Algorithmic Economics CIS 6930/CIS 4930 Spring 2018

Lecture — Jan. 11 - Jan 16

Lecturer: Dr. Meera Sitharam Scribe: Rahul Prabhu

1 Introduction

Before going into the definition of what a game is, we first define some background terms.

1.1 Background

Definition 1 (Topological space[1]) *A topological space, also called an abstract topological space, is a set X together with a collection of open subsets T that satisfies the four conditions:*

1. *The empty set $\varnothing$ is in T.*
2. *X is in T.*
3. *The intersection of a finite number of sets in T is also in T.*
4. *The union of an arbitrary number of sets in T is also in T.*

Definition 2 (Convex Set[5]) *A convex set is a region where, for every pair of points within the region, every point on the straight line segment that joins the two points is also in the set.*

Definition 3 (Closed Set[2]) *A topological space that contains all its limit points.*

Definition 4 (Compact space[4]) *A topological space that is closed and bounded.*

Definition 5 (Concave function[3]) *$f : \Omega \to \mathbb{R}$ is concave if for any x and y in the interval and for any $\alpha \in [0, 1]$*

$$f((1 - \alpha)x + \alpha y) \geqslant (1 - \alpha)f(x) + \alpha f(y)$$

Alternatively, a function $f : \Omega \to \mathbb{R}$ is concave if the set $\forall_t\ \Omega_t = \{x : f(x) \geqslant t\}$ is convex.

1.2 Games

In this section we introduce basic terminology in game theory.

Definition 6 (Strategy Space) *A strategy space in an n player game is $\Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_n$, where Σ_i is the set of strategies for player i.*

Definition 7 (Strategy Profile) *A strategy profile in an n player game is $s \in \Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_n$.*

Definition 8 (Game) *A game is a tuple $([n], S, U)$, where $[n]$ is the set of players, $S = \Sigma_1 \times \Sigma_2, \times \cdots \times \Sigma_n$ is the strategy space, $U = \bigcup_i U_i$, and $U_i : \Sigma_1 \times \Sigma_2, \times \cdots \times \Sigma_n \to \mathbb{R}$ is the utility function for player i.*

Definition 9 (Finite Game) *A game in which there are a finite number of players and each player has a finite number of strategies (Σ_i for each player is finite).*

Definition 10 (Pure Strategy) *A player picks only one of discrete possible choices of strategy from Σ.*

Definition 11 (Mixed Strategy) *A mixed strategy is a probability density function defined over a strategy space.*

Mixed strategy space of a pure strategy game can be be represented as convex combinations of the original strategies.

Lemma 12 *The mixed strategy space corresponding to a pure strategy space is convex.*

Definition 13 (Symmetric game) *A two player game is symmetric if $A = B^T$, where A and B are the payoff matrices of the two players.*

Definition 14 (Zero sum game) *A game is called a zero sum game if $\Sigma_i u_i(s) = 0$*

Definition 15 (Countering strategy) *A strategy profile s' counters s if*

$$\forall_i \ \forall_t \ u_i(s_1, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots s_n) \geqslant u_i(s_1, \ldots s_{i-1}, t, s_{i+1}, \ldots s_n)$$

Note: s_{-1} is used to denote $(s_1, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots s_n)$

Definition 16 (Nash Equilibrium) *A Nash equilibrium in a game is a strategy that counters itself.*

Definition 17 (Pareto Equilibrium) *Pareto Equilibrium is a strategy (profile) s such that*

$$\forall_i u_i(s_i, s_{-i}) \geqslant u_i(t, s_{-i}) \forall_t : \forall_j u_j(t, s_{-j}) \geqslant u_j(s_j, s_{-j})$$

Note that the extra qualification for t filters strategies that hurt others.

Definition 18 (Dominant strategy) *A strategy is dominant for a player i if the following holds*

$$\forall_{s_i} \forall_t \ u_i(s_{-i}, s_i) > u_i(s_{-i}, t)$$

2 Nash Equilibrium

Theorem 19 (Existence of a Nash equilibrium[7]) *If a game has a convex strategy profile space $\Sigma_1 \times \Sigma_2 \times \ldots \Sigma_n$ and concave utility function u_i, then the game has a Nash equilibrium.*

Note that Nash's theorem gives a sufficient condition for the existence of a Nash equilibrium. While Nash's theorem gives a way to verify whether a mixed strategy game has a Nash equilibrium or not, it cannot be used to analyze pure strategy finite games, since the utility functions are discrete and hence not continuous and convex. There exist alternate sufficient conditions for the existence of Nash equilibrium in pure strategy games. One such condition is shown in the following theorem.

It is interesting to note that often, there is a simple mechanism that converts a game into a dominant strategy and hence strategy-proof game; this is especially interesting in games without perfect information - by making them strategy-proof no one has any incentive to hide anything and it doesn't matter that it is not a perfect-information game any more.

For instance, consider auctions as games where, there is a product that is being auctioned and there are several bidders for this product. Each bidder or player in the game has a certain valuation of the product. If a player wins the bid, his utility is the difference between his bid and his valuation. If he doesn't win, his utility is zero. A first price auction where, the highest bidder wins and pays his bid amount is generally a game without perfect information. I.e., players generally do not know each other's valuation of the product. This game can be altered by making it a second price or Vickrey auction where, the highest bidder wins, but only pays the amount of the second highest bid. This compels players to bid their valuation of the product. As they only pay the second price amount, they will always have a positive utility.

Theorem 20 (Existence of Nash Equilibrium for Pure strategy games) *In an n player game, if every player has a dominant strategy, then there exists a Nash equilibrium.*

A dominant strategy implies that the game is strategy proof and that players don't have to care about other players' strategy at all. This makes the game uninteresting and most real world games do not have dominant strategies. Next we give the proof of Nash's theorem. But before that, we discuss the following theorems which will be used in the proof of Nash's theorem.

Theorem 21 (Brouwer's fixed point theorem[8]) *A continuous function $f : \Omega \to \Omega$ on a compact convex set of a topological space has a point x where $f(x) = x$.*

Theorem 22 (Kakutani's fixed point theorem [6]) *Let Ω be a non-empty, compact and convex subset of some Euclidean space R^n. Let $f : \Omega \to 2^\Omega$ be a set-valued function on Ω with a closed graph and the property that $f(x)$ is non-empty and convex for all $x \in \Omega$. Then Ω has a point x such that $x \in f(x)$.*

Proof: [Existence of Nash's equilibrium]
Define $\Delta : \Omega \to 2^\Omega$, where $\Omega = \Sigma_1 \times \ldots \times \Sigma_n$.

In particular for strategy profile s in Ω, $\Delta(s)$ will be the set of all strategy profiles s' that counter s. To do this, we define $\Delta(s)$ as the intersection of $\Delta_i(s)$ where $\Delta_i(s)$ is the set of strategies s' that counter s for the i^{th} player. A strategy profile s' counters a strategy profile s for the i^{th} player if $\forall_t \; u_i(s'_i, s_{-i}) > u(r_i, s_{-i})$. Each $\Delta_i(s)$ is convex since the utility function for the i^{th} player is concave. so $\Delta(s)$ is a finite intersection of convex sets and hence also convex. Since the utility functions are continuous, the function Δ is continuous. So Δ is a function that satisfies the conditions of Kakutani's theorem So there is a s such that $s \in \Delta(s)$, i.e, s counters itself and hence is a Nash equilibrium. ■

3 Analysis of Games

3.1 Prisoner's Dilemma

Prisoner's dilemma is a 2 player game where the prisoners are on trial for a crime. Each prisoner has two choices, confessing or staying silent. Their pay off for each of these choices are as given in Table 1.

A/B	Confess	Silent
Confess	4, 4	1, 5
Silent	5, 1	2, 2

Table 1: Prisoner's Dilemma

The prisoner's dilemma game has a pure strategy Nash equilibrium when both prisoners confess. Though this is does not give the best payoff for either player, in any other state of the game, each player has something to gain by changing their strategy. Next we give an algorithm

3.2 Matching Pennies

The matching pennies is a 2 player game where two coins are tossed and if both coins have the same side (i.e., both heads or both tails) player A wins, otherwise player B wins. The payoff matrix for this game is shown in Table 2.

As can be seen from the payoff matrix, there is no pure strategy Nash equilibrium for this game. However, as we'll discuss later, the game has a mixed strategy Nash equilibrium.

A/B	Confess	Silent
Confess	1, -1	-1, 1
Silent	-1, 1	1, -1

Table 2: Matching Pennies

3.3 Tragedy of the commons

Tragedy of the commons is an n player game where there is a finite amount of resource that the n players need to share. If player i uses x_i amount of resource, then their utility is given by

$$\begin{aligned} u_i(x) &= 0 && \text{if } (\Sigma_i x_i > 1) \\ &= x_i(1 - \Sigma_j x_j) && \text{otherwise} \end{aligned} \tag{1}$$

To maximize his own utility, each player tries to utilize as much of the resources as possible. However, if the total consumption increases beyond the available resources, nobody gets anything.

To understand the Nash equilibrium in this game, let us consider the game from the perspective of a single player i. Let us assume that $t = \Sigma_{j \neq i} x_j$ resource is utilized by all other players. From i's perspective, this is a simple optimization problem. Consuming x resource results in a utility of $x(1-t-x)$ which gives an optimal utility of $x = \frac{1-t}{2}$. This yields a stable state when each player is being selfish and maximizing their own utility, i.e., $x_i = \frac{(1-\Sigma_{j \neq i} x_j)}{2}$. This has a unique solution when $x_i = 1/(n+1)$ for all i.

Notice that this solution has extremely low resource utilization. The utility for each player is $x_i = \frac{(1-\Sigma_{j \neq i} x_j)}{2} = \frac{1}{(n+1)^2}$. Thus, the total resource utilization is only $1/n$. Each user, by being selfish overuses the resource, driving the overall utilization down.

3.4 Pollution Game

This is an extension of the prisoner's dilemma to n players. In this game there are n countries. Each country can either pass legislation to control pollution or choose not to do so. Assume that passing legislation costs 3 units for each country. Not passing legislation adds a cost of 1 for every country in the game.

The way the game is set up, the cost of controlling pollution is clearly higher than the cost of not controlling it. Suppose k countries out of the n choose not to control pollution. The cost incurred by each of these k countries is k, but the cost incurred by the other $n-k$ countries is $k+3$ each.

The only Nash equilibrium in this game is when no country passes legislation and the cost for each country is n. But, a far better solution to the game is when all countries pass legislation and the cost incurred by each country is only 3.

References

[1] Topological space, 2017. `http://mathworld.wolfram.com/TopologicalSpace.html` [Online; accessed 16-January-2018].

[2] Wikipedia contributors. Closed set — wikipedia, the free encyclopedia, 2017. `https://en.wikipedia.org/w/index.php?title=Closed_set&oldid=807441968` [Online; accessed 19-January-2018].

[3] Wikipedia contributors. Concave function — wikipedia, the free encyclopedia, 2017. `https://en.wikipedia.org/w/index.php?title=Concave_function&oldid=807907041` [Online; accessed 16-January-2018].

[4] Wikipedia contributors. Compact space — wikipedia, the free encyclopedia, 2018. `https://en.wikipedia.org/w/index.php?title=Compact_space&oldid=819937624` [Online; accessed 19-January-2018].

[5] Wikipedia contributors. Convex set — wikipedia, the free encyclopedia, 2018. `https://en.wikipedia.org/w/index.php?title=Convex_set&oldid=821236852` [Online; accessed 22-January-2018].

[6] Shizuo Kakutani. A generalization of brouwer's fixed point theorem. 1941.

[7] John F Nash. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

[8] David Roger Smart. *Fixed point theorems*, volume 66. CUP Archive, 1980.

4 Analysis of Games

4.1 Battle of the Sexes

Battle of the sexes is a two player game where a boy and a girl , are deciding how to spend their evening. They have two options, going to a baseball game or going to a softball game. The boy prefers baseball and the girl prefers softball, but they would like to spend the evening together rather than separately. The payoffs for each of these scenarios is captured in the table payoff matrix shown in Table 3.

Girl/Boy	Baseball	Softball
Baseball	5, 6	1, 1
Softball	2, 2	6, 5

Table 3: Battle of the Sexes

In this game, the strategy profiles where the boy and the girl spend the evening separately from each other are not stable. Assuming that the other player's strategy remains unchanged, each player has a choice of switching their strategies and getting a better payoff. The two strategy profiles where the players spend the evening together are stable. Assuming the other player's strategy is unchanged, each player is getting the best payoff with their current choice of strategies. Thus this game has two Nash equilibria.

4.2 Cars at an Intersection

In this game, two cars approach an intersection at the same time. If both attempt to cross at the same time, there will be a fatal collision. The payoff matrix for this game is as shown in Table 4

1/2	Cross	Stop
Cross	-100, -100	1, 0
Stop	0, 1	0, 0

Table 4: Cars at an Intersection

Considered as a pure strategy game, this game has two Nash equilibria and correspond to the states where one of the player crosses. Considered as a mixed strategy game, there is a third Nash equilibrium where both players cross with an extremely low probability, say ϵ and crash with a probability of ϵ^2. The pure strategy Nash eqilibria have a payoff of 1. The mixed strategy equilibrium has a low (expected) pay off and a positive chance of crashing.

4.3 Auctions

Auctions are n player games where each player is bidding for a product being auctioned. Each player i has an internal valuation of the product v_i, which is typically not known to other players. Each player bids an amount b_i for the product and the highest bidder wins. Such auctions are sometimes called first price auctions. The utility for each player is as given below.

$$u_i = \begin{cases} v_i - b_i, & \text{if } b_i > max_{j \neq i} b_j. \\ 0, & \text{otherwise.} \end{cases}$$

If the players knew the valuation of other players, all the player with the highest valuation has to do is bid slightly more than the valuation of the second highest bidder and get a positive utility. This leads to a nice mechanism design called the Vickrey auction.

4.3.1 Vickrey Auction

Vickrey auctions, sometimes called second price auctions, are similar to first price auctions. Each player has a valuation v_i of the product and bids a value b_i and the highest bid wins, but the winner only pays the only the amount of the second highest bid. The utility for each player is as given below.

$$u_i = \begin{cases} v_i - max_{j \neq i} b_j, & \text{if } b_i > max_{j \neq i} b_j. \\ 0, & \text{otherwise.} \end{cases}$$

In Vickrey auctions knowledge of other players' utility does not affect anyone's bid. Each player essentially can bid their valuation and expect to get a positive utility. Thus Vickrey auctions are a dominant strategy game and have a Nash equilibrium.

4.4 Pricing Game - Indivisible Version

Suppose that there are two sellers and three buyers in a market. Buyers $B1$ and $B2$ have access to Seller $S1$ and buyers $B2$ and $B3$ have access to seller $S2$ as shown in the figure below. The sellers are selling a product that each of the buyers is interested in and are willing to pay a maximum of \$1 for it. Given a choice, the buyers want to buy the product at the lowest price possible. Here we discuss the indivisible version of the game, meaning that buyers buy all the goods they are interested in from a single seller. Relaxing this condition gives us the divisible version described in Section 7.1.

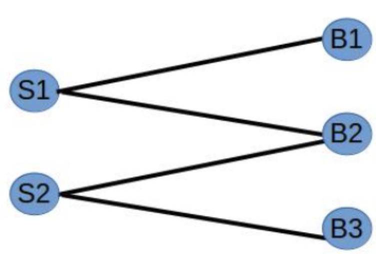


Figure 1: Pricing game

The players in the game are the sellers whose sell the product at a price P_i. We assume that the sellers cannot price differentiate, i.e., a seller cannot sell a product at different prices to different buyers. For the sake of simplicity, we also assume that the seller does not incur any cost for producing the products. Thus, their utility depends solely on how many of the product they sell $u_i = t_i \times P_i$, where t_i is the number of products player i has sold.

In this setup, buyers $B1$ and $B3$ have no choice in who they can buy from and they have to pay whatever the seller demands. Buyer $B2$ has a choice and will invariably pick the one with the lowest price. So, each player tries to price his product lower than the other player in order to get $B2$ to buy from them. They try to price their products lower and lower till one of them reaches \$0.5 at which point, the other player raises his price to \$1. The way it is set up, this game has no Nash equilibrium.

5 Mixed Strategy Games

Given a two player pure strategy game $G : ([n], S, U)$, $S = \Sigma_1 \times \Sigma_2$ is the strategy space, $Sigma_1 = \{s_1^1, s_1^2\}$, $U = U_1 \cup U_i$, and $U_i : \Sigma_1 \times \Sigma_2 \to \mathbb{R}$ is the utility function for player i. We define a mixed strategy game $G' : ([n], S', U')$. The mixed strategy for player i is given as $\Sigma_i = \{\lambda s_1^1 + (1-\lambda)s_1^2\}$, mixed utility function, $u_1^1(s_1^1, s_2^1)$ is defined as follows.

$$\begin{aligned} u_1^1(\lambda s_1^1 + (1-\lambda)s_1^2, \gamma s_2^1 + (1-\gamma)s_2^2) = &\lambda\gamma u(1,1) + \\ &\lambda(1-\gamma)u(1,2) + \\ &(1-\lambda)\gamma u(2,1) + \\ &(1-\lambda)(1-\gamma)u(2,2) \end{aligned}$$

$$\sum_j P_j E_{p-i}(u_i(s_i^j, s_{-j}))$$

$$u_1^1(m_i, m_{-i}) = E_{s_i, m_i} E_{s_{-i} m_{-i}} u_i(s_i, s_{-i})$$

$$\begin{aligned} u_1(m_1, m_2) = &\lambda\gamma u_1(1,1) \ + \\ &\lambda(1-\gamma)u_1(1,2) \ + \\ &(1-\lambda)\gamma u_1(2,1) \ + \\ &(1-\lambda)(1-\gamma)u_1(2,2) \end{aligned}$$

Proposition 23 *Mixed utility function is n-linear for an n-player game.*

Proposition 24 *If mixed strategy m_{-i} is fixed, mixed utility function is linear in m_i.*

Definition 25 *For a mixed strategy game, the support of a mixed strategy is those pure strategies that appear in the mixed strategy. i.e., $supp_i(s_i) = \{P_j^i : \lambda_j^i \neq 0\}$.*

For each user i and for each strategy j in $supp_i(s)$, $u_i(s_i, s_{-i}) = u_i(P_j^i, s_{-i})$, $s_i = \Sigma_i \lambda_j^i P_j^i$ and $\Sigma_i \lambda_j = 1$. Thus, if we know the supp for each player, finding the actual weights λ_j can be done by solving a linear system. However, finding supports is hard and the best known algorithm to find the supports, the Lemke-Howsen algorithm runs in exponential time, unless the game is a zero sum game (see Section 7). The following example illustrates this for the matching pennies game.

5.1 Matching Pennies

For each player $i \in \{A, B\}$, the strategy support is given by $\{s_i^1, s_i^2\}$ corresponding to playing heads or tails respectively. Let λ indicate the player 1's proportion of playing heads in his strategy (weight). $(1-\lambda)$ thereby indicates player 1's proportion of playing tails. Let γ and $(1-\gamma)$ indicate player

2's weights. Mixed strategy for player 1 is given by $m_1 = \lambda s_1^1 + (1-\lambda)s_1^2$ and for player 2, $m_2 = \gamma s_2^1 + (1-\gamma)s_2^2$. We can solve the system as follows:

$$\begin{aligned}
u_1(s_1^1, \gamma s_2^1 + (1-\gamma)s_2^2) &= u_1(s_1^2, \gamma s_2^1 + (1-\gamma)s_2^2) \\
&= \gamma u_1(s_1^1, s_2^1) + (1-\gamma)u_1(s_1^1, s_2^2) \\
&= \gamma u_1(s_1^2, s_2^1) + (1-\gamma)u_1(s_1^2, s_2^2) \\
&= \gamma \times 1 + (1-\gamma) \times -1 \\
&= \gamma \times -1 + (1-\gamma) \times 1
\end{aligned}$$

Solving these equations we get $\gamma = 0.5$. Similarly, λ is obtained as 0.5.
Note: The matching pennies example was worked out by Aditya Mahadev Prakash.

5.2 Nash Equilibrium in Mixed Strategy Games

Note that the mixed utility function (as described above) is not concave and hence we cannot use Nash's theorem to analyze the game for the existence of a Nash equilibrium. As stated in the following theorem, it turns out that we don't really need concavity of the utility function for Nash's theorem to hold.

Theorem 26 (Stronger Nash's theorem) *A game $\{P, (\Sigma_i)_{i\in P}, (u_i)_{i\in P}\}$ has a Nash equilibrium if*

i. $X_i\Sigma_i$ is convex.

ii. $\forall_i u_i$ are continuous.

iii. u_i is concave in the i^{th} argument.

Corollary 27 *A mixed strategy Nash equilibrium exists for any finite game.*

Earlier, we used the Kakutani's fixed point theorem to prove the Nash's theorem. This involved defining a function from one strategy to a set of strategies. To prove the Stronger Nash's theorem, we instead use Brouwer's fixed point theorem. To do this, we need to define a function that satisfies the requirements in Theorem 26. We define such a function next:

$$f(s) : \left(\underset{x}{\overset{max}{arg}}[u_i(r, s_{-i}) - ||s_i - r||^2]\right)_i$$

But, before we can use this function, we need to prove that the max defined in the function is unique and that the function $f(s)$ is continuous.

Fact 28 $f(s) : \left(\underset{x}{\overset{max}{arg}}[u_i(r, s_{-i}) - ||s_i - r||^2]\right)_i$ *is unique.*

Proof: Any strictly concave function has a unique maximum. Notice that $u_i(r, s_{-i}) = \sum_{s\in\Omega} r_s u_s(s_{-i})$, where $u_s(s_{-i})$ is the utility function of r and $||r - s_{-i}||^2$ is strictly concave and thus has a unique maximum. ■

Fact 29 $\forall_i f_i(s)$ *is continuous.*

Proof: If for a class of maximization problems $\underset{x}{\overset{max}{arg}}(g(\alpha, x))$ are unique, then as a function of α, $\underset{x}{\overset{max}{arg}}$ is continuous. ■

Now, we can use this function and Brouwer's fixed point theorem to prove that a Nash equilibrium exists for mixed strategy games.

6 Symmetry of Games and Nash Equilibria

In 1951 Nash proved the following theorem [9].

Theorem 30 *Symmetric games have symmetric Nash equilibrium.*

Given an asymmetric game, we can construct a new symmetric game with the property that it has a Nash equilibrium if and only if the original game has a Nash equilibrium. Thus Theorem 30 is useful in non-symmetric games too.

Claim 31 *An asymmetric two player game G with payoff matrices A and B respectively for the two players, has a pure strategy Nash equilibrium if and only if the two player game G' with the following payoff matrices has a pure strategy Nash equilibrium.*

$$A' = \begin{bmatrix} 0 & B^T \\ A & 0 \end{bmatrix} B' = \begin{bmatrix} 0 & A \\ B^T & 0 \end{bmatrix}$$

7 Two Player Zero-Sum Games

A two player game is a zero sum game if the sum of payoffs of the two players is zero for any choice of strategies. Typically, in such games, only the payoff of the row player is specified as the matrix A. Each entry in A represents the gain of the row player and the loss of the column player.

Corollary 27 states that a Nash equilibrium exists for all finite games. Here we show that for the particular case of zero-sum games, we can find the Nash equilibrium using linear programming.

Let λ and γ be the probability distributions for the strategies of the row and column players respectively that lead to a Nash equilibrium. Here we think of λ as a row vector and γ as a column vector. The expected utility value for the row player can be expressed as

$$u_1 = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix} \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_n \end{bmatrix}$$

Consider a strategy λ_1 for the row player. The expected payoffs for different strategies of the column player will be λA. Once λ is known, the column player will want to minimize his loss and essentially play strategies that correspond to the minimum entries in λA. Thus the best strategy can be found using the following linear program.

$$\begin{aligned} u^{row}_{max} &= \max_{\lambda \geqslant 0} u \\ \sum_j \lambda_j &= 1 \\ \forall_i (\lambda A)_i &\geqslant u \end{aligned}$$

Similarly, the best strategy for the column player can be found using the following linear program.

$$\begin{aligned} u^{column}_{min} &= \min_{\gamma \geqslant 0} u \\ \sum_j \gamma &= 1 \\ \forall_i (A\gamma)_i &\leqslant u \end{aligned}$$

Theorem 32 *The optimum solution for the linear programs described above gives us the λ and γ that form the Nash equilibrium for the two player zero-sum game.*

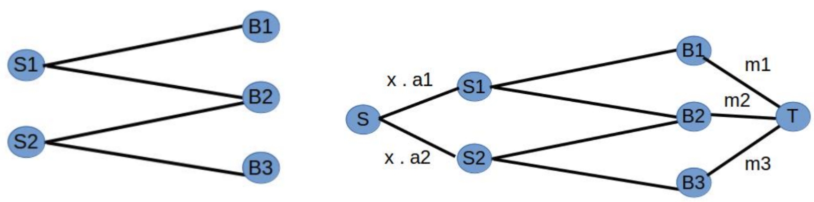


Figure 2: Bipartite graph G and the corresponding network N for a pricing game.

Proof: Let λ^* and γ^* be strategies of the two players respectively that lead to the optimum solutions to the two linear programs. Since this is a Nash equilibrium, we can argue that $u_{max}^{row} = u_{max}^{column} = u_1$. If the players play these strategies, then the row player cannot increase his win as the column player is guaranteed by his strategy to not lose more than u_{max}^{column}. Similarly, the column player cannot decrease his loss, as the row player is guaranteed to win u_{max}^{row} by his strategy. So, the strategies are at equilibrium. ■

7.1 Pricing Game - Divisible Version

This game is similar to the pricing game described in Section 4.4, except that each buyer can buy part of their goods from one seller and the rest from other sellers. This game is set up as follows. Let S be the set of sellers and B be the set of buyers. Buyer i has m_i amount of money and has access to $S_i \subseteq S$ sellers. Seller j has a_j amount of goods and they sell their goods at p_j.

Buyers don't care which seller they buy from, but will try to maximize the total amount of goods they can purchase with the money they have. For the given pricing strategy $P = [p_1, p_2, \ldots, p_n]$, buyer i is only interested in the cheapest subset $S_i' \subseteq S_i$ that exhausts m_i. We call this the *optimal basket* of goods for buyer i at price p.

Definition 33 (Market Clearing Price) *A pricing strategy P is said to be* market clearing *or* equilibrium price *if there exists an assignment of optimal basket to each buyer so that there is neither surplus or deficiency of goods or demand is equal to supply.*

Now we describe the algorithm for finding the market clearing prices. We create a bipartite graph $G = (A, B, E)$ where B - the set of buyers - and A - the set of sellers are connected by edges E whenever a buyer can access a seller (Figure 2. Because of the assumptions made, each vertex in G has a non-zero degree. For $S \subseteq A$ of goods, let $a(S)$ denote the total amount of goods in S, i.e., $a(S) = \sum_{j \in S} a_j$. For a subset $T \subseteq B$ of buyers, let $m(T) = \sum_{i \in T} m_i$ denote the total money possessed by all buyers in T.

Let $\Gamma(S)$ denote the set of buyers who are interested in buying goods S or $\Gamma(S) = \{i \in B | S_i \cap S \neq \varnothing\}$

Definition 34 (Feasible Price) *A uniform price x is feasible if $\forall\ S \subseteq A,\ x \cdot a(s) \leqslant m(\Gamma(s))$, where $\Gamma(s)$ is the set of buyers that have access to seller s and $a(s) = \Sigma_{j \in S} a_j$.*

Definition 35 (Tightness) *We say that the feasible price x^* is tight if the condition in the above definition become a strict equality.*

Theorem 36 (Max-Flow Min-Cut Theorem:) *The max-flow min-cut theorem states that in a flow network, the maximum amount of flow passing from the source to the sink is equal to the total weight of the edges in the minimum cut, i.e. the smallest total weight of the edges which if removed would disconnect the source from the sink [10].*

Lemma 37 *A uniform price x is feasible if and only if all the goods can be partitioned in such a way that each buyer gets all the goods they are interested in.*

Proof:
$\rightarrow$ If $\exists\ S \subseteq A$ where $x \cdot a(S) > m(\Gamma(S))$, the goods in S cannot be sold because the buyers interested in these goods do not have enough money to purchase it.
$\leftarrow$ We also create a network N by adding a source and a sink vertex to G as shown in Figure 2. Assign a capacity of infinity to all the edges in G. For each edge from the source s to vertex $j \in A$ introduce capacity of $x \cdot a_j$. For each edge from vertex $i \in B$ to sink t, introduce capacity of m_i.

Clearly, a way of selling all goods corresponds to a feasible flow in N that saturates all edges going out of s. We will show that if x is feasible, then such a flow exists in N. By the max-flow min-cut theorem, if no such flow exists, then the minimum cut must have capacity smaller than $x \cdot a(A)$. Let S be the set of goods on the $s-$side of a minimum cut. Since edges (j, i) for good $j \in S$ have infinite capacity, $\Gamma(S)$ must also be on the $s-$side of this cut. Therefore, the capacity of this cut is at least $x \cdot a(A - S) + m(\Gamma(S))$. If this is less than $x.a(A)$ then $x.a(S) > m(\Gamma(S))$, thereby contradicting the feasibility of x.

With respect to a feasible x, if a set S is tight, then on selling all goods in S, the money of buyers in $\Gamma(S)$ will be fully spent. Therefore, x constitute market clearing prices for goods in S. The idea is to look for such a set S, allocate goods in S to $\Gamma(S)$, and recurse on the remaining goods and buyers.

7.2 Algorithm to obtain market clearing price

The algorithm starts with $x = 0$ which is feasible, and raises x continuously, always maintaining its feasibility. It stops when a non empty set goes tight. Let x^* be the smallest value of x at which this happens and let S^* be the maximal tight set. x^* is the maximum value for which the min-cut is between the source and the sellers. It can be obtained by binary search. Once x^* is obtained find the set of nodes that can reach t in residual graph of this flow. This set, W, is the $t-$side of the unique maximal min-cut in N at $x = x^*$. Then, $S^* = A - W$, the set of goods on the s side of this cut.

In the next iteration, the algorithm removes S^* and $\Gamma(S^*)$, initializes the price of the goods in $A - S^*$ to x^*, and raises prices until a new set goes tight. The algorithm continues in this manner until all goods have been assigned prices. It can be shown that the above algorithm computes equilibrium prices and allocations in polynomial time.

8 Homework

1. Prove Claim 31.
2. In Section 4.3.1 we claimed that Vickrey auction is a dominant strategy game. Prove this and show that the dominant strategy is for each player to bid their valuation.

References

[9] John Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.

[10] Wikipedia contributors. Max-flow min-cut theorem — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Max-flow_min-cut_theorem&oldid=841036458, 2018. [Online; accessed 22-May-2018].

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018
Lecture — Jan. 30 - Feb 1	
Lecturer: Dr. Meera Sitharam	Scribe: Aditya Mahadev Prakash

9 Recap

1. Stronger version of Nash theorem was discussed and proved.
2. First price and Vickrey auction was discussed. In first price auction, if all the valuations $V_i, i \in \{1, \ldots, n\}$ are known, the winning price is $P_{\text{win}} = V_{(n-1)} + \epsilon$. Vickrey auction forces all the bidders to bid at their true valuation, which is a Nash equilibrium.
3. Sufficient conditions for Nash equilibrium: convex strategy space, utility function should be concave
4. Any mixed strategy game has a Nash equilibrium. The example of matching pennies was discussed.
5. Buyer-seller problem and market clearing price was discussed.
6. Nash theorem- every finite game has a mixed Nash equilibrium

10 Basics of NP-completeness

Definition 38 (Problem:) *By a problem we refer to a yes or no problem, i.e., a function $A(\dot{)}$ from a set of inputs to {Yes,No}. An instance of a problem is a given input x. The size $|x|$ of an instance x is the number of digits in the encoding of x.[11]*

Definition 39 (Algorithm:) *An algorithm T is a well-defined set of instructions which produce a specific output state denoted YES. An algorithm is said to solve the problem A if $T_1 = \{x : A(x) = YES\}$, i.e., it reaches the state YES precisely on the correct set of inputs. Note that the definition is not symmetric with respect to replacing YES by NO.[11]*

Definition 40 (Polynomial Time Algorithm:) *Running time complexity of an algorithm defined in terms of its input size n is polynomial if $T(n) \leqslant O(n^k)$ for some constant k.*

Definition 41 (Non-deterministic Polynomial (NP) Algorithm:) *Class of algorithmic **decision** problems that have a **non-deterministic** algorithm that runs in time **polynomial** in the size of the input in the worst case.*

Definition 42 (Non Deterministic Algorithm:) *For the algorithm defined above, if we are provided a witness/guess/certificate y, where $|y| < |x|^k$, the algorithm is non deterministic if at least for one of its guesses, $A(x) = D(x)$*

Definition 43 (Deterministic Polynomial (P) Algorithm:) *lass of algorithmic **decision** problems that have a **deterministic** algorithm that runs in time **polynomial** in the size of the input in the worst case.*

Definition 44 (P versus NP Problem:) *We know that P⊂NP, however, it has not yet been proved whether P⊆NP or P⊊NP.*

Definition 45 (NP-hard:) *A problem A is easier than B if there is a polynomial algorithm which can translate every possible input of A to an input of B such that all A-inputs for which the A-answer is Yes and only those are mapped to B-inputs for which the B-answer is Yes. An algorithmic decision problem is NP-hard if $\forall x \in NP$ if x is reducible to $y \in P$.*

Definition 46 (NP-complete:) *A decision problem that is NP-hard and in NP or a problem which is harder in the above sense than all NP problems is NP complete.*

In a 2 person game, determining the existence of a Nash equilibrium is NP-complete. Given payoff matrices A and B for player 1 and 2, construct game with payoff matrix C. The first game has a Nash equilibrium if and only if the second game has a Nash equilibrium.
Note: Nobody knows if determination of existence of potentially non-symmetric Nash equilibria for symmetric games is NP-complete or not.

11 Nash and Correlated Equilibrium: Some complexity considerations

Given a game $G = \{\{1, \ldots, n\}, \Sigma_1 \times \cdots \times \Sigma_n, u_i(\Sigma_1 \times \cdots \times \Sigma_n) \forall i \in \{1, \ldots, n\}\}$ the following problems:

1. Whether there exists at least 2 Nash equilibria?
2. a Nash equilibrium in which player 1 has utility $\geqslant t$
3. a Nash equilibrium in which 2 players have utility $\geqslant t$
4. A Nash equilibrium whose support size $\geqslant n_s$
5. A Nash equilibrium whose support contains a strategy s
6. A Nash equilibrium whose support does not contain a strategy s

are NP-complete [11].

12 Lemke-Howson Algorithm for 2 player symmetric games

We will show that in 2 player symmetric games, finding a symmetric Nash is PPAD-complete.

Before we begin the algorithm it can be seen that there is a polynomial time reduction from Nash to symmetric Nash. Given two matrices A and B, define $C = \left(\begin{smallmatrix} 0 & A \\ B^T & 0 \end{smallmatrix}\right)$ and let (x, y) be the symmetric equilibrium of this game. It is easy to see that for (x, y) to be the best response to itself, y must be a best response to x and x must be a best response to y. Hence x and y constitute a Nash equilibrium to the original game.

Consider a symmetric utility matrix A of size $n \times n$. Consider a convex polytope P defined by $2n$ inequalities $Az \leqslant 1$, $z \geqslant 0$. It is a non-empty bounded polytope since $z = 0$ is a solution and since it is to be assumed that all coefficients of A are non-negative and no column is zero. Also assume that all vertices are non degenerate or all vertices lies on precisely n constraints. We say that a strategy i is *represented* at vertex z if either $A_i z = 1$ or $z_i = 0$ or both, that is, if at least one of the two inequalities of the polytopes associated with strategy i is tight at z.

Suppose that at a vertex z, all strategies are represented. This would happen at $z = 0$– but suppose it is not. Then for the strategies i with $z_i > 0$, it must be the case that $A_i z = 1$. Now define a vector x as follows:

$$x_i = \frac{z_i}{\sum_{i=1}^{n} z_i} \tag{2}$$

x_i's are well defined since we assume $z \neq 0$ and they all add up to 1, thereby constituting a mixed strategy. We claim that x is a symmetric Nash equilibrium.

12.1 Example

$$A = \begin{bmatrix} 0 & 3 & 0 \\ 0 & 0 & 3 \\ 2 & 2 & 2 \end{bmatrix}$$

The convex polytope for the above matrix is depicted in Figure 3.

The only vertices where all strategies are represented are $z = (0, 0, 0)$ and $(0, \frac{1}{6}, \frac{1}{3})$ and the latter corresponds to Nash equilibrium $x = (0, \frac{1}{3}, \frac{2}{3})$

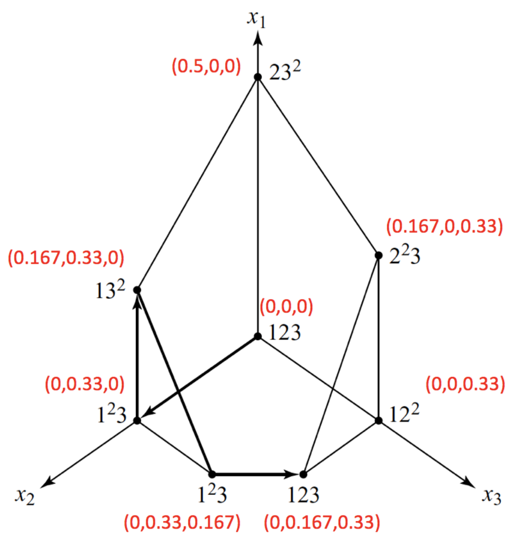


Figure 3: Convex polytope of strategies. Vertex 1^23 indicates that strategy 1 is represented twice, i.e. $z_1 = 0$ and $A_1z = 0$. Figure edited from [12].

12.2 Pivot Method for finding Nash Equilibrium[12]

Fix a strategy, say strategy n, and consider the set V of all vertices of P for which all strategies are represented except possibly for strategy n. This set of vertices is nonempty, because it contains the vertex (0,0,0), so let us start there a path $\langle v_0 = 0, v_1, v_2, \ldots \rangle$ of vertices in the set V. Since we assume that P is non-degenerate, there are n vertices adjacent to every vertex, and each one is obtainable by relaxing one of the tight inequalities and making some other inequality tight. So, consider n vertices adjacent to $v_0 = (0, 0, 0)$. In one of these vertices, z_n is non-zero and all other variables are 0, so this vertex is also in V; call it v_1.

At v_1 all strategies are represented except for strategy n and one strategy $i < n$ is represented twice. By either relaxing $z_i = 0$ or $A_iz = 1$, we obtain two vertices in V adjacent to v_1. One of them is the vertex we came from v_0 and the other is bound to be a new vertex $v_2 \in V$.

If at v_2 all strategies are represented, then it is a Nash equilibrium and we are done. Otherwise, there is a strategy j that is represented twice at v_2, and there are two other vertices in V adjacent to v_2 and correspond to these two inequalities. One of them is v_1 and the other v_3 and so on.

In this path, no vertex v_i can be repeated, because repeating v_i would mean that there are three vertices that are adjacent to v_i obtainable by relaxing a constraint associated with the doubly represented strategy which is impossible. Since the polytope is finite, it cannot go on forever. It can only stop at a vertex where each strategy is represented once, other than v_0 which is the Nash equilibrium.

References

[11] Itzhak Gilboa and Eitan Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1(1):80 – 93, 1989.

[12] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018
Lecture — Feb. 6 - Feb 8	
Lecturer: Dr. Meera Sitharam	Scribe: Rahul Prabhu

13 The Complexity Classes PPAD, FNP and TFNP

In 1994, Christos Papadimitriou introduced the complexity class Polynomial Parity Arguments on Directed graphs (PPAD) to characterize the set of problems for which we know that a solution exists but it is hard to find these solutions [13]. This is an interesting complexity class as it contains the problem of finding Nash equilibrium.

Definition 47 (PPAD) *PPAD or Polynomial Parity Arguments on Directed graphs is a complexity class whose solution space is characterized by the following.*

7. *A directed graph is defined on a finite but exponentially large set of vertices.*
8. *Each vertex has indegree and outdegree at most one.*
9. *Given a string, it is a computationally easy problem to (a) tell if it is indeed a vertex of the graph, and if so to (b) find its neighbors (one or two of them), and to (c) tell which one is the predecessor and/or which one is the successor (i.e., identify the direction of each edge).*
10. *There is one known source (vertex with no incoming edges) called the standard source.*
11. *Any sink of the graph (a vertex with no outgoing edges), or any source other than the standard one, is a solution of the problem.*

PPAD is the class of problems where existence is known but the proof is non-constructive, by which we mean that it is long. Note that the decision versions of the problems in PPAD take constant time and hence PPAD is not quite the same as NP and showing PPAD completeness is weaker evidence of intractability than showing NP-completeness. The following lemma illustrates this.

Lemma 48 *Let* $\Phi = \{G : \text{every Nash equilibrium of the game } G \text{ satisfies some } \phi(x)\}$. *If* Φ *is NP-hard, NP = CO-NP.*

Proof: If Φ is in NP, there exists a polynomial time reduction that reduces SAT to Φ. We start with a formula in $\overline{SAT}$ and map it to a problem instance in $\overline{\Phi}$ using this reduction in polynomial time. Since the problem instance in $\overline{\Phi}$, which is of the form $\exists$ a Nash equilibrium that doesn't satisfy $\phi(x)$, can be checked in polynomial time. This would mean that we have found a short proof for a problem in CO-NP. ■

Now we define two additional complexity classes, FNP and TFNP and relate them to PPAD.

Definition 49 [Function NP] *A binary relation* $R : X \times Y \to \{0, 1\}$*, where* $\exists k : |y| \leqslant |x|^k$*, with input sets* X, Y *is in FNP if and only if there is a deterministic polynomial time algorithm,* $A(x, y)$ *such that*

$$A(x, y) = R(x, y).$$

Example 1 [Examples of FNP Problems]

- **Longest Common Subsequence**: *Given two words $S = (s_1, ..., s_n), T = (t_1, ..., t_n)$ what is the longest common subsequence of S, T. Here the input is $x = \{S, T\}$ and $y = LCS(S, T)$. $|y| < \max\{|S|, |T|\}$. We can test this property in polynomial time using dynamic programming.*

- **CNF-SAT**: *Given a boolean circuit and collection of literals $C = \{\phi, L\}$ give a truth assignment to L satisfies ϕ. Here $x = C$ and $y = \{0,1\}^{|L|}$ is a truth assignment. We can test this property in $|\phi|$ time by evaluating each clause and taking the conjunction.*

Claim 50 $[NP \subset FNP]$

For the proof, take the deterministic verifier for any set in NP to be the FNP set.

Definition 51 **[Total FNP]** *Given a polynomial time checkable predicate $R(x, y)$ and an input x output a y such that $R(x, y) = 1$ when such a y exists. (Or, equivalently, state that no such y exists.)*

Definition 52 (MAX CNF-SAT) *The decision problem of MAX CNF-SAT(k, ϕ, L) is to decide whether k clauses of a CNF circuit ϕ over literal set L can be satisfied. A function version of this problem, FMAX CNF-SAT(ϕ, L), asks for the assignment that satisfies the maximum number k s.t. MAX CNF-SAT(k, ϕ, L) is true. A maximally satisfying assignment always exists since the number of clauses is finite.*

Lemma 53 *MAX CNF-SAT is in TFNP.*

Note that SAT is NOT IN TFNP: not all propositions have a satisfying assignment. The function problem FMAX CNF-SAT$(\phi, L) \in$ TFNP. So immediately, NP $\not\subset$ TFNP $\subseteq$ FNP.

14 Sperner's Lemma

In this section we introduce the Sperner's Lemma which is a combinatorial analog of the Brouwer's fixed point theorem and is equivalent to it. We start by defining some background terms.

Definition 54 **[Simplex]** *A $k-$dimensional simplex is the convex hull of $k+1$ points in $\mathbb{R}^k$ which have differences with any single point in the collection that are linearly independent.*

Let's see an example

Example 2 *[Triangle] Take 3 points in general position in $\mathbb{R}^2$, P_1, P_2, P_3. If the three points are collinear then the resulting configuration is not a simplex (it is a lower dimensional simplicial complex) as the difference with, say, P_1 for P_2 is just a multiple of $P_3 - P_1$ by the definition of collinearity. Meanwhile, if the three point are not collinear then we have a triangle, which is a (the) $2-$dimensional simplex.*

Often, what is important to study is not so much the specific layout of a simplex, but the relations between points in the simplex. In the case of a single simplex, the relationships are defined by *m-faces*, intuitively the edges, faces, and surfaces of the simplex, which define groupings of points. Any k points in a $k-$simplex are co-$k-$facial and so the groupings can be described as exclusions of the missing $k+1$th point: so there are $kC1 = k$ difference $k-$faces.

Exercise 1 *Show that for a $k-$simplex there are $\binom{k+1}{m+1}$ distinct $m-$faces.*

Definition 55 **[Triangulation of a Simplex]** *A triangulation of a $k-$simplex $\mathcal{A}$ into n triangles is an arbitrary arrangement of $k-$simplices $\{S_i\}_{i=1}^n$ such that*

- $\bigcup_{i=1}^{n} S_i = \mathcal{A}$*: they partition* $\mathcal{A}$*.*

- $S_i^\circ \bigcap S_j^\circ = \varnothing$*: the interiors of the triangles are mutually disjoint.*

- *The triangles intersect only along lower-dimension ($j < k$) simplices ($j-$faces).*

The definition is a bit wordy, but the triangulation is a very fundamental thing that our intuition tends to be right about. So proofs require checking the conditions carefully but to come up with the proof working with the idea of a triangulation (mentioned above) is usually good enough.

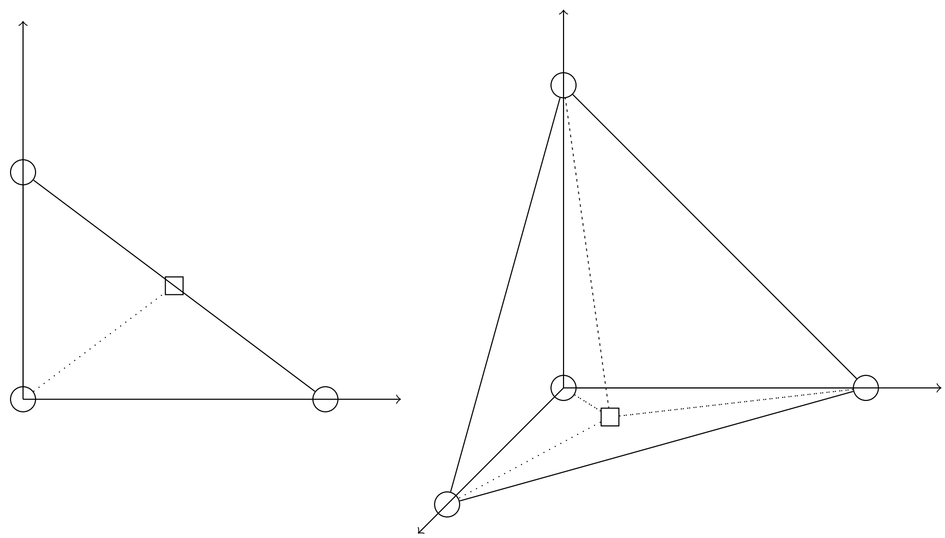


Figure 4: Illustration of $k-$simplices for $k = 2, 3$ embedded in $\mathbb{R}^2$ and $\mathbb{R}^3$. Sperner's lemma is actually a topological statement so viewing the simplices we discuss through a graph lens is encouraged. The dotted lines and $\square$'s represent a potential triangulation.

Definition 56 [Sperner Coloring] *A Sperner Coloring (or Sperner labelling) is a labelling of a triangulation of a $n-$simplex such that*

- *The points of the original simplex are labelled uniquely (1 through $n + 1$) with an ordered labelling.*

- *Any point in the triangulation on a face of the simplex is labelled with a label from one of the points of the original simplex along that face.*

- *The interior points are labelled with any of the labels.*

An example of a Sperner Labelling (a coloring with symbols) is shown in Figure 5.

Lemma 57 [Sperner's Lemma] *Any triangulation of an $n-$dimensional simplex with an $n-$color Sperner Labelling contains an odd number of* **panchromatic simplices.**

Proof: We prove the Sperner's lemma using induction on the number of dimensions n. The base case is a $1-$simplex where it is easy to show that this is true. We start with one color at one end point and end with another at the other end point and therefore must switch between the colors an odd number of times.

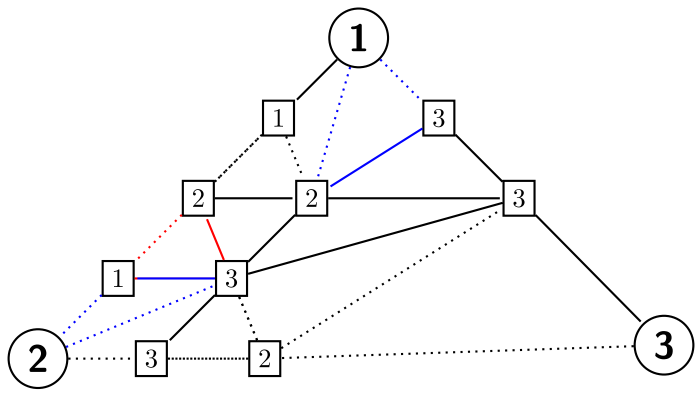


Figure 5: A simple Sperner coloring on a $2-$simplex. Chirality of the panchromatic triangles guaranteed by Sperner's Lemma is indicated by red and blue edges. Openness is indicated by the dotted lines. Note that dotted lines are also colored. Some panchromatic triangles can be entered from different edges, but none can be entered from the same edge along 2 different paths.

Now we show that the Sperner's lemma is true for a $2-$simplex. Along a given edge of the original simplex, label edges as open or closed depending on whether they are $2-$color or not. Apply the same labelling throughout the greater triangulation. On the boundary of the original simplex is a unique triangle (of the triangulation) adjacent to the face of a given open edge. Of the faces of this triangle, at most 1 is open. Following the open edges this way, label the faces traversed as closed as we follow the path. Then, from a given initial edge either we have a path through the triangulation that leads back to the given edge we began with, or we have a path that leads to a triangle in which the only open edge is the one by which we arrived. In the latter case we have found an $n-$colored simplex. In the former case we have 'closed' two edges on the outer face of the original simplex.

Since there are an odd number of open faces of the triangulation on any face of the original simplex, their must be an edge that we enter and cannot leave in the above fashion (by parity). This corresponds to at least 1 n-colored simplex of the triangulation. Similar arguments can be made for higher dimensions and the induction holds. ■

14.1 Sperner's Lemma and Brouwer's Fixed Point Theorem

Now we show the relation between Sperner's lemma and Brouwer's fixed point theorem. Let us begin by recalling Brouwer's fixed point theorem

Theorem 58 [Brouwer] *Any continuous function $f : D \to \mathbb{R}$ mapping a convex, compact domain into itself ($f(D) \subset \mathbb{R}$) has a point x_0 such that $f(x_0) = x_0$.*

In Brouwer's fixed point theorem we have the continuity condition assumed:

$$\forall x \in D, \forall \epsilon > 0 \exists \delta_{(x,\epsilon)} : \forall y ||x - y|| < \delta \Rightarrow ||f(y) - f(x)|| < \epsilon$$

where $|| \cdot ||$ indicates the appropriate norm. Here the $\delta(x, \epsilon)$ can be made universal for the whole domain.[1] So we will try to create a Sperner labelling of points in the domain in such a way that

[1] Since we are in a compact domain D we can cover D by a finite collection of open balls, use $\delta_\epsilon = \max_{U \in \mathcal{C}} \sup_{x \in U} \delta_{(x,\epsilon)}$, to represent a 'global' δ associated with each epsilon (continuous functions are uniformly continuous over compact sets). Since the δ is finite for each x (recall compactness, diameter of a compact convex set) the sup exists and is finite. This is a technical note that makes the proof easier.

the uniform continuity property immediately gives us the Brouwer Fixed Point theorem. First we need an intermediate definition that will make the particular rule for making the Sperner Labelling clearer.

Definition 59 **[Barycentric Coordinates]** *Given an $n-$simplex S the barycenter coordinates of a point x in the convex hull of S, $\overline{S}$, is a convex combination of the vertices of the simplex. That is*

$$x = \sum_i \alpha_i p_i \in \overline{S}$$

has barycentric coordinates $\vec{\alpha}$.

Now let's define a Sperner Labeling procedure for continuous functions.

Definition 60 **[Continuous Sperner Labelling]** *For a continuous map f into its $n-$dimensional compact, convex domain D let the continuous Sperner labelling be the function $L : D \to [n+1]$ defined by $L(x) = \min\{k \ |(f(x))_k \leqslant x_k\}$.*

The subscript j is used to denote the entry in the jth coordinate.

Claim 61 **[Continuous Sperner $\to$ Sperner]** *Given a continuous function into its compact, simplex domain Δ the continuous Sperner labelling function maps any points on a face of the original simplex to a legally labelled symbol in the Sperner labelling of that simplex.*

Proof: Fix an ordered labelling the the vertices of the simplex. Each of the points p^i on the vertices v^i are uniquely labelled by the continuous Sperner since $p^i_k = 0$ unless $k = i$. For points on the faces use barycentric coordinates: if $x \in F \ |F = \{x|\alpha_k = 0\}$ then $L(x) \in [n]\backslash k$ which is the definition of the Sperner Labelling. ∎

Theorem 62 **[Brouwer on Simplex]** *Let $f : \Delta \to \Delta$ be a continuous function over a simplex Δ. Then $\exists x_0 \mid f(x_0) = x_0$.*

Proof: Take a collection of sequences in $\Delta, \{S_i\}_{i=1}^{\infty}$ s.t $\lim_{n\to\infty} S_n$ is dense in Δ and $S_i \subset S_{i+1}$ — S_{i+1} is a refinement of S_i. Apply the continuous Sperner labelling to S_i. By Sperner's lemma there is a panchromatic cell in S_i, call it T_i. This means that the points labeled j in T_i, p^{ij} satisfies $f(p^{ij})_j < p^{ij}_j$. Since $\lim_n S_n$ is dense in Δ there is an n such that $T_i^{\circ} \bigcap S_n$ is nonempty. For this n repeat the argument with $\Delta = T_i$ to get another (contained) panchromatic cell. Take the sequence of points formed by all of the panchromatic cells obtained this way associated with the n^{th} entry (an infinite set provided by Sperner's Lemma) and call each one $\{r_{i,n}\}_{i=1}^{\infty}$. Since the size of the panchromatic triangles is shrinking there is a common limit among all of these subsequences: x^*.

Now we will show that $f(x^*)_i = x^*_i$ for all i. Since $r_{N,n}$ is labeled n we have that $f(r_{N,n})_n < (r_{N,n})_n$ and passing to the limit $f(x^*)_n \leqslant (x^*)_n$ for all n. But no index can be strictly less, since the bound on the other indices would for the barycentric coordinates to add up to a number strictly less that 1. Therefore, we must have equality in each index in the limit. ∎

References

[13] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498 – 532, 1994.

15 Introduction

Previously, we proved that mixed Nash was reducible to the "End of the line" problem via the $Nash(mixed) \leqslant Brouwer \leqslant Sperner \leqslant PPAD$ reduction. Now we will show the PPAD-completeness of Sperner, Brouwer, and Nash following a $PPAD \leqslant Sperner \leqslant Brouwer \leqslant Nash(mixed)$ reduction in the opposite direction.

16 PPAD-completeness of Sperner

In the definition of the PPAD problem, we are given a directed, acyclic graph (DAG) on a finite, but exponentially large set of vertices with a circuit that tells us if we are at a vertex in the graph and the neighboring vertices in a computationally easy way. However, we are unable transform the entire explicit PPAD graph to a Sperner instance in polynomial time because the DAG is exponentially large. This problem of succinctness forces us to transform the circuit for computing neighboring vertices when given a vertex to a circuit that generates the Sperner coloring of a neighboring vertex when given a colored vertex.

Our goal is to identify a piecewise linear, single dimensional subset L of the cube, corresponding to the PPAD graph. We define $m = n + 4$. Then we map non-isolated nodes on the PPAD graph map to pairs of segments (main and axillary segments) such that $u \in 0, 1^n$. The main segment will be defined by points $u_1 = (8\langle u \rangle + 2, 3, 3)$ and $u_1^{'} = (8\langle u \rangle + 6, 3, 3)$ and the auxiliary segment by $u_2 = (3, 8\langle u \rangle + 6, 2^m - 3)$ and $u_2^{'} = (3, 8\langle u \rangle + 10, 2^m - 3)$. From there, we connect these segments by adding an orthonormal path connecting the end of the main segment to the beginning of the auxiliary segment using breakpoints $u_3 = (8\langle u \rangle + 6, 8\langle u \rangle + 6, 3)$ and $u_4 = (8\langle u \rangle + 6, 8\langle u \rangle + 6, 2^m - 3)$. Next, we map edges to orthonormal paths. The edge between u and v maps to an orthonormal path connecting the end of the auxiliary segment of u with the beginning of main segment of v using the breakpoints $(8\langle v \rangle + 2, 8\langle u \rangle + 10, 2^m - 3)$ and $(8\langle v \rangle + 2, 8\langle u \rangle + 10, 3)$. The resulting directed, orthonormal line is L.

Claim 63 *Two points p, $p^{'}$ of L are closer than $3 * 2^{-m}$ in Euclidean distance only if they are connected by a part of L that has length $8 * 2^{-m}$ or less.*

Claim 64 *Given the circuits P, N of the PPAD instance, and a point x in the cube, we can decide in polynomial time if x belongs to L.*

Claim 65 *u is a sink in PPAD graph $\iff$ L is disconnected at $u_2^{'}$ and u is a source in PPAD graph $\iff$ L is disconnected at u_1*

Now we have embedded the generic PPAD graph in $[0, 1]^3$ and now we will look to reduce this to 3D (dual) Sperner [14].

With dual Sperner, we color the center of the cubelets rather than the vertices of the subdivisions where the solution to dual-Sperner is a vertex of the subdivision such that all colors are present in the surrounding centers of cubelets. We defined K_{ijk} to be the center of the cubelet whose least significant corner has coordinates $(i, j, k) * 2^{-m}$.

Lemma 66 *If the canonical simplicization of the dual graph has a panchromatic simplex, then this simplex contains a vertex of the subdivision that is panchromatic [14].*

We define the canoncial boundary coloring to be different than normal Sperner coloring[14]:

$$\begin{cases} K_{ijk} \leftarrow 0, \text{ if any of } i, j, k \text{ is } 2^m - 1 \\ K_{ijk} \leftarrow 1, \text{ if } i = 0, \text{ unless already colored} \\ K_{ijk} \leftarrow 2, \text{ if } j = 0, \text{ unless already colored} \\ K_{ijk} \leftarrow 3, \text{ if } k = 0, \text{ unless already colored} \end{cases}$$

Lemma 67 *The modified boundary coloring of dual-Sperner still guarantees existence of panchromatic simplex [14].*

We then use the following two coloring rules for the reduction[14]:

- All cube lets get color 0 unless they touch line L in the embedded PPAD graph
- All cube lets surrounding line L at any point are given colors 1, 2 and 3 to prevent line L from touching color 0.

We place colors 1, 2 and 3 clockwise around L where color 3 appears twice. We must also be aware of which cubelets around L that we color with color 3 during a walk as to prevent any panchromatic vertices at the turns by making sure that the pair of colored 3 cubelets lies above L for the main segment for u and below for the main segment for v [14]. The coloring on the connecting segments is impossible to efficiently decide locally so we must assume that all edges (u, v) of the PPAD graph join an odd u with an even v. We place the pair of color 3 cubelets below the main segment for even u's and above for odd v's. The resulting walk will only give us panchromatic vertices at the ends.

Claim 68 *A point in the cube is panchromatic in our coloring if and only if [14]:*

- *an endpoint $u_2^{'}$ of a sink vertex u of the PPAD graph, or*
- *an endpoint u_1 of a source vertex $u! = 0^n$ of the PPAD graph*

Claim 69 *Given the description of generic PPAD graph, there exists a polynomial-size circuit computing the coloring of every cubelet K_{ijk} in Sperner. [14]*

17 PPAD-completeness of Brouwer

Claim 70 *The boundary coloring of the dual-Sperner instance is no longer legal, but no new panchromatic points were introduced by the modification because color 0 composes most of the cube.*

Proof: The points that were not, but potentially could become new panchromatic after the modification are those where: $x_1, x_2, orx_3 = 1 - 2^{-m}$, but since the ambient space is the ambient color 0 and line L is far from the boundary, this will not happen. ■

For the reduction from dual-Sperner to Brouwer, we define a Brouwer instance on the cube defined by the convex hull of the centers of the cubelets and map the colors to the direction of the displacement vector $f(x) - x$ where $\alpha = 2^{-2m}$[14]:

$$\begin{cases} \text{color 0 (green) [ambient]} \longrightarrow (-1, -1, -1) * \alpha \\ \text{color 1 (yellow)} \longrightarrow (1, 0, 0) * \alpha \\ \text{color 2 (red)} \longrightarrow (0, 1, 0) * \alpha \\ \text{color 3 (blue)} \longrightarrow (0, 0, 1) * \alpha \end{cases}$$

We then extend f on the remaining cube by interpolation and canonically triangulating the cube. We compute displacement f at some x by finding x's simplex S then[14]:

if $x = \sum_{i=1}^{4} w_i * x_i$, where x_i are the corners of S, we define $f(x) - x := \sum_{i=1}^{4} w_i * (f(x_i) - x_i)$

Claim 71 *Let x be a 2^{-3m}-approximate Brouwer Fixed Point of f. Then the corners of the simplex S containing x must have all colors/displacements[14].*

18 PPAD-completeness of Nash (mixed)

Definition 72 (Graphical game) *games defined to capture sparse player interactions such as those arising under various geographical, communication, or other constraints, where players are considered nodes in the graph and a player's payoff is only affected by its own strategy and the strategies of its indirection neighbors[14].*

Definition 73 (Separable multiplayer games) *(polymatrix games) graphical games with edge-wise separable utility functions. Each edge is a 2-player game and the player's payoff is given by the sum of payoffs from all adjacent edges: $\sum_{i-1}^{n} x_u^T A^{(u,v_i)} x_{v_i}$ where n is the number of adjacent edges[14].*

We begin the reduction at Brouwer, a stylized, discrete version of Brouwer's fixpoint theorem. The problem once again is succinctness, as we do not have an explicit Brouwer function lookup table where we would be able to easily perform a polytime reduction to a Nash payoff matrix. This is because the PPAD graph we started with was exponentionally large so our initial Sperner reduction and the subsequent Brouwer reduction were based on transforming the circuit (which we can do in polynomial time) rather than transforming the entire explicit input. We follow this pattern for the reduction to polymatrix Nash where we take the circuit that converts a Brouwer function value and find a polytime transformation to a circuit that computes a player's response in a graphical, seperable, multiplayer game such that finding a Brouwer fix point will correspond to a Nash equilibrium of this graphical game.

18.1 Binary Computation with Games

We present a function ϕ derived from the mapping of a unit cube to itself. We define a three player game with players x, y, and z where x and y only care about their inputs and z only cares about x and y. We are able to define payoff matrices for z such that we can create logical AND, OR, and NOT gates. We define ϕ as a Boolean Circuit, a directed, acyclic graph of such gates with $3n$ input bits (x, y, z) representing the 3D coordinate of the cubelet in question, and two output bits, representing the applied displacement of the four possible displacements from the Brouwer instance. In this scenario each bit has a binary strategy set $0, 1$. Any function ϕ meeting the previously stated criteria can be represented as such a circuit problem. Our goal: given a Boolean circuit describing ϕ, find a fixpoint of ϕ. [14]

18.2 Real Arithmetic with Games

We define a game where three of the players will choose the three numbers needed to represent a point in the cube. The other players analyze these coordinates to determine the exact cubelet that houses the point by simulating the circuit to compute the displacements at the cubelet and neighboring cubelets. If the point is a fixpoint of ϕ, this is a Nash equlibrium and the three players have no incentive to change their mixed strategy.

Table 5: List of Gadgets [14]

copy	$z = x$
addition	$z = min(1, x + y)$
subtract	$z = max(0, x - y)$
set equal to constant	$z = max(0, min(1, \alpha))$
multiply by constant	$z = max(0, min(1, \alpha * x))$
comparison	$\begin{cases} 1, \text{ if } x > y \\ 0, \text{ if } x < y \\ *, \text{ if } x = y \end{cases}$

When looking at the "other players" from the game mentioned above, the strategy of an "other" player should result in a Nash equilibrium based on the inputs of the three original players if and only if the point is a fixpoint, meaning there is some arithmetic relationship between the inputs and the mixed strategy of the output player that results in such a Nash equilibrium. We are looking for a game that can do real arithmetic to find this relationship.

18.3 Gadgets

We are able to break up the larger game into sub-games by using basic arithmetic and logic operations to define this relationship. Each of these sub-games will have a Nash equilibrium based on the operation (multiplication, addition, subtraction, AND, OR, NOT, etc...) [15].

Addition Gadget A game with properties:

- 4 players
- 2 pure strategies 0,1 so mixed strategy [0,1]
- Payoffs
  - $u(w : 0) = Pr[x : 1] + Pr[y : 1]$
  - $u(w : 1) = Pr[z : 1]$
  - $u(z : 0) = 0.5$
  - $u(z : 1) = 1 - Pr[w : 1]$

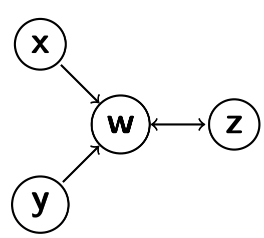


Figure 6: The addition gadget is defined by input players x and y, output player z, and auxiliary player w

Note that when constructing a gadget, for example multiplication, where $y = 1$ if $x > 1/2$ and

$y = 0$ if $x < 1/2$, the output is undetermined when $x = 1/2$ and we must maintain this behavior [14]. Thus, such a multiplication gadget is non separable while all the others stated above are separable. We will only use separable gadgets in our reduction.

18.4 Fixed Point Computation

So any circuit composed of the separable gates can be implemented with a separable game. This circuit, which we'll call a game-inspired straight-line program, need not be a DAG circuit, but does allow feedback[14].

Suppose function $f : [0,1]^k \to [0,1]^k$ is computed by a game-inspired straight-line program. We can construct a polymatrix game whose Nash equlibria are in a many-to-one and onto correspondence with the fixed points of f so we are able to try to reduce PPAD to simply finding a fixed point of such a game-inspired straight-line program[14].

The game consists of three *variables* (x, y, z), whose mixed strategies represent a coordinate in the unit cube $[0,1]^3$. This coordinate is the input of a series of operators which extract the most significant bit of the coordinate to determine the cubelet that the point is in. Based on these inputs and a series of gadgets, we could compute the outputs of the circuit for ϕ for a cubelet and its adjacent ones, deciding whether this is a fixpoint and hence a Nash equilibrium. This holds except in the scenario noted above where $x = 1/2$ resulting in indeterminate output from the operators when the point lies on the boundary of the cubelet, which could result in a game with no Nash equilibria. Therefore, instead of using the point (x, y, z) as our input of ϕ we use a series of points around (x, y, z) to get the average displacement $(\Delta x, \Delta y, \Delta z)$ for (x, y, z) [15]. We consider the "good set" of coordinates to be those which do not lie on the boundary of any subdivision. We discard all others. This prevents and indeterminable output. We then close the loop by adding the components of the average displacement to the three *variables* to complete the graphical game in which the *variables'* strategies affect them.

Claim 74 *The average displacement is zero if and only if we are near a fixpoint [14]*

Claim 75 *The points on the unit cube where the average displacement is the zero vector are Nash equlibria and hence panchromatic and can be recovered in polynomial time given (x, y, z) [14].*

Theorem 76 *Given a polymatrix game G there exists a ϵ^* such that[14]:*

1. $|\epsilon^*| = poly(|G|)$

2. *given a ϵ^*-Nash equilibrium of G we can find in polynomial time an exact Nash equilibrium of G.*

Proof:
1. Polymatrix Nash $\equiv$ Polymatrix Nash
2. Polymatrix Nash is PPAD-complete ■

References

[14] Constantinos Daskalakis. Topics in algorithmic game theory. 2011.

[15] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018
Lecture 6	
Lecturer: Dr. Meera Sitharam	Scribe: Kenan Yildirim

19 Social Welfare and Social Choice

$$A = \text{set of alternatives } \{1, \ldots, n\}$$

$$\lambda_i = \text{player } i\text{'s ordering of } A$$

$$\lambda = (\lambda_1, \ldots, \lambda_n)$$

$$\lambda(a,b) = \begin{cases} 0 & \text{if } b \text{ is preferred to } a \text{ in } \lambda \\ 1 & \text{if } a \text{ is preferred to } b \text{ in } \lambda \end{cases}$$

Definition 77 (Social welfare function) *A function $F : \lambda^n \to \lambda$ is called a social welfare function. F aggregates the preferences of all voters into a total social order of the candidates.*

Definition 78 (Social choice function) *A function $F : \lambda^n \to A$ is called a social choice function. F aggregates the preferences of all voters into a social choice of a single candidate.*

20 Arrow's Theorem

There are a few properties that are desirable in social welfare functions:

Definition 79 (Unanimity) *A social welfare function F satisfies unanimity if, when all voters have the same preferences, then the aggregate social preference is the same.*

$$\forall a, b, i \quad (a > b)_{\lambda_i} \to (a > b)_{\lambda}$$

$$\textit{where } \lambda = F(\lambda)$$

Definition 80 (Independence of irrelevant alternatives) *A social welfare function F satisfies independence of irrelevant alternatives if the social preference between any two alternatives a and b depends only on the voters' preferences between a and b.*

$$\forall a, b \in A \quad \& \quad \lambda = (\lambda_1, \ldots, \lambda_n) \quad \& \quad \lambda' = (\lambda'_1, \ldots \lambda'_n)$$

$$\forall i[\lambda_i(a,b) = \lambda'_i(a,b)] \implies \lambda(a,b) = \lambda'(a,b)$$
$$\textit{where } \lambda = F(\lambda) \quad \& \quad \lambda' = F(\lambda')$$

Definition 81 (Dictatorship) *Voter i is a dictator in a social welfare function F if*

$$\forall \lambda = (\lambda_1, \ldots, \lambda_n), F(\lambda) = \lambda_i$$

That is, the aggregate social choice is always the preference of the dictator. It is generally desired that a voting system have no dictators.

Unfortunately, it is has been shown that it is impossible to satisfy all three of these properties in voting systems with more than just two candidates. We will introduce the theorem behind this and prove it.

Theorem 82 (Arrow's Theorem) *Every social welfare function F with $|A| \geqslant 3$ that satisfies unanimity and independence of irrelevant alternatives has a dictator.*

Proof: Let F be a social welfare function which satisfies unanimity and independence of irrelevant alternatives.

Claim 83 (Pairwise neutrality) *Let $\lambda = (\lambda_1, \ldots, \lambda_n)$ and $\lambda' = (\lambda'_1, \ldots, \lambda'_n)$ such that $\forall i \lambda_i(a,b) = \lambda'_i(c,d)$. Then, $\lambda(a,b) = \lambda'(c,d)$.*

Without loss of generality, $\lambda(a > b) = 1$ and $b \neq c$. For each i, combine λ_i and λ'_i to create λ^*_i. Put c before a and d after b, preserving the internal order of (a,b) and (c,d).

$$\lambda^* = F(\lambda^*_1, \ldots, \lambda^*_m)$$

$$\text{From unanimity, } (c > a)_{\lambda*} \text{ and } (b > d)^*_\lambda$$

$$\text{Using transitivity, } (c > d)_{\lambda*}$$

$$\text{Using independence, } (c > d)_{\lambda'}$$

Take $a \neq b$ and define λ^i in which exactly the first i players prefer a to b. Let i^* be such that $(b > a)_{\lambda^{i*}_j}$ for $j < i^*$ and $(a > b)_{\lambda^{i*}_j}$ for $j \geqslant i^*$. We will show that i^* is a dictator for F.

Take $c \neq d \neq e$. Define $\lambda' = (\lambda'_1, \ldots, \lambda'_n)$ where $\lambda'_i = \lambda_i$ except we move e to the top of λ'_i for $i < i^*$ and to the bottom of λ'_i for $i > i^*$. For λ'_{i*} we move e so that $(c > e > d)_{\lambda'_{i*}}$. By independence of irrelevant alternatives, we have not changed the relative rankings betweem c and d. Also notice how the relative ranking of c and e is the same as for a and b in λ^{i^*}, but the preferences for e and d are the same as a and b in λ^{i^*-1}. Thus, by the pairwise neutrality claim, $\lambda(c,e) = 1$ and $\lambda(e,d) = 1$, and by transitivity $\lambda(c,d) = 1$. ■

21 Gibbard-Satterthwaite Theorem

We will now take a look at dictatorships in the context of social choice functions. First, we will introduce some definitions.

Definition 84 (Monotonicity) *f is monotonic if*

$$f(\lambda_i \lambda_{-i}) = a \neq a' = f(\lambda'_i \lambda_{-i})$$
$$\implies (a' < a)_{\lambda_i} \text{ and } (a' > a)_{\lambda'_i}$$

Definition 85 (Strategically manipulated) *A social choice function f can be strategically manipulated by voter i if $\exists \lambda$ such that $(a < a')_{\lambda_i}$ where $a = f(\lambda)$ and $a' = f(\lambda_1, \ldots, \lambda'_i, \ldots, \lambda_n)$. This means that if voter i prefers a' to a, they can ensure that a' is socially chosen over a by misrepresenting their preference to be λ'_i). f is called incentive compatible if it cannot be manipulated.*

Definition 86 (Pareto efficient) *A social choice function f is Pareto efficient if whenever an alternative a is at the top of every individual i's ranking, λ_i, then $f(\lambda) = a$.*

Theorem 87 (Gibbard-Satterthwaite Theorem) *If $|A| \geqslant 3$, any monotone, Pareto efficient social choice function has a dictator.*

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
b	$\ldots$	b	a	a	$\ldots$	a			a
a	$\ldots$	a	b	.		.			.
.		.	.	.		.	$\rightarrow$	a	.
.		.	.	.		.			b
.		.	.	.		.			.
.		.	.	b	$\ldots$	b			.

Figure 7: Situation before b is raised above a in λ_n

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
b	$\ldots$	b	b	a	$\ldots$	a			b
a	$\ldots$	a	a	.		.	$\rightarrow$		a
.		.	.	.		.		b	.
.		.	.	.		.			.
.		.	.	b	$\ldots$	b			.

Figure 8: Situation after b is raised above a in λ_n

We will prove this theorem using the proof presented in [16].

Proof: Consider any two alternatives a and b, and a profile of rankings in which a is ranked highest and b is ranked lowest for every individual. Pareto efficiency implies that the social choice as this profile is a.

Consider now changing individual 1's ranking by raising b in it one position at a time. By monotonicity, the social choice remains equal to a so long as b is below a in 1's ranking. But when b finally does rise above a, monotonicity implies that the social choice either changes to b or remains equal to a. If the latter occurs, then begin the same process with individual 2, then 3, etc. until for some individual n, the social choice does change from a to b when individual n prefers b to a. See figures 7 and 8.

Now consider Figures 9 and 10. Figure 9 is derived from Figure 7 by moving a to the bottom of λ_i for $i < n$ and moving it to the second last position in λ_i for $i > n$. Figure 10 is derived similarly from Figure 8. We will argue that these changes do not affect the social choices. First, note that the social choice in Figure 10 must, by monotonicity, be b because the social choice in 8 is b and no individual's ranking of b vs. any other candidate changes in the move from Figure 8 to Figure 10. Next, note that the profiles in Figures 9 and 10 differ only in the individual n's ranking of a and b. So, because the social choice in Figure 10 is b, the social choice in Figure 9 must, by monotonicity, be either a or b. But if the social choice in Figure 9 is b, then by monotonicity, the social choice in Figure 7 must be b, a contradiction. Hence, the social choice in Figure 9 is a.

Consider candidate c. Because the profile of ranking in Figure 11 can be obtained from the Figure

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
b	$\ldots$	b	a	.		.			a
.		.	b	.		.	$\rightarrow$		b
.		.	.	.		.		a	.
.		.	.	a	$\ldots$	a			.
a	$\ldots$	a	.	b	$\ldots$	b			.

Figure 9: Derived from Figure 7 by moving a to the bottom of λ_i for $i < n$ and moving it to the second last position in λ_i for $i > n$.

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
b	$\ldots$	b	b	.		.			b
.		.	a	.		.			.
.		.	.	.		.	$\rightarrow$	b	.
.		.	.	a	$\ldots$	a			a
a	$\ldots$	a	.	b	$\ldots$	b			.
									.

Figure 10: Derived from Figure 8 by moving a to the bottom of λ_i for $i < n$ and moving it to the second last position in λ_i for $i > n$.

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
.		.	a	.		.			a
.		.	c	.		.			.
.		.	b	.		.	$\rightarrow$	a	.
c	$\ldots$	c	.	c	$\ldots$	c			.
b	$\ldots$	b	.	a	$\ldots$	a			.
a	$\ldots$	a	.	b	$\ldots$	b			.

Figure 11: Introducing c to Figure 9.

9 profile without changing the ranking of a vs. any other candidate in any individual's ranking, the social choice in Figure 11 must, by monotonicity, be a.

Consider the profile of rankings in Figure 12 derived from the Figure 11 profile by interchanging the rankings of candidates a and b for individuals $i > n$. Because this is the only difference between the profiles, and because the social choice in Figure 11 if a, the social choice in Figure 12 must, by monotonicity, be either a or b. But the social choice in Figure 12 cannot be b because candidate c is ranked above b in every individual's ranking, and monotonicity would then imply that the social choice would remain b even if c were raised to the top of every individual's ranking, contradicting Pareto efficiency. Hence the social choice in Figure 12 is a.

Note that an arbitrary profile of rankings with a at the top of individual n's ranking can be obtained from the profile in Figure 12 without reducing the ranking of a vs. any other candidate in any individual's ranking. Hence, monotonicity implies that the social choice must be a whenever a is at the top of individual n's ranking. So, we may say that individual n is a dictator for candidate a. Because a was arbitrary, we have shown that for each alternative $a \in A$, there is a dictator for a. But there cannot be distinct dictators for distinct candidates, hence there is a single dictator for all

λ_1	$\ldots$	λ_{n-1}	λ_n	λ_{n+1}	$\ldots$	λ_N		Social Choice	Social Order
.		.	a	.		.			a
.		.	c	.		.			.
.		.	b	.		.			.
.		.	.	.		.	$\rightarrow$	a	c
.		.	.	.		.			.
.		.	.	.		.			.
c	$\ldots$	c	.	c	$\ldots$	c			b
b	$\ldots$	b	.	b	$\ldots$	b			.
a	$\ldots$	a	.	a	$\ldots$	a			.

Figure 12: Interchanging the rankings of candidates a and b for individuals $i > n$

candidates. ■

References

[16] Philip J Reny. Arrows theorem and the gibbard-satterthwaite theorem: a unified approach. *Economics Letters*, 70(1):99–105, 2001.

22 Price of Anarchy

The price of anarchy (PoA) tells us how good or bad a Nash Equilibrium is. Formally, we have

$$PoA = \frac{min_{S\in NE}V(S)}{max_S V(S)}.$$

Here we define some function $V : S \to \mathbb{R}$. The smaller this ratio is, the larger the price you are paying for using a distributive/non-cooperative/selfish strategy instead of central optimization. When playing a pure strategy game, we have

$$PoA = \frac{min_{S\in PNE}V(S)}{max_S V(S)}.$$

There is also a price of stability (PoS), and it is defined as

$$PoS = \frac{max_{S\in NE}V(S)}{max_S V(S)}.$$

Price of anarchy and price of stability satisfy the following properties.

$$PoA(PSNE) \leqslant PoA(MSNE) \leqslant PoA(CNE) \leqslant 1$$
$$PoS(PSNE) \leqslant PoS(MSNE)$$
$$PoA() \leqslant PoS()$$

PSNE, MSNE, CNE stands for pure strategy, mixed strategy and cooperative Nash Equilibrium, respectively. The last property states that for any kind of game we are playing, the PoA for that game is no larger than the PoS for that game.

22.1 Congestion Game

We will start with a congestion game example. Let
- N be the number of players, and $(\Sigma_i)_{i\in N}$. So $\Sigma_i \subseteq 2^E$;
- E be the set of edges, and each $e \in E$;
- $d_e(l(E))$ be the delay function. $d_e = f_e$ and for now, assume f_e is a linear function $f_e(x) = a_e x + b_e$ and $f_e(x) = x$;
- $l_e(S)$ be the load function of a strategy and $l_e(S) = |\{i : e \in S_i\}|$.

The cost of each player is therefore $c_i(s) = \Sigma_{e\in N} l_e(s)$. The price of anarchy is therefore $\frac{max_{S\in NE}V_1(S)}{min_S V_1(S)}$. The numerator and the denominator are switched since it is a congestion game.

Theorem 1 there are linear congestion games with $|n| \geqslant 3$ with pure price of anarchy for V_1 being $\frac{5}{2}$, a best to worst ratio.

Theorem 2 for any linear congestion game, the pure price of anarchy ($\forall S^* \in NASH \Rightarrow \frac{V_1(S^*)}{V_1(P)}$) of V_1 is less than or equal to $\frac{5}{2}$.

Lemma for any $\alpha, \beta \geqslant 0, \alpha(\beta + 1) \leqslant \frac{1}{3}\beta^2 + \frac{5}{3}\alpha^2$.

Now we will attempt to prove theorem 2.

Proof of theorem 2 let S^* be a Nash Equilibrium and let P be an optimal strategy. Cost for player i is $c_i(S_i^*) = \Sigma_{e \in S_i^*} l_e(S^*)$, with $l_e(S^*)$ being the number of players choosing channel e in the strategy profile S^*.

We want to bound, so we have $V_i(S^*) = \Sigma c_i(S^*) = \Sigma_{e \in E} l_e^2(S^*)$ with respect to $V_1(P) = min_S V_1(S) = \Sigma_i c_i(P) = \Sigma_{e \in E} l_e^2(P)$. At S^* Nash, the cost of player i should not decrease when switched to P.

We have $c_i(S^*) = \Sigma_{e \in S_i^*} l_e(S^*) \leqslant \Sigma_{e \in P_i}(S_{-i}^*, P) \leqslant \Sigma_{e \in P_i}(l_e(S^*)+1)$, and $V_1(S^*) = \Sigma_{i \in N} c_i(S^*) \leqslant \Sigma_{i \in N} \Sigma_{e \in P_i}(l_e(S^*)+1) = \Sigma_{e \in E} l_e(P)(l_e(S^*)+1) \leqslant \frac{1}{3}\Sigma_{e \in E} l_e^2(S^*) + \frac{5}{3}\Sigma_{e \in E} l_e^2(P) = V_1(S^*) \leqslant \frac{1}{3}V_1(S^*) + \frac{5}{3}V_1(P)$.

An optimal strategy P is for each player to select his or her first strategy. A worst case Nash S^* is for each player to select his or her second strategy. At this Nash Equilibrium, each player has cost 5, while the cost for each player is 2 in the optimal allocation.

22.1.1 Smoothness

A game is called (λ, μ)-smooth for $\lambda > 0$ and $\mu \leqslant 1$ if, for every pair of states $s, s' \in \Sigma$, we have

$$\Sigma_{i \in N} c_i(s_i', s_{-i}) \leqslant \lambda \cdot cost(s') + \mu \cdot cost(s)$$

Smoothness directly gives a bound for the PoA.

Theorem in a (λ, μ)-smooth game, the PoA for pure Nash Equilibria is at most $\frac{\lambda}{1-\mu}$.

Proof let s be the worst PNE and $s' = s^*$ be an optimum solution. Then:

$$\begin{aligned} cost(s) = \Sigma_{i \in N} c_i(s) &\leqslant \Sigma_{i \in N} c_i(s_i^*, s_{-i}) \text{ (as } s \text{ is NE)} \\ &\leqslant \lambda \cdot cost(s^*) + \mu \cdot cost(s^*) \text{ (by smoothness)} \end{aligned}$$

On both sides subtract $\mu \cdot cost(s)$, this gives

$$(1-\mu) \cdot cost(s) \leqslant \lambda \cdot cost(s^*)$$

and rearranging yields

$$\frac{cost(s)}{cost(s^*)} \leqslant \frac{\lambda}{1-\mu}$$

■

22.2 Fair Cost Sharing Game

Now we introduce Fair Cost Sharing Game.

- Set N of n players, set R of m resources
- Player i allocates some resources, i.e., strategy set $\Sigma_i \subseteq 2^R$
- Resource $r \in R$ has fixed cost $c_r \geqslant 0$
- Cost c_r is assigned in equal shares to the players allocating r (if any).

Fair cost sharing games are congestion games with delays $d_r(x) = \frac{c_r}{x}$.

Social cost turns out to be the sum of costs of resources allocated by at least one player:

$$cost(S) = \Sigma_{i \in N} c_i(S) = \Sigma_{i \in N} \Sigma_{r \in S_i} d_r(n_r) = \Sigma_{r \in R, n_r \geqslant 1} n_r \cdot \frac{c_r}{n_r} = \Sigma_{r \in R, n_r \geqslant 1} c_r$$

Theorem every fair cost sharing game is $(n, 0)$-smooth. Thus, the PoA is upper bounded by n. The class of fair cost sharing games is tight, i.e., there are games in which the PoA for pure Nash Equilibria is exactly n.

PoA is large, but pure Nash Equilibrium is not necessarily unique. What do other pure NE cost, what about the best one?

Price of Stability for Nash Equilibria:
- Consider Σ^{PNE} as the set of pure Nash Equilibria of a game Γ
- Price of Stability is the ratio:

$$PoS = \frac{min_{s' \in \Sigma^{PNE}} cost(s')}{cost(s*)}$$

PoS is a best-case ratio and measures how much the best PNE costs in comparison to an optimal state of the game.

22.3 Potential Function and Social Delay (and a little bit on the exam question #3)

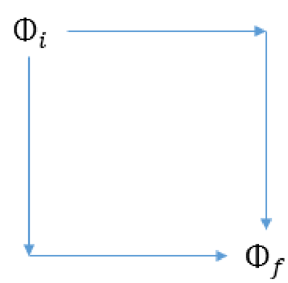
A change in social delay is the change in delay when player i goes from strategy S to S' and is as follows:

$$\Delta(s_i, s_i', s_{-i}) = \Sigma_{e \in (s_i' \setminus s_i) \cup (s_i \setminus s_i')} d_e(l_e(S')) - d_e(l_e(S))$$

Change in potential function is defined as follows:

$$\Phi(S') - \Phi(S) = \Sigma_e (\Sigma_{k=1}^{l_e(S')} d_e(k) - \Sigma_{k=1}^{l_e(S)} d_e(k))$$

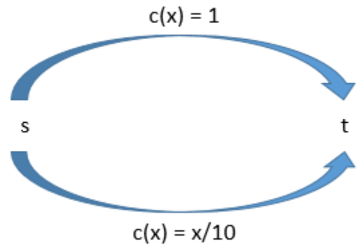
Proven by Dr. Sitharam on Piazza, these two equations are equal. However, the social delay function is not always the potential function. In the case of exam question #3, normalizing the social delay function yields exactly the potential function. Furthermore, intuitively, a potential function must satisfy the following property, illustrated in a diagram below:



To have a function be a potential function, it must be able to choose any Manhattan path to the destination while resulting in the same value at the destination. In this diagram, one can take the down-right path or the right-down path and end up at the same potential function.

23 Network/Routing Games and Braess's Paradox

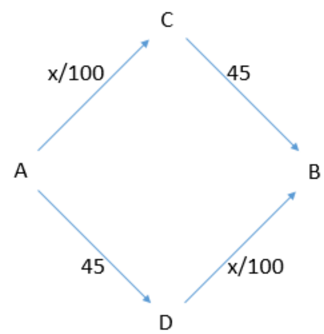
Many literatures on the inefficiency of equilibria concerns routing games, because routing games shed light on an important practical problem: how to route traffic in a large communication network that has no central authority. Here we have Pigou's example:



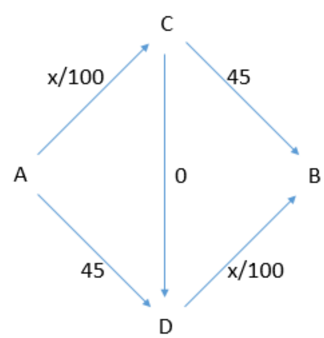
- Traffic wants to flow from s to t
- Edges have latency or delay. The upper edge has a latency of 1 unit, while the lower edge depends on how many people chooses it.

All players are making private decisions. We have a lot of players now, and each wants to minimize his or her latency.

In order to reduce his or her latency, each player has to reason about the latency of the lower edge. However, this does not have a stable outcome because it depends on how many people are using that edge. Let us look at another example:



Here, the only Nash Equilibrium is when there is an equal amount of players going through top and bottom route. Under any other circumstances, players would want to switch from the slower route to the faster route. Therefore, any other situation where $x \neq \frac{N}{2}$ isn't a Nash Equilibrium, with x being the number of players occupying either route. Suppose a 0 weight bridge connects C and D:



One would think that this bridge would improve the traffic flow. However, this bridge impedes traffic flow. Discovered by Dietrich Braess from Ruhr University, Germany, he noticed that if every player makes the optimal self-interested decision as to which route is the quickest, the shortcut could be chosen too often for players to have the shortest travel times possible. The idea behind this is that the Nash Equilibrium may not equate with the best overall flow through a network.

A more general version of this paradox is the Pigou's Paradox. This paradox, in short, states that expanding a road system as a remedy to congestion is ineffective and often even counterproductive.

Homework

1. Verify that $f_e(l_e(S)) = a_e l_e(S) + b_e$ for theorem 1.
2. Verify that every player choosing his or her second strategy is a Nash Equilibrium.

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018
Lecture — Mar 20 - Mar 22	
Lecturer: Dr. Meera Sitharam	Scribe: Keshava Raghunathan

24 Introduction

Mechanisms with money are a way to "solve" dictatorship issues with Social Choice and Social Order voting systems that are based on a linear ordering of the options as inputs to the Social Choice/Social Order function.

24.1 Description of the game

The game consists of:
A group of players N
A group of alternatives/options A
Each player has a valuation $v_i \in V_i$ for each $a \in A$
Each player has some "money" m_i
The utility function for player i is defined as $u_i(a, m_i) = v_i(a) - m_i$

25 What is a mechanism?

A way to change the game so that some facet of it changes *e.g.*,
- Add a Nash Equilibrium to a game that does not have one.
- Make not having perfect information irrelevant.
Mechanisms are formally defined in a later section.

26 Example: Vickrey auction

26.1 Game Description

A Vickrey auction can be described in the following way:

$$N : \text{A set of players}$$
$$A : \text{ "i-wins"}, i \in N$$
$$v_i(a) = \begin{cases} w_i & \text{if a="i wins"}, \\ 0 & \text{otherwise} \end{cases}$$
$$\text{If } a = \text{"i-wins", then } m_i = p$$

Hence,

$$u_i(a) = \begin{cases} w_i - p & \text{if a="i wins"}, \\ 0 & \text{o/w} \end{cases}$$

26.2 2^{nd} Price Mechanism

"The player with the highest bid wins and pays the second-highest bid."

26.3 Strategy-proofing

Theorem 88 *2^{nd} price auction is strategy-proof.*

Proof: Let i be the winner. For i,
Case 1: Changing to any value above the 2^{nd} bid will not change the outcome.
Case 2: Going below the 2^{nd} bid will cause $u_i = w_i - p$ to drop to 0.

For all other $j \neq i$,
Case 1: Raising above v_i will cause a loss of $v_j - v_i$.
Case 2: Changing to any value below v_i will not change outcome.

Hence, no player has an incentive to bid anything other than their valuation. The game is strategy-proof. ■

27 Direct Mechanisms

Definition 89 (Direct Mechanism) *A direct mechanism is where*
$f : V_1 \times V_2 \times \cdots \to A$ *Social choice function from valuations instead of ordering*
$P_i : V_1 \times V_2 \times \cdots \to \Re$ *Price for each player*

Definition 90 (Strategy-proof Mechanism) *A mechanism is Strategy-proof/Incentive-compatible if $\forall i, \forall v_i \in V_i$:*

$$\text{If } a = f(V_i, V_{-i}) \text{ and } a^{'} = f(V_i^{'}, V_{-i})$$
$$\text{then}$$
$$u_i = V_i(a) - P_i(V_i, V_{-i}) \geqslant V_i^{'}(a) - P_i(V_i^{'}, V_{-i}) = u_i^{'}$$

For any value function, a player cannot gain from reporting a different valuation.

27.1 VCG Mechanisms

A mechanism is a VCG mechanism if:

$$f(V_1 V_2 \dots V_n) = \underset{a \in A}{\operatorname{argmax}} \sum_{i \in N} u_i(a)$$
$$P(V_1 V_2 \dots V_n) = h_i(V_{-i}) - \sum_{j \neq i} V_j(f(V_1 V_2 \dots V_n))$$

for any function h_i that does not depend on V_i.

Theorem 91 *$\forall h_i$, a VCG mechanism is strategy-proof.*

Proof: Fix a player i and a value vector V_{-i} for the other players. Let V_i be the true valuation and $V_i^{'}$ be a false valuation for the ith player and

$$\begin{aligned} a &= f(V_i, V_{-i}) \\ a^{'} &= f(V_i^{'}, V_{-i}) \end{aligned}$$

$$\begin{aligned} u_i &= v_i - p_i \\ &= V_i(a) - [h_i(V_{-i}) - \sum j \neq i V_j(a)] \\ &= V_i(a) + \sum_{j \neq i} V_j(a) - h_i(V_{-i}) \\ &= \sum_i V_i(a) - h_i(V_{-i}) \end{aligned} \tag{3}$$

Similarly,

$$\begin{aligned} u_i^{'} &= v_i^{'} + \sum_{j \neq i} V_j(a^{'}) - h_i(V_{-i}) \\ &= \sum_k V_k(a^{'}) - h_i(V_{-i}) \end{aligned} \tag{4}$$

Since a maximizes f,

$$\sum_{k \in N} V_k(a^{'}) \leqslant \sum_{k \in N} V_k(a) \tag{5}$$

$$\therefore u_i \geqslant u_i^{'} \tag{6}$$

■

27.2 Grove's Mechanism

Definition 92 (Individual Rationality) *If a player participates their utility is non-negative.*

$$\forall i,\ u_i = V_i(V_1 V_2 \ldots V_n) - P_i(V_1 V_2 \ldots V_n) \geqslant 0 \tag{7}$$

Definition 93 (No positive transfer)

$$P_i(V_1 V_2 \ldots V_n) \geqslant 0 \tag{8}$$

The Grove function is defined as follows:

$$h_i(V_{-i}) = \max_{b \in A} \sum_{j \neq i} V_j(b) \tag{9}$$

$$P_i(V_1 V_2 \ldots V_n) = \underbrace{\max_{b \in A} \sum_{j \neq i} V_j(b)}_{\text{Let b* be the maximizer}} - \sum_{j \neq i} V_j(f(V_1 V_2 \ldots V_n)) \tag{10}$$

Theorem 94 *A VCG mechanism with the Grove function has the above two properties (Individual Rationality and No positive transfer), assuming $V_i(a) \geqslant 0$.*

Proof: For Individual Rationality, we can see that

$$\begin{aligned}
u_i(a) =& V_i(a) - P_i(a) \\
=& V_i(a) + \sum_{j \neq i} V_j(a) - \sum_{j \neq i} V_j(b*) \\
\geqslant& \sum_{j} V_j(a) - \sum_{j} V_j(b*) \\
\geqslant& 0
\end{aligned} \tag{11}$$

To see that the mechanism also enforces No positive transfer,

$$P_i = \sum_{j \neq i} V_j(b*) - \sum_{j \neq i} V_j(a) \geqslant 0 \tag{12}$$

■

27.2.1 Revisiting the Vickrey auction

$$A = \{\text{“i wins”},\ i \in N\}$$

$$V_i(a) = \begin{cases} w_i & \text{if } a = \text{“i wins”} \\ 0 & \text{otherwise} \end{cases}$$

$$f = \sum_{j \in N} V_j(\text{“k wins”}) = w_k$$

Also, note that k is the argmax value for “k wins”. f will choose the player with maximum value of w_k. For i w/ max value w_i,

$$\sum_{j \neq i} V_j(\text{“i wins”}) = 0$$

$$\sum_{j \neq i} V_j(b^*) = w_s$$

where w_s is the second highest bid value.

27.2.2 Auction with k identical items

$$A = \{\text{“S wins”},\ S \subseteq N, |S| = k\}$$

$$V_i(\text{“S wins”}) = \begin{cases} w_i & \text{if } i \in S \\ 0 & \text{o/w} \end{cases}$$

At the max value of f S includes the k players with the highest valuation. Fix player i and calculate

$$\begin{aligned}
P_i =& \underbrace{\max_{S'} \sum_{j \notin S} \theta_j(S')}_{\text{Grove's function}} - \sum_{j \neq i} V_j(S) \\
=& w_{k+1}
\end{aligned}$$

So the price that the k winners pay should be the $k+1^{th}$ highest bid.

27.2.3 Public project

$$A = \{\text{“build”}, \text{ “not build”}\}$$
$$C = \text{Cost of building}$$

Each player has a worth w_i of the bridge

$$w_i = \begin{cases} w_i & \text{if } A\text{=“build”} \\ 0 & \text{o/w} \end{cases}$$

To ensure Individual Rationality and No positive transfer, the social choice function should be

$$\text{If } \sum_i w_i \geqslant C \rightarrow \text{“build”}$$
$$\text{If } \sum_i w_i < C \rightarrow \text{“not build”}$$

To ensure fair payment:

$$P_i = \begin{cases} C - \sum_{j \neq i} w_j & \text{if } w_i \geqslant C - \sum_{j \neq i} w_j \\ 0 & \text{o/w} \end{cases}$$

This ensures

$$\sum_i P_i \leqslant C \tag{13}$$

If we add another “reasonable” condition that the cost of the bridge is covered,

$$\sum_i P_i \geqslant C \tag{14}$$

(13) implies Individual Rationality and (14) implies No positive transfer. Both cannot be simultaneously satisfied with the condition (people may pay even if the bridge is not built). An example is shown to illustrate the mechanics:

Example 3 $C = 3 \quad w_1, w_2 \in \{0, 2\}$

$$w_1 = w_2 = 2 \textit{ then } P_1(2) + P_2(2) \geqslant 3 \textit{(Cost of bridge covered)} \tag{15}$$
$$w_1 = w_2 = 0 \textit{ then } P_1(2) + P_2(2) \geqslant 0 \textit{(No positive transfer)} \tag{16}$$

Given (15) *and* (16)*, either:*

$$P_1(2) + P_2(0) \geqslant 3/2 \tag{17}$$
$$P_1(0) + P_2(2) \geqslant 3/2 \tag{18}$$

must be true. One player does not care if the bridge is built. In either case, the bridge will not be built ($\because \sum_i w_i < C$). However, from (17) *and* (18) *someone still paid. This contradicts Individual Rationality, and therefore the third condition cannot be enforced.*

27.2.4 Buying edges in a network

The game is to find the shortest (least expensive) path from S to E, with a mapping from players to edges.

$$v_c(p) = \begin{cases} -c_e & if e \in P \\ 0 & o/w \end{cases}$$

$$f : \text{Shortest path w.r.t. costs}$$

$$P_e = \underbrace{\underbrace{\text{Cost of the shortest path in the graph w/o } e}_{h_i} - \underbrace{\text{Shortest path in the graph without } e\text{'s cost}}_{\Sigma}}_{\text{Grove's function}}$$

This mechanism is strategy-proof. No player needs to hide their real valuation and it satisfies Individual Rationality and No positive Transfer.

28 Introduction to Indirect Mechanisms

All the games considered so far have the players have a true public valuation. This is not necessarily true, as players might announce a valuation that is not their true valuation.

28.1 Hidden Valuation game description

A mechanism is considered where players have a private information space $T_1 \times T_2 \dots T_n$ $T_i : A \to R$.
The player's "action" space is $X_1 \times X_2 \dots X_n$
Player strategies: Each row and column for a player is a space function, $s_i(t_i)$ $s_i : T_i \to X_i$
Previously, we considered the strategies themselves to be actions.
Player utilities: $u_i(t_i, s_i(t_{-i}))$

28.2 Revelation Principle

Theorem 95 *If $\exists$ a strategy-proof mechanism that implements a scoial choice function f, then $\exists$ a direct strategy-proof mechanism implementing f.*

This is the same as saying, "If you design a mechanism where players have no incentive to lie, then you have a mechanism where players have no incentive to hide."

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018
Lecture — March 26 - March 30	
Lecturer: Dr. Meera Sitharam	Scribe: Christopher Kodadek

29 Brief Review of Mechanism Design

Mechanism Design is interested in designing economic mechanisms with the goal of implementing desired social choices in a strategic setting, with the assumption that different members of society act rationally. Common scenarios/examples of mechanism design include:

12. **Elections:** Each voter has his own preferences, and the outcome of an election is a single social choice: a candidate being elected
13. **Markets:** Each participant has his/her own preferences, and the outcome is a single social choice: the reallocation of money and goods
14. **Auctions:** Auction rules define the social choice: the identity of a winner
15. **Government Policy:** Enacting laws, and public projects. Each citizen has his/her set of preferences, but a single choice is made: which laws are enacted, and which projects are commenced

A *direct mechanism* consists of two parts:
• a social choice function $V_1 \times ... \times V_n \to A$
• a collection of payment functions where $p_i : V_1 \times ... \times V_n \to \mathbb{R}$ is the amount player i pays.

29.1 Key Terms

Definition 96 (Social Choice) *An aggregation of the preferences of the different participants toward a single joint decision.*

Definition 97 (Strategy Proofness / Incentive Compatibility) *:*

A mechanism is $\overline{\textit{Strategy-proof/Incentive-compatible}}$ *if* $\forall i, \forall v_i \in V_i$*:*
If $a = f(V_i, V_{-i})$ *and* $a' = f(V_i', V_{-i})$
then
$u_i = V_i(a) - P_i(V_i, V_{-i}) \geqslant V_i(a') - P_i(V_i', V_{-i}) = u_i'$

Definition 98 (Indirect Mechanism) *Players have:*

Types-Spaces: $T_1, ..., T_n$
Private Information: $t_i \in T_i$ $t_i : A \to \mathbb{R}$
Actions: $X_i, ..., X_n$
Strategies: $\mathcal{S}_i : T_i \to X_i$
Utilities: $u_i(t_i, \mathcal{S}_{-i}(t_i))$

Definition 99 (Direct Mechanism) *Players have:*

Social Choice Function f: $V_i \times V_2 \times ... \times V_n \to A$
Payment: $P_i : V_i \times ... \times V_n \to \mathbb{R}$
Utility: $u_i() \to V_i() - P_i$

Definition 100 (Greedy Algorithm) *An algorithm which makes a locally optimal choice at each stage with the hope of finding a global optimum.*

30 Revelation Principle

Definition 101 (Incentive Compatible) *Player i whose valuation is v_i would prefer telling the truth to the mechanism rather than possibly "lie" $v_i^{'}$, since this gives him higher utility.*

For every player i, every $v_i \in V_1, ..., v_n \in V_n$ and every $v_i^{'} \in V_i$
Telling the truth: $a = f(v_i, v_{-i})$
Lying: $a^{'} = f(v_i^{'}, v_{-i})$
Value of telling truth is greater than value of lying: $v_i(a) - p_i(v_i, v_{-i}) \geqslant v_i(a^{'}) - p(v_i^{'}, v_{-i})$

Theorem: If $\exists$ an arbitrary mechanism that implements f in **dominant strategies**, then $\exists$ an **incentive compatible (strategy-proof) direct** mechanism that implements f.
The payouts of the players in the incentive compatible mechanism are identical to the payouts at equilibrium of the original mechanism.

31 Bayesian Nash Equilibrium

Definition 102 *A strategy of a player i is a function $s_i : T_i \to X_i$. A profile of strategies $s_1, ..., s_n$ is a Bayesian-Nash Equilibrium if for every* ***player*** *i and every* ***type*** *t_i we have that $s_i(t_i)$ is the best response that i has to $s_{-i}()$ when his type is t_i, in expectation over the types of other players.*

Given n players:
•Type spaces: $T_1, ..., T_n$
•Distributions: $D_1, ..., D_n$ on type spaces $T_1, ..., T_n$
•Action spaces: $X_1, ..., X_n$
• Strategies: $s_i : T_i \to X_i$
•Alternative set A
•Players' valuations functions $v_i : T_i \times A :\to \mathbb{R}$
•Outcome function $a : X_1 \times ... \times X_n \to \mathbb{R}$
•Payment functions $p_1, ..., p_n$ where $p_i : X_1 \times ... \times X_n \to \mathbb{R}$
•Expected Utility:
$E_{D_{-i}}[u_i(t_i, s_i(t_i), s_{-i}(t_{-i})] \geqslant E_{D_{-i}}[u_i(t_i, x_i^{'}, s_{-i}(t_{-i})]$
where $E_{D_{-i}}$ denotes the expectation over the other types t_{-i} being chosen according to distribution D_{-i}
Implements social function $f : T_1 \times ... \times T_n \to A$

The main idea is that each player i must choose his action x_i when knowing t_i but not the other t_j's but rather only knowing the prior distribution D_j on each other t_j. The behavior of player i in such a setting is captured by a function that specifies which action x_i is taken for every possible type t_i.

32 Revenue Equivalence

Theorem (The Revenue Equivalence Principle):
Under certain weak assumptions, for every two Bayesian-Nash implementations of the same social choice functions f, we have that if for some type t_i^0 of player i, **the expected payment of player i is the same in the two mechanisms, then it is the same for every value of t_i.**
In particular, if for each player i there exists a type t_i^0 where the two mechanisms have the same expected payment for player i, then the two mechanisms have the same expected payments from each player and their expected revenues are the same.

32.1 Auction Example

First Price Auction:
Single Item
Two players: "Alice" and "Bob"
Private values of item: $t_{Alice} = a = Value_a$; $t_{Bob} = b = Value_b$
Bids: Alice's x, Bob's y
Distribution: D_{Bob} over b and D_{Alice} over a
Typespaces: T_{Alice} of Alice, and T_{Bob} of Bob $\in +\mathbb{R}$
Possible Outcomes: Alice wins, Bob wins
If Bob Wins: $v_{Alice} = 0$ and $v_{Bob} = t_{Bob}$
If Alice Wins: $v_{Alice} = t_{Alice}$ and $v_{Bob} = 0$
Win Criteria: if $x \geqslant y$ then Alice wins, else Bob wins
Payment Function: Alice pays 0 when she loses, and x when she wins; Bob pays 0 when he loses, and y when he wins

Find strategies: S_{Alice} for Alice and S_{Bob} for Bob such that they are Bayesian Nash Equilibrium, and best-replies to each other.

Lemma: In a first price auction among two players with prior distributions of the private values a, b uniform over the interval $[0, 1]$, the strategies $x(a) = a/2$ and $y(b) = b/2$ are in Bayesian-Nash Equilibrium.

Since $x < y$ if and only if $a < b$,
then the winner is the player with the highest private value.

Proof: Consider Alice's optimal response x to Bob's strategy $y = b/2$, when Alice has value a. $u_{Alice} = 0$ if shes loses, and $u_{Alice} = a - x$ if she wins, and pays x.
Thus, expected $u_{Alice} = Pr[Alicewinswithbidx] \cdot (a - x)$
Given Bob's strategy $y = b/2$, Alice wins when $x \geqslant b/2$ This happens with a probability of $2x$ for $0 \leqslant x \leqslant 1/2$ and 1 for $x \geqslant 1/2$ and 0 for $x \leqslant 0$.
To optimize the value of x, find the maximum of the function $2x(a - x)$ over the range $0 \leqslant x \leqslant 1/2$
Find the maximum by taking the derivative with respect to x, and equating it to 0, which gives $2a - 4x = 0$ whose solution is $x = a/2$

Second Price - Vickrey Auction
•Single Item
•Players: Alice, and Bob
•Value: Player i has value w_i of item
•Utility: $u_i = w_i - p$ where p is price paid for item, or 0 if the player loses
•Whoever bids the most, pays the price of the second highest bidder.
Given this game set-up, if Alice knew Bob's valuation y, then Alice would make her valuation $x = y + \epsilon$, where ϵ is some very small number, assuming $x \leqslant p$

Lemma: $x = a/2$ and $y = b/2$ are Bayes' Nash Equilibrium

Proof: Consider which x is optimal response to $y = b/2$
Utility for Alice is 0 (loss) or $a - x$ (win)
$u_a(x) = Pr[x > y(b)]$
If Alice bids $\geqslant 1/2$ she will win, so consider $0 \leqslant x \leqslant 1/2$
Probability $D_b[x > y(b)]$ for $y(b) = b/2$ and $x \in [0, 1/2]$
Probability $D_b[2x > b] = 2x$ and $u_a(x) = 2x(a - x)$
$max(2x(a - x))$
$0 \leqslant x \leqslant 1/2$
$(\sigma 2x(a - x))/(\sigma x) = 2a - 4x = 0$
$x = a/2$ and $y = b/2$

Revenue Equivalence From Both Auctions
Revenue First Price Auction: $max(a/2, b/2)$
Revenue Second Price Auction: $min(a, b)$
When a and b are chosen uniformly from $[0, 1]$, the expected value of $min(a, b) = max(a/2, b/2) = 1/3$
Thus, both auctions generate equivalent revenue in expectation.
This is a general circumstance for every two Bayesian-Nash implementations of the same social choice function.

33 Combinatorial Auctions

Definition 103 ***Combinatorial Auction:*** *An auction such that there is a set of m indivisible items that are concurrently auctioned among n bidders. Bidders have preferences regarding subsets - bundles - of items. Every bidder i has a valuation function v_i that describes his preferences in monetary terms.*

Definition 104 ***Valuation v:*** *is a real-valued function for each subset S of items, $v(S)$ is that value that bidder i obtains from winning the bundle. A valuation must have "free disposal" i.e., be monotone: for $S \subseteq T$, $v(S) \leqslant v(T)$, and $v(\emptyset) = 0$*
Valuation v_i of bidder i is private information.

Definition 105 ***An allocation*** *of the items among bidders is $S_1, ..., S_n$ where $S_i \bigcap S_j = \emptyset$ for every $i \neq j$.*
The social welfare obtained by an allocation is $\sum_i v_i(S_i)$.
A socially efficient allocation is an allocation with maximum social welfare among all allocations.

Definition 106 ***Single Minded Valuation:*** *A valuation v is single minded if there exists a bundle of items S^* and a value $v^* \in \Re^+$ such that $v(S) = v^*$ for all $S \supseteq S^*$, and $v(S) = 0$ for all other S. A single-minded bid is the pair (S^*, v^*)*

Goal: Design a mechanism that will find the socially efficient allocation.
Mechanism must be:
•Incentive Compatible
•Computationally Efficient
•Attains a value which is within $\sqrt{m}$ of the optimal solution.

Difficulties:
• *Computational complexity:* The allocation problem is NP-Complete even for simple special cases.
• *Representation and communication:* The valuation functions are exponential size objects since they specify a value for each bundle.
• *Strategies:* Analysis of strategic behavior of bidders.

34 NP-Completeness for Epsilon Approximation of Allocation Problem

Recall, an allocation gives disjoint sets of items S_i to each bidder i, and aims to maximize social welfare $\sum_i v_i(S_i)$

Definition 107 (Allocation Problem) *:*
INPUT: *(S_i^*, v_i^*) for each bidder $i = 1, ..., n$*
OUTPUT: *A subset of winning bids $W \subseteq 1, ..., n$ such that for every $i \neq j \in W, S_i^* \cap S_j^* = \varnothing$ (i.e. the winners are compatible with each other) with maximum social welfare $\sum_{i \in W} v_i^*$*
•This problem is a "weighted-packing" problem, and is NP-Complete, and is reduced from the Independent-Set problem.
•Consequently, the allocation problem has no polynomial time $O(M^{1/2-\epsilon})$ approximation for any $\epsilon > 0$

35 Greedy Algorithm for $\sqrt{n}$ Approximate Allocation

The Greedy Mechanism for Single-Minded Bidders:
Initialization:
• Reorder the bids such that $v_1^*/\sqrt{|S_1^*|} \geqslant v_2^*/\sqrt{|S_2^*|} \geqslant ... \geqslant v_n^*/\sqrt{|S_n^*|}$ • $W \leftarrow \varnothing$

For $i = 1...n$ do: if $S_i^* \cap (\cup_{j \in W} S_j^*) = \varnothing$ then $W \leftarrow W \cap i$
Output:
Allocation: The set of winners is W.
Payments: For each $i \in W, p_i = v_j^*/\sqrt{|S_j^*|/|S_i^*|}$ where j is the smallest index such that $S_k^* \cap S_j^* = \varnothing$ (if no such j exists, then $p_i = 0$)
When computing payments, j is chosen as the bidder who lost because of player i.

Theorem
The greedy mechanism is efficiently computable, incentive compatible, and produces a $\sqrt{m}$ approximation of the optimal social welfare.

36 Greedy Algorithm Proofs

Incentive Compatibility:

Lemma
Any mechanism for single-minded bidders in which losers pay 0 is incentive compatible if and only if it satisfies the following two conditions:
•**Monotonicity -** A bidder who wins with bid (S_i^*, v_i^*) keeps winning for any $v_i^{'} > v_i^*$ and for any $S_i^{'} \subset S_i^*$
•**Critical Payment -** A bidder who wins pays the minimum value needed for winning: the infimum of all values $v_i^{'}$ such that $(S_i^*, v_i^{'})$ still wins

Monotonicity implied since increasing v_i^* or decreasing $|S_i^*|$ will cause i to be visited sooner.
Critical payment is met, since: $1 : i$ wins as long as he appears in the greedy order before some j who wants an item in his bundle.
$2 :$ Payment computed is precisely the value which causes i to swap places with j in the greedy order.

Proof
• Utility of a player is always non-negative
The proof is primarily based on the fact that the player will always only pay the critical value p assuming that their bid (S, v) with value v is greater than p.
If the valuation $v < p$, then the player will not win, and pay 0
If $v < p$ and $v^{'} > v$, but $v^{'} > p$, then $(S, v^{'})$ will win, but this is not necessarily beneficial because this would be an inaccurate representation of the package value.

Approximation Factor:

Lemma
Let OPT be an allocation (i.e. set of winners) with maximum value of $\sum_{i \in OPT} v_i^*$ and let W be the output of the algorithm, then $\sum_{i \in OPT} v_i^* \leqslant \sqrt{m} \sum_{i \in W} v_i^*$

Proof

For each $i \in W$ let $OPT_i = j \in OPT, j \geqslant i | S_i^* \cap S_j^* \neq \emptyset$ be the set of elements in OPT that did not enter W because of i.

Clearly $OPT \subseteq \cup_{i \in W} OPT_i$ and the lemma will follow once we prove for every $i \in W, \sum_{j \in OPT_i} v_j^* \leqslant \sqrt{m} v_i^*$

Estimate $\sum_{j \in OPT_i} v_j^* \leqslant v_i^* / \sqrt{S_i^*} \sum_{j \in OPT_i} \sqrt{S_j^*}$

Then use Cauchy-Schwarz inequality:

$\sum_{j \in OPT_i} \sqrt{|S_j^*|} \leqslant \sqrt{OPT_i} \sqrt{\sum_{j \in OPT_i} |S_j|}$

Every S_j^* for $j \in OPT_i$ intersects S_i^*.

Since OPT is an allocation, these intersections are disjoint, and thus $|OPT_i| \leqslant |S_i^*|$. Since OPT is an allocation $\sum_{j \in OPT_i} |S_j| \leqslant m$.

Thus $\sum_{j \in OPT_i} \sqrt{|S_j^*|} \leqslant \sqrt{|S_j^*|} \sqrt{m}$ and plugging into the Cauchy-Schwarz Inequality, gives the claim $\sum_{j \in OPT_i} v_j^* \leqslant \sqrt{m} v_i^*$

Mechanisms without Money

Previously, we considered a number of environments where it was possible to enforce an optimal social outcome given that the environments involved some transfer of "money." By "money," we mean that preference orderings are specified using quasi-linear valuations and cost / payment functions. We saw that in the general case, implementing a mechanism with money becomes an NP-Hard problem. In particular, this situation occured with VCG mechanisms in combinatorial auctions.

However, there are many environments in which no transfer of money can be made, and as such, preferences are specificed strictly as an ordering. Arrow's Thorem and the Gibbard-Satterthwaite Theorem have already shown us that in this scenario, any mechanism / social choice function will lead to a dictatorship. This leads us to instead impose conditions on the set of preferences in a game, in an effort to design good mechanisms. Here, we consider such problems and study guarantees of optimality and strategy-proofness. As it turns out, mechanisms without money can be thought of as a set of measures that can work to characterize the performance of approximated solutions to NP-Hard problems.

37 Single-Peaked Preferences

We can look specifically at the class of problems where there is a set of individuals whose preferences can be mapped to a singular point in some space.

Definition The notion of single-peaked preferences is as follows:

- There is a one-dimensional ordering of alternatives: $A = [0, 1]$
- Each individual i has a **single peak** $x_i^* \in A$, s.t. $\forall a, b \in A$:

$$x_i^* \geqslant a > b \Rightarrow a \succ_i b$$

37.1 Median Voter Scheme

It turns out that a straightforward way to make a fair decision in the case of single-peaked preferences is to just pick the median point of all the agents' preferences. Formally, we define a **Median Voter Scheme** with a social choice function F having the property that it is *strategy-proof*, *onto*, and *anonymous* i.f.f there exists $y_1, \ldots, y_n \in A$, s.t. for all $(x_1^*, \ldots, x_n^*)$:

$$F(x_1^*, \ldots, x_n^*) = \text{median}(x_1^*, \ldots, x_n^*, y_1, \ldots, y_n)$$

Of course, our environment doesn't need to be limited to $[0, 1]$ or even $\mathbb{R}^1$. Instead, we can use any metric space where individuals have a preferred point within the space. To make this more concrete, we can look at the k-facilities problem.

37.2 k-Facilities Problem

Definition We have k facilities that we want to decide the location of, based on where each individual i wants a facility to be (x_i). However, each of the n individuals may report that they each want a facility at a different location y_i rather than their desired location x_i.

For the 1-Facility problem, where the space is defined on a $[0, 1]$ line, the median voter scheme (choosing the median of all reported locations $y_1, \ldots, y_n$) is an optimal and strategy-proof mechanism. To see this, consider the case where there are three players with preferences x_1, x_2, and x_3 placed on a line so that $x_1 < x_2 < x_3$. If x_1 reported y_1 to be to their left so that $y_1 < x_1$, this would not affect the median being x_2; if they reported it to the right of x_2 so that $x_2 < y_1 < x_3$, the median would become y_1 but it would be farther away from x_1 than if $y_1 < x_2$. A similar argument holds for the other two players. As such, no player has a reason to report a location different from their preferred location.

In the case of the 2-Facility problem on a $[0, 1]$ line however, the optimal solution is not strategy-proof – an individual can mis-represent their preferred position in order to bring one of the two facilities closer to them. This same issue ends up extending into any generalized metric space for $k \geqslant 2$ facilities, with any number of players.

It turns out though that for the 1-Facility problem in a general metric space, the optimal solution isn't strategyproof, and a strategy-proof solution is a dictatorship. However, there are known bounds on two types of strategy-proof mechanisms relative to the optimal solution:

$$\text{Deterministic Dictatorship} : cost \leqslant (n-1)OPT$$

$$\text{Randomized Dictatorship} : cost \leqslant 2OPT$$

In other words, a randomized mechanism will not be significantly worse than the optimal solution. In the case of the facilities problem, a randomized mechanism is a social choice function F mapping the location profile $(y_1, \ldots, y_n)$ to a distribution over sets of k facilities. An additional benefit of finding an approximate randomized mechanism is computational efficiency: a randomized mechanism is solvable in polynomial size of the input, whereas the exact optimum is an NP-Hard problem. This property has connections with the notion of finding the *core* of a game, which will be highlighted in the next section.

38 Cooperative Games

As we just saw, it is possible to characterize the quality of mechanisms obtained for a game where we specifically impose conditions on the preferences of players. Once again, we notice that the notion of finding the optimal social welfare / cost in a combinatorial optimization game ends up being an NP-Hard problem, whereas approximating it to a reasonable accuracy turns out to be solvable in polynomial size of the input.

In the study of cooperative games, this idea becomes much clearer with the notion of finding the *core* of a game: a stable, optimal solution where no individual can benefit from breaking from a coalition with other individuals. Determining whether some game has a core (the core-emptyness problem) is an NP-Hard problem with the strict formulation of a core. Relaxing this formulation, much like relaxing the constraints on optimality in the prior section, allows us to formulate approximate core-emptyness as a linear programming problem, solvable in polynomial time.

38.1 Definition

Similar to non-cooperative games, in a Cooperative Game there is a set of players N and a valuation $v(S)$, where S is a coalition. We also define a **coalition structure** CS, which is a partition of N.

With this in mind, we can define the optimum for the game G as:

$$\mathrm{OPT}(G) = \max_{CS} \sum_{S \in CS} v(S) \tag{19}$$

There are a few properties and classifications for cooperative games we can look at. In particular, a cooperative game is:

Simple: if $v(S) \in 0, 1$

Monotone: if for any $S \subseteq T \subseteq N$:

$$v(S) \leqslant v(T) \tag{20}$$

Superadditive: if for disjoint $S, T \subseteq N$:

$$v(S) + v(T) \leqslant v(S \cup T) \tag{21}$$

Convex: if for $S \subseteq T \subseteq N$ and $i \in N \backslash T$:

$$v(S \cup i) - v(S) \leqslant v(T \cup i) - v(T) \tag{22}$$

38.2 Core of a Game

In cooperative games, the *core* can be interpreted as a scenario in which no agent can benefit from breaking away from the coalition. First, we define an **imputation** as a vector $\mathbf{x} \in \mathbb{R}^n$ which satisfies *efficiency*: $\sum_{i \in S} x_i = v(S)$, $\forall S \in CS$. We say that an imputation $\mathbf{x}$ is in the core of a game if:

$$\sum_{i \in S} x_i \geqslant v(S), \forall S \subseteq N \tag{23}$$

Example of a Core for 3 players : We then take x_i to be the individual valuation of a player i ($x_i = v(\{i\})$, and $v(S)$ to be the valuation of a coalition of players. For a 3-player game, the valuation of a full coalition $v(\{1, 2, 3\}) = v(N) = x_1 + x_2 + x_3$. Then, we can rewrite equation 23 into the following three inequalities:

$$\begin{aligned}
x_1 + x_2 \geqslant v(\{1, 2\}) \to x_3 \leqslant v(N) - v(\{1, 2\}) \\
x_2 + x_3 \geqslant v(\{2, 3\}) \to x_1 \leqslant v(N) - v(\{2, 3\}) \\
x_1 + x_3 \geqslant v(\{1, 3\}) \to x_2 \leqslant v(N) - v(\{1, 3\})
\end{aligned}$$

The core then becomes the feasible region defined by the full system of constraints.

38.3 Core-Emptiness and Core-Approximation

It's possible for the core of a game to be empty, in that there is no imputation $\mathbf{x}$ which satisfies the system of linear constraints defining the core. Determining whether the core is non-empty in the general case turns out to be NP-Hard.

We can relax the definition of a core with that of a γ-*approximate core*:

Definition A vector $\mathbf{x} \in \mathbb{R}^N$ is in the γ-core of the game if it satisfies:

> **γ-Budget balance**: $v(N) \geqslant \sum_{i \in N} x_i \geqslant \gamma v(N)$, $\gamma \leqslant 1$
>
> **Core Property**: $\sum_{i \in S} x_i \geqslant v(S), \forall S \subseteq N$

This formulation ends up being solvable as a polynomial-size LP problem.

38.4 Shapley Value

The Shapley value can be considered as a "fair" way to share cost, and has the benefit that it always exists for any given game unlike the core which can be empty. The Shapley value may also not be within the core of a game which does have a non-empty core. Another issue with the core is that if it is non-empty, then there could be more than one imputation which satisfies it – creating the additional problem of determining a way to decide which satisfying imputation should be picked. The Shapley value eliminates this issue by providing just a single solution.

To define the Shapley Value, we need to look at the marginal contribution that a player i adds if they were to join a coalition S among the set of players N. This is simply:

$$m_i(S) = v(S \cup \{i\}) - v(S) \tag{24}$$

Note that the order in which players join a coalition can affect their marginal contribution m_i. We define the predecessors of i in joining S to be:

$$P_i(\sigma) = \{j \in N | \sigma(j) < \sigma(i)\}, \sigma \in \Pi(N) \tag{25}$$

where $\Pi(N)$ is a permutation of players. Then,

$$m_i(\sigma) = m_i(P_i(\sigma)). \tag{26}$$

Definition If the ordering of players is chosen uniformly at random, then the Shapley value for some player i is

$$\phi_i = \mathbb{E}[m_i(\sigma)] = \frac{1}{n!} \sum_{\sigma \in \Pi(N)} m_i(\sigma) \tag{27}$$

The Shapley Value has the following properties:

> **Efficiency**: $\sum_{i \in N} \phi_i = v(N)$
>
> **Symmetry**: Players who contribute equally are paid equally: $\phi_i = \phi_j$ if $m_i(S) = m_j(S)$
>
> **Dummy**: A player who adds no value should not be paid anything: $\phi_i = 0$ if $m_i = 0$
>
> **Additivity**: $\phi_i(\sigma_1) + \phi_i(\sigma_2) = \phi_i(\sigma_1 + \sigma_2)$

Theorem 108 *Any value that satisfies efficiency, symmetry, dummy, and additivity is the Shapley Value.*

HW: Prove this theorem.

39 Online Algorithms

Keeping in theme with this idea of approximate algorithms to hard optimization problems and characterizing the trade-offs made when using one, we can look at the topic of online algorithms. Although only briefly covered towards the end of the week, online algorithms are algorithms which operate without seeing its full input all at once: the input arrives "online," or dynamically over time. These are in contrast to offline algorithms which see the input all at once. We can characterize the performance of an online algorithm to an offline algorithm using the notion of **competitive ratio**. To illustrate this, we can look at the problem of matching on a bipartite graph.

Let there be a bipartite graph $G = (U, V, E)$, where $|U| = n$. The vertices U are known in advance (offline), while the vertices V arrive dynamically (online) along with their edges E. We'd like to maximize the size of the matching in G. We define the competitive ratio as:

$$\frac{ALG(G, \pi)}{OPT(G)} \geqslant \alpha \tag{28}$$

where $\alpha \leqslant 1$, G is the game, and π is the input order that arrives online.

Keeping in mind the previous pattern we've observed when characterizing the complexity and performance of various algorithms / mechanisms, it turns out for this problem, the competitive ratio of several online algorithms are as follows:

Deterministic algorithm $\leqslant 1/2$

Random ranking algorithm $= 1 - \frac{1}{\epsilon}$

It turns out that there is an upper bound for any randomized, online algorithm:

Theorem 109 *No randomized algorithm has a competitive ratio that is greater than* $1 - \frac{1}{\epsilon} + O(1)$.

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018

Lecture 11 April 16 - April 20

Lecturer: Dr. Meera Sitharam — Scribe: Archit Rathi

40 Introduction

Near the end of the previous lecture, we discussed briefly about the concept of online algorithms. Quite simply, online algorithms are algorithms that function without seeing the all the input. Instead, the input arrives in pieces or one by one, and the algorithm performs its actions on whatever portion of the input its received. Clearly, an algorithm that has all its input upon start (offline algorithms) will tend to perform better than online algorithms. If we consider the offline algorithm as the optimal solution, we know have a way to understand the performance of an online algorithm:

$$\frac{\text{Online Algorithm}}{\text{Optimal (Offline) Algorithm}}$$

The ratio between the performance of the online algorithm and the optimal is called a **competitive ratio**. We see that the competitive ratio can be no greater than 1(where the offline algorithm has optimal performance).

41 Bipartite Graph Matching

Let's establish some definitions first:

Definition 110 (Bipartite) *A bipartite graph is a graph whose vertices can be split into two different sets, where each vertex is never connected to another vertex in its own set. For example, for sets U and V, the vertices in U can only connect to the vertices in V and vice versa.*

Definition 111 (Matching) *Matching exists in a graph no set of edges share any vertices. A graph is matching if each vertex either has 0 or 1 edge connected to it.*

Image we had two sets of vertices (U and V), which combined made up a bipartite graph. Our setup would then be:

- All the vertices in V are known at the start of the algorithm(offline)
- The vertices in U arrive one by one. Any vertex $u \in U$ comes with all the edges it's connected to
- As soon as the new vertex $u \in U$ arrives, it is matched arbitrarily

Given this, we can update the competitive ratio given in the introduction to be more pertinent. If A is the online algorithm and O is the optimal, then:

$$\frac{A(G, \pi)}{O(G)}$$

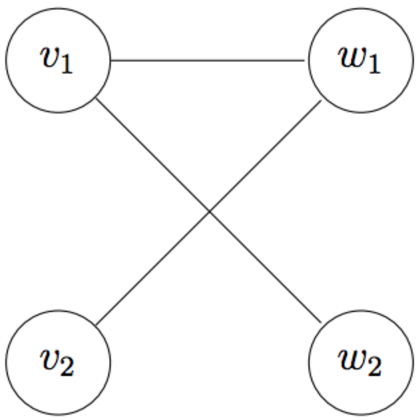
where G is the graph and π is the order the online input arrives in.

42 Maximum Matching

Theorem 1 The competitive ratio of any deterministic online matching algorithm is at most $\frac{1}{2}$.

Definition 112 (Deterministic Algorithm) *A deterministic algorithm is an algorithm that always outputs the same thing given a certain input, regardless of the machine operating it.*

To understand why Theorem 1 is true, consider at the example below[17]:



The vertices v_1, v_2 are both known at the beginning of the algorithm. When vertex w_1 arrives, the algorithm has the option between edges (v_1, w_1) or (v_2, w_1). Imagine that the algorithm arbitrarily chose the first of the two, but then vertex w_2 arrived and its only connection was v_1, which is already matched. In this scenario, the maximum matching would have been 2, but our algorithm had a matching of 1. In every deterministic algorithm, the online algorithm will have a matching at most $\frac{1}{2}$ the maximum possible matching. Every maximal matching has a size of at least one half off that of maximum matching.

43 Algorithmic Ranking

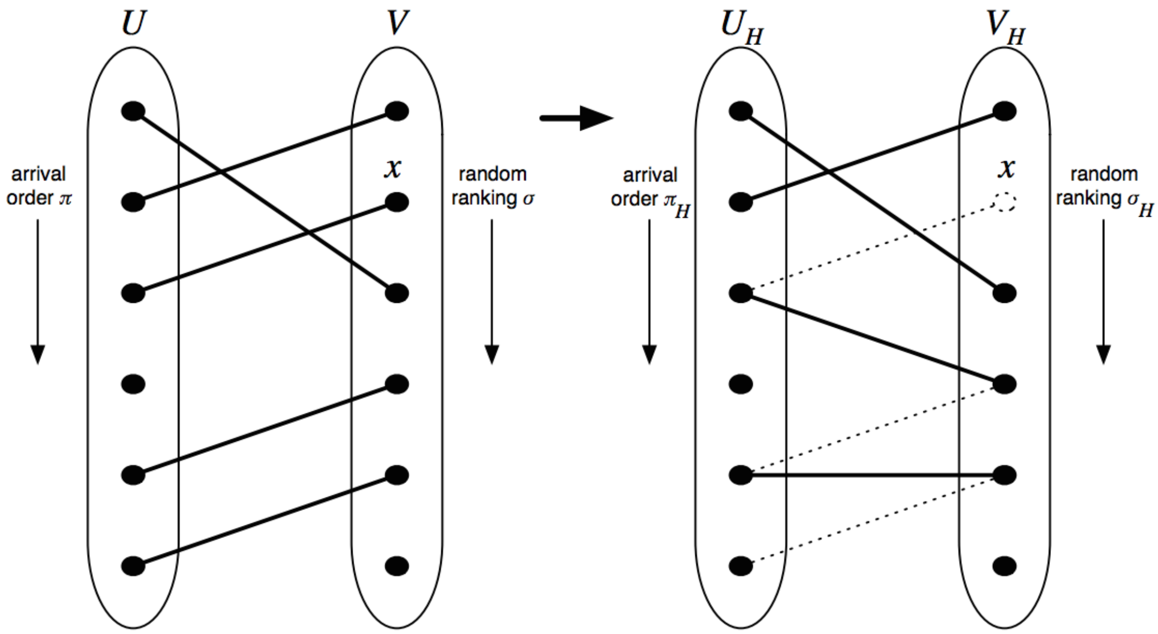
So far, all the online algorithms we've analyzed have been purely deterministic. If a deterministic algorithm can only achieve a competitive ratio of $\frac{1}{2}$, then maybe a procedure that involves some randomization could potentially do better. As a result, the Ranking algorithm was created, and with it, a maximum competitive ratio of $1 - \frac{1}{e}$ was achieved.

Theorem 2 Ranking achieves a competitive ratio of $1 - \frac{1}{e}$.

Let's attempt to prove this, using the proof we learned in class and [18]

Consider a bipartite graph, G, with the two sets U and V, where U is the set that comes online. At the beginning of the algorithm, a random permutation of V is create and is called σ and the order the online input arrives in is called π. It is critical that V is randomly permuted. If it is not, than the upper bound is the same as that of a deterministic algorithm. Upon arrival of every vertex $u \in U$, we attempt to find a matching for u. If no matching is achievable, we match u to a vertex $v \in V$ that minimizes $\sigma(v)$. Using this notation, let Ranking(G, σ, π) represent the matching of graph G with online order π and ranking σ

Lemma 1 *If $x \in G$, $H = G/\{x\}$, and π_H, σ_H are π, σ restricted to H, then the matchings between Ranking(G, σ, π) and Ranking(H, σ_H, π_H) differ by a single alternating path starting at vertex x*



We can see **Lemma 1** occur in the image above. After vertex x is removed, the matching from Ranking(G, σ, π) differs to the matching from Ranking(H, σ_H, π_H) by a single alternating path. If we keep removing vertices, that are not in the matching, we see that the competitive ratio comes from graphs with a perfect matching. For the rest of these notes, we'll assume G has a perfect matching m^*.

Lemma 2 *Fix $u \in U$ and $v = m^*(u)$. If v is not matched by Ranking(G, σ, π), the u is matched to vertex v' such that $\sigma(u') \leqslant \sigma(u)$*
Proof: If v is not matched by Ranking(G, σ, π), then when u arrives, its eligible neighbors include v. Following that, there must exist a v' where $\sigma(u') \leqslant \sigma(u)$

Lemma 2 helps us prove **Lemma 3** below.

Lemma 3 *Let x_t be the probability that a vertex of V with $\sigma(u) = t$ is matched. Then $1 - x_t \leqslant \frac{1}{n}\sum_{1 \leqslant s \leqslant t} x_s$.*

In the Lemma 3 above, x_t is the probability that a matching occurs, so therefore $1 - x_t$ is the probability that a matching doesn't occur. With this lemma, we can now prove Theorem 1. Since G has perfect matching, the competitive ratio is basically

$$\underset{n \to \infty}{inf} \frac{1}{n} \sum_{1 \leqslant s \leqslant n} x_s$$

where $n = |U|$. We can now rewrite Lemma 3 as

$$S_t(1 + \frac{1}{n}) \geqslant 1 + S_{t-1}$$

where

$$S_t = \sum_{1 \leqslant s \leqslant t} x_s$$

The infimum clearly occurs when the inequalities are tight. From here, we get

$$S_t = \sum_{s=1}^{t} (1 - \frac{1}{n+1})^s$$

which then leads to a competitive ratio of at least

$$\frac{1}{n} \sum_{s=1}^{n} (1 - \frac{1}{n+1})^s = 1 - (1 - \frac{1}{n+1})^n$$

which asymptotically approaches $1 - \frac{1}{e}$ as n approaches infinity.

Proof of Lemma 3: Suppose $v \in V$ with rank $t = \sigma(u)$. Remember that $1 - x_t$ is the probability that v is not matched. Let u be a vertex where $v = m^*(u)$ and $R_{t-1} \subset U$ mean the set of vertices of U matched by the algorithm to the vertices of V who have a rank less than $t - 1$. The cardinality of R_{t-1} is $\sum_{1 \leqslant s \leqslant t-1} x_s$. We can now use Lemma 1: if v is not matched, then u is a part of R_{t-1}. Unfortunately, u and R_{t-1} are not independent and therefore this proof is incorrect.

Lemma 4 *Let $u \in U$ and $v = m^*(u)$. If σ' is a permutation, then σ_i is the permutation we get from σ' by removing vertex v and putting it back where its rank is i. If v is wasn't matched in the Ranking of σ', then, for every i, u is matched by Ranking(σ_i) to a vertex v_i whose rank $\sigma_i(v_i)$ is at most $\sigma'(v)$*

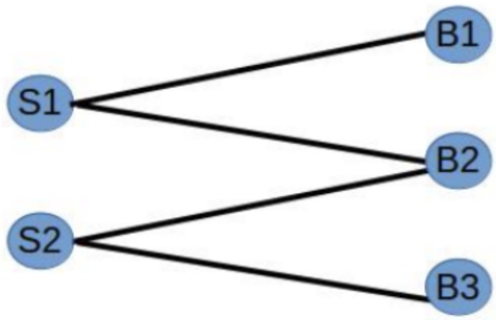
Using Lemma 4, we can now correct our proof. Let's say we have a random permutation σ and another permutation σ' that we can get by randomly picking a vertex $v \in V$, taking it out of σ and putting it back so the rank is t. Using Lemma 4, we see that if v is not matched by Ranking(σ'), then u is matched to Ranking(σ) with a vertex, $\bar{v}$ such that $\sigma(\bar{v}) \leqslant t$. This basically means that $u \in R_t$ and that u is now independent of R_t.

44 Ranking Example

As an end to these notes, let's consider a quick application of the Ranking algorithm: Google Adwords.

In a search engine like Google, companies will often bid on certain keywords when buying ads. It is up to Google to decide which companies to show and how to much to charge them. Imagine bidders are represented by V and the keywords by U. As each keyword comes one by one, the bids placed by the company for it are revealed, upon which Google decides which company to choose for that keyword. This example is essentially an online bipartite matching example where every bid is either 1 or 0 and the budget of every company is 1.

Another example is the pricing game we discussed earlier in this course. The image below should give you a reminder of what the game was. (It was discussed in Lecture Notes 2, Section 1.4)



As we can see, this is another Bipartite graph. In this case we could view the Sellers as set V and the buyers as set U. As more and more buyers come in, we have to decide how to match the buyers to sellers for maximal matching.

References

[17] Tim Roughgarden *CS261: A Second Course in Algorithms Lecture 14: Online Bipartite Matching* 2016.

[18] Benjamin Birnbaum and Claire Mathieu. *On-line Bipartite Matching Made Simple*. 2013.

Algorithmic Economics CIS 6930/CIS 4930	Spring 2018

Lecture — Apr. 24

Lecturer: Dr. Meera Sitharam — Scribe: Victoria Crawford

45 Dividing an Interval

Suppose we would like to divide the interval $[0,1]$ among n people. By divide the interval $[0,1]$, we mean find $\alpha_1, ..., \alpha_n$ such that each α_i is a finite union of disjoint intervals of $[0,1]$, which we will call a "piece", and $\cup_{i=1}^{n} \alpha_i = [0,1]$. α_i is the piece of $[0,1]$ that is given to the ith person. Further, suppose that each person i has a valuation function v_i that takes pieces of $[0,1]$ to how much that person values that piece, and has the following properties:

16. $v_i(\alpha) \in [0,1]$ for any piece α
17. $v_i(\alpha + \beta) = v_i(\alpha) + v_i(\beta)$ for any pieces α, β
18. If $\alpha \subseteq \beta$, then $v_i(\alpha) \leqslant v_i(\beta)$.
19. $v_i([0,1]) = 1$
20. $v_i(\varnothing) = 0$

The following are two ideas of fairness that we can use to evaluate the division of the interval:

Definition 113 (Proportionality:) *A division $\alpha_1, ..., \alpha_n$ is proportional if for every person i, $v_i(\alpha_i) \geqslant \frac{1}{n}$.*

Definition 114 (Envy-Free) *A division $\alpha_1, ..., \alpha_n$ is envy-free if for every person i, and for every other person $j \neq i$, $v_i(\alpha_i) \geqslant v_i(\alpha_j)$.*

In proportionality, each person believes they have received at least their fair share of the interval. In envy-freeness, there doesn't exist a person who prefers a different person's piece over their own. Envy-freeness implies proportionality, but the reverse is not true (see homework).

Lemma 115 *If a division $\alpha_1, ..., \alpha_n$ is envy-free, then it is proportional.*

Proof: Consider any person i. By properties (2) and (4) of the valuation functions, $\sum_{j=1}^{n} v_i(\alpha_j) = v_i([0,1]) = 1$. By the pigeonhole principle, there must exist j such that $v_i(\alpha_j) \geqslant \frac{1}{n}$. Since $\alpha_1, ..., \alpha_n$ is envy-free, $v_i(\alpha_i) \geqslant v_i(\alpha_j)$, which implies that $v_i(\alpha_i) \geqslant \frac{1}{n}$. ■

We now look at two examples of interval division.

Cake cutting In cake cutting, the interval represents an entire cake. The cake is rectangular and cut by parallel lines into n connected pieces that are then given to n people. Different people may value parts or portions of the cake differently. For example, one person may prefer the parts with more icing on the end to those in the center, and the reverse may be true of another person.

Roommate rent sharing In roommate rent sharing, we have three roommates and three rooms. The interval is the division of the rent into the three rooms. Not only do we need to split the rent among the rooms, but assign people to rooms. Different people may have different preferences among the rooms. In addition, they may be willing to pay different amounts for each room. For this application, properties (3)-(5) of valuation functions would be dropped since people prefer lower rents.

45.1 Applying Sperner's Lemma

Does there always exist a division of the interval that is envy-free? In the case of cake cutting, there is. In fact, we may show this by using Sperner's Lemma. We will specifically consider the case where there are three people. Recall that Sperner's Lemma, for the special case of triangles, is stated as follows:

Lemma 116 *Suppose that we have a triangulated triangle, and a Sperner Coloring of the vertices of the triangulation in three colors. Then there exists at least one triangle with a vertex of each color.*

Suppose that we have a cake represented by the interval $[0, 1]$, and people $\{A, B, C\}$ each with valuation functions satisfying the conditions listed in Section 45. Consider any triangle with vertices $(1, 0, 0), (0, 1, 0), (0, 0, 1)$, and any triangulation of that triangle. Each vertex (x, y, z) in the triangulation can be identified with a cut of the cake $[0, x], [x, x + y], [x + y, x + y + z] = [x + y, 1]$. Assign a person to each vertex in such a way that every elementary triangle has vertices assigned A, B, and C. Let 1,2, and 3 be the pieces of the cake from left to right. "Color" each vertex with the piece of the cake that the person who is assigned to the vertex prefers the most. By properties (3)-(5) of the valuation functions, this is a Sperner Coloring. Therefore we can apply Sperner's Lemma to say there exists some triangle where each person values a different cut of the cake the most. Since we can make the triangulation arbitrarily small, the limit gives a division that is envy-free.

45.1.1 Sperner's Lemma for Roommate Rent Sharing

A similar approach will work to show the existence of an envy-free rent and room allocation for the roommate rent sharing problem. However, notice that if we colored the triangle in the same way as in cake cutting that this would not necessarily produce a Sperner Coloring since a lower rent would be preferred. This issue can be dealt with by a new version of Sperner's Lemma on the dual simplex, where edges and vertices are switched.

46 Cake Cutting Algorithms

While Sperner's Lemma can be used to show the existence of a fair division for cake cutting, finding the fair division is another matter. There exists a number of algorithms in order divide a cake in a fair way among n people. Throughout this section, we assume the Robertson-Webb model for cake cutting [24].

46.0.1 Cut-and-Choose

This algorithm is specifically for cake division where there are only two people, $\{A, B\}$. First, A divides the cake into α, β such that $v_A(\alpha) = v_A(\beta) = \frac{1}{2}$. Then B picks $x \in \{\alpha, \beta\}$ such that $v_B(x)$ is greatest. Finally, A picks the remaining piece.

Lemma 117 *Cut-and-Choose produces a division that is envy-free (and therefore also proportional).*

46.0.2 Dubins-Spanier

This algorithm takes place over n rounds, each on a subset of the cake $[a, 1] \subseteq [0, 1]$, starting with $[0, 1]$. Each round goes as follows: First, each player i makes a mark m_i on the cake $[a, 1]$ such that $v_i([a, m_i]) = \frac{1}{n}$. The player i with the left most mark cuts off $[a, m_i]$ as their piece. The procedure repeats with the remaining cake $[m_i, 1]$.

Lemma 118 *Dubins-Spanier produces a division that is proportional in $\Theta(n^2)$ time.*

46.0.3 Evan-Paz

For this algorithm, we assume $n = 2^k$. The following procedure is recursively called on portions of the cake $[a, b]$ along with subsets of people assigned to that portion, starting with the entire cake $[0, 1]$ and all people: First, if there is only one person for the portion of cake, give the person the entire portion as their piece. Otherwise, have every person i make a mark x_i on $[a, b]$ such that $v_i([a, x_i]) = \frac{1}{2}v_i([a, b])$. Let y be the $\frac{r}{2}$ mark from the left of the portion, where there are r people assigned to this portion. Divide the portion $[a, b]$ into $[a, y]$ and $[y, b]$. Those players i that made a mark $x_i \leqslant y$, assign to the left portion, and others to the right portion. Follow the same procedure with the two new portions and the people assigned to them.

Lemma 119 *Evan-Paz produces a division that is proportional in $\Theta(n \log(n))$ time.*

In fact, Evan-Paz is provably optimal among divisions that are proportional.

Theorem 120 *Edmonds and Pruh 2006 [20]: Any proportional protocol needs $\Omega(n \log(n))$ operations for cake cutting.*

46.0.4 Selfridge-Conway

For this algorithm, we assume $n = 3$. This algorithm takes place over three stages.

In the first stage, person 1 cuts the cake into three pieces of equal value according to v_1. Next, player 2 trims a single piece so that the two largest according to v_2 are now tied according to v_2. Call the untrimmed cake Cake 1, and the cake without the trimmings Cake 2.

In the second stage, person 3 first picks which of the three pieces of Cake 1 they prefer. If it wasn't the trimmed piece, then person 2 gets the trimmed piece. If it was the trimmed piece, then person 2 gets to pick their preferred one of the two remaining pieces. Let person x be the one who now has the trimmed piece, $x \in \{2, 3\}$. Let $y = \{2, 3\} \backslash \{x\}$.

In the final stage, y cuts Cake 2 into three equal pieces according to v_y. Then, the pieces of the cake are chosen by x, 1, and y according to v_x, v_1, and v_y respectively, in that order.

Lemma 121 *Selfridge-Conway produces a division that is envy-free (and therefore proportional).*

While there does exist an algorithm that produces an envy-free cake cutting for n people, it may take arbitrarily long.

Theorem 122 *Brams and Taylor 1995 [19]: There exists an unbounded envy-free cake cutting algorithm.*

47 Indivisible Goods

So far, division has occurred on an interval that could be broken into arbitrarily small pieces. Consider a different situation, where we have a set G which is composed of m goods. There are n people, and the ith person has a valuation function v_i on subsets of goods. We seek to find an allocation $A = \{A_1, ..., A_n\}$ where $A_1, ..., A_n$ is a partition of G, and the ith player receives goods A_i. In this case, finding divisions that are proportional and/or envy-free is infeasible, as the following theorem shows.

Theorem 123 *Nisan and Segal 2002 [22]: Let A be an allocation of G, $e_{ij}(A) = \max\{0, v_i(A_j) - v_i(A_i)\}$, and $e(A) = \max\{e_{ij}(A) : i, j \in \{1, ..., n\}\}$. Then every protocol that finds an allocation minimizing $e(A)$ uses an exponential number of bits of communication in the worst case.*

48 Motivation of Fairness

We have previously assumed that the behavior of each person has only been motivated by the value of what they receive, i.e. they are rational. However, this is not necessarily true, and some people may in fact be motivated by fairness [23] [21].

Consider an experiment where there are two people, A and B, and A is offering B x money in dollars, $x \in [0, 100]$. If B accepts, B gets x dollars and A gets $100 - x$ dollars. If B declines, then neither get any money. If we assume that B is rational, B's utility will look like:

$$u_B = \begin{cases} x & \text{accept} \\ 0 & \text{decline} \end{cases}$$

And so B will always accept the offer if $x > 0$. However, experiments show that this is not how all people behave [21]. If x is too low, B may decline the offer since it is unfair, even though B would have more money if she accepted. In this case, a more accurate utility would be, for some $\alpha \geqslant 0$:

$$u_B = \begin{cases} x - \alpha((100 - x) - x) & \text{accept} \\ 0 & \text{decline} \end{cases}$$

In addition, B may be altruistic and not want to get more than A. In this case, B's utility could be for some $\alpha, \beta \geqslant 0$:

$$u_B = \begin{cases} x - \alpha((100 - x) - x) - \beta(x - (100 - x)) & \text{accept} \\ 0 & \text{decline} \end{cases}$$

49 Homework

21. Find a counter-example to the claim that if a division is proportional, then it is envy-free.

References

[19] Steven J Brams and Alan D Taylor. An envy-free cake division protocol. *The American Mathematical Monthly*, 102(1):9–18, 1995.

[20] Jeff Edmonds and Kirk Pruhs. Cake cutting really is not a piece of cake. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 271–278. Society for Industrial and Applied Mathematics, 2006.

[21] Ernst Fehr and Simon Gächter. Fairness and retaliation: The economics of reciprocity. *Journal of economic perspectives*, 14(3):159–181, 2000.

[22] Noam Nisan and Ilya Segal. The communication complexity of efficient allocation problems. *Draft. Second version March 5th*, pages 173–182, 2002.

[23] Matthew Rabin. Incorporating fairness into game theory and economics. *The American economic review*, pages 1281–1302, 1993.

[24] Jack Robertson and William Webb. *Cake-cutting algorithms: Be fair if you can.* AK Peters/CRC Press, 1998.