Multi-Agent Reinforcement Learning

Solution Concepts for Games

Stefano V. Albrecht, Filippos Christianos, Lukas Schäfer
Slides by: Leonard Hinckeldey

Multi-Agent Reinforcement Learning: Foundations and Modern Approaches

Stefano V. Albrecht, Filippos Christianos, Lukas Schäfer

Published by MIT Press
(print version will be released on 17 December 2024)

This lecture is based on *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches* by Stefano V. Albrecht, Filippos Christianos and Lukas Schäfer

The book can be downloaded for free at www.marl-book.com.

Lecture Outline

- Joint policy and expected return
- Equilibrium solution concepts
- Additional solution criteria
- Complexity of computing equilibria

Solution Concepts for Games

MARL Problem	=	**Game Model** *e.g. normal-form game, stochastic game, POSG*	+	**Solution Concept** *e.g. Nash equilibrium, social welfare, ...*

What does it mean to act optimally in a multi-agent system?

- Maximizing returns of one agent is no longer enough to determine a solution
- We must consider the **joint policy** of all agents
- This is less straightforward, and there are many different solution concepts

Joint Policy and Expected Return

Joint Policy

The joint policy is the combination of all agents' policies.

- $\pi = (\pi_1, ..., \pi_n)$ is the joint policy

The probability of a joint action under joint policy π can be defined as:

$$\pi(a^\tau | h^\tau) = \prod_{j \in I} \pi_j(a_j^\tau | h_j^\tau)$$

Note

This definition assumes probabilistic independence between agents' policies.

Additional Notation

In the multi-agent setting, we add the following notation:

- $\hat{h} = \{(s^\tau, o^\tau, a^\tau)_{\tau=0}^{t-1}, s^t, o^t\}$ is the **full history** containing:
  - s^τ, states up to $t-1$
  - o^τ, joint observations up to $t-1$
  - a^τ, joint actions up to $t-1$
  - s^t and o^t at current time step t
- $\sigma(\hat{h}^t) = (o^0, ..., o^t)$ is a function that returns the joint observation history from the **full history**
- $\mathcal{O}(o^t | a^{t-1}, s^t)$ is the joint observation probability

History Based Expected Return

Given a joint policy π, we can define the expected return for agent i under π as the probability-weighted sum of returns for agent i under all possible **full histories.**

- Let $\hat{H}$ be a set containing all full histories $\hat{h}^t$ for $t \to \infty$
- then the expected return for agent i under joint policy π is given by:

$$U_i(\pi) = \lim_{t\to\infty} \mathbb{E}_{\hat{h}^t \sim (\mu, \mathcal{T}, \mathcal{O}, \pi)} \left[u_i(\hat{h}^t) \right]$$

$$= \sum_{\hat{h}^t \in \hat{H}} \Pr(\hat{h}^t | \pi) u_i(\hat{h}^t)$$

History Based Expected Return - Continued

The probability of a full history $\Pr(\hat{h}^t|\pi)$ is:

$$\Pr(\hat{h}^t|\pi) = \mu(s^0)\mathcal{O}(o^0|\emptyset, s^0)\prod_{\tau=0}^{t-1}\pi(a^\tau|\hat{h}^\tau)\mathcal{T}(s^{\tau+1}|s^\tau, a^\tau)\mathcal{O}(o^{\tau+1}|a^\tau, s^{\tau+1})$$

$u_i(\hat{h}^t)$ is the discounted return for agent i in the **full history**

$$u_i(\hat{h}^t) = \sum_{\tau=0}^{t-1}\gamma^\tau R_i(s^\tau, a^\tau, s^{\tau+1})$$

Recursive Expected Returns

Expected returns under a **joint policy** can also be defined recursively, analogously to the Bellman recursion.

$$V_i^{\pi}(\hat{h}) = \sum_{a \in A} \pi(a \mid \sigma(\hat{h})) Q_i^{\pi}(\hat{h}, a)$$

We can use this to define a Q function for the individual agent i as follows:

$$Q_i^{\pi}(\hat{h}, a) = \sum_{s' \in S} \mathcal{T}(s' \mid s(\hat{h}), a) \left[\mathcal{R}_i(s(\hat{h}), a, s') + \gamma \sum_{o' \in O} O(o' \mid a, s') V_i^{\pi}(\langle \hat{h}, a, s', o' \rangle) \right]$$

- $s(\hat{h})$ denotes the last state in $\hat{h}$ such that $s(\hat{h}) = s^t$

Recursive Expected Returns - Continued

- $V_i^\pi(\hat{h})$ is the **value** or **expected return** for agent i when agents follow **joint policy** π
- $Q_i^\pi(\hat{h}, a)$ is the **expected return** for agent i when agents execute **joint action** a after $\hat{h}$ and follow π thereafter
- Given the definition for $V_i^\pi(\hat{h})$ and $Q_i^\pi(\hat{h}, a)$, we can define the expected return for agent i at the initial state s^0 as:

$$U_i(\pi) = \mathbb{E}_{s_0 \sim \mu, o_0 \sim \mathcal{O}(\cdot|\emptyset, s_0)} \left[V_i^\pi(\langle s_0, o_0 \rangle) \right]$$

Equilibrium Solution Concepts

Best Response

The **best-response** policy is the policy that maximizes the expected return for agent i against a given set of policies for all other agents $\pi_{-i} = (\pi_1, ..., \pi_{i-1}, \pi_{i+1}, ..., \pi_n)$.

- A best response for agent i to π_{-i} is a policy π_i that maximizes the expected return for agent i when facing π_{-i}

$$\mathrm{BR}_i(\pi_{-i}) = \arg\max_{\pi_i} U_i(\langle \pi_i, \pi_{-1} \rangle)$$

- Where $\langle \pi_i, \pi_{-1} \rangle$ is the entire **joint policy**

Minimax

Minimax is a solution concept defined for **two-agent zero-sum** games. Recall that one agent's reward is the negative of the other agent's reward.

- The existence of minimax solution in **normal-form** games was first proven by John von Neumann (1928)
- Minimax solutions also exist in **two-agent zero-sum stochastic games** with finite episode lengths like chess and Go.

Minimax Definition

In a two-agent, zero-sum game, a joint policy $\pi = (\pi_i, \pi_j)$ is a minimax solution if

$$
\begin{aligned}
U_i(\pi) &= \max_{\pi_i'} \min_{\pi_j'} U_i(\pi_i', \pi_j') && (1) \\
&= \min_{\pi_j'} \max_{\pi_i'} U_i(\pi_i', \pi_j') && (2) \\
&= -U_j(\pi). && (3)
\end{aligned}
$$

- Equation 1 is the minimum expected return agent i can guarantee against any opponent
- Equation 2 is the minimum expected return agent j can **force** on agent i
- A minimax solution is the **best response** to the **worst-case** opponent
- (π_i, π_j) is a minimax solution if $\pi_i \in \mathrm{BR}_i(\pi_j)$ <u>and</u> $\pi_j \in \mathrm{BR}_j(\pi_i)$

Minimax via Linear Programming

We can obtain a minimax solution for non-repeated zero-sum normal-form games by solving two linear programs, one for each agent.

$$
\begin{aligned}
\text{minimize} \quad & U_j^* & \\
\text{subject to} \quad & \sum_{a_i \in A_i} \mathcal{R}_j(a_i, a_j) x_{a_i} \leq U_j^* & \forall a_j \in A_j \\
& x_{a_i} \geq 0 & \forall a_i \in A_i \\
& \sum_{a_i \in A_i} x_{a_i} = 1 &
\end{aligned}
$$

- Minimizing agent j's return U_j^*
- Such that no single action of agent j can receive a greater return than U_j^* when agent i follows $\pi_i(a_i) = x_{a_i}$

Nash Equilibrium

The Nash equilibrium solution concept applies the idea of a **mutual best response** to general-sum games with two or more agents.

- John Nash (1950) proved the existence of such a solution for **general-sum non-repeated normal-from games**
- No agent i can improve its expected returns by changing its policy π_i assuming other agents policies remain fixed

$$\forall i, \pi_i' : U_i(\pi_i', \pi_{-i}) \leq U_i(\pi)$$

- Each agent's policy in the Nash equilibrium is a **best response** to all other agent's policies

Nash Equilibrium in Matrix Games

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

Prisoners Dilemma

	A	B
A	10	0
B	0	10

Coordination Game

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

Rock Paper Scissors

Can you identify the Nash equilibria?

Nash Equilibrium in Matrix Games

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

Prisoners Dilemma
NE at D, D (-3, -3)

	A	B
A	10	0
B	0	10

Coordination Game

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

Rock Paper Scissors

Nash Equilibrium in Matrix Games

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

Prisoners Dilemma
NE at D, D (-3, -3)

	A	B
A	10	0
B	0	10

Coordination Game
Two NE's at A, A (10) and B, B (10)

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

Rock Paper Scissors

Nash Equilibrium in Matrix Games

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

Prisoners Dilemma
NE at D, D (-3, -3)

	A	B
A	10	0
B	0	10

Coordination Game
Two NE's at A, A (10) and B, B (10)

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

Rock Paper Scissors
NE is to choose actions uniformly at random with expected return 0

Folk Theorem in Repeated Normal-Form Games

Folk theorems provide solutions for **repeated normal-form games** showing that any **set** of **feasible** and **enforceable** expected returns $\hat{U}$ can be achieved by an equilibrium solution if agents are far-sighted (γ close to 1).

- $\hat{U}$ is **feasible** if it can be achieved by a joint policy π
- $\hat{U}$ is **enforceable** if each $\hat{U}_i$ is at least as great as the agent's minmax value v_i

$$v_i = \min_{\pi_{-i}^m m} \max_{\pi_i^m m} U_i(\pi_i^m m, \pi_{-i}^m m)$$

- When $\hat{U}$ is enforceable no agent has an incentive to deviate from π, deviating results in other agents **enforcing** the minmax value $v_i \leq \hat{U}_i$

ϵ-Nash Equilibrium

Exact Nash equilibria are difficult to compute:

- In settings with more than two players, action probabilities may be irrational numbers
- Exact Nash equilibria are often computationally too expensive (more on slide 31)
- We can relax the conditions by requiring that no agent can improve its expected return by more than some amount $\epsilon > 0$
- In a general-sum game with n agent, a joint policy π is an ϵ-Nash equilibrium for $\epsilon > 0$ if:

$$\forall i, \pi'_i : U_i(\pi'_i, \pi_{-i}) - \epsilon \leq U_i(\pi)$$

ϵ-Nash Equilibrium can be far from Nash Equilibrium

	C	D
A	100,100	0,0
B	1,2	1,1

- Unique Nash equilibrium at A, C
- ϵ-Nash equilibrium when $\epsilon = 1$ at B, D and A, C
- ϵ-Nash equilibrium may not be a good approximation for the true Nash equilibrium

  $\Rightarrow$ Returns under ϵ-Nash equilibrium can differ greatly from returns under the Nash equilibrium

Correlated Equilibrium

We have previously assumed that the agent's policies are probabilistically independent, which can limit expected returns.

- **Correlated equilibria** allow for correlated policies
- π_c is a central policy that provides a probability distribution across all agents' actions
- Agents can follow action 'recommendations' $\pi_c(a)$ or deviate by choosing another action, represented by action modifier ξ_i
- then a **correlated equilibrium** can be defined as:

$$\sum_{a \in A} \pi_c(a)\mathcal{R}_i(\langle \xi_i(a_i), a_{-i} \rangle) \leq \sum_{a \in A} \pi_c(a)\mathcal{R}_i(a)$$

- Nash equilibria are a special case of correlated equilibria

Correlated Equilibrium Chicken Game

	S	L
S	0,0	7,2
L	2,7	6,6

Non-Correlated Nash Equilibrium:

- Deterministic: $\pi_i(S) = 1, \pi_j(S) = 0 \rightarrow (7, 2)$ and $\pi_i(S) = 0, \pi_j(S) = 1 \rightarrow (2, 7)$
- Probabilistic: $\pi_i(S) = \frac{1}{3}, \pi_j(S) = \frac{1}{3} \rightarrow \approx (4.66, 4.66)$

Correlated Equilibrium Chicken Game

	S	L
S	0,0	7,2
L	2,7	6,6

Correlated Nash Equilibrium:

- $\pi_c(L, L) = \pi_c(S, L) = \pi_c(L, S) = \frac{1}{3}$ and $\pi_c(S, S) = 0$
- Expected return $= 7 \cdot \frac{1}{3} + 2 \cdot \frac{1}{3} + 6 \cdot \frac{1}{3} = 5$
- Assuming knowledge of π_c, if agent i receives recommendation L they know agent j will choose either S or L with probability 0.5
- Thus expected return is $0.5 \cdot 6 + 0.5 \cdot 2 = 4$, which is greater than deviating from the action where the agent i has an expected return $0.5 \cdot 0 + 0.5 \cdot 7 = 3.5$

Coarse Correlated Equilibrium

Coarse correlated equilibria are more general than correlated equilibria where the action modifier ξ_i is not conditioned on the recommended action given by π_C.

- In correlated equilibrium $\xi_i : A_i \to A_i$, such that it takes the recommended action and provides an alternative action
- Coarse correlated equilibrium ξ_i is a constant action
- In other words the agent needs to choose to deviate from the recommended action before seeing it
- The coarse correlated equilibrium plays an important role in no-regret learning discussed in later slides

Correlated Equilibrium via Linear Programming

Similar to a minimax we can solve for correlated equilibria using a linear program for each agent i:

$$
\begin{aligned}
\text{maximise} \quad & \sum_{a \in A} \sum_{i \in I} x_a \mathcal{R}_i(a) \\
\text{subject to} \quad & \sum_{\substack{a \in A \\ a_i = a_i'}} x_a \mathcal{R}_i(a) \geq \sum_{\substack{a \in A \\ a_i = a_i''}} x_a \mathcal{R}_i(a'', a_{-i}) && \forall i \in I, a_i', a_i'' \in A_i \\
& x_a \geq 0 && \forall a \in A \\
& \sum_{a \in A} x_a = 1
\end{aligned}
$$

- The constraint ensures that no agent can increase their return by deviating from the action a_i' sampled under the joint policy $\pi(a) = x_a$, to some other action a_i'

Shortcomings of Equilibrium Solutions

Equilibrium solutions have been adopted as standard solution concepts in MARL but have limitations.

Shortcomings of Equilibrium Solutions

Equilibrium solutions have been adopted as standard solution concepts in MARL but have limitations.

- Sub-optimality:
  - Nash equilibria do not always maximize expected returns
  - E.g., In Prisoner's Dilemma, (D,D) is Nash but (C,C) yields higher returns

Shortcomings of Equilibrium Solutions

Equilibrium solutions have been adopted as standard solution concepts in MARL but have limitations.

- Sub-optimality:
  - Nash equilibria do not always maximize expected returns
  - E.g., In Prisoner's Dilemma, (D,D) is Nash but (C,C) yields higher returns
- Non-uniqueness:
  - There can be multiple (even infinite) equilibria, each with different expected returns

Shortcomings of Equilibrium Solutions

Equilibrium solutions have been adopted as standard solution concepts in MARL but have limitations.

- Sub-optimality:
  - Nash equilibria do not always maximize expected returns
  - E.g., In Prisoner's Dilemma, (D,D) is Nash but (C,C) yields higher returns
- Non-uniqueness:
  - There can be multiple (even infinite) equilibria, each with different expected returns
- Incompleteness:
  - Equilibria for sequential games don't specify actions for off-equilibrium paths i.e. paths not specified by equilibrium policy $\Pr(\hat{h}|\pi) = 0$
  - If there is a temporary disturbance that leads to an **off-equilibrium** path, the equilibrium policy π does not specify actions to return to a **on-equilibrium** path

Refinement Concepts

Pareto Optimality

To address some of these limitations, we can add additional solution requirements such as **Pareto optimality**.

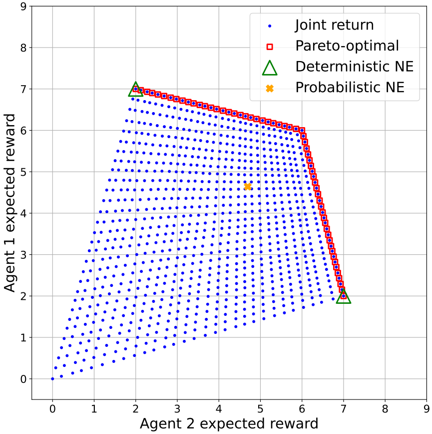
A joint policy π is **Pareto-optimal** if it is not **Pareto-dominated** by any other joint policy. A joint policy π is Pareto-dominated by another policy π' if

$$\forall i : U_i(\pi') \geq U_i(\pi) \quad \text{and} \quad \exists i : U_i(\pi') > U_i(\pi).$$

Intuition

A joint policy is **Pareto-optimal** if there is no other joint policy that improves the expected return for at least one agent without reducing the expected return for any other agent.

Pareto-Optimal Solution in the Chicken Game



	S	L
S	0,0	7,2
L	2,7	6,6

- The figure shows the discretized space of joint policies for the chicken matrix game
- Each blue dot represents the expected joint return obtained by a joint policy

Social Welfare and Fairness

To further constrain the space of desirable solutions, we can consider social welfare and fairness concepts.

Welfare optimality:

$$W(\pi) = \sum_{i \in I} U_i(\pi)$$

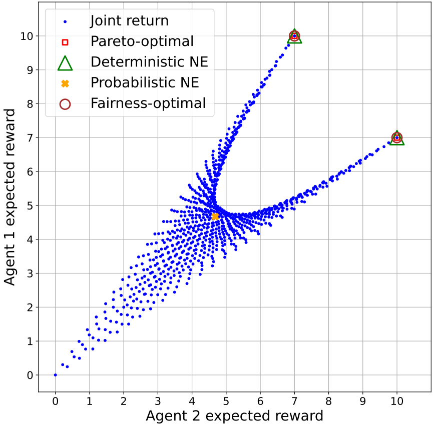
- A joint policy π is **welfare-optimal** if $\pi \in \arg\max_{\pi'} W(\pi')$

Fairness optimality:

$$F(\pi) = \prod_{i \in I} U_i(\pi)$$

- A joint policy π is **fairness-optimal** if $\pi \in \arg\max_{\pi'} F(\pi')$

Fairness in Battle of the Sexes



	A	B
A	10,7	2,2
B	0,0	7,10

- 2 agents agreeing to meet at either location A or B, with each agent having a preference for one or the other location
- A, A and B, B are both Nash equilibria and fairness optimal
- In the chicken game, there is only 1 Pareto optimal and fairness optimal solution

No Regret

Regret measures the difference between the rewards an agent received versus the rewards it would have received by choosing a different action in past episodes.

- In non-repeated normal-form games (assuming actions of other agents are fixed) **regret** is:

$$Regret_i^z = \max_{a_i \in A_i} \sum_{e=1}^{z} \left[\mathcal{R}(\langle a_i, a_{-i}^e \rangle) - \mathcal{R}_i(a^e) \right]$$

- Let a^e denote the joint action in episode $e = 1, ..., z$
- There is no regret if regret is at most 0 as $z \to \infty$
- In general-sum games with n agents, the agents have no regret if:

$$\forall i : \lim_{z \to \infty} \frac{1}{z} Regret_i^z \leq 0$$

No Regret in Prisoners Dilemma

Episode e	1	2	3	4	5	6	7	8	9	10
Action a_1^e	C	C	D	C	D	D	C	D	D	D
Action a_2^e	C	D	C	D	D	D	C	C	D	C
Reward $\mathcal{R}_1(a^e)$	-1	-5	0	-5	-3	-3	-1	0	-3	0
Reward $\mathcal{R}(\langle C, a_2^e \rangle)$	-1	-5	-1	-5	-5	-5	-1	-1	-5	-1
Reward $\mathcal{R}(\langle D, a_2^e \rangle)$	0	-3	0	-3	-3	-3	0	0	-3	0

- Agent 1 receives total reward -21, always playing D would have resulted -15
- Thus, $\text{Regret}_1^{10} = -15 + 21 = 6$

Generalizing No-Regret to Stochastic Games and POSGs

For each agent i we introduce:

- A finite space of policies Π_i from which agent i can select a policy
- Let π^e denote the joint policy from episode $e = 1, \ldots, z$ with $\pi_i^e \in \Pi_i$ for all $i \in I$
- Agent i's regret for not having chosen the best policy across episodes is then defined as

$$Regret_i^z = \max_{\pi_i \in \Pi_i} \sum_{e=1}^{z} \left[U_i(\langle \pi_i, \pi_{-i}^e \rangle) - U_i(\pi^e) \right]$$

Note

This equation is equivalent to the previous equation for normal-form games if each Π_i is a set of **deterministic** policies corresponding to an action $a_i \in A_i$

Complexity of Computing Equilibria

Complexity of computing equilibria

Normal-form games provide a complexity *lower bound* for more complex games.

- Two-agent non-repeated zero-sum games have **polynomial-time** minimax solutions via linear programming
- Correlated equilibria in non-repeated general-sum normal-form games can also be computed in **polynomial time** via linear programming
- Nash equilibria computation is more complex due to the independence assumption and cannot be done using linear programming

Problem

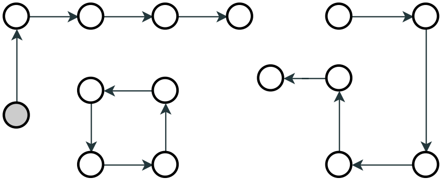
Finding Nash equilibria (NASH problem) is a **total search** problem and has been proven to be **PPAD complete**

END-OF-LINE PPAD Complexity

The END-OF-LINE problem is PPAD complete, meaning all other problems in PPAD, including the NASH problem, can be reduced to it

END-OF-LINE Definition: Let $G(k) = (V, E)$ be a directed graph consisting of

- a finite set V containing 2^k nodes (each node is represented as a bit-string of length k)
- a finite set $E = \{(a, b) \mid a, b \in V\}$ of directed edges (from node a to node b, for $a, b \in V$) such that:
  - if $(a, b) \in E$ then $\exists a' \neq a : (a', b) \in E$ and $\nexists b' \neq b : (a, b') \in E$
- The goal is to find a node $e \neq s$ in this graph using two functions:
  - Parent(v) and Child(v), which return the parent or child node of v, respectively



- The PPAD ”parity argument” ensures the existence of a sink node corresponding to a given source node (in grey)
- If a source node is given we know node *e* must exist
- To find *e* we can start at source and repeatedly call Child(v) until we find *e*
- As the graph scales 2^k this means finding *e* may require **exponential time** in the worst case.

Complexity Considerations for MARL

- **Reduction to NASH:** Computing Brouwer fixed points and other PPAD problems are reducible to the NASH problem, indicating there are no known efficient (polynomial time) algorithms for solving NASH
- **Approximate ϵ-Nash Equilibrium:** PPAD-completeness holds for both approximate solutions ($\epsilon > 0$) and exact solutions ($\epsilon = 0$), with approximations often necessary due to potentially irrational equilibria
- **Implications for MARL:** MARL algorithms are unlikely to be a silver bullet for finding Nash equilibria efficiently
- **Research Focus in MARL:** Research often targets identifying exploitable structures in certain game types, but MARL may still require **exponential** time when such structures are unavailable.

Summary

We covered:

- Best Response and minimax
- Equilibrium solutions
- Additional solution criteria
- Complexity of finding Nash equilibria

Next we'll cover:

- MARL in games