Algorithmic game theory

Martin Balko

8th lecture

November 30th 2023

Applications of regret minimization

Concluding the story of NE

Concluding the story of NE

- We learned that Nash equilibria (NE) always exist.

Concluding the story of NE

- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.

Concluding the story of NE

- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.
- Therefore we came up with "relaxations" of NE.

Concluding the story of NE

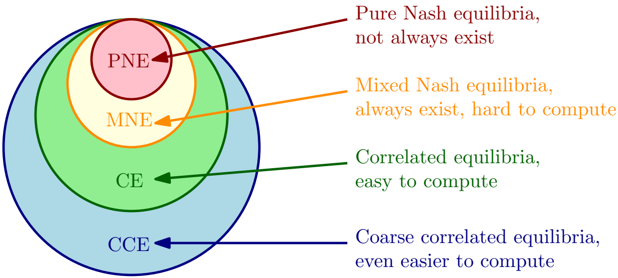
- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.
- Therefore we came up with "relaxations" of NE. Correlated equilibria (CE) look particularly interesting as we can compute them in polynomial time using linear programming.

Concluding the story of NE

- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.
- Therefore we came up with "relaxations" of NE. Correlated equilibria (CE) look particularly interesting as we can compute them in polynomial time using linear programming.
- Using external regret minimization, we can apply no-regret dynamics to converge to coarse correlated equilibria (CCE).

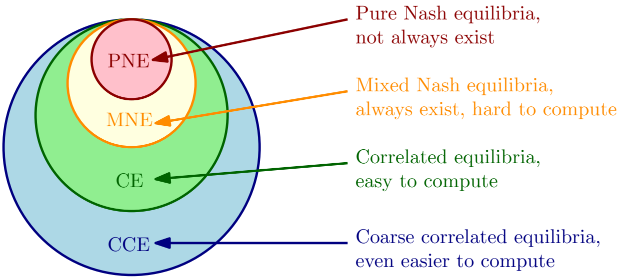
Concluding the story of NE

- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.
- Therefore we came up with "relaxations" of NE. Correlated equilibria (CE) look particularly interesting as we can compute them in polynomial time using linear programming.
- Using external regret minimization, we can apply no-regret dynamics to converge to coarse correlated equilibria (CCE).



Concluding the story of NE

- We learned that Nash equilibria (NE) always exist. However, there seem to be no polynomial algorithm for computing NE.
- Therefore we came up with "relaxations" of NE. Correlated equilibria (CE) look particularly interesting as we can compute them in polynomial time using linear programming.
- Using external regret minimization, we can apply no-regret dynamics to converge to coarse correlated equilibria (CCE).



- Today, we introduce a new notion of regret that will converge to CE.

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. The cumulative loss of A is $L_A^T = \sum_{t=1}^T \ell_A^t$.

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. The cumulative loss of A is $L_A^T = \sum_{t=1}^T \ell_A^t$. The cumulative loss of 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. The cumulative loss of A is $L_A^T = \sum_{t=1}^T \ell_A^t$. The cumulative loss of 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 $\mathcal{A}_X$.

Weather					**Loss**
Algorithm					**1**
Umbrella					**1**
Sunscreen					**3**

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. The cumulative loss of A is $L_A^T = \sum_{t=1}^T \ell_A^t$. The cumulative loss of 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 $\mathcal{A}_X$.

Weather					Loss
Algorithm					1
Umbrella					1
Sunscreen					3

- Our goal is to minimize the external regret $R_A^T = L_A^T - L_{min}^T$.

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 0.3: 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 $t \in \{1, \dots, T\}$.
for every step $t = 1, \dots, T$
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 $\{p^t : t \in \{1, \dots, T\}\}$.

The No-swap-regret dynamics

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

Algorithm 0.4: 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 $t \in \{1, \dots, T\}$.
for every step $t = 1, \dots, T$
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 $\{p^t : t \in \{1, \dots, T\}\}$.

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

Games in extensive form

Games in extensive form

Games in extensive form

- In normal-form games, players act simultaneously resulting in a static description of a game.

Games in extensive form

- In normal-form games, players act simultaneously resulting in a static description of a game.
- Today, we describe a different representation of games which provides a dynamic description where players act sequentially.

Games in extensive form

- In normal-form games, players act simultaneously resulting in a static description of a game.
- Today, we describe a different representation of games which provides a dynamic description where players act sequentially.
- Instead of tables, we describe games using trees.

Zdroj: https://cz.pinterest.com

Games in extensive form

- In normal-form games, players act simultaneously resulting in a static description of a game.
- Today, we describe a different representation of games which provides a dynamic description where players act sequentially.
- Instead of tables, we describe games using trees.

Zdroj: https://cz.pinterest.com

- For some of these games, we show how to compute NE.

Example: normal-form of chess

Example: normal-form of chess

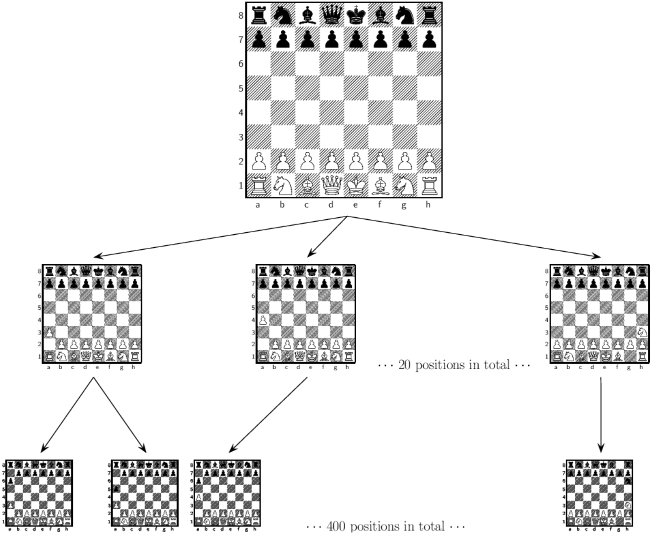
Source: https://edition.cnn.com/

- Chess as a normal-form game: Each action of player $i \in \{\text{black}, \text{white}\}$ is a list of all possible situations that can happen on the board together with the move player i would make in that situation. Then we can simulate the whole game of chess in one round.

Example: extensive form of chess

Example: extensive form of chess

- Root corresponds to the initial position of the chessboard. Each decision node represents a position on the chessboard and its outgoing edges correspond to possible moves in such a position.

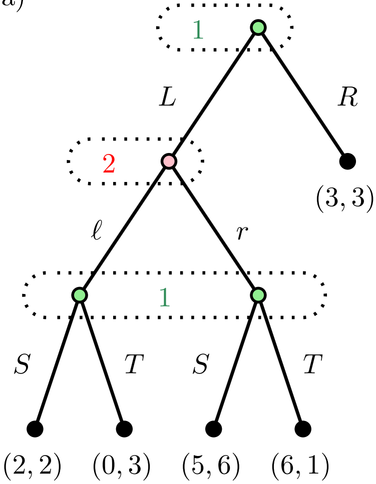


Example

Example

- An example of an imperfect-information game in extensive form (part (a)) and its normal-form (part (b)).

(a)



(b)

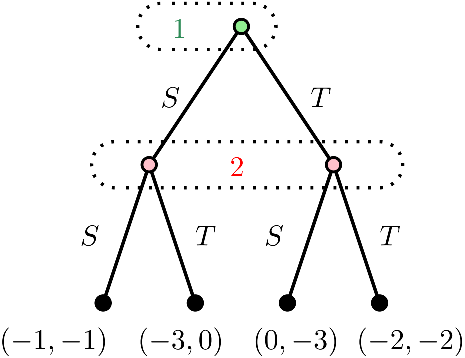
	(ℓ)	(r)
(L, S)	(2,2)	(5,6)
(L, T)	(0,3)	(6,1)
(R, S)	(3,3)	(3,3)
(R, T)	(3,3)	(3,3)

Example: Prisoner's dilemma

Example: Prisoner's dilemma

- Prisoner's dilemma in extensive form (part (a)) and its normal-form (part (b)).

(a)



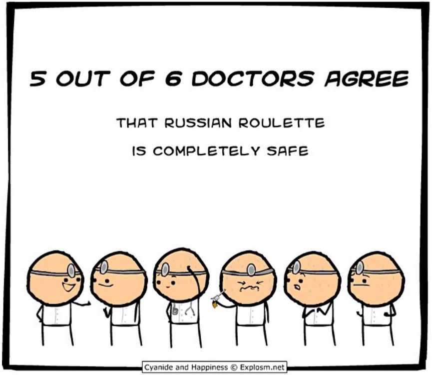
(b)

	T	S
T	(-2,-2)	(0,-3)
S	(-3,0)	(-1,-1)

Example: Russian roulette

Example: Russian roulette

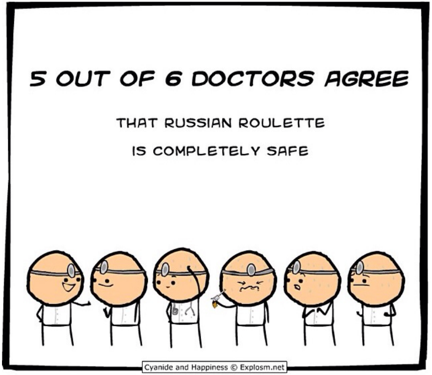
- We have two players with a six-shot revolver containing a single bullet. Each player has two moves: shoot or give up. If player gives up, he loses the game immediately. If he shoots, then he either dies or survives, in which case the other player is on turn.



Source: https://www.memedroid.com/

Example: Russian roulette

- We have two players with a six-shot revolver containing a single bullet. Each player has two moves: shoot or give up. If player gives up, he loses the game immediately. If he shoots, then he either dies or survives, in which case the other player is on turn.



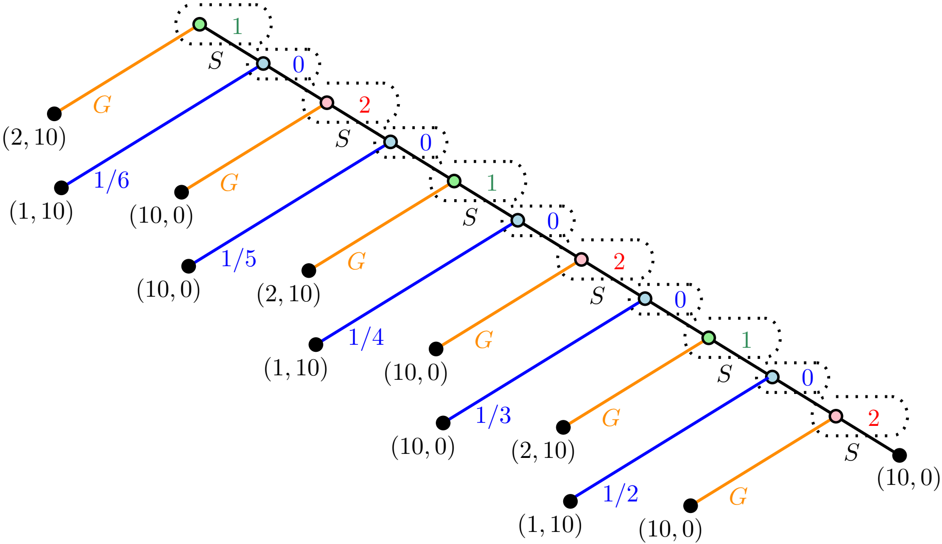
Source: https://www.memedroid.com/

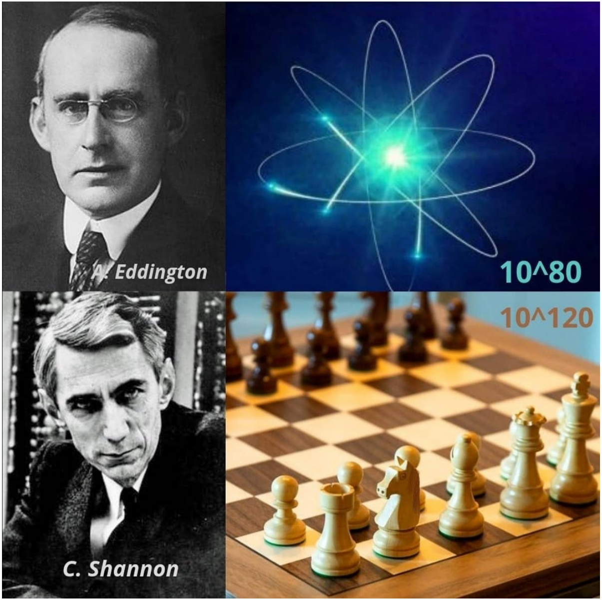
- Consider that player 1 has payoffs $(10, 2, 1)$ for (Win, Loss, Death) and that player 2 has payoffs $(10, 0, 0)$.

Example: Russian roulette

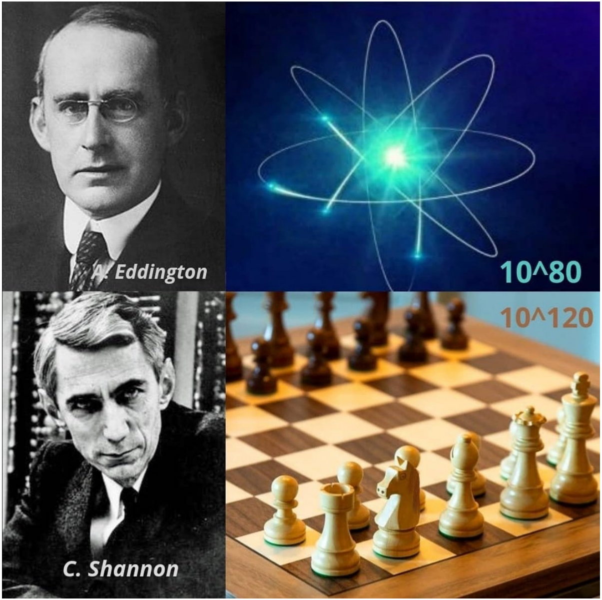
Example: Russian roulette

- The Russian roulette in the extensive form using the random player.





Source: https://twitter.com/curiosite12



Source: https://twitter.com/curiosite12

Thank you for your attention.