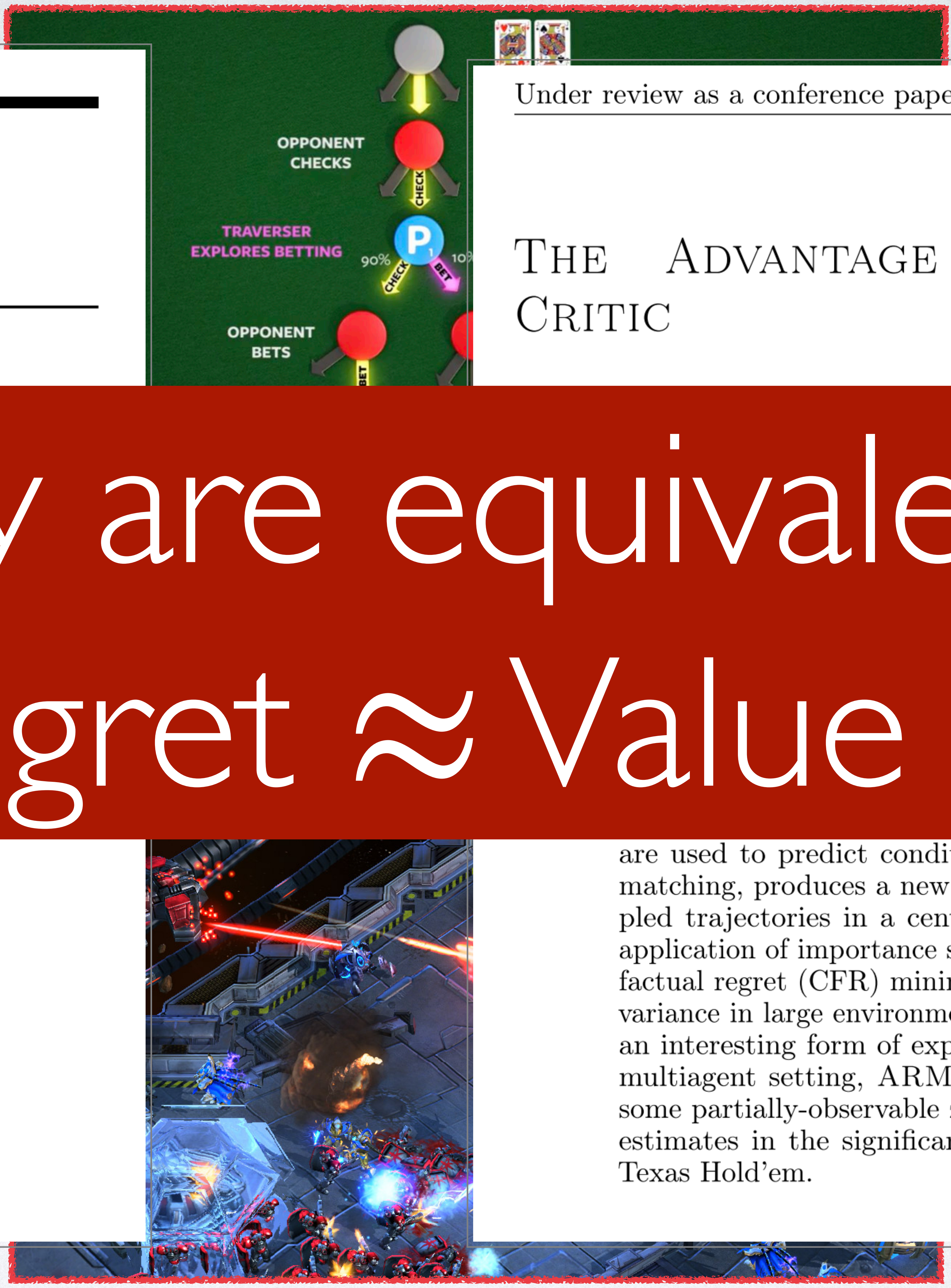
Sriram Srinivasan[*,1] srsrinivasan@
Marc Lanctot[*,1] lanctot@
Viniciu vza

Karl Tuyls[1] karltuyls@
Rémi Munos[1] munos@

...@google.com. [1]DeepMind. *These author

Abstract

Optimization of parameterized policies for reinforcem tant and challenging problem in artificial intelligence. Among the most common approaches are algorithms based on gradient ascent of a score function representing discounted return. In this paper, we examine the role of these policy gradient and actor-critic algorithms in partially-observable multiagent environments. We show several candidate policy update rules and relate them to a foundation of regret minimization and multiagent learning techniques for the one-shot and tabular cases. We apply our method to *model-free* multiagent reinforcement learning in adversarial sequential decision problems (zero-sum imperfect information games), using RL-style function approximation. We evaluate on commonly used benchmark Poker domains, showing performance against fixed policies and empirical convergence to approximate Nash equilibria in self-play with rates similar to or better than a baseline model-free algorithm for zero-sum games, without any domain-specific state space reductions.



Under review as a conference paper at ICLR 2021

THE ADVANTAGE REGRET-MATCHING ACTOR-CRITIC

CT

ole in online learning, equilibrium ent learning (RL). In this paper, thod for no-regret learning based vior: Advantage Regret-Matching aving past state-action data, AR- aying through them to reconstruct These retrospective value estimates are used to predict conditional advantages which, combined with regret matching, produces a new policy. In particular, ARMAC learns from sampled trajectories in a centralized training setting, without requiring the application of importance sampling commonly used in Monte Carlo counterfactual regret (CFR) minimization; hence, it does not suffer from excessive variance in large environments. In the single-agent setting, ARMAC shows an interesting form of exploration by keeping past policies intact. In the multiagent setting, ARMAC in self-play approaches Nash equilibria on some partially-observable zero-sum benchmarks. We provide exploitability estimates in the significantly larger game of betting-abstracted no-limit Texas Hold'em.

They are equivalent

Regret ≈ Value !

s

e of

te

e is

me,

the problem problem, auto-curricula.

Input: a joint strategy $(\pi^1, \ldots, \pi^N)$

Best response based methods: StarCraft type

Online Learning and No-Regret

- **The settings of online learning:**
  - The algorithm picks a strategy $p^t \in \Delta_{|A|}$ at time step t
  - The adversary/nature picks cost vector $c^t : A \to [0,1]$
  - An action a^t is drawn from p^t, and the algorithm incurs cost of $c^t(a^t)$
  - **Full-information settings:** observe the entire cost vector c^t. **Bandit settings:** only observe selected $c^t(a^t)$
  - **Oblivious adversary:** c^t only depends on t. **Adaptive adversary:** c^t depends on $\{t, (p^1, \ldots, p^t), (a^1, \ldots, a^{t-1})\}$
  - **Goal:** we try to learn how should we adapt our algorithms, learn from mistakes [remember Alan Turing] !
- **The (external) regret of a sequence of actions w.r.t action $a \in A$:**

$$R_T(a) = \frac{1}{T}\left(\sum_{t=1}^{T} c^t\left(a^t\right) - \sum_{t=1}^{T} c^t(a)\right)$$

- **A no-(external)-regret algorithm $\mathscr{A}$ is said to be no-regret (also Hannan consistent) if:**

$$\lim_{T\to\infty} \mathbf{E}\left[R_T^{\mathscr{A}}(a)\right] = \frac{1}{T}\left(\sum_{i=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(a^t\right)\right] - \sum_{t=1}^{T} c^t(a)\right) = 0, \forall a \in A$$

No-Regret Learning

- **No-regret to the best action sequence $\sum_{t=1}^{T} \min_{a \in A} c^t(a)$ in hindsight is impossible.**
  - When the adversary exploits you, in hindsight, a zero-loss best action sequence is always possible. If we known c^t before each iteration we can have $\sum_{t=1}^{T} \min_{a \in A} c^t(a) = 0$, then the regret explodes.

$$\mathbf{E}\left[R_T^{\mathscr{A}}(a)\right] = \frac{1}{T}\left(\sum_{i=1}^{T} \mathbf{E}_{a^t \sim p^t}\left[c^t\left(a^t\right)\right] - \sum_{t=1}^{T} \min_{a \in A} c^t(a)\right) = \mathcal{O}(T) \neq \mathcal{O}(1)$$

not sublinear!

  - However, no-regret to the best action (not sequence!) in hindsight is possible:

$$\underbrace{\sum_{t=1}^{T} \min_{a \in A} c^t(a)}_{\text{best action sequence}} \quad vs. \quad \underbrace{\min_{a \in A} \sum_{t=1}^{T} c^t(a)}_{\text{best action in hindsight}}$$

- **In games, no-regret is the minimal requirement for rationale.**

No-Regret Learning towards Coarse Correlated Equilibrium

- In games, no-regret is the minimal requirement for rationale.
- Recall no-regret is

$$\lim_{T\to\infty} \mathbf{E}\left[R_T^{\mathscr{A}}(a)\right] = \frac{1}{T}\left(\sum_{i=1}^{T} \mathop{\mathbf{E}}_{a^t\sim p^t}\left[c^t\left(a^t\right)\right] - \sum_{t=1}^{T} c^t(a)\right) = 0, \forall a \in A$$

- **Theorem: if all players adopt no-regret algorithms such that $\mathbf{E}[R_T^{\mathscr{A}_i}(a)] \leq \epsilon,\ \forall i \in [k], a \in S_i$, then the average distribution $\sigma = \sum_{t=1}^{T}\prod_{i=1}^{k}\frac{p_i^t}{T}$ is an $\epsilon-$CCE, i.e., $\mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i(s)\right] \leq \mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i\left(s_{-i}, s_i'\right)\right] + \epsilon$**

- Proof:

$$\mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i(s)\right] - \mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i\left(s_{-i}, s_i'\right)\right] = \frac{1}{T}\sum_{t=1}^{T}\mathop{\mathbf{E}}_{s\sim\sigma^t}\left[C_i(s)\right] - \frac{1}{T}\sum_{t=1}^{T}\mathop{\mathbf{E}}_{s\sim\sigma^t}\left[C_i\left(s_{-i}, s_i'\right)\right]$$

$$= \frac{1}{T}\left(\sum_{t=1}^{T}\mathop{\mathbf{E}}_{a^t\sim p_i^t}\left[c_i^t\left(a^t\right)\right] - \sum_{t=1}^{T} c_i^t\left(s_i'\right)\right)$$

$$= \mathbf{E}\left[R_T^{\mathscr{A}_i}\left(s_i'\right)\right] \leq \epsilon$$

Note that in game playing, we have:

$$c_i^t(a) = \mathbf{E}_{s\sim\sigma^t}\left[C_i\left(s_{-i}, a\right)\right]$$

No-Regret Learning in Zero-Sum Games

- We have showed that if all players play no-regret methods, they can reach a CCE.
- We can further show that no-regret players will reach the Nash in zero-sum games.
  - For better clarity. Let's assume row player chooses an action $I_t \in \{1,\ldots,N\}$, mixed strategy $\mathbf{p}_t = \left(p_{1,t}, \ldots, p_{N,t}\right)$
  - Column player instead of deciding c^t, let's assume they choose an action $J_t \in \{1,\ldots,M\}$ and $\mathbf{q}_t = \left(q_{1,t}, \ldots, q_{M,t}\right)$
  - Assuming both players adopt no-regret algorithm regardless of what the opponent does, such that

$$\limsup_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{n} \ell\left(I_t, J_t\right) - \min_{i=1,\ldots,N} \frac{1}{T}\sum_{t=1}^{n} \ell\left(i, J_t\right)\right) \le 0$$

  - Recall that the Nash value of a two-player zero-sum game is $\max_{\mathbf{q}} \min_{\mathbf{p}} \sum_{i=1}^{N} \sum_{j=1}^{M} p_i q_j \ell(i,j)$
- We have the main theorem that

Theorem: assuming that in a two-player zero-sum game, if both players play no-regret methods, then $\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \ell\left(I_t, J_t\right) = V = \max_{\mathbf{q}} \min_{\mathbf{p}} \sum_{i=1}^{N} \sum_{j=1}^{M} p_i q_j \ell(i,j)$ **almost surely.**

No-Regret Learning in Zero-Sum Games

Theorem: assuming that in a two-player zero-sum game, if both players play no-regret methods, then $\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \ell\left(I_t, J_t\right) = V = \max_{\mathbf{q}} \min_{\mathbf{p}} \sum_{i=1}^{N}\sum_{j=1}^{M} p_i q_j \ell(i,j)$ **almost surely.**

Proof: 1) we first should that regardless of what column player plays, if the row plays play no-regret method, his loss will be no more than maximin value (i.e., worst case scenario to row player) $\lim\sup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \ell\left(I_t, J_t\right) \leq V$.

Since the row player adopts no-regret method we only need to show:

$$\min_{i=1,\ldots,N} \frac{1}{T}\sum_{t=1}^{T} \ell\left(i, J_t\right) = \min_{\mathbf{p}} \frac{1}{T}\sum_{t=1}^{T}\Big(\sum_{i=1}^{N} p_i \ell(i,j)\Big)$$

pure strategy is a special mixed strategy

$$= \min_{\mathbf{p}} \frac{1}{T}\sum_{t=1}^{T} \bar{\ell}\left(\mathbf{p}, J_t\right)$$

change of notation

$$= \min_{\mathbf{p}} \sum_{j=1}^{M}\Big(\frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{\{J_t=j\}}\bar{\ell}(\mathbf{p}, j)\Big)$$

empirical mean on column player's action

$$= \min_{\mathbf{p}} \bar{\ell}\left(\mathbf{p}, \hat{\mathbf{q}}_T\right)$$

expectation over the empirical mean of column player

$$\leq \max_{\mathbf{q}} \min_{\mathbf{p}} \bar{\ell}(\mathbf{p}, \mathbf{q}) = V$$

done!

No-Regret Learning in Zero-Sum Games

Theorem: assuming that in a two-player zero-sum game, if both players play no-regret methods, then $\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \ell\left(I_t, J_t\right) = V = \max_{\mathbf{q}} \min_{\mathbf{p}} \sum_{i=1}^{N}\sum_{j=1}^{M} p_i q_j \ell(i,j)$ **almost surely.**

Proof: 2) we proved that $\min_{i=1,\ldots,N} \frac{1}{T}\sum_{t=1}^{T} \ell\left(i, J_t\right) \leq V$, and since row player plays no-regret

$$\limsup_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{n} \ell\left(I_t, J_t\right) - \min_{i=1,\ldots,N} \frac{1}{T}\sum_{t=1}^{n} \ell\left(i, J_t\right)\right) \leq 0$$

we can know that

$$\text{lim sup}_{T\to\infty} \tfrac{1}{T}\textstyle\sum_{t=1}^{T} \ell\left(I_t, J_t\right) \leq V$$

for row player, we can do the same for the column player

$$\liminf_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \ell\left(I_t, J_t\right) \geq \min_{\mathbf{p}} \max_{\mathbf{q}} \bar{\ell}(\mathbf{p}, \mathbf{q}) = V$$

By von Neumann's minimax theorem, we prove that

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \ell\left(I_t, J_t\right) = V$$

This mean no-regret players self-play will reach to Nash equilibrium in two-player zero-sum games.

No-Regret Learning in Zero-Sum Games

- We have showed that no-regret players will reach Nash equilibrium value.
- Furthermore, we can show that they reach to the Nash strategy.

Theorem: In a two-player zero-sum game, if both players play no-regret methods, then $\hat{p}_{i,T} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{\{I_t=i\}}, \quad \hat{q}_{j,T} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{\{J_t=j\}}$ **almost surely converge to the set of Nash equilibrium.**

Proof: in the previous proof, we have shown that

$$\min_{\mathbf{p}} \bar{\ell}\left(\mathbf{p}, \hat{\mathbf{q}}_T\right) \leq V, \quad \max_{\mathbf{q}} \bar{\ell}\left(\hat{\mathbf{p}}_T, \mathbf{q}\right) \geq V$$

and due to the uniqueness of V value in zero-sum games

$$\min_{\mathbf{p}} \bar{\ell}\left(\mathbf{p}, \hat{\mathbf{q}}_T\right) = \max_{\mathbf{q}} \bar{\ell}\left(\hat{\mathbf{p}}_T, \mathbf{q}\right) = V$$

and because of

$$\bar{\ell}\left(\hat{\mathbf{p}}_T, \hat{\mathbf{q}}_T\right) \geq \min_{\mathbf{p}} \bar{\ell}\left(\mathbf{p}, \hat{\mathbf{q}}_T\right), \quad \max_{\mathbf{q}} \bar{\ell}\left(\hat{\mathbf{p}}_T, \mathbf{q}\right) \geq \bar{\ell}\left(\hat{\mathbf{p}}_T, \hat{\mathbf{q}}_T\right)$$

finally, we prove that

$$\bar{\ell}\left(\hat{\mathbf{p}}_T, \hat{\mathbf{q}}_T\right) = V$$

This means that the empirical mean of $\hat{\mathbf{p}}_T, \hat{\mathbf{q}}_T$ is the Nash

Fictitious Play is Not No-Regret

- No-regret can lead to CCE in general-sum, NE in two-player zero-sum.
- But, what exactly is a no-regret algorithm? How can we behave to achieve no-regret?

$$\limsup_{T\to\infty}\left(\frac{1}{T}\sum_{t=1}^{n}\ell\left(I_t, J_t\right) - \min_{i=1,\ldots,N}\frac{1}{T}\sum_{t=1}^{n}\ell\left(i, J_t\right)\right) \leq 0$$

- Surprisingly, Fictitious play is not no-regret !
  - Recall that fictitious play, also known as **Follow-the-Leader**, is to take the best response to the average cumulative loss.

$$I_t = \underset{i=1,\ldots,N}{\operatorname{argmin}}\left(\frac{1}{t-1}\sum_{t=1}^{t-1}\ell\left(i, J_t\right)\right)$$

  - Consider $N = 2$ actions, let J_t be chosen such that $\ell(1,J_t) = (1/2,0,1,0,1,\ldots)$ and $\ell(2,J_t) = (1/2,1,0,1,0,\ldots)$
  - Then the accumulative loss for both actions is T/2, and the FP will suffer a loss of T, thus has constant regret.
  - There are many variants that make FP no-regret, for example Follow-the-Perturbed-Leader:

$$I_t = \underset{i=1,\ldots,N}{\operatorname{argmin}}\left(\frac{1}{t-1}\sum_{t=1}^{t-1}\ell\left(i, J_t\right) + Z_{i,t}\right),\ \mathbf{Z}_t \text{ any i.i.d random vectors of size } N$$

No-Regret Algorithms: MWU

- Fictitious play is not no-regret !
- Let's introduce a true no-regret algorithm: Multiplicative Weight Update [Freund 1999].
  - MWU has many names: *Hedge, Exponential Weights Algorithms, Randomised Weighted Majority*

$$p_{i,t} = \frac{\exp\left(-\eta \sum_{s=1}^{t-1} \ell\left(i, J_s\right)\right)}{\sum_{k=1}^{|A|} \exp\left(-\eta \sum_{s=1}^{t-1} \ell\left(k, J_s\right)\right)}$$

  - **Equivalently,** one can think of the following iterative process:

$$p_t = \frac{w^t}{\sum_{a \in A} w^t(a)}, \quad w^{t+1}(a) = w^t(a) \cdot \exp\left(-\eta \ell(a, J_t)\right), w^1(a) = \mathbf{1} \; \forall a \in A$$

    - Large η means more exploitation, small η means more exploration.
  - **Equivalently**, one can get MWU by the following maximum entropy framework (a common trick in RL).

$$\arg \min_{\boldsymbol{p} \in \Delta_{|A|}} \bar{\ell}(\boldsymbol{p}, J_s) + 1/\eta \cdot \sum_i p_i \log p_i$$

similar to soft Q-learning !

No-Regret Algorithms: MWU

- Let's introduce a true no-regret algorithm — Multiplicative Weight Update.
  - Equivalently, one can get MWU by the following maximum entropy framework (a common trick in RL).

$$\arg\min_{p\in\Delta_{|A|}} \bar{\ell}(\boldsymbol{p}, J_s) + 1/\eta \cdot \sum_i p_i \log p_i = \frac{\exp\left(-\eta \sum_{s=1}^{t-1} \ell\left(i, J_s\right)\right)}{\sum_{k=1}^{|A|} \exp\left(-\eta \sum_{s=1}^{t-1} \ell\left(k, J_s\right)\right)}$$

$$\arg\min_p \ell(p, J_s) + 1/\eta \cdot \sum_i p_i \log p_i$$

$$= \sum_i p_i \, \ell(i, J_s) + 1/\eta \cdot \sum_i p_i \log p_i$$

$$= 1/\eta \sum_i p_i \left(\eta \cdot \ell(i, J_s) + \log p_i\right)$$

$$= 1/\eta \, E_p[\log \exp(\eta \cdot \ell(i, J_s)) + \log p_i]$$

$$= 1/\eta \, E_p[-\log \exp(-\eta \cdot \ell(i, J_s)) + \log p_i]$$

$$= 1/\eta \, E_p\left[\log \frac{p_i}{\exp(-\eta \cdot \ell(i, J_s))}\right]$$

$p_i^* = \exp(-\eta \cdot \ell(i, J_s))$ to minimize

Therefore $P = \exp(-\eta \cdot \ell(i, J_s)) / \sum_i \exp(-\eta \cdot \ell(i, J_s))$

No-Regret Algorithms: MWU

- Let's introduce a true no-regret algorithm — Multiplicative Weight Update.

$$p_t = \frac{w^t}{\sum_{a\in A} w^t(a)}, \quad w^{t+1}(a) = w^t(a)\cdot \exp\big(-\eta\ell(a, J_t)\big), w^1(a) = \mathbf{1}\ \forall a \in A$$

- Let's now show MWU is indeed a no-regret method
  - Let $\mathbf{opt} = \min_{a\in A}\sum_{t=1}^{n} \ell\left(a, J_t\right)$ and $\mathbf{v}^t = \sum_{a\in|A|} p_t(a)\cdot\ell(a, J_t)$, thus the regret-bound is $\lim_{T\to\infty}\frac{1}{T}\left(\sum_{t=1}^{T} v^t - \mathbf{opt}\right)$
  - We first bound the denumerator $\sum_{a\in A} w^t(a) \geq w^t(a^*) = \exp(-\eta\cdot\mathbf{opt})$ multiplication rule of exp function

$$\geq (1-\epsilon)^{\mathbf{opt}}$$ assume $1-\epsilon = e^{-\eta}$

$$\sum_a w^{t+1}(a) = \sum_a w^t(a)\cdot\exp\big(-\eta\ell(a, J_t)\big) := \sum_a w^t(a)\cdot(1-\epsilon)^{\ell(a,J_t)}$$ assume$1-\epsilon = e^{-\eta}$

$$\leq \sum_a w^t(a)\cdot(1-\epsilon\ell(a, J_t))$$ Taylor expansion

$$= \sum_a w^t(a)\cdot(1-\epsilon\mathbf{v}^t)$$ definition of $\mathbf{v}^t = \sum_{a\in|A|}\frac{w^t(a)}{\sum_a W^t(a)}\cdot\ell(a, J_t)$

No-Regret Algorithms: MWU

- Regret bound of MWU
  - Let $\mathbf{opt} = \min_{a \in A} \sum_{t=1}^{n} \ell\left(a, J_t\right)$ and $\mathbf{v}^t = \sum_{a \in |A|} p_t(a) \cdot \ell(a, J_t)$, thus the regret-bound is $\frac{1}{T}\left(\sum_{t=1}^{T} v^t - \mathbf{opt}\right)$
  - Merge the upper and lower bound in the last slides

$$(1-\epsilon)^{\mathbf{opt}} \leq \sum_{a \in A} w^t(a) \leq \sum_{a} w^1(a) \prod_{t=1}^{T} (1-\epsilon \mathbf{v}^t) = |A| \prod_{t=1}^{T} (1-\epsilon \mathbf{v}^t)$$

$$\mathbf{opt} \cdot \ln(1-\epsilon) \leq \ln|A| + \sum_{t=1}^{T} \ln\left(1-\epsilon \mathbf{v}^t\right)$$

$$\mathbf{opt} \cdot \left(-\epsilon-\epsilon^2\right) \leq \ln|A| + \sum_{t=1}^{T} \left(-\epsilon \mathbf{v}^t\right)$$

Taylor expansion $\ln(1-x) = -x - x^2/2$

  - Set $\epsilon = \sqrt{\ln|A|/T}$ we conclude the proof

$$\frac{1}{T}\left(\sum_{t=1}^{T} v^t - \mathbf{opt}\right) \leq \frac{1}{T}\left(\epsilon T + \frac{1}{\epsilon}\ln|A|\right) = \frac{1}{T}(\sqrt{T\ln|A|} + \sqrt{T\ln|A|}) \leq 2\sqrt{\frac{\ln|A|}{T}} \to 0$$

Note: the log term is great for many real-world problems!

No-Regret Algorithms: Online Double Oracle

Online Double Oracle

Le Cong Dinh[*,1,2], Yaodong Yang[*,1,4], Nicolas Perez-Nieves[3], Oliver Slumbers[4], Zheng Tian[4], David Henry Mguni[1], Haitham Bou Ammar[1], Jun Wang[1,4]

1. Nash is unexploitbale, but when a player always plays Rock, you should play Paper rather than (1/3, 1/3, 1/3).
2. Double Oracle/PSRO assumes both players play the worst-case scenario, can be too pessimistic during training.
3. Online learning provides a framework about how to exploit opponents through minimising regret.

Algorithm 1 Double Oracle (McMahan et al., 2003)

```
Input: A set Π, C strategy set of players
Π_0, C_0: initial set of strategies
for t = 1 to ∞ do
  if Π_t ≠ Π_{t−1} or C_t ≠ C_{t−1} then
    Solve the NE of the subgame G_t:
    (π*_t, c*_t) = arg min_{π∈Δ_{Π_t}} arg max_{c∈Δ_{C_t}} π^⊤ A c      too pessimistic
    Find the best response a_{t+1} and c_{t+1} to (π*_t, c*_t):
        a_{t+1} = arg min_{a∈Π} a^⊤ A c*_t
        c_{t+1} = arg max_{c∈C} π*_t^⊤ A c
    Update Π_{t+1} = Π_t ∪ {a_{t+1}}, C_{t+1} = C_t ∪ {c_{t+1}}
  else if Π_t = Π_{t−1} and C_t = C_{t−1} then
    Terminate
  end if
end for
```

What we want:

if opponents play $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_T$, we want the player to have $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_T$ s.t.

$$\lim_{T\to\infty} \frac{R_T}{T} = 0, \quad R_T = \max_{\pi\in\Delta_\Pi} \sum_{t=1}^{T} \left(\boldsymbol{\pi}_t^\top \boldsymbol{A}\boldsymbol{c}_t - \boldsymbol{\pi}^\top \boldsymbol{A}\boldsymbol{c}_t\right)$$

What we know:

hedge algorithm/multiplicative weight update can achieve no-regret property if one follows the below update

$$\boldsymbol{\pi}_{t+1}(i) = \boldsymbol{\pi}_t(i)\frac{\exp\left(-\mu_t \boldsymbol{a}^{i\top}\boldsymbol{A}\boldsymbol{c}_t\right)}{\sum_{i=1}^{n} \boldsymbol{\pi}_t(i)\exp\left(-\mu_t \boldsymbol{a}^{i\top}\boldsymbol{A}\boldsymbol{c}_t\right)}, \forall i \in [n]$$

the regret of MWU is $\mathcal{O}(\sqrt{\log(n)/T})$

No-Regret Algorithms: Online Double Oracle

Algorithm 1 Double Oracle (McMahan et al., 2003)

Input: A set Π, C strategy set of players
Π_0, C_0: initial set of strategies
for $t = 1$ to ∞ **do**
  if $\Pi_t \neq \Pi_{t-1}$ or $C_t \neq C_{t-1}$ **then**
    Solve the NE of the subgame $\boldsymbol{G}_t$:
    $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*) = \arg\min_{\boldsymbol{\pi} \in \Delta_{\Pi_t}} \arg\max_{\boldsymbol{c} \in \Delta_{C_t}} \boldsymbol{\pi}^\top \boldsymbol{A}\boldsymbol{c}$
    Find the best response $\boldsymbol{a}_{t+1}$ and $\boldsymbol{c}_{t+1}$ to $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*)$:
      $\boldsymbol{a}_{t+1} = \arg\min_{\boldsymbol{a} \in \Pi} \boldsymbol{a}^\top \boldsymbol{A}\boldsymbol{c}_t^*$
      $\boldsymbol{c}_{t+1} = \arg\max_{\boldsymbol{c} \in C} \boldsymbol{\pi}_t^{*\top} \boldsymbol{A}\boldsymbol{c}$
    Update $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_{t+1}\}, C_{t+1} = C_t \cup \{\boldsymbol{c}_{t+1}\}$
  else if $\Pi_t = \Pi_{t-1}$ and $C_t = C_{t-1}$ **then**
    Terminate
  end if
end for

Algorithm 2: Online Single Oracle Algorithm

Input: Player's pure strategy set Π
Init. effective strategies set: $\Pi_0 = \Pi_1 = \{\boldsymbol{a}^j\}, \boldsymbol{a}^j \in \Pi$
for $t = 1$ to T **do**
  if $\Pi_t = \Pi_{t-1}$ **then**
    Compute $\boldsymbol{\pi}_t$ by the MWU in Equation (5)
  else if $\Pi_t \neq \Pi_{t-1}$ **then**
    Start a new time window T_{i+1} and
    Reset $\boldsymbol{\pi}_t = [1/|\Pi_t|, \ldots, 1/|\Pi_t|]$, $\bar{\boldsymbol{l}} = \boldsymbol{0}$
  end if
  Observe $\boldsymbol{l}_t$ and update the average loss in T_i:
  $\bar{\boldsymbol{l}} = \sum_{t \in T_i} \boldsymbol{l}_t / |T_i|$
  Calculate the best response: $\boldsymbol{a}_t = \arg\min_{\boldsymbol{\pi} \in \Pi} \langle \boldsymbol{\pi}, \bar{\boldsymbol{l}} \rangle$
  Update the set of strategies: $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_t\}$
end for
Output: $\boldsymbol{\pi}_T, \Pi_T$

Intuition: *maintain a time window T_i to track opponent's strategy, if no new best response can be found, then keep exploiting, otherwise refresh the time window to catch up with the latest change*

No-Regret Algorithms: Online Double Oracle

Algorithm 2: Online Single Oracle Algorithm

Input: Player's pure strategy set Π
Init. effective strategies set: $\Pi_0 = \Pi_1 = \{\boldsymbol{a}^j\}, \boldsymbol{a}^j \in \Pi$
for $t = 1$ to T **do**
  if $\Pi_t = \Pi_{t-1}$ **then**
    Compute $\boldsymbol{\pi}_t$ by the MWU in Equation (5)
  else if $\Pi_t \neq \Pi_{t-1}$ **then**
    Start a new time window T_{i+1} and
    Reset $\boldsymbol{\pi}_t = \left[1/|\Pi_t|, \ldots, 1/|\Pi_t|\right], \ \bar{\boldsymbol{l}} = \mathbf{0}$
  end if
  Observe $\boldsymbol{l}_t$ and update the average loss in T_i:
  $\bar{\boldsymbol{l}} = \sum_{t \in T_i} \boldsymbol{l}_t / |T_i|$
  Calculate the best response: $\boldsymbol{a}_t = \arg\min_{\boldsymbol{\pi} \in \Pi} \langle \boldsymbol{\pi}, \bar{\boldsymbol{l}} \rangle$
  Update the set of strategies: $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_t\}$
end for
Output: $\boldsymbol{\pi}_T, \Pi_T$

1. OSO is a no-regret algorithm.

Theorem 4 (Regret Bound of OSO). *Let $\boldsymbol{l}_1, \boldsymbol{l}_2, \ldots, \boldsymbol{l}_T$ be a sequence of loss vectors played by an adversary, and $\langle \cdot, \cdot \rangle$ be the dot product, OSO in Algorithm 2 is a no-regret algorithm with*

$$\frac{1}{T}\left(\sum_{t=1}^{T} \langle \boldsymbol{\pi}_t, \boldsymbol{l}_t \rangle - \min_{\boldsymbol{\pi} \in \Pi} \sum_{t=1}^{T} \langle \boldsymbol{\pi}, \boldsymbol{l}_t \rangle\right) \leq \frac{\sqrt{k \log(k)}}{\sqrt{2T}},$$

where $k = |\Pi_T|$ is the size of effective strategy set in the final time window.

2. Putting OSO into self-play settings, we get Online Double Oracle which can solve Nash.

- Recall that in two-player zero-sum game, if two no-regret methods self play, the outcome will leads to a Nash equilibrium!
- We just prove that.

Algorithm 3: Online Double Oracle Algorithm

Input: Full pure strategy set Π, C
Init. effective strategies set: $\Pi_0 = \Pi_1, C_0 = C_1$
for $t = 1$ to T **do**
  Each player follows the OSO in Algorithm 2 with their respective effective strategy sets Π_t, C_t
end for
Output: $\boldsymbol{\pi}_T, \Pi_T, \boldsymbol{c}_T, C_T$

Theorem 5. *Suppose both players apply OSO. Let k_1, k_2 denote the size of effective strategy set for each player. Then, the average strategies of both players converge to the NE with the rate:*

$$\epsilon_T = \sqrt{\frac{k_1 \log(k_1)}{2T}} + \sqrt{\frac{k_2 \log(k_2)}{2T}}.$$

In situation where both players follow OSO with Less-Frequent Best Response in Equation (6) and $\alpha^i_{t-|\bar{T}_i|} = \sqrt{t - |\bar{T}_i|}$, the convergence rate to NE will be

$$\epsilon_T = \sqrt{\frac{k_1 \log(k_1)}{2T}} + \sqrt{\frac{k_2 \log(k_2)}{2T}} + \frac{\sqrt{k_1} + \sqrt{k_2}}{\sqrt{T}}.$$

No-Regret Algorithms: Online Double Oracle

Exploitability on the Spinning Top games

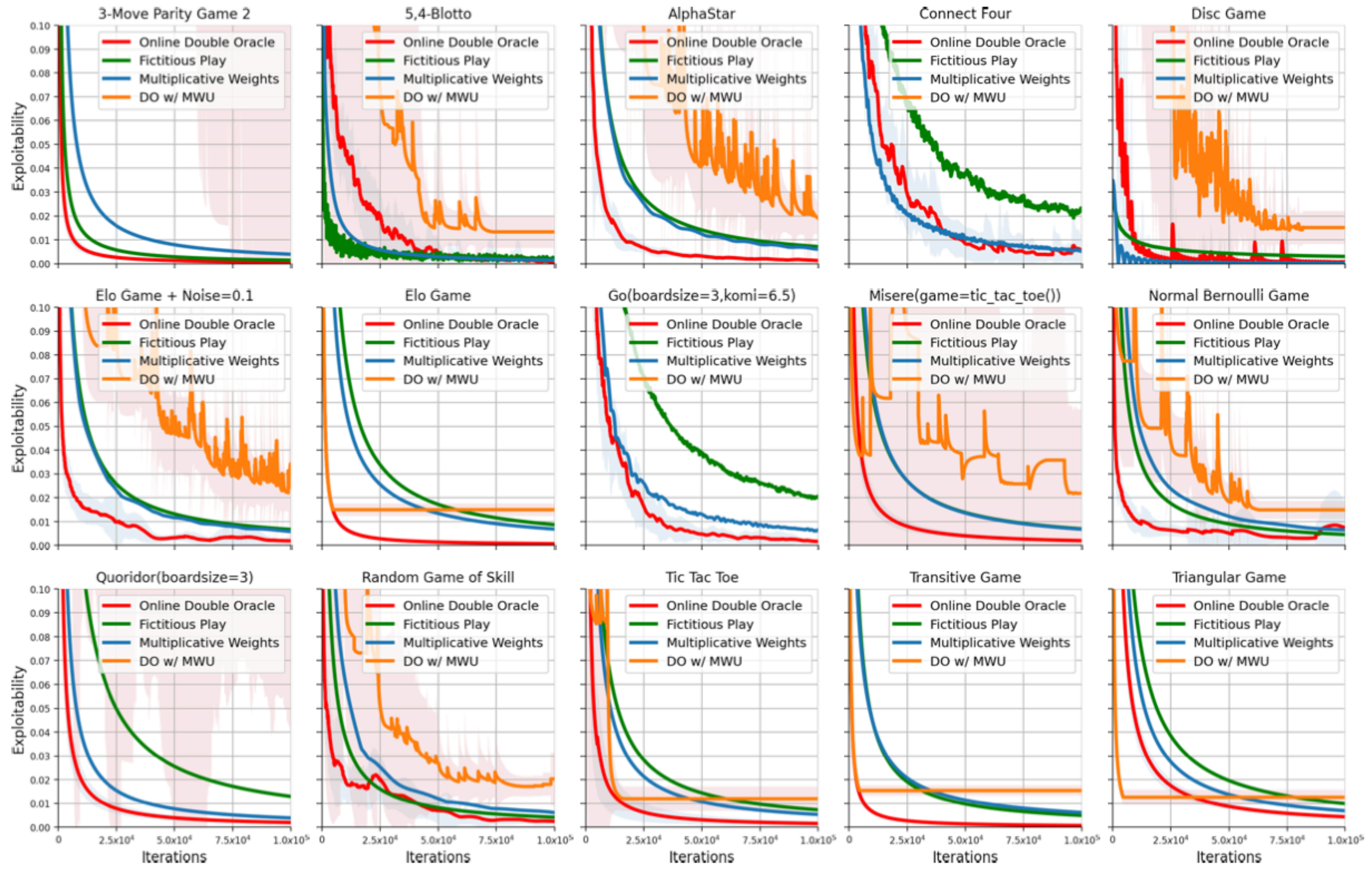
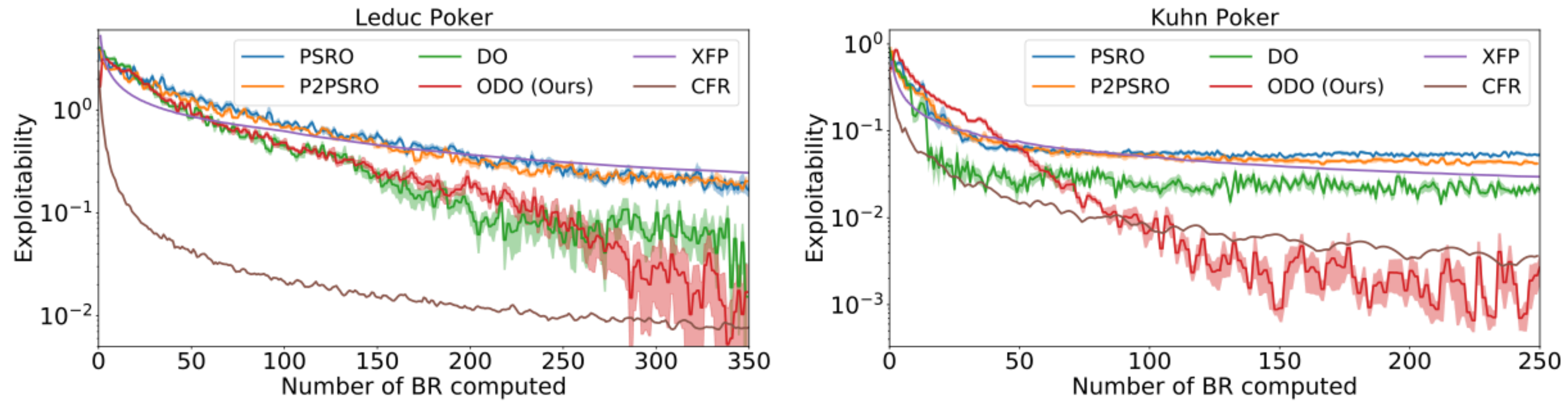


Figure 1: Performance comparisons under self-plays

Exploitability on Poker

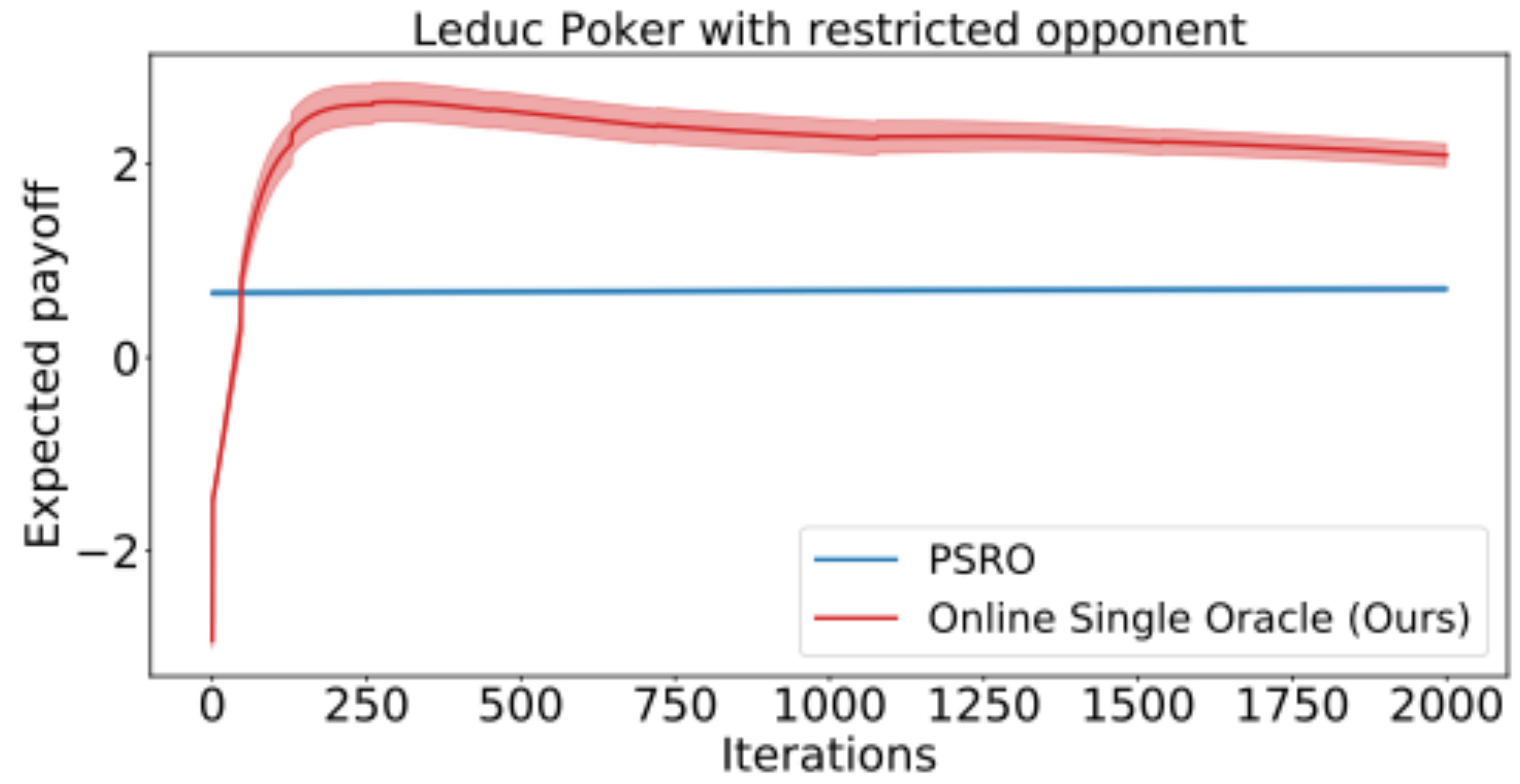


(a) Exploitability on Leduc Poker

(b) Exploitability on Kuhn Poker

Figure 3: Performance comparisons in exploitability on Poker games.

Play with an imperfect opponent



(a) Leduc Poker

No-Regret Algorithms for Correlated Equilibrium

- No-(external)-regret players can lead to CCE in multi-player general-sum games, and Nash in two-player zero-sum games.
- The last missing piece: how about correlated equilibrium?

Definition 5.3.2 *Let σ be a distribution over $S = S_1 \times \cdots \times S_k$. Then σ is a* correlated equilibrium *if*

$$\mathop{\mathbf{E}}_{s \sim \sigma}[c_i(s)|s_i] \le \mathop{\mathbf{E}}_{s \sim \sigma}[c_i(s_{-i}, s_i')|s_i]$$

for all $i \in [k]$ and for all $s_i, s_i' \in S_i$.

- We can also define CE through a switching function (play action b when I plan to play a)

Theorem 7.3.2 *σ is a correlated equilibrium if and only if for all $i \in [k]$ and $\delta : S_i \to S_i$,*

$$\mathop{\mathbf{E}}_{s \sim \sigma}[C_i(s)] \le \mathop{\mathbf{E}}_{s \sim \sigma}[C_i(s_{-i}, \delta(s_i)]. \tag{7.3.1}$$

$$\delta(x) = \begin{cases} b & \text{if } x = a \\ x & \text{otherwise} \end{cases}$$

  - $\Rightarrow$: if a CE, then any switching will incur large cost
  - $\Leftarrow$: we can rewrite (7.3.1) into $\sum_{x \in S_i} \left(\Pr[s_i = x] \cdot \mathop{\mathbf{E}}_{s \sim \sigma}[C_i(s) \mid s_i = x] \right) \le \sum_{x \in S_i} \left(\Pr[s_i = x] \cdot \mathop{\mathbf{E}}_{s \sim \sigma}\left[C_i\left(s_{-i}, \delta(s_i)\right) \mid s_i = x\right] \right)$

both sides only differ when $x = a$

$$\Pr[s_i = a] \cdot \mathop{\mathbf{E}}_{s \sim \sigma}[C_i(s) \mid s_i = a] \le \Pr[s_i = a] \cdot \mathop{\mathbf{E}}_{s \sim \sigma}\left[C_i(s_{-i}, b) \mid s_i = a\right]$$

$$\Longrightarrow \mathop{\mathbf{E}}_{s \sim \sigma}[C_i(s) \mid s_i = a] \le \mathop{\mathbf{E}}_{s \sim \sigma}\left[C_i(s_{-i}, b) \mid s_i = a\right]$$

No-Regret Algorithms for Correlated Equilibrium

- We can also define CE through a switching function (play action b when I plan to play a)

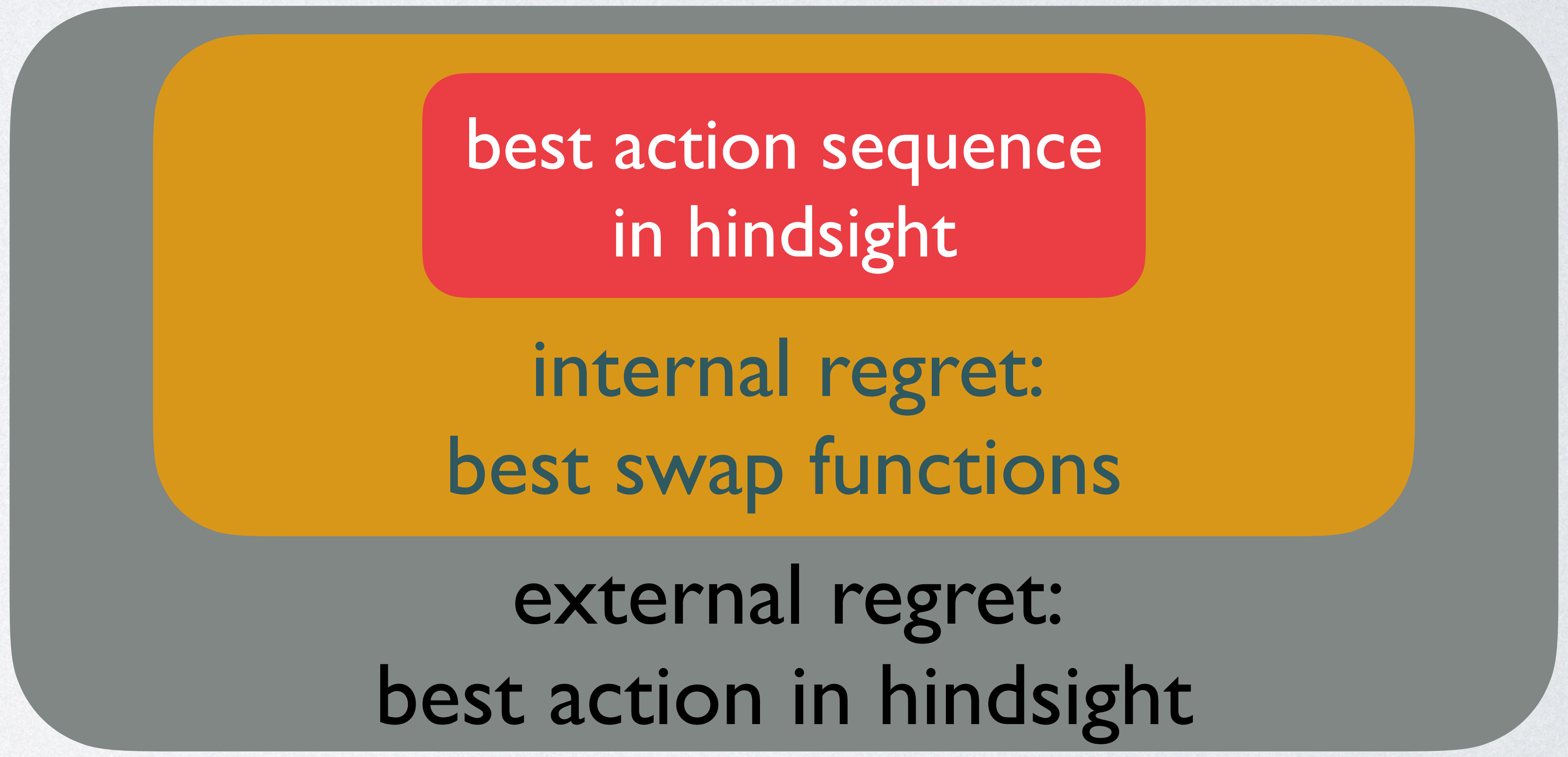
Theorem 7.3.2 *σ is a correlated equilibrium if and only if for all $i \in [k]$ and $\delta : S_i \to S_i$,*

$$\mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s)] \le \mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s_{-i}, \delta(s_i))]. \tag{7.3.1}$$

- We removed the condition requirement of CE at the price of switching function.
- Best action in hindsight is equivalent to a switching function that maps $a^t \to a^*$.
- The new definition relates to swap-regret:

Definition 8.2.2 *The* swap regret *of a sequence of actions $a^1, a^2, \ldots, a^t$ with respect to a switching function $\delta : A \to A$ is*

$$S_T(\delta) = \frac{1}{T}\left(\sum_{t=1}^{T} c^t(a^t) - \sum_{t=1}^{T} c^t(\delta(a^t))\right)$$



- No-(swap)-regret implies no-(external)-regret. Not the other way round.
- To remember: no internal regret $\to$ CE, no external regret $\to$ CCE.

No-Regret Algorithms for Correlated Equilibrium

- For algorithm $\mathscr{A}$, the expected swap regret w.r.t a $\delta : A \to A$ is defined by

$$\mathbf{E}\left[S_T^{\mathscr{A}}(\delta)\right] = \frac{1}{T}\left(\sum_{t=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(a^t\right)\right] - \sum_{t=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(\delta\left(a^t\right)\right)\right]\right)$$

  - Compare to external-regret $\mathbf{E}\left[R_T^{\mathscr{A}}(a)\right] = \frac{1}{T}\left(\sum_{i=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(a^t\right)\right] - \sum_{t=1}^{T} c^t(a)\right)$

- No-swap-regret means $\lim_{T\to\infty} \mathbf{E}[S_T^{\mathscr{A}}(\delta)] = 0, \forall \delta.$

- **Theorem: if all players adopt no-swap-regret algorithms s.t.** $\mathbf{E}[S_T^{\mathscr{A}_i}(\delta)] \le \epsilon, \forall i \in [k], \forall \delta$ **,**
then the average distribution $\sigma = \sum_{t=1}^{T} \prod_{i=1}^{k} \frac{p_i^t}{T}$ **is an** $\epsilon-$**CE, i.e.,** $\mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i(s)\right] \le \mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i\left(s_{-i}, \delta(s_i)\right)\right] + \epsilon$

- proof:

$$\mathop{\mathbf{E}}_{s\sim\alpha}\left[C_i(s)\right] - \mathop{\mathbf{E}}_{s\sim\sigma}[C_i(s_{-i}, \delta(s_i))]$$

Note that $c^t(a) = \mathbf{E}_{s\sim\sigma^t}\left[C_i\left(s_{-i}, a\right)\right]$

$$= \frac{1}{T}\sum_{t=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(a^t\right)\right] - \frac{1}{T}\sum_{t=1}^{T} \mathop{\mathbf{E}}_{a^t \sim p^t}\left[c^t\left(\delta\left(a^t\right)\right)\right]$$

$$= \mathbf{E}[S_T^{\mathscr{A}_i}(\delta)] \le \epsilon$$

No-Swap-Regret Algorithms for Correlated Equilibrium

- **Theorem: if all players adopt no-swap-regret algorithms s.t.** $\mathrm{E}[S_T^{\mathcal{A}_i}(\delta)] \le \epsilon, \forall i \in [k], \forall \delta$ **, then the average distribution** $\sigma = \sum_{t=1}^{T} \prod_{i=1}^{k} \frac{p_i^t}{T}$ **is an** $\epsilon-$**CE, i.e.,** $\mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i(s)\right] \le \mathop{\mathbf{E}}_{s\sim\sigma}\left[C_i\left(s_{-i}, \delta(s_i)\right)\right] + \epsilon$
- The goal is then to develop no-swap-regret algorithms like MWU for external regret.
- In turns out we can transform no-external-regret algorithm into no-swap-regret.
  - The idea: for each action, maintain a no-regret algorithm q_i^t such as MWU
  - Let different no-regret algorithm $(q_1^t, \ldots, q_n^t)$ reach a consensus p^t (see how later).
  - "Lie" to each no-regret algorithm the loss of $p^t(i)c^t(i)$ instead of $c^t(i)$.
- The goal of no-swap-regret is then written as

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} p^t(i)c^t(i) - \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} p^t(i)c^t(\delta(i)) = o(1)$$

No-Swap-Regret Algorithms for Correlated Equilibrium

- The goal of no-swap-regret is then written as

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} p^t(i)c^t(i) - \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} p^t(i)c^t(\delta(i)) = o(1)$$

- Because for each action, we maintain a no-regret algorithm, thus we have

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} q_j^t(i)\left(\boxed{p^t(j)c^t(i)}\right) - \frac{1}{T}\sum_{t=1}^{T} p^t(j)c^t(\delta(j)) \boxed{\leq R_T^j(\delta(j))} = o(1)$$

this is because no regret w.r.t $\delta(j)$

the cost that is lied

playing $\delta(j)$ in hindsight and internal regret is always smaller than external regret

- Sum over $n = |A|$ will not influence the total regret, since no T term is involved.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} \boxed{\sum_{j=1}^{n} q_j^t(i)\left(p^t(j)\right.}\left.c^t(i)\right) - \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{n} p^t(j)c^t(\delta(j)) \leq \sum_{j=1}^{n} R_T^j(\delta(j)) = o(1)$$

- Recall $q_j^t(i)$ is the probability that the j-th no-regret algorithm pick the i-th action. If we make $p^t(i) = \sum_{j=1}^{n} q_j^t(i)p^t(j)$, then we use the above equation prove no-swap-regret.

DONE!

Thanks!