Algorithmic game theory

Martin Balko

7th lecture

November 23rd 2023

Applications of regret minimization

Our notation

Our notation

- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.

Our notation

- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \ldots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \ldots, T$.

Our notation

- Agent A with actions $X = \{1, \dots, N\}$ selects a probability distribution $p^t = (p_1^t, \dots, p_N^t)$ at every step $t = 1, \dots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \dots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \dots, T$.
- The agent A receives loss $\ell_A^t = \sum_{i=1}^N p_i^t \ell_i^t$ at step t.

Our notation

- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \ldots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \ldots, T$.
- The agent A receives loss $\ell_A^t = \sum_{i=1}^N p_i^t \ell_i^t$ at step t. His cumulative loss after all steps is $L_A^T = \sum_{t=1}^T \ell_A^t$.

Our notation

- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \ldots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \ldots, T$.
- The agent A receives loss $\ell_A^t = \sum_{i=1}^{N} p_i^t \ell_i^t$ at step t. His cumulative loss after all steps is $L_A^T = \sum_{t=1}^{T} \ell_A^t$. The cumulative loss of action i is $L_i^T = \sum_{t=1}^{T} \ell_i^t$.

Our notation

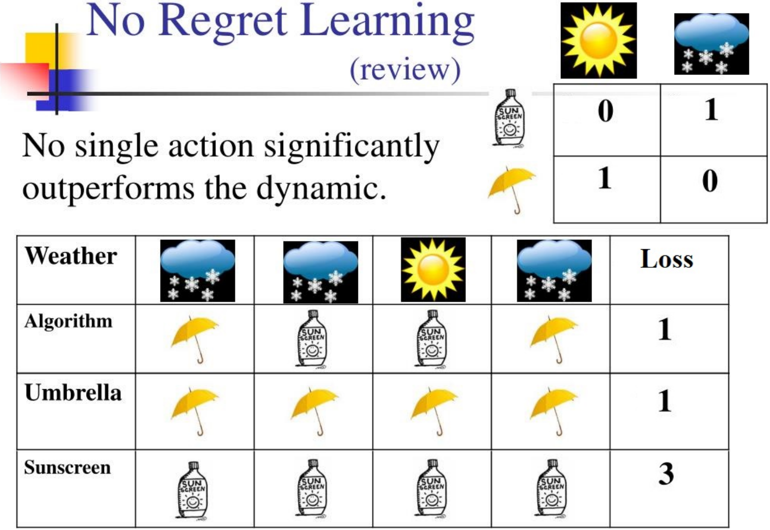
- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \ldots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \ldots, T$.
- The agent A receives loss $\ell_A^t = \sum_{i=1}^N p_i^t \ell_i^t$ at step t. His cumulative loss after all steps is $L_A^T = \sum_{t=1}^T \ell_A^t$. The cumulative loss of action i is $L_i^T = \sum_{t=1}^T \ell_i^t$.
- Given a comparison class $\mathcal{A}_X$ of agents A_i that select a single action i in all steps, we let $L_{min}^T = \min_{i \in X}\{L_{A_i}^T\}$ be the minimum cumulative loss of an agent from $\mathcal{A}_X$.

Our notation

- Agent A with actions $X = \{1, \ldots, N\}$ selects a probability distribution $p^t = (p_1^t, \ldots, p_N^t)$ at every step $t = 1, \ldots, T$.
- After that, the adversary environment selects a loss vector $\ell^t = (\ell_1^t, \ldots, \ell_N^t) \in [-1, 1]^N$ at every step $t = 1, \ldots, T$.
- The agent A receives loss $\ell_A^t = \sum_{i=1}^N p_i^t \ell_i^t$ at step t. His cumulative loss after all steps is $L_A^T = \sum_{t=1}^T \ell_A^t$. The cumulative loss of action i is $L_i^T = \sum_{t=1}^T \ell_i^t$.
- Given a comparison class $\mathcal{A}_X$ of agents A_i that select a single action i in all steps, we let $L_{min}^T = \min_{i \in X}\{L_{A_i}^T\}$ be the minimum cumulative loss of an agent from $\mathcal{A}_X$.
- Our goal is to minimize the external regret $R_A^T = L_A^T - L_{min}^T$.

Example

Example



Source: No regret algorithms in games (Georgios Piliouras)

The polynomial weights algorithm

The polynomial weights algorithm

Algorithm 0.2: POLYNOMIAL WEIGHTS ALGORITHM(X, T, η)

Input : A set of actions $X = \{1, \ldots, N\}$, $T \in \mathbb{N}$, and $\eta \in (0, 1/2]$.
Output : A probability distribution p^t for every $t \in \{1, \ldots, T\}$.
$w_i^1 \leftarrow 1$ for every $i \in X$,
$p^1 \leftarrow (1/N, \ldots, 1/N)$,
for $t = 2, \ldots, T$
do $\begin{cases} w_i^t \leftarrow w_i^{t-1}(1 - \eta \ell_i^{t-1}), \\ W^t \leftarrow \sum_{i \in X} w_i^t, \\ p_i^t \leftarrow w_i^t / W^t \text{ for every } i \in X. \end{cases}$
Output $\{p^t \colon t \in \{1, \ldots, T\}\}$.

The polynomial weights algorithm

Algorithm 0.3: POLYNOMIAL WEIGHTS ALGORITHM(X, T, η)

Input : A set of actions $X = \{1, \ldots, N\}$, $T \in \mathbb{N}$, and $\eta \in (0, 1/2]$.
Output : A probability distribution p^t for every $t \in \{1, \ldots, T\}$.
$w_i^1 \leftarrow 1$ for every $i \in X$,
$p^1 \leftarrow (1/N, \ldots, 1/N)$,
for $t = 2, \ldots, T$

$$\textbf{do} \begin{cases} w_i^t \leftarrow w_i^{t-1}(1 - \eta \ell_i^{t-1}), \\ W^t \leftarrow \sum_{i \in X} w_i^t, \\ p_i^t \leftarrow w_i^t / W^t \text{ for every } i \in X. \end{cases}$$

Output $\{p^t : t \in \{1, \ldots, T\}\}$.

Theorem

For any sequence of loss vectors, we have $R_{\mathrm{PW}}^T \leq 2\sqrt{T \ln N}$.

The No-regret dynamics

The No-regret dynamics

- "Players in a normal-form game play against each other by selecting actions according to the Polynomial-weights algorithm."

The No-regret dynamics

- "Players in a normal-form game play against each other by selecting actions according to the Polynomial-weights algorithm."

Algorithm 0.6: No-regret dynamics(G, T, ε)

Input : A normal-form game $G = (P, A, C)$ of n players, $T \in \mathbb{N}$, and $\varepsilon > 0$.
Output : A prob. distribution p_i^t on A_i for each $i \in P$ and $t \in \{1, \ldots, T\}$.
for every step $t = 1, \ldots, T$

do $\begin{cases} \text{Each player } i \in P \text{ independently chooses a mixed strategy } p_i^t \\ \text{using an algorithm with average regret at most } \varepsilon\text{, with actions} \\ \text{corresponding to pure strategies.} \\ \text{Each player } i \in P \text{ receives a loss vector } \ell_i^t = (\ell_i^t(a_i))_{a_i \in A_i}\text{, where} \\ \ell_i^t(a_i) \leftarrow \mathbb{E}_{a_{-i}^t \sim p_{-i}^t}[C_i(a_i; a_{-i}^t)] \text{ for the product distribution} \\ p_{-i}^t = \prod_{j \neq i} p_j^t. \end{cases}$

Output $\{p^t : t \in \{1, \ldots, T\}\}$.

Application: Modern proof of the Minimax Theorem

Application: Modern proof of the Minimax Theorem

- A new proof of the Minimax theorem.

Application: Modern proof of the Minimax Theorem

- A new proof of the Minimax theorem.
- A zero-sum game $G = (\{1, 2\}, A, C)$ with $A_1 = \{a_1, \ldots, a_m\}$, $A_2 = \{b_1, \ldots, b_n\}$ is represented with an $m \times n$ matrix M where $M_{i,j} = -C_1(a_i, b_j) = C_2(a_i, b_j) \in [-1, 1]$.

Application: Modern proof of the Minimax Theorem

- A new proof of the Minimax theorem.
- A zero-sum game $G = (\{1,2\}, A, C)$ with $A_1 = \{a_1, \ldots, a_m\}$, $A_2 = \{b_1, \ldots, b_n\}$ is represented with an $m \times n$ matrix M where $M_{i,j} = -C_1(a_i, b_j) = C_2(a_i, b_j) \in [-1, 1]$.
- The expected cost $C_2(s)$ for player 2 equals $x^\top M y$, where x and y are the mixed strategy vectors.

Application: Modern proof of the Minimax Theorem

- A new proof of the Minimax theorem.
- A zero-sum game $G = (\{1,2\}, A, C)$ with $A_1 = \{a_1, \ldots, a_m\}$, $A_2 = \{b_1, \ldots, b_n\}$ is represented with an $m \times n$ matrix M where $M_{i,j} = -C_1(a_i, b_j) = C_2(a_i, b_j) \in [-1, 1]$.
- The expected cost $C_2(s)$ for player 2 equals $x^\top My$, where x and y are the mixed strategy vectors.

- The Minimax theorem then states

$$\max_{x \in S_1} \min_{y \in S_2} x^\top My = \min_{y \in S_2} \max_{x \in S_1} x^\top My.$$

Source: https://www.privatdozent.co/

Application: Coarse correlated equilibria

Application: Coarse correlated equilibria

- Recall that a prob. distribution p on A is a correlated equilibrium if

$$\sum_{a_{-i} \in A_{-i}} C_i(a_i; a_{-i}) p(a_i; a_{-i}) \leq \sum_{a_{-i} \in A_{-i}} C_i(a_i'; a_{-i}) p(a_i; a_{-i})$$

for every player $i \in P$ and all pure strategies $a_i, a_i' \in A_i$.

Application: Coarse correlated equilibria

- Recall that a prob. distribution p on A is a correlated equilibrium if

$$\sum_{a_{-i} \in A_{-i}} C_i(a_i; a_{-i}) p(a_i; a_{-i}) \leq \sum_{a_{-i} \in A_{-i}} C_i(a_i'; a_{-i}) p(a_i; a_{-i})$$

  for every player $i \in P$ and all pure strategies $a_i, a_i' \in A_i$.
- In other words,

$$\mathbb{E}_{a \sim p}[C_i(a) \mid a_i] \leq \mathbb{E}_{a \sim p}[C_i(a_i'; a_{-i}) \mid a_i].$$

Application: Coarse correlated equilibria

- Recall that a prob. distribution p on A is a correlated equilibrium if

$$\sum_{a_{-i} \in A_{-i}} C_i(a_i; a_{-i}) p(a_i; a_{-i}) \leq \sum_{a_{-i} \in A_{-i}} C_i(a_i'; a_{-i}) p(a_i; a_{-i})$$

  for every player $i \in P$ and all pure strategies $a_i, a_i' \in A_i$.
- In other words,

$$\mathbb{E}_{a \sim p}[C_i(a) \mid a_i] \leq \mathbb{E}_{a \sim p}[C_i(a_i'; a_{-i}) \mid a_i].$$

- We define an even more tractable concept.

Example: Coarse correlated equilibrium

Example: Coarse correlated equilibrium

- Giving probability $1/6$ to each red outcome gives coarse correlated equilibrium in the Rock-Paper-Scissors game.

	Rock	Paper	Scissors
Rock	(0,0)	(-1,1)	(1,-1)
Paper	(1,-1)	(0,0)	(-1,1)
Scissors	(-1,1)	(1,-1)	(0,0)

Example: Coarse correlated equilibrium

- Giving probability $1/6$ to each red outcome gives coarse correlated equilibrium in the Rock-Paper-Scissors game.

	Rock	Paper	Scissors
Rock	(0,0)	(-1,1)	(1,-1)
Paper	(1,-1)	(0,0)	(-1,1)
Scissors	(-1,1)	(1,-1)	(0,0)

- Then, the expected payoff of each player is 0 and deviating to any pure strategy gives the expected payoff 0.

Example: Coarse correlated equilibrium

- Giving probability $1/6$ to each red outcome gives coarse correlated equilibrium in the Rock-Paper-Scissors game.

	Rock	Paper	Scissors
Rock	(0,0)	(-1,1)	(1,-1)
Paper	(1,-1)	(0,0)	(-1,1)
Scissors	(-1,1)	(1,-1)	(0,0)

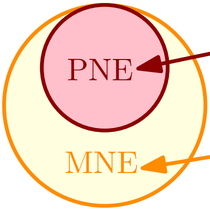
- Then, the expected payoff of each player is 0 and deviating to any pure strategy gives the expected payoff 0.
- It is not a correlated equilibrium though.

Hierarchy of Nash equilibria

Hierarchy of Nash equilibria

Pure Nash equilibria,
not always exist

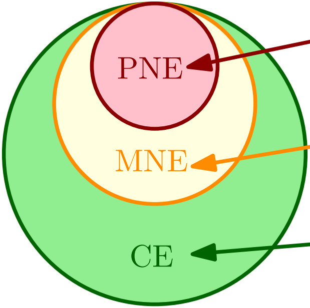
Hierarchy of Nash equilibria



Pure Nash equilibria,
not always exist

Mixed Nash equilibria,
always exist, hard to compute

Hierarchy of Nash equilibria

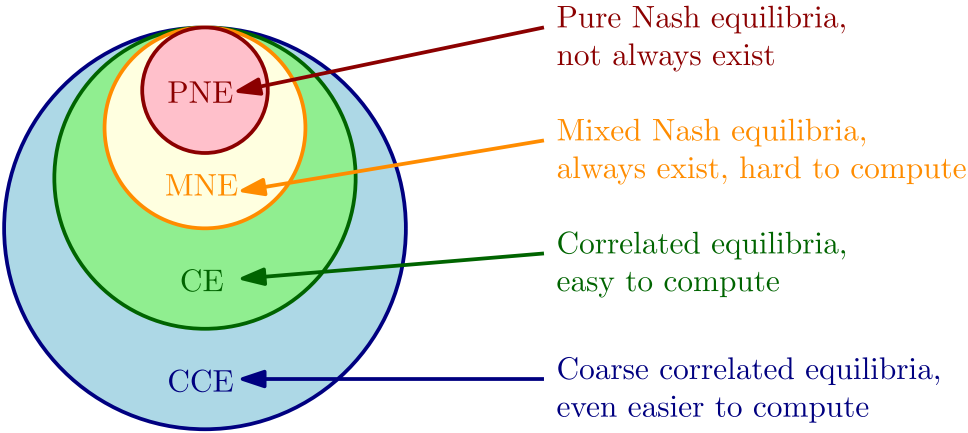


Pure Nash equilibria,
not always exist

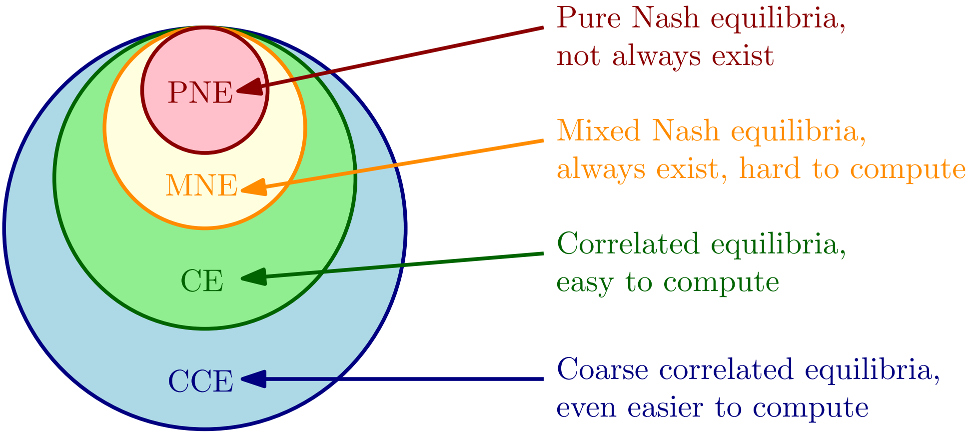
Mixed Nash equilibria,
always exist, hard to compute

Correlated equilibria,
easy to compute

Hierarchy of Nash equilibria



Hierarchy of Nash equilibria



- In general normal-form game, no-regret dynamics converges to a coarse correlated equilibrium.

The No-swap-regret dynamics

The No-swap-regret dynamics

- Using swap regret instead of external regret, we get:

The No-swap-regret dynamics

- Using swap regret instead of external regret, we get:

Algorithm 0.9: NO-SWAP-REGRET DYNAMICS(G, T, ε)

Input : A normal-form game $G = (P, A, C)$ of n players, $T \in \mathbb{N}$, and $\varepsilon > 0$.
Output : A prob. distribution p_i^t on A_i for each $i \in P$ and $t \in \{1, \ldots, T\}$.
for every step $t = 1, \ldots, T$
do Each player $i \in P$ independently chooses a mixed strategy p_i^t using an algorithm with average swap regret at most ε, with actions corresponding to pure strategies.
Each player $i \in P$ receives a loss vector $\ell_i^t = (\ell_i^t(a_i))_{a_i \in A_i}$, where $\ell_i^t(a_i) \leftarrow \mathbb{E}_{a_{-i}^t \sim p_{-i}^t}[C_i(a_i; a_{-i}^t)]$ for the product distribution $p_{-i}^t = \prod_{j \neq i} p_j^t$.
Output $\{p^t : t \in \{1, \ldots, T\}\}$.

The No-swap-regret dynamics

- Using swap regret instead of external regret, we get:

Algorithm 0.10: NO-SWAP-REGRET DYNAMICS(G, T, ε)

Input : A normal-form game $G = (P, A, C)$ of n players, $T \in \mathbb{N}$, and $\varepsilon > 0$.
Output : A prob. distribution p_i^t on A_i for each $i \in P$ and $t \in \{1, \ldots, T\}$.
for every step $t = 1, \ldots, T$
do {
Each player $i \in P$ independently chooses a mixed strategy p_i^t using an algorithm with average swap regret at most ε, with actions corresponding to pure strategies.
Each player $i \in P$ receives a loss vector $\ell_i^t = (\ell_i^t(a_i))_{a_i \in A_i}$, where $\ell_i^t(a_i) \leftarrow \mathbb{E}_{a_{-i}^t \sim p_{-i}^t}[C_i(a_i; a_{-i}^t)]$ for the product distribution $p_{-i}^t = \prod_{j \neq i} p_j^t$.
}
Output $\{p^t : t \in \{1, \ldots, T\}\}$.

- No-swap-regret dynamics then converges to a correlated equilibrium.


WHEN YOU FIND THE NASH EQUILIBRIUM
NO RAGRETS
memegenerator.net

Thank you for your attention.