Harvard CS 1360, Spring 2025
Instructor: Ariel Procaccia
Scribe: Taig Singh

Equilibrium Computation

—

Lecture 2

Nash Equilibrium Computation Complexity

To introduce the complexity class that Nash equilibrium computation falls under, we will begin by defining the End of the Line problem.

Definition 1 (End of the Line problem)**.** The input is a directed graph $G = (V, E)$ with $V = \{0, 1\}^n$, where every vertex has at most one predecessor and at most one successor. The edges E are implicitly given by a polynomial-time computable functions $f_p : \{0, 1\}^n \to \{0, 1\}^n$ and $f_s : \{0, 1\}^n \to \{0, 1\}^n$ that respectively return the predecessor and successor of a given vertex (if they exist). Given a source vertex (no predecessor), the problem is to find a sink (no successor).

For any input to End of the Line, the existence of a sink vertex is guaranteed - but how do you find it?

Definition 2 (TFNP (total function NP))**.** The complexity class TFNP includes problems that are guaranteed to have a solution, and this solution can be checked in polynomial time.

An example of a problem in TFNP is the factoring problem - given an integer $n \geq 2$, find a prime factor of n. A solution always exists and the validity of any factorization can be quickly checked.

Definition 3 (PPAD (polynomial parity arguments on directed graphs))**.** The complexity class PPAD includes all problems in TFNP that have polynomial-time reductions to the End of the Line problem.

This complexity class was introduced by Christos Papadimitriou (1949 - present) in 1994. He was an influential theoretical computer scientist and a founder of algorithmic game theory. A depiction of how PPAD relates to the other complexity classes you have learned about is shown in Figure 1.

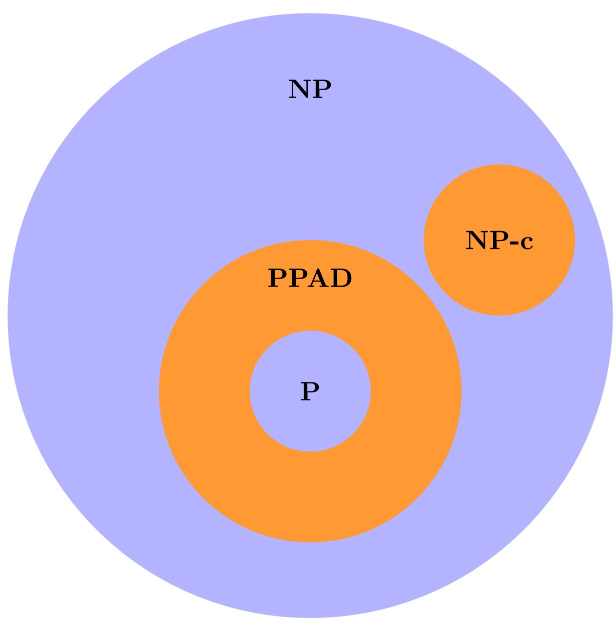


Figure 1: How PPAD relates to other complexity classes

Note that PPAD problems cannot be NP-complete as problems that are NP-complete are decision problems, but there is always a solution to a problem in PPAD.

Theorem 1. *For all $n \geq 2$, computing an (approximate) Nash equilibrium in an n-player normal-form game is PPAD-complete.*

Note that while finding a Nash equilibrium in an n-player normal-form game is PPAD-complete and is not NP-complete as a solution always exists, the problem of finding a second Nash equilibrium for that game is NP-complete.

An Introduction to Linear Programming

From here, we will take a brief interlude to discuss linear programming: an optimization technique through which solutions can be identified in polynomial time. This will later help us show that a few equilibrium concepts are in P.

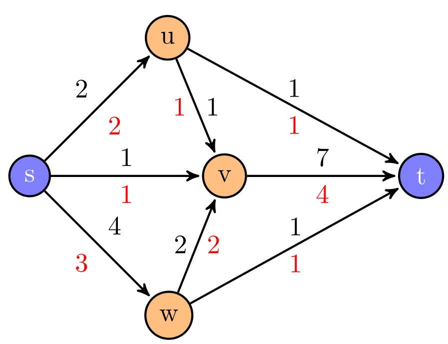
Definition 4 (Linear Programming)**.** Linear programming (LP) is an optimization technique used to find the best outcome within a mathematical model, subject to a set of linear constraints. The objective function and all constraints are linear. An LP problem takes on the following form:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & A\mathbf{x} = \mathbf{a}, \\ & B\mathbf{x} \leq \mathbf{b} \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the optimization variable, and $\mathbf{c} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $\mathbf{a} \in \mathbb{R}^m$, $B \in \mathbb{R}^{k \times n}$, $\mathbf{b} \in \mathbb{R}^k$ are the problem data.

Linear programs can be solved in polynomial time using interior-point methods.

Example 1 (The Max Flow Problem)**.** In the max flow problem, we are given a directed graph $G = (V, E)$ with a source s and a sink t, and a capacity α_{xy} for each $(x, y) \in E$. A flow is a function $f : E \to \mathbb{R}^+$ that satisfies $f_{xy} \leq \alpha_{xy}$ (the flow from x to y is at most the capacity from x to y) for all $(x, y) \in E$ and for all $x \in V \backslash s, t$, $\sum_{(y,x)\in E} f_{yx} = \sum_{(x,z)\in E} f_{xz}$ (the flow into x is equal to the flow out of x). The value of a flow is $\sum_{(s,x)\in E} f_{sx}$. We wish to find a value-maximizing flow. In the example below, the value of the max flow is 6, where the capacities are displayed above edges in black and the flow is displayed below edges in red.



We can formulate the max flow problem for this example using an LP:

$$
\begin{aligned}
\max \quad & f_{su} + f_{sv} + f_{sw} \\
\text{s.t.} \quad & 0 \le f_{su} \le 2 \\
& 0 \le f_{sv} \le 1 \\
& 0 \le f_{sw} \le 4 \\
& 0 \le f_{uv} \le 1 \\
& 0 \le f_{wv} \le 2 \\
& 0 \le f_{ut} \le 1 \\
& 0 \le f_{vt} \le 7 \\
& 0 \le f_{wt} \le 1 \\
& f_{su} = f_{uv} + f_{ut} \\
& f_{sv} + f_{uv} + f_{wv} = f_{vt} \\
& f_{sw} = f_{wv} + f_{wt}
\end{aligned}
$$

Equilibrium Concepts in P

Going back to our discussion of the complexity of equilibrium concepts, we will first restrict the game (all normal-form games to zero-sum games) and then expand the solution (Nash equilibria to correlated equilibria) to identify equilibrium concepts in P. We will leverage our new tool of linear programming to show these concepts can be computed in polynomial time.

Definition 5 (Two-Player Zero-Sum Games)**.** In two-player zero-sum games, it holds that for every strategy profile $\mathbf{s}$,

$$u_1(\mathbf{s}) = -u_2(\mathbf{s})$$

Definition 6 ((Randomized) Maximin and Minimax)**.** The Maximin (randomized) strategy of player 1 is

$$x_1^\star \in \arg \max_{x_1 \in \Delta(S_1)} \min_{s_2 \in S_2} u_1(x_1, s_2)$$

The Minimax (randomized) strategy of player 2 is

$$x_2^\star \in \arg \min_{x_2 \in \Delta(S_2)} \max_{s_1 \in S_1} u_1(s_1, x_2)$$

In player 1's Maximin strategy, they are finding a strategy x_1 that maximizes their utility knowing that player 2 will choose a strategy s_2 that will minimize player 1's utility. In player 2's Minimax strategy, they are finding a strategy x_2 that minimizes player 1's utility knowing that player 1 will choose a strategy s_1 that will maximize player 1's utility.

Example 2. Consider a two-player zero-sum game with the following payoff matrix:

$-1, 1$	$2, -2$
$2, -2$	$-2, 2$

If $x_1^\star = (p, 1-p)$, we wish to find p. If player 1 plays $x_1 = (p, 1-p)$, then if player 2 plays $s_2 = 0$

$$u_1(x_1, s_2) = -p + 2 \cdot (1-p) = -3p + 2$$

and if player 2 plays $s_2 = 1$, then

$$u_1(x_1, s_2) = 2p - 2 \cdot (1-p) = 4p - 2$$

Note that the first expression is smaller when $p \ge \frac{4}{7}$ and the second expression is smaller when $p \le \frac{4}{7}$. Thus, player 1 wishes to choose p that maximizes

$$\min_{s_2 \in S_2} u_1(x_1, s_2) = \begin{cases} -3p + 2 & p \geq \frac{4}{7} \\ 4p - 2 & p < \frac{4}{7} \end{cases}$$

Therefore player 1 will choose $p = \frac{4}{7}$ to maximize this value, and thus $x_1^\star = (\frac{4}{7}, \frac{3}{7})$

The Maximin strategy is computed via LP (and the minimax strategy is computed analogously):

$$\begin{aligned} \max \quad & w \\ \text{s.t.} \quad & \forall s_2 \in S, \quad \sum_{s_1 \in S} p(s_1) u_1(s_1, s_2) \geq w, \\ & \sum_{s_1 \in S} p(s_1) = 1, \\ & \forall s_1 \in S, \quad p(s_1) \geq 0. \end{aligned}$$

Here, w is the maximum utility player 1 can achieve when player 2 minimizes player 1's utility. The first set of constraints says that every strategy player 2 plays must result in a utility for player 1 of at least w. The second and third constraints are just trivial constraints on the probabilities in player 1's mixed strategy.

Theorem 2 (von Neumann 1928)**.** *Every 2-player zero-sum game has a unique value v such that player 1 can guarantee utility at least v and player 2 can guarantee utility at least $-v$.*

John von Neumann (1903 - 1957) was a founder of game theory. He was also known for revolutionary contributions to mathematics, physics, computer science, and the Manhattan Project.

Proof. We will give this proof by starting with Nash's Theorem. Let (x_1, x_2) be a Nash equilibrium and denote $v = u_1(x_1, x_2)$. For every $s_2 \in S_2$, $u_1(x_1, s_2) \geq v$ as (x_1, x_2) is a Nash equilibrium so player 1 can guarantee at least v by playing x_1. Similarly, every $s_1 \in S_1$, $u_2(s_1, x_2) \geq -v$ so player 1 can guarantee at least $-v$ by playing x_2. □

This proof will be given from scratch later in the course.

Definition 7 (Correlated Equilibrium)**.** Let $N = \{1, 2\}$ for simplicity. Let p be a distribution over all pairs of strategies S^2. Now, assume a mediator chooses (s_1, s_2) according to p and reveals s_1 to player 1 and s_2 to player 2. Player 1 is best responding if they are playing their most preferred strategy knowing the conditional distribution of player 2's strategies, given that player 1 got s_1. In other words, player 1 is best responding if for all $s_1' \in S$,

$$\sum_{s_2 \in S} P(s_2|s_1) u_1(s_1, s_2) \geq \sum_{s_2 \in S} P(s_2|s_1) u_1(s_1', s_2)$$

However, since player 1 got s_1 from the mediator, they know the distribution over strategies for player 2 is

$$P(s_2|s_1) = \frac{P(s_1 \wedge s_2)}{P(s_1)} = \frac{p(s_1, s_2)}{\sum_{s_2' \in S} p(s_1, s_2')}$$

Substituting this expression above, we get that player 1 is best responding if for all $s_1' \in S$,

$$\sum_{s_2 \in S} p(s_1, s_2) u_1(s_1, s_2) \geq \sum_{s_2 \in S} p(s_1, s_2) u_1(s_1', s_2)$$

p is a correlated equilibrium (CE) if both players are best responding.

Note that Nash equilibria are special cases of correlated equilibria when each player's actions are drawn from an independent distribution. In Nash equilibria, conditioning on s_1 provides no additional information about s_2 and vice versa, but both players are still best responding to each other.

Example 3 (Game of Chicken)**.** Consider the classic game of chicken with the following payoff matrix:

	Dare	Chicken
Dare	$0, 0$	$4, 1$
Chicken	$1, 4$	$3, 3$

Social welfare is defined as the sum of utilities. The pure NE in this game are (C, D) and (D, C) in which case the social welfare is 5. The mixed NE is $(1/2, 1/2)$ in which case the social welfare is 4. However, the optimal social welfare is 6.

Consider the following correlated equilibrium: (D, D) played with probability 0, (D, C) played with probability $\frac{1}{3}$, (C, D) played with probability $\frac{1}{3}$, (C, C) played with probability $\frac{1}{3}$. In this case, the social welfare is $\frac{16}{3}$, which is higher than both the pure and mixed NEs! To verify that this is indeed a correlated equilibrium, we can go through each possible strategy for each player and verify that neither player would be better off switching their strategy knowing the conditional distribution of the other player's strategies.

To implement a CE, one would need to implement the mediator. In the case of the correlated equilibrium above in the game of chicken, this can be implemented by putting two "chicken" balls and one "dare" ball in a hat, and having each blindfolded player pick a ball. This implementation achieves the desired joint distribution because (D, D) can never be selected, and (C, C), (C, D), and (D, C) are all equally likely to be selected.

We can compute CEs via linear programming in polynomial time:

$$
\begin{aligned}
& \text{find} \quad \forall s_1, s_2 \in S, \quad p(s_1, s_2) \\
\text{s.t.} \quad & \forall s_1, s_1' \in S, \quad \sum_{s_2 \in S} p(s_1, s_2) u_1(s_1, s_2) \geq \sum_{s_2 \in S} p(s_1, s_2) u_1(s_1', s_2), \\
& \forall s_2, s_2' \in S, \quad \sum_{s_1 \in S} p(s_1, s_2) u_2(s_1, s_2) \geq \sum_{s_1 \in S} p(s_1, s_2) u_2(s_1, s_2'), \\
& \sum_{s_1, s_2 \in S} p(s_1, s_2) = 1, \\
& \forall s_1, s_2 \in S, \quad p(s_1, s_2) \in [0, 1].
\end{aligned}
$$

Here, we are finding the distribution p over S^2 that satisfies the four sets of constraints. The first set of constraints ensures that player 1's strategies are the best responses given the conditional distributions of player 2's strategies under p. Similarly, the second set of constraints ensures that player 2's strategies are the best responses given the conditional distributions of player 1's strategies under p. The third and fourth sets of constraints merely ensure p is a probability distribution over S^2.