Algorithmic Game Theory

Notes by Arjun Maneesh Agarwal

based on course by Prajakta Nimbhorkar

1. Games and Equilibria : We are assuming games where everyone is acting selfishly. When everyone doesn't wish to switch strategies, we get an equilibrium.
   - We will see two-player games and multiplayer games.
   - Equilibrium and existence and computation.
2. Mechanism Design : Given multiple selfish agents, we need to make a mechanism that encourages (or discourages using hardness) the behavior we wish to encourage.
   - We will see auctions, voting etc.
3. Fair Division: This is an allocation problem where we want to allocate some resources in a 'fair' (or less unfair) way.
   - Distributing divisible and indivisible resources fairly among agents, given valuation.
   - EF, EQ, PROP and its relaxations.

Table of Contents

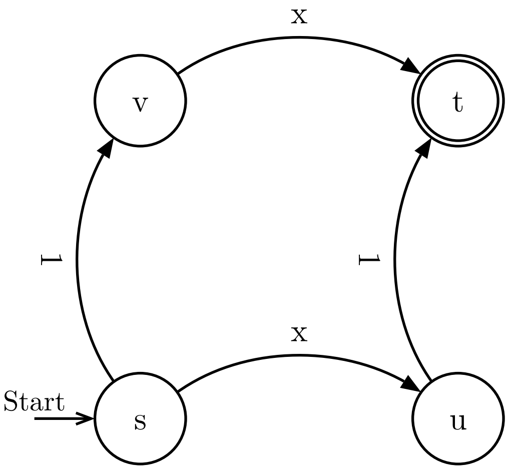
1. Games and Equilibria

We make the following assumptions:

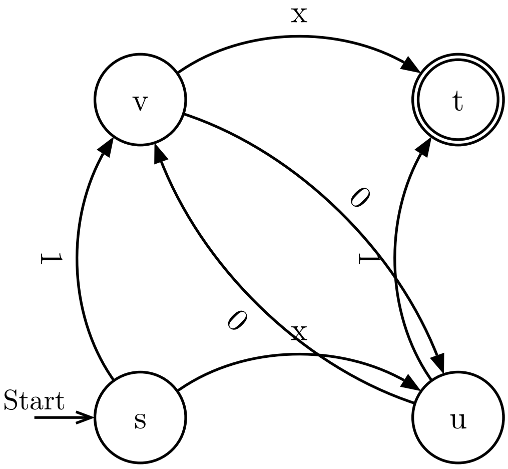
1. Rationality/selfishness : Each agent attempts to maximize own payoff and believes others will do so as well as well has knowledge of same behavior being
2. Intelligence: Agents have enough computational resources to take into account the strategies and behaviors of others.
3. Common Knowledge: P, S_i, μ are known to everyone and everyone knows that everyone knows them and so on.

1.1. Braess's Paradox

Taking the example of traffic, let the network be:



In this case, the equilibrium is 1.5 hours as traffic is divided in the two paths. Let's say the authorities build a bridge:



This will lead to a (weak) equilibrium of 2 hours as all agents take the route s-v-u-t.

This is called Braess's Paradox.

Side Note

We have actually seen this example occur almost verbatim in Seoul where the demolition of a bridge in 2005, solved a 2 decade old traffic problem.

Furthermore, Steinberg and Zangwill, 1983; showed that in a random network, Braess's paradox is approximately 50% likely to occur in an random network.

It is also a consideration in google maps where the navigation sometimes suggests a longer route, not to avoid a pre-existing traffic jam, but to avoid creating one. (This is why google map is not referentially transparent and may suggest two people different routes between same places at same time)

We also would like to define the term 'cost of anarchy' here which refers to the ratio of equilibrium payouts (or cost) divided by optimal payouts(or cost). In this case, anarchy was $\frac{2}{1.5} = \frac{4}{3}$ times costlier.

1.2. Prisoner's Dilemma

		P2	
		C	D
P1	C	$4, 4$	$5, 1$
	D	$1, 5$	$2, 2$

Here both players have incentive to deviate from the mutually best case as confessing (strongly) dominates denial and hence, is a Nash equilibrium.

Let's define these words formally.

Definition: Game

To define a game we need to know the players and strategies for each player. Thus, a game $G = (P, S, \mu)$ where for i in P, S_i in S denotes the set of possible strategies for player i.

In a particular play of the game denoted as s, i plays strategy s_i forming the strategy profile or game vector $(s_1, s_2, \ldots, s_n)$.

We denote the payoffs or costs of the game for player i using a function $\mu_i : S_1 \times S_2 \times \ldots \times S_n \to \mathbb{R}$ which takes the game vector and tells the payoff for player i. $\mu : S_1 \times S_2 \times \ldots \times S_n \to \mathbb{R}^n; \mu(s) = (\mu_1(s), \mu_2(s), \ldots, \mu_n(s))$ is the payoff vector returning function.

Finally, for conciseness, we sometimes denote the choices for all players except i as s_{-i}. Thus, $\mu_i(s_i, s_{-i})$

Definition: Strongly Dominant Strategy

Player i's strategy $\hat{s_i}$ is a **Strongly Dominant Strategy** to the strategy s_{-i} of other players if:

$$\mu_i(\hat{s}_i, s_{-i}) > \mu_i(s_i', s_{-i}) \forall s_i' \in S_i$$

Definition: Weakly Dominant Strategy

Player i's strategy $\hat{s_i}$ is a **Weakly Dominant Strategy** to the strategy s_{-i} of other players if:

$$\mu_i(\hat{s}_i, s_{-i}) \geq \mu_i(s_i', s_{-i}) \forall s_i' \in S_i$$

and

$$\exists s_i' \in S_i \text{ s.t.} \mu_i(\hat{s}_i, s_{-i}) > \mu_i(s_i', s_{-i})$$

Definition: Weakly Dominant Strategy / Best Response(BR)

Player i's strategy $\hat{s_i}$ is a **best response** to the strategy s_{-i} of other players if:

$$\mu_i(\hat{s}_i, s_{-i}) \geq \mu_i(s_i', s_{-i}) \forall s_i' \in S_i$$

or if $\hat{s_i}$ solves $\max_{S_i} \mu_i(s_i, s_{-i})$.[1]

Definition: Strongly Dominated Strategy Equilibrium

Informally, this is when all players are playing dominant strategies.

It is a strategy profile $(s_1^*, s_2^*, ..., s_N^*)$ is a SDSE if for each i, her choice s_i^* is strongly dominant to all .

Definition: Nash Equilibrium(NE)

Informally, this is when all players are playing BR to each other.

Formally, it is a strategy profile $(s_1^*, s_2^*, ..., s_N^*)$ is a NE if for each i, her choice s_i^* is a best response to s_{-i}^*.

[1] Sometimes we can replace the actual strategies by others by p, which is i's belief of others' choices. This is often done in econ, but here, one of our assumptions (Common knowledge) allows us to not need it. The BR word is also from Econ.

Remark

The reason Nash equilibrium is important is as it leads to no-regrets as no individual can do strictly better by deviating holding others fixed. Second, as it is a self-fulfilling prophecy (we will see what this means).

1.3. Pollution Control

Let there be n countries with the cost of pollution control being 3. If a country doesn't control pollution, it adds cost 1 to each country.

This leads to the dark case of no one controlling pollution. We can check that if only k countries control pollution, all have incentive to not control.

Definition

A dominant strategy is what is best for a player regardless of whatever strategy the rest of the players choose to adopt.

1.4. RPS

			P2	
		R	P	S
	R	$0, 0$	$1, -1$	$-1, 1$
P1	P	$1, -1$	$0, 0$	$-1, 1$
	S	$0, 0$	$-1, 1$	$1, -1$

There is no pure Nash equilibrium here. But allowing randomized strategy, we can get an equilibrium.

Definition: Mixed Strategy

A mixed strategy p_i is a randomization over i's pure strategies.

- $p_i(s_i)$ is the probability i assigns to the pure strategy s_i.
- $p_i(s_i)$ could be zero for some i.
- p_i could be one (in case of a pure strategy).

Definition: Mixed Strategy Profile

A mixed strategy profile is a tuple $(p_1, p_2, ..., p_n)$ where each p_i is a mixed strategy for player i. It represents the joint behavior of all players where each one plays a randomized strategy independently.

Definition: Expected Payoff

The expected payoff for player i is:

$$\sum_{(s_1,\ldots,s_n)\in S_1\times\ldots\times S_n} \mu_i(s_i,s_{-i})\prod_{a=1}^{n} p_a(s_a)$$

Definition: Mixed Strategy Nash Equilibrium

A mixed strategy profile $(p_1^*, p_2^*, \ldots, p_n^*)$ is a mixed strategy NE (MSNE) if for each player i, p_i^* is a BR to p_{-i}^*. That is:

$$\mu_i(p_i^*, p_{-i}^*) \geq \mu_i(\sigma_i, \sigma_{-i}^*) \forall \sigma_i \in \text{probability distribution over } S_i$$

Nash's Theorem 1.1. *Every bimatrix game has a (pure or mixed) Nash equilibrium.*

Theorem 1.2. *For zero-sum games, it is polytime; otherwise, it is PPAD-hard.*

PPAD-hard $\overset{?}{\Rightarrow}$ NP-Hard $\Longleftrightarrow$ NP $\overset{?}{=}$ co-NP. This is open and unknown.

1.5. Sealed Bids Auctions

There is one seller and n buyers and one item that the seller wants to sell.

Buyers have non-negative valuations over this item v_i.

The rules of this auction are that each buyer submits a sealed bid b_i be the bid amounts. The highest bid gets the item (in case of ties, by the lower index).

If buyer i gets the item, and pays t_i, then their $\mu_i = v_i - t_i$ and payoff for other buyers is 0.

Definition: First Price Auctions

In first price auction, $t_i = b_i$.

Definition: Second Price Auction

Buyer who gets the items (say i) pays the second highest bid

Theorem 1.3. *Second price auction is weakly revealing (that is it is a weakly dominant strategy to bid truthfully).*

Proof. In such an auction setting (regret less, envy less, fee less auctions), our payoff is zero unless we win.

What agents are hence trying to avoid is the so called ‘winner’s curse. Hence, we can restrict the analysis to bidding as if we know we will win.

In that case, bidding your value is weakly dominant as if everyone is bidding their value; you can’t do better by

- bidding lower than your value, $\tilde{v} < v$, as another agent may have valuation v^* such that $\tilde{v} < v^* < v$ and will now win instead of you.
- bidding above your valuation, $\tilde{v} > v$, as another agent may have valuation v^* such that $\tilde{v} > v^* > v$ and you will win instead of them; However, with the winners curse.

■

Side Note

This is the only weakly dominant equilibrium. Although, this is by far not the only nash equilibrium. $(v_1, 0, ..., 0)$ is a NE; $(v_2, v_1, 0, ..., 0)$ is a NE; and so on.

This is called a Bayesian Nash Equilibrium as given our beliefs of the nature of the world, this maximizes our payoff. (Here the belief is that all valuations (in some radius around ours) are likely and that we may win).

1.6. Battle of the Sexes

Definition

We represent a situation where two agents must simultaneously take an action where each of them prefers one option over the other but prefer coordination over doing different things. And example occurrence would be[2]:

		Wife	
		Cricket	Music
Husband	Cricket	2,1	0,0
	Music	0,0	1,2

We can try to brute force this. Let’s look at the payoffs for both the players

$$\begin{aligned}
\mu_1(\sigma_1, \sigma_2) &= \sigma_1(C)\sigma_2(C)\mu_i(C, C) + \\
&\sigma_1(C)\sigma_2(D)\mu_i(C, D) + \\
&\sigma_1(D)\sigma_2(C)\mu_i(D, C) + \\
&\sigma_1(D)\sigma_2(D)\mu_i(D, D) \\
&= 2\sigma_1(C)\sigma_2(C) + \sigma_1(D)\sigma_2(D) \\
&= 2\sigma_1(C)\sigma_2(C) + (1 - \sigma_1(C))(1 - \sigma_2(C)) \\
&= 3\sigma_1(C)\sigma_2(C) - \sigma_1(C) - \sigma_2(C) + 1
\end{aligned}$$

[2] Which is extremely stereotypical and doesn’t represent the author and hopefully the prof’s views.

And similarly

$$\mu_2 = 3\sigma_1(C)\sigma_2(C) - 2\sigma_1(C) - 2\sigma_2(C) + 2$$

Using $\mu_1(\sigma_1^*, \sigma_2^*) \geq \mu_1(\sigma_1, \sigma_2 *)\forall\sigma_1 \in \Delta(S_1)$

$$\sigma_1^*(C)[3\sigma_2(C) - 1] \geq \sigma_1(1)[3\sigma_2(C) - 1]\forall\sigma_1(c) \in (0, 1)$$

Similarly

$$\sigma_2^*(C)[3\sigma_1(C) - 2] \geq \sigma_2(1)[3\sigma_1(C) - 2]\forall\sigma_2(c) \in (0, 1)$$

If $3\sigma_2^*(C) - 1 > 0 \Rightarrow \langle(1, 0), (0, 1)\rangle$.

If $3\sigma_2^*(C) - 1 < 0 \Rightarrow \langle(0, 1), (1, 0)\rangle$.

If $3\sigma_2^*(C) - 1 = 0 \Rightarrow \langle(\frac{2}{3}, \frac{1}{3}), (\frac{1}{3}, \frac{2}{3})\rangle$.

1.7. Properties of MSNE

Expected payoff for player i will be:

$$\begin{aligned}\mu_i(p_1, p_2, ..., p_n) &= \sum_{(s_1,...,s_n)\in S_1\times...\times S_n} \mu_i(s_i, s_{-i}) \prod_{a=1}^{n} p_a(s_a) \\ &= \sum_{s_i\in S_i} \sum_{s_{-i}\in S_{-i}} p_i(s_i)p_{-i}(s_{-i})\mu_i(s_i, s_{-i}) \\ &= \sum_{s_i\in S_i} p_i(s_i) \sum_{s_{-i}\in S_{-i}} p_{-i}(s_{-i})\mu_i(s_i, s_{-i}) \\ &= \sum_{s_i\in S_i} p_i(s_i)\mu_i(s_i, p_{-i})\end{aligned}$$

This makes the utility of a player a convex combination of their pure strategy payoffs.

Definition: Convex Combination

A convex combination of $a_1, a_2, ..., a_n$ is $\sum_{i=1}^{n} \lambda_i a_i$ where $\lambda_i \in [0, 1]$ and $\sum_{i=1}^{n} \lambda_i = 1$.

An obvious observation is that convex combination of $a_1, ..., a_n \leq \max\{a_i\}$.

Which implies that the payoff with mixed strategy is less than equal to max payoff with a pure strategy s_i. This implies:

$$\max_{\sigma_i\in\Delta(S_i)} \mu_i(\sigma_i, \sigma_{-i}) = \max_{s_i\in S_i} \mu_i(s_i, \sigma_{-i})$$

Theorem 1.4. *$(\sigma_1^*, ..., \sigma_n^*)$ is a MSNE if and only if $\forall i \in N$*

1. $\mu_i(s_i, \sigma_{-i}^*)$ is the same for all $s_i \in$ support of σ_i^*.

2. $\mu_i(s_i, \sigma_{-i}^*) \geq \mu_i(s_i^*, \sigma_{-i}^*)\forall s_i' \notin$ support of $\sigma_i^*, s_i \in$ support of σ_i^*.

Remark: Implications

Each player gets the same payoff for any pure strategy in the support of the MSNE strategy.

2. Equilibrium Computation

2.1. Computing a SDSE

Example

			P_2	
		D	E	F
	A	4,3	5,1	6,2
P_1	B	2,1	8,4	3,6
	C	3,0	9,6	2,8

Notice, E is dominated by F as $\forall i, \mu_2(F, s_{-i}) > \mu_2(E, s_{-i})$. THis practically makes the game:

		P_2	
		D	F
	A	4,3	6,2
P_1	B	2,1	3,6
	C	3,0	2,8

Now A dominates B, C and after elimination, D will dominate E and we will be left with the nash equilibrium of $4, 3$.

Also note, (A, D) is a PSNE.

Note, this algorithm by no means gives all the nash equilibrium, if there are multiple. We don't run into such issues with SDSE as only one can exist by the definition of Strict dominance.

Example

			P_2	
		D	E	F
	A	3,1	0,1	0,0
P_1	B	0,1	4,1	0,0
	C	1,1	1,1	5,0

We can eliminate F as D, E are strictly better.

		P_2	
		D	E
	A	3,1	0,1
P_1	B	0,1	4,1
	C	1,1	1,1

Here, we can eliminate C as if P_2 plays D or E, we are better off playing A or B respectively.

Another way to argue the same is $\mu_1\left(\frac{A+B}{2}, s_{-i}\right) > \mu_1(C, s_{-i})$.

Side Note

Here, playing C is called the Bayesian Equilibrium as it is safe.

Example

			P_2	
		X	Y	Z
	A	7,7	4,2	1,8
P_1	B	2,4	5,5	2,3
	C	8,1	3,2	0,0

We can get the guarantee finding the nash equilibrium by identifying the highest entry per row and column wrt the respective players.

			P_2	
		X	Y	Z
	A	$\underline{7}$,7	4,2	1,$\overline{8}$
P_1	B	2,4	$\underline{5}, \overline{5}$	2,3
	C	$\underline{8}$,1	3,$\overline{2}$	0,0

Making $(5, 5)$ the PSNE.

Algorithm 2.1. For the column player, mark the maximum entry for them.

For the row player, mark the maximum entry for them.

The intersections are PSNEs.

2.2. Two Player Zero Sum Games

Definition: Two Player Zero Sum Games

A game with:

- $N = \{1, 2\}$
- $S = \{S_1, S_2\}$
- $\mu = \{\mu_1, \mu_2\}$
- $\mu_1(s_1, s_2) = -\mu_2(s_1, s_2)$

In these games, we only need to specify the payoffs for one player. We, by convention, do this for the row player.

$$
\begin{array}{c|c|c|c|}
\multicolumn{1}{c}{} & \multicolumn{3}{c}{B} \\
\cline{2-4}
 & 2 & 1 & 1 \\
\cline{2-4}
A & -1 & 1 & 2 \\
\cline{2-4}
 & 1 & 0 & 1 \\
\cline{2-4}
\end{array}
$$

For any strategy, i of P_1 and P_2 will choose a strategy such that,

- P_1 chooses $\max_i \min_j a_{ij}$ (maximin)
- P_2 will choose $\min_j \max_i a_{ij}$. (minimax)

If maximin and minimax are equal, we are done and the value is the nash equilibrium.

Definition: Maximin-Minimax

The maximin value refers to $\max_{i \in S_1} \min_{j \in S_2} a_{ij}$.

The minimax value refers to $\min_{j \in S_2} \max_{i \in S_1} a_{ij}$.

If a PSNE exists, we will get it by this process.

If there is no PSNE, this obviously doesn't work.

$$
\begin{array}{cc|c|c|c|}
\multicolumn{2}{c}{} & \multicolumn{3}{c}{\mathbf{P2}} \\
\multicolumn{2}{c}{} & \multicolumn{1}{c}{R} & \multicolumn{1}{c}{P} & \multicolumn{1}{c}{S} \\
\cline{3-5}
 & R & 0 & -1 & 1 \\
\cline{3-5}
\mathbf{P1} & P & 1 & 0 & -1 \\
\cline{3-5}
 & S & -1 & 1 & 0 \\
\cline{3-5}
\end{array}
$$

We can also discuss Saddle points.

Definition: Saddle Point

(i, j) is a saddle point of a matrix A if $a_{ij} \geq a_{kj} \forall k$ and $a_{ij} \leq a_{il} \forall l$

Theorem 2.2. *If i, j and k, h are saddle points, then (i, h) and k, j are also saddle points.*

Proof.

$$a_{ij} \geq a_{kj} \geq a_{kh} \geq a_{ih} \geq a_{ij}$$

And we are done by squeeze theorem. ■

Definition: Mixed Strategy in 2 player zero sum game

Let $|S_1| = m$ and $|S_2| = m$. Let $x = (x_1, x_2, ..., x_n)$ be a mixed strategy for P_1 and $y = (y_1, y_2, ..., y_n)$ be a mixed strategy for P_2.

The expected payoff is $\sum_{i=1}^{n} \sum_{j=1}^{m} x_i y_j a_{ij} = x^T Ay$

We can define maximin and minimax values here as:

Definition: Maximin-minimax

maximin value $= \max_{x \in \Delta(S_1)} \min_{y \in \Delta(S_2)} x^T Ay$

minimax value $= \min_{y \in \Delta(S_2)} \max_{x \in \Delta(S_1)} x^T Ay$

Lemma 2.3.

$$\min_{y \in \Delta(S_2)} x^T Ay = \min_{j \in S_2} \sum_{i=1}^{m} a_{ij} x_i$$

Proof. It is obvious that

$$\min_{y \in \Delta(S_2)} x^T Ay \leq \min_{j \in S_2} \sum_{i=1}^{m} a_{ij} x_i$$

as $S_2 \subseteq \Delta(S_2)$.

To do the other way round,

$$x^T Ay = \sum_{j=1}^{n} y_j \sum_{i=1}^{m} x_j a_{ij}$$

$$\geq \sum_{i=1}^{n} y_j \min_{k \in S_2} \sum_{i=1}^{m} x_i a_{ik}$$

$$= \min_{k \in S_2} \sum_{i=1}^{m} x_I a_{ik} \cdot \sum_{i=1}^{n} y_j$$

$$= \min_{k \in S_2} \sum_{i=1}^{m} x_I a_{ik}$$

Thus, by squeeze theorem, we are done. ■

2.3. Linear Programming for the the win

We make an LP for the row player as:

The row player has to to maximize $\min_{j \in S_2} \sum_{i=1}^{m} a_{ij} x_i$ with the constrains $\sum_{i=1}^{m} x_i = 1, x_i \geq 0 \forall i \in \{1, 2, ..., m\}$.

Equivalently, maximize z with the constrains $z \leq \sum_{i=1}^{m} a_{ij} x_i \forall j \in \{1, 2, ..., n\}$, $\sum_{i=1}^{n} x_i = 1$, $x_i \geq 0 \forall i \in [m]$.

Exercise : Column Player's LP

Define the column players LP.

Solution. minimize w such that $w \geq \sum_{j=1}^{n} a_{ij} y_j \forall i \in [m]$, $\sum_{j=1}^{n} y_j = 1$, $y_j \geq 0$ forall $j \in [n]$. ■

2.3.1. Duel of LP

Example : Toy LP

max $3x_1 + 2x_2 + x_3$ with constraints

1. $x_1 + x_2 + x_3 \leq 2$
2. $3x_1 + x_3 \leq 4$
3. $x_1 + x_2 + x_3 \leq 5$
4. $x_1, x_2, x_3 \geq 0$

We can get an upper bound by $3 * \text{eq } 3 \Rightarrow 3x_1 + 3x_2 + 3x_3 \leq 15$.

We can get a better upperbound by $\text{eq } 1 + \text{eq } 2 \Rightarrow 3x_1 + 3x_2 + 3x_3 \leq 9$.

The idea is that as atleast one coefficient is bigger, the sum is bigger and is an upperbound.

But doing this for all combinations is going to be painfully time consuming. Let's try to do this algebraically and let the coefficients of equation 1,2,3 be y_1, y_2, y_3 in the linear combination.

$$x_1(y_1 + 3y_2 + y_3) + (y_1 + y_3)x_2 + (y_2 + y_3)x_3 \leq 2y_1 + 4y_2 + 5y_3$$

This leads to a LP

Minimize $2y_1 + 4y_2 + 5y_3$ with the constraints:

- $y_1 + 3y_2 + y_3 \geq 3$
- $y_1 + y_3 \geq 2$
- $y_2 + y_3 \geq 1$

Theorem 2.4. *The Row player LP and Column player LP are duel pairs.*

Proof. DIY. It's just transposing stuff here and there. ■

2.4. Strong Duality

Optimum value of f is the optimal value of its primal duel.

Ler $x^* = (x_1^*, x_2^*, x_3^*, ..., x_m^*), z^*$ be the optimum values for Row player.

z^* must attain equality at some $j^* \in [n]$ that is $z^* = \sum_i a_{ij^*} x_i^*$ and $z^* \leq \sum_i a_{ij} x_i^* \forall j \in [n]$.

By lemma, $z^* = \min_{j \in S_2} \sum_{i=1}^n x_i^* = \min_{y \in \Delta(S_2)} {x^*}^T Ay$ and similarly, $w^* = \max \sum_{j=1}^n a_{ij^*} y_i = \max_{x \in \Delta(S_2)} x^T A y^*$.

Thus,

$$\min_{y \in \Delta(S_2)} {x^*}^T Ay = \max_{y \in \Delta(S_2)} x^T A y^*$$

This is a nash equilibrium as no player can unilaterally increase payoff by moving.

2.5. Existence of Nash Equilibrium (mixed) in finite strategic form games

Ler $(N, < s_i >, < \mu_i >)$ be the game where $N = \{1, 2, ..., n\}$, S_i is strategy set for player i, with $|S_i| = m \forall i$ and and $\mu_i : S_i \times S_{-i} \to \mathbb{R}$ is the payoff function for player i.

Mixed strategy σ_i is a probability distribution over S_i.

Δ_i of $\Delta(S_i)$ denotes the set of all mixed strategies for player i. $\Delta = \Delta_1 \times \Delta_2 \times ... \times \Delta_m$.

Expected payoff from δ_i for player i is

$$\mu_i(\sigma_i) \triangleq \sum_{(s_1, ..., s_n) \in S_1 \times S_2 ... S_n} \left(\prod_{j=1}^n \sigma_j(s_j) \right) \mu_i(s_1, ..., s_n)$$

Definition

σ^* is a Nash Equilibrium if for every player i,

$$\mu_i(\sigma^*) \geq \mu_i(s_i, \sigma_{-i}^*) \forall s_i \in S_i$$

Brouwer's Fixed Point Theorem 2.5. *Let B be a closed, bounded convex set. Let $f :: B \to B$ be a continuous function, then $\exists x \in B$* s.t. $f(x) = x$

We want to define B, f such that the Nash Equilibrium is the fixed point of f. Define $B = \Delta$.

Define f such that $\sigma \in \Delta$ is not a NE then $f(\sigma) \neq \sigma$ NS IF σ^* is a NE then $f(\sigma^*) = \sigma^*$.

Attempt 1: Define $f(\sigma) = \rho$ where $\rho \in \Delta, \rho = (\rho_1, \rho_2, ..., \rho_n)$ such that

$$\rho_i = \arg \max_{\delta_i \in \Delta_i} \mu_i(\sigma_{-i}, \delta_i)$$

Recall, this is called the best response.

The problem is, this is not a function and if we apply a tie breaking rule, it is not continuous. We could use the powerset as the domain, but that would require Katakumi's fixed point which is more analytical than we wish to be.

Example : Matching Pennies

		P_2	
		H	T
P_1	H	1, −1	−1, 1
	T	−1, 1	1, −1

Let row player choose the mixed strategy $(p, 1-p)$ and column player choose $(q, 1-q)$. The problem is

$$p > \frac{1}{2} \Rightarrow q = 0$$
$$p < \frac{1}{2} \Rightarrow q = 1$$
$$p = \frac{1}{2} \Rightarrow 0 \leq q \leq 1$$

This is an counter example to the continuity of f.

Modifying f: We first define a gain function of player i for deviation to s_{ij} from σ_i.

$$g_{ij} = \max\{\mu_i(\sigma_{-i}, s_{ij}) - \mu_i(\sigma), 0\}$$

We now define

$$f_{ij}\sigma = \frac{\sigma_{ij} + g_{ij}(\sigma)}{\sum_{k=1}^{m}(\sigma_{ik} + g_{ik}(\sigma))} = \frac{\sigma_{ij} + g_{ij}(\sigma)}{1 + \sum_{k=1}^{m} g_{ik}(\sigma)}$$

We want

$$\sum_{k=1}^{m} f_{ik}(\sigma) = 1$$

We will define

$$\begin{aligned} f(\sigma) &:= \rho \\ f(\sigma) &= f(\sigma_{11}, ..., \sigma_{1m}, ..., \sigma_{n1}, ...\sigma_{nm}) \\ &= (\rho_1, ..., \rho_{1m}, ..., \rho_{n1}, ..., \rho_{nm}). \end{aligned}$$

Let σ^* be a fixed point of f.

$$f_{ij} = \sigma^*_{ij}$$
$$\sigma^*_{ij} = \frac{\sigma_{ij} + g_{ij}(\sigma)}{1 + \sum_{k=1}^{m} g_{ik}(\sigma)}$$
$$\Longleftrightarrow g_{ij}(\sigma^*) = 0 \forall j$$

TODO. *Will complete from Solan...*

Thus, by Brower's, we are done.

2.6. Two Player Non-Zero Sum Games

Definition

Given players $1, 2$ and strategy sets S_1, S_2 and payoff matrices R for player 1 and C for player 2. A mixed strategy (x^*, y^*) is Nash Equilibrium if and only if

$$x^{*^T} R y^* \geq (R y^*)_i \forall i \in [n]$$
$$x^{*^T} C y^* \geq \left(x^{*^T} R\right) \forall i \in [n]$$

where the LHS are expected payoffs. Also (M_i) is the i^{th} position in the matrix. We could also write this as $e_i^T R y^*$ and $x^{*^T} R e_i$ respectively.

Notice, if one switches say R for $R' = R + a$ that is

$$\begin{bmatrix} r_{1,1} + a & \dots & r_{1,m} + a \\ r_{2,1} + a & \dots & r_{2,m} + a \\ \vdots & \vdots & \vdots \\ r_{n,1} + a & \dots & r_{n,m} + a \end{bmatrix}$$

Claim 2.6. Set of NE for (R', C) are same as that of (R, C). This implies, by symmetry, that $(R, C + b)$ has same NEs as (R, C).

Claim 2.7. Set of NE remain unchanged for $\left(\frac{1}{a} R, C\right)$ wrt (R, C) and similarly the set of NE remains unchanged for $\left(R, \frac{1}{b} C\right)$ wrt (R, C).

This allows us to only deal with $R_{i,j}, C_{i,j} \in [0, 1]$.

2.6.1. NE by Support enumeration

Let $S \subseteq [n]$, $T \subseteq [m]$ be set of strategies in the support of a NE that is $x_i > 0 \quad \forall i \in S$, $y_i > 0 \quad \forall i \in T$.

We will write a LP for this, named **LP[S,T]**.

$$\text{probability constraints}\begin{cases} x_i \geq 0 \quad \forall i \in [n] \\ y_i \geq 0 \quad \forall i \in [n] \\ \sum_{i=1}^{n} x_i = 1 \\ \sum_{i=1}^{m} y_i = 1 \end{cases}$$

$$\text{NE Constraints}\begin{cases} (Ry)_i \geq (Ry)_j & \forall i \in S, j \in [n] \\ \left(x^T C\right)_i \geq \left(X^T C\right)_j & \forall i \in T, j \in [m] \end{cases}$$

Claim 2.8. A solution to the LP say (x^*, y^*) is a NE.

Proof. If x^*, y^* is not a NE then one of the players has an incentive to deviate to another pure strategy, which violates the NE constraints ■

NE algorithm by support enumeration

```
for each $S \subseteq [n], T \subseteq [m]$:
    if **LP[S,T]** has a feasible solution $(x, y)$
        return $(x, y)$
```

This clearly has a $2^{n+m}\operatorname{poly}(n+m)$ running time.

This is clearly not polytime. As we shall see, that the solution lies in complexity class PPAD and is hence, believed not to be possible in polytime.

2.6.2. Lemke-Howson Algorithm

Characterization of NE for this algorithm is using the translation and scaling transforms and using the fact that the max achievable payoff is always 1 and the payoffs are ≥ 0. We say $\left(\frac{x^*}{\sum^n x_i^*}, \frac{y^*}{\sum^m y_i^*}\right)$ is a NE if:

$$\begin{aligned} x_i^* = 0 \quad &\text{or} \quad (Ry^*)_i = 1 \quad \forall i \in [n] \\ y_i^* = 0 \quad &\text{or} \quad \left(x^{*T} C\right)_i = 1 \quad \forall i \in [m] \end{aligned}$$

The idea is that instead of the payoff's being variables, we can sort of normalize them to 1 and then make a system of linear equations using it. This will

Definition: Polytope, Polyhedron, Half-Space, Vertex

A bounded **polyhedron** is **polytope**.

Intersection of **half-spaces** is called a **polyhedron**.

Given a n dimensional plane, the plane can be split into two parts by a $n-1$ dimensional hyper-plane. This is called **half-space**. For example $ax + b > 0$ defines a half-space in 2D while $a_1x_1 + a_2x_2 + a_3x_3 \geq 0$ defines a half-space in 3D.

Vertex/Extreme Points in $\mathbb{R}^k$ is the intersections of k linearly independent hyper-planes.

Consider the polytope:

$$P = \left\{(x, y) \mid x_i \underset{i \in [n]}{\geq} 0, y_i \underset{i \in [m]}{\geq} 0, (Ry)_i \leq 1, \left(x^T C\right)_i \leq 1\right\}$$

We are assuming non-degeneracy here.

Definition: Non-Degeneracy

Any set of $> m + n$ constraints do not meet at one point.

While the polytope itself doesn't define nash equilibria but some of it's vertex do. We will hop from vertex to vertex up to some condition to get nash equilibria.

At Nash equilibrium, by the stability condition, n of the equations in

$\left\{x_i \underset{i \in [n]}{\geq} 0, (Ry)_i \leq 1\right\}$

and m of the equations in

$\left\{y_i \underset{i \in [m]}{\geq} 0, \left(x^T C\right)_i \leq 1\right\}$.

Thus, $m + n$ many equalities the NE must have. That implies NE is a vertex of P.

The vertex $x, y = \left(\vec{0}_n, \vec{0}_n\right)$ is not a NE. This is called an artificial equilibrium.

A vertex $x > 0$ $Ry_i < 1$ but $x_2 = 0, R_2 y = 1, \ldots$ is not a NE. Strategy 1 has +ve prov but not max payoff.

Let $(\hat{x}, \hat{y})$ be a vertex. Define a set of labels for each vertex. Define a set of labels for each vertex $(\hat{x}, \hat{y})$ has a label $i \in [n]$ if either $x_i = 0$ or $R_i y = 1$. Also $(\hat{x}, \hat{y})$ has a label $n + j$ if $y_2 = 0$ or $x^T C^{(j)} = 1$.

Claim 2.9. $(\tilde{x}, \tilde{y})$ is a NE if and only if H has all $m + n$ labels and $(\tilde{x}, \tilde{y}) \neq \left(\vec{0}_n, \vec{0}_m\right)$ Under the non-degeneracy assumption.

Goal To find $(\tilde{x}, \tilde{y}) \neq \left(\vec{0}_n, \vec{0}_m\right)$ and has all the labels.

So let's start at $(x_0 y_0) = \left(\vec{0}_n, \vec{0}_m\right)$.

Fix label J, relax the constraint $x_1 = 0$; $x_2 = \ldots = x_n = 0, y_1 = \ldots = y_n = 0 x_1 > 0$.

This new vertex, say (x_1, y_1) doesn't have label 1, but has a duplicate label $\in \{n + 1, \ldots, n + m\}$, say $k = 1$.

We relax the corresponding to the old duplicate label and proceed.

At any point, say t if (x_t, y_t) have all the labels, output it as a NE.

Claim 2.10. For any $i \neq j$, $(x_i, y_i) \neq (x_j, y_j)$

Proof. (x_t, y_t) has all labels and (x_0, y_0) is $(0, 0)$.

All intermediate vertices (x_i, y_i) with duplicate labels k_i have only two neighbors, corresponding to the duplicates. With only two ways of coming, there are only two ways of going, ensuring that we can never repeat vertices. ■

Corollary 2.11. *Every game has odd number of nash equilibrium.*

Corollary 2.12. *The odds in a mixed nash equilibrium are rational.*

2.7. Mixed Nash Equilibrium in 2 Player Games

$N = \{1, 2\}, S_1 = [n], S_2 = [m]$ with payoff matrices R, C.

Characterization of MNE is

$$\begin{gathered} P : \text{set of points } (x, y) \in \mathbb{R}^{m+n} \text{ s.t.} \\ x \geq 0 \forall i \in [n] \\ R_i y \leq 1 \forall i \in [n] \\ y_j \geq 0 \forall j \in [m] \\ x^T C_j \geq 0 \forall j \in [m] \end{gathered}$$

$(x, y) \in P$ is a MNE if and only if

$$\begin{gathered} \forall i \in [n], \text{either } x_i = 0 \text{ or } R y_i = 1 : \text{Label } i \text{ cases} \\ \forall j \in [n], \text{either } x_j = 0 \text{ or } x^T C_j = 1 : \text{Label } n + j \text{ cases} \end{gathered}$$

(x, y) has all labels in $[m + n]$.

Non-Degeneracy assumption ⇒ exactly $m + n$ labels for any vertex of P.

Note $(0, 0)$ has all labels but is not a NE.

Lemke-Hawson Algorithm

```
Start at $(0, 0)$, fix label 1.
Relax the constraint $\equiv$ label 1 (ie $x = 0$)
Goto next vertex $\left(x^{(1)}, y^{(1)}\right)$
    Either all labels exist ⇒ MNE
Or a duplicate label $k$ exists
    Relax the "old" constraint $\equiv$ label $k$
Goto $\left(x^{(2)}, y^{(2)}\right)$
so on…
Let $\left(x^{(0)}, y^{(0)}\right), …, \left(x^{(t)}, y^{(t)}\right)$ be the visited vertices.
```

```
Lemke-Hawson Algorithm
10  All $(x^{(i)}, y^{(i)}), 0 < i < t$ miss label 1.
```

This algorithm is clearly exponential time as our polytope may have exponential vertices.

The edges we visit form a graph, specifically a path where the first and last vertex have degree 1 and the other vertices have degree 2.

2.8. Complexity of algorithm

As with most problems in computer science, we want to know how hard it is. After all after being saddened by not getting a polytime algorithm, we want to show it is unlikely that anyone else will[3].

The problem in showing that it is NP lies in the definition of NP itself.

Definition: NP

NP is a class of decision problems with yes/no answer and there exists a polytime verifiable certificate for a yes answer.

Our problem is not a decision problem. So what do we do?

Definition: Functional NP (FNP)

If there is a certificate (solution), output one.

Note, the certificate should be polytime verifiable.

Clearly, MNE $\in$ FNP. So can we show MNE is FNP-Complete.

Theorem 2.13. *If MNE is FNP-Complete, NP = co-NP*

Side Note

It is believed that NP $\neq$ co-NP, not as strongly as P $\neq$ NP but strongly enough. It is still open nonetheless.

Example

Take your favorite NP-complete problem. The instructor took Hamiltonian Path.

Let's there be a reduction from Hamiltonian Path to MNE.

[3] Unless P = NP which is still unlikely.

Input will be a graph G in which we want to compute the Hamiltonian path.

Reduction will be something that takes a graph G and converts it into an instance of 2-Player Game Γ with the property:

G has a Hamiltonian path $\iff \Gamma$ has a MNE which maps back to “yes”

G has no Hamiltonian path $\iff \Gamma$ has a MNE which maps back to “no”

Let’s take a slightly less ambitious goal.

Definition: TFNP

Total FNP is the class of FNP problems that always have a solution.

This still doesn’t help us out as there are no known TFNP problems.

So we go down to PPAD, a class contained in TFNP.

Definition: PPAD

Polynomial Parity Argument Directed version is a complexity class defined by a canonical problem called the **end of line** problem.

Definition: End of Line

Given G possibly exponential sized graph with polytime algorithm (or circuit) to determine neighbors of a given vertex.

Every vertex has in-degree ≤ 1 and out-degree ≤ 1.

Q: Given a source, find a sink (any sink, not the one corresponding to the source!)

Theorem 2.14. *NME is PPAD complete*

High Level Proof. NME is in PPAD by the Lemke-Hawson algorithm. We do this reduction in 2 steps.
1. Reduce End of Line to (approx) Brower’s Fixed Point.
2. Reduce approx Brower Fixed Point to approx MNE.

We will do the first part by an intermediate step called **Sperner Lemma**.

Sperner's Lemma 2.15. *Given a lattice as an input with the lattice points colored in three colors with every boundary being forbidden to use one color. The intermediate vertices can have any color (from the 3).*

We define a triangle as 3 points in the same cell. That is a $1, 1, \sqrt{2}$ *right angle triangle where lengths are unit.*

Sperner's Lemma states that there exists a triangle with all vertices having distinct colors.

Definition: Sperner's Lemma Problem

Given a lattice as an input with the lattice points colored in three colors with every boundary being forbidden to use one color. The intermediate vertices can have any color (from the 3).

We define a triangle as 3 points in the same cell. That is a $1, 1, \sqrt{2}$ right angle triangle where lengths are unit.

Find: a triangle with all vertices having distinct colors.

We can solve Brower Fixed Point on $f : [0, 1]^2 \to [0, 1]^2$, we want x s.t. $f(x) = x$.

We can just declare $0^\circ - 120^\circ$ as red, $120^\circ - 240^\circ$ as yellow and $240^\circ - 360^\circ$ as blue and color everything by this. We will use Sperner to find the approximate fixed point.

So our flow chart is

$$\text{End of Line} \to \text{Sperner Problem} \to \text{Approx Brower Fixed Point}$$
$$\to \text{Arithmetic Circuit} \to \text{Game} \to \text{MNE}$$

■

2.9. Alternatives to Nash Equilibrium

Some issues with Nash Equilibria are:

- It is hard to compute (unless PPAD $\subseteq$ P, which is unlikely)
- Many Nash Equilibria may exist, it is hence difficult to predict players behavior.
- Payoff at Nash Equilibrium may be much smaller than optimum (cost of anarchy)

Remark

For example, in Prisoner's Dilemma:

		P2	
		C	D
P1	C	$-5, -5$	$-1, -10$
	D	$-10, -1$	$-2, -2$

The optimal cases is $-2, -2$ but the Nash equilibrium is $-5, -5$ which makes the cost of anarchy 2.5 and is not desirable from players point of view.

Problem 2.16. Can we circumvent the hardness of Nash Equilibrium by computing say the ε-Nash Equilibrium?

Definition: ε-Nash Equilibrium

$(\sigma_1^*, ..., \sigma_n^*)$ is a ε −NE if

$$\forall i \in [n] \quad \mu_i(\sigma_i^*, \sigma_{-i}^*) \geq \mu_i(\sigma_i', \sigma_{-i}^*) - \varepsilon \forall \sigma_i' \in \Delta_i$$

Unfortunately, this is also PPAD complete for all ε as our proof for PPAD-hardness for NE uses approximate Brower's fixed point and more approximation won't really allow for any ease in computation.

Theorem 2.17. *Around each NE, there are ε-NE but converse may not hold as ∃ games where an ε-NE is 'far' from any NE.*

2.9.1. Correlated Equilibria

Example

		P2	
		C	M
P1	C	$2, 1$	$0, 0$
	M	$0, 0$	$1, 2$

The NE are clearly C and M pure and $\left(\left(\frac{2}{3}, \frac{1}{3}\right)\left(\frac{1}{3}, \frac{2}{3}\right)\right)$ mixed. The expected payoffs of the equilibrium are:

$$(2, 1), (1, 2), \left(\frac{2}{3}, \frac{2}{3}\right)$$

respectively where the mixed equilibrium is worse for both as they are assigning some probability to the undesirable CM, MC options.

These are called Correlated equilibria.

Definition: Corelate Equilibrium

Algorithm 2.18. . Given n-players $[n]$, strategy sets $S_1, ..., S_n$.

- A coordinator declares a probability distribution on $S_1 \times ...S_n$.
- The coordinator 'privately' samples the joint distribution for a strategy combination.
- The coordinator tells player i to play s_i

Note: The player may choose to or not to heed the advice.

A distribution D on $S_1 \times ... \times S_n$ is a CE if $\forall i, \forall b_i, b_i' \in S_i$

$$\mathbb{E}_{s\sim D}[\mu_i(b_i, s_{-i}) \mid s_i = b_i] \geq \mathbb{E}_{s\sim D}[\mu_i(b_i', s_{-i}) \mid s_i = b_i]$$

Basically, the player is better off heeding the advice of the coordinator given everyone else is heeding the advice of the coordinator.

Example : Chicken

		P2	
		D	C
P1	D	$0, 0$	$5, 1$
	C	$1, 5$	$4, 4$

This game comes from a dumb things teenagers did 'back in the day'[4] where they would drive towards each other at some speed and the first person to hit the breaks (Chicken out) would lose. The issue is if two hot headed (or stupid, albeit it is hard to tell the difference) drive into each other.

We can see the Nash Equilibria are $(A, B), (B, A), \left(\frac{1}{2}A, \frac{1}{2}B\right)$ whereas we can have a CE of $\frac{1}{3}(A, B), \frac{1}{3}(B, A), \frac{1}{3}(B, B)$.

As John Green once said: "Turtles all the way down!", we will say Equilibria all the way down.

$$\text{Dominant Equilibria} \subseteq \text{Weakly Dominant Equilibria} \subseteq \text{Nash Equilibria} \subseteq \text{Correlated Equilibria}$$

where the equilibria from Nash onwards are guaranteed to exist and Correlated is easy to compute.

Theorem 2.19. *CE form a convex set*

Algorithm 2.20. We can compute CE using LP:

$$\sum_j A_{ij}p_{ij} \geq \sum_j A_{i'j}p_{ij}$$

$$\sum_j B_{ij}p_{ij} \geq \sum_i B_{ij'}p_{ih}$$

$$\sum_{i,j} p_{ij} = 1$$

$$p_{ij} > 0$$

[4] Nowadays, teenagers have better (read safer) things to do in their time. Unfortunately, social media fame, friendship graphs, min-maxing for university/NEET-JEE/Olympiads etc are much harder to study.

3. Fair Division

Definition: Division

$\langle X_1, X_2, ..., X_n \rangle$ a division of $[0,1]$ if

$$X_i \subset [0,1],$$
$$X_i \cap X_j = \emptyset$$
$$\bigcup_{i=1}^{n} X_i = [0,1]$$

Definition: Envy Free

Given n agents and their valuation functions v_i, and we want to divde a cake $[0,1]$ then $\langle X_1, X_2, ..., X_n \rangle$ a EF division if

$$\forall i, j \in [n]; v_i(X_i) \geq v_i\left(X_j\right)$$

Problem 3.1. Envy Free Cake Division on 2 agents

Solution

This is rather easy. We ask one agent to cut and other to choose.

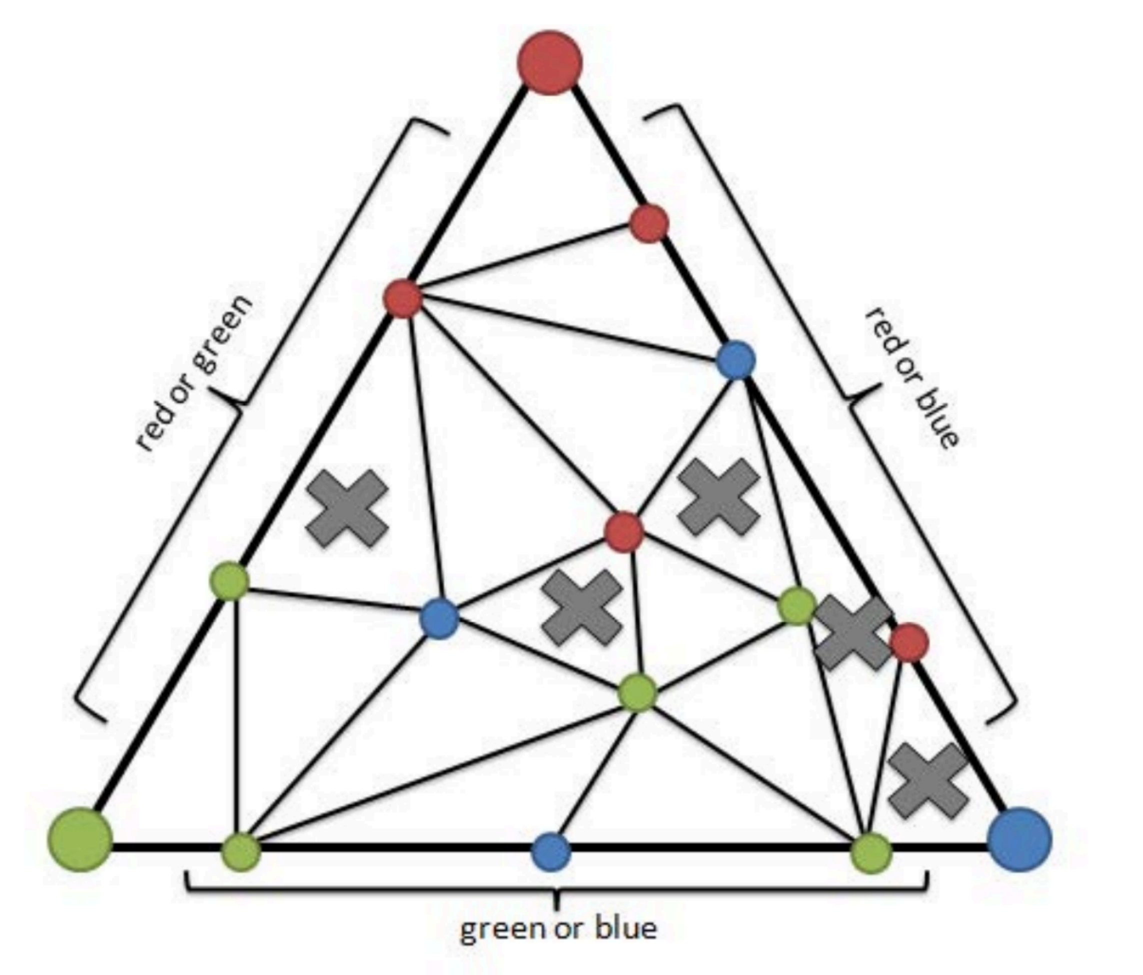
Another idea is that I keep moving the knife and when an agent is satisfied say at a, I cut the cake there and give them the piece $[0,a]$ as

Problem 3.2. Envy Free Cake Division with 3 agents

Problem 3.3. Given an apartment with n rooms and rent k, we want to divide the apartment between n roomates with valuation functions $v_i : [n] \times [0,k] \to [0,1]$.

3.1. Detour: Sperner’s Lemma

Definition: Sperner Coloring on Triangles



Given a triangle ABC divided into triangles, a coloring of vertices such that

- Each of the three vertices A, B, and C of the initial triangle has a distinct color.
- The vertices that lie along any edge of triangle ABC have only two colors, the two colors at the endpoints of the edge. For example, each vertex on AC must have the same color as A or C.

Sperner's Lemma in Triangles 3.4. *Given any Spenumberolored triangulation, there exists odd numver of fully colored elementary triangle (has all three colors.)*

Corollary 3.5. *Given any Sperner colored triangulation, there exists atleast one of fully colored elementary triangle.*

Definition: Simplex

A 0 simplex is just a point.

A 1 simplex is a line.

A 2 simplex is a triangle.

A 3 simplex is a tetrahedron.

So on and so fourth.

Definition: Facet

Choosing n vertices from a $n+1$-simplex is called a facet.

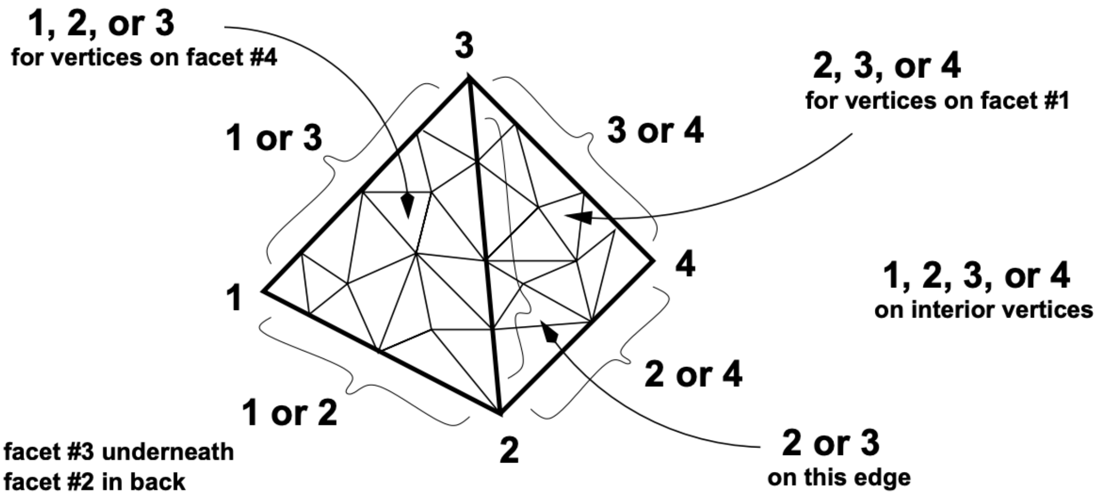


Figure 1: Sperner Tetrahedron

Definition: Sperner Coloring

Given a n simplex divided into n simplices, we call a cohering Sperner if:

- Each of the boundary vertices of the initial simplex has a distinct color.
- The vertices that lie along any facet i (named after the boundary vertex not in the facet) doesn't have color i.

Sperner's Lemma 3.6. *Any Facet of a sperner colored simplex is also sperner colored, also, number of rainbow elementary n-simplices is odd.*

Proof.

Definition: Door

Door is a facet of elementary simplex if it has labels $\langle 1 \ldots n \rangle$. For a 3 simplex (triangle), it is edges with vertices colored $(1, 2)$.

Lemma 3.7. *We have an odd number of doors per facet.*

We will now induct on n.

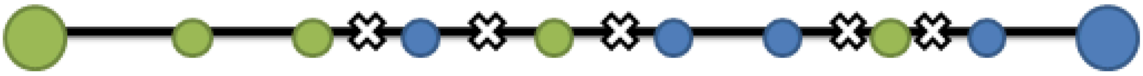


Figure 2: Proof for 1D case

(B) For 1 simplex, We are obviously done as $1 - 2$ rainbow will be there.

(S) Let Sperner hold for $n - 1$ simplex.

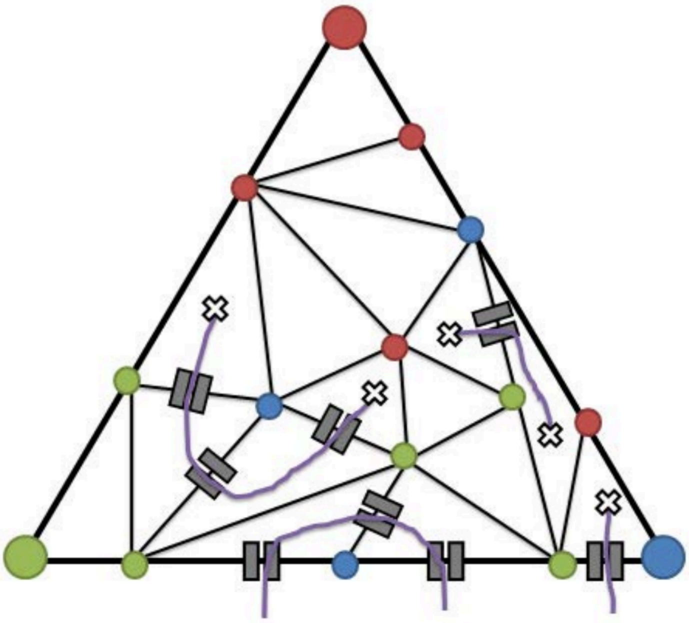


Figure 3: Visualization for the inductive step in 2D

For an n simplex, The n facet is $n - 1$ simplex and by induction, we will have odd number of doors via Sperner.

We can now enter via doors and lock them behind us. If we end up in a room, it is rainbow.

If we exit the building, we will close the door we left through and close it behind us.

If some rainbow room is unreachable, we claim that if we were to start there, we would end up walking to another unreachable rainbow room. Thus, the unreachable rooms come in pairs.

Thus, we have odd number of rainbow rooms, proving the Sperner's Lemma. ■

3.2. Back to Fair Division

3.2.1. Cake-Cutting Problem

We will solve a general version of Problem 3.2 also called the cake-cutting problem. For n agents, We consider any $n - 1$ vertical cuts parallel to the side of the cake, dividing the cake into n pieces. Such a way of cutting is called a cut-set.

Idea

Observe that in a particular cut-set, if we ask all players which part do they prefer and get all different answers, then that cut-set gives an envy-free division (because every one gets the piece he/she prefer the most). We will use Sperner's lemma to prove that there must exists such cut-set, as well as finding one.

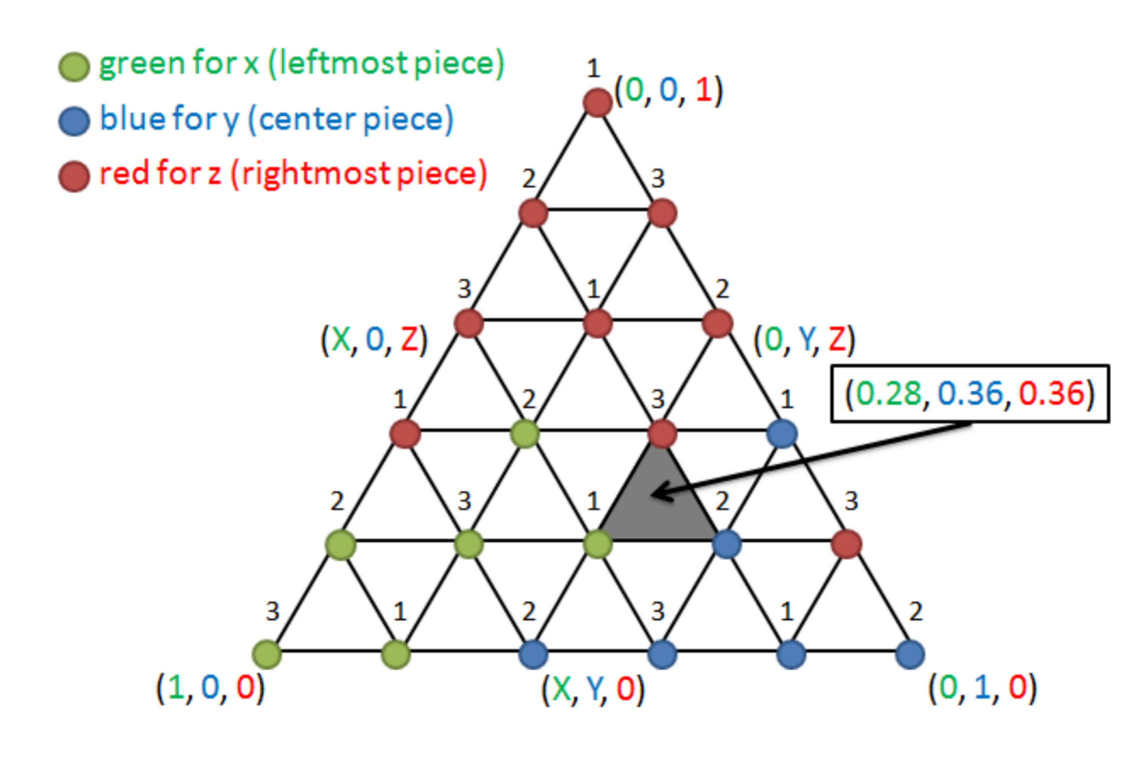
The idea is that Sperner allows us to guarantee that there is a triangle part with all vertices of different color. So given continuity aka small change in cut-set causes small change in preference; we can make a Sperner coloring of the linear programming simplex and get an approximate. Tha's exactly what the solution is.

Solution. Consider every possible cut-set that divides the cake into n (maybe empty) pieces. Let x_i be the proportion of ith leftmost part. We have $x_1 + x_2 + ... + x_n = 1$ and $x_i \geq 0$ for every $i = 1, 2, ..., n$.

In the n-dimensional space, consider a polytope formed by linear programming $x_1 + x_2 + ... + x_n = 1$ and $x_i \geq 0$ for every $i = 1, 2, ..., n$. The resulted polytope is a regular $(n - 1)$-simplex.

Then, we triangulate that simplex into smaller regular $(n - 1)$ simplices with each having side length less than ε, for a small enough ε, as well as writing numbers $1, 2, ..., n$ on the vertices in a way that every elementary $(n - 1)$-simplex has vertices with all different numbers.

Example



In that figure, the resulted polytope is a regular triangle (regular 2-simplex). We then divide it into k^2 smaller regular triangles, for some integer k as big as we want. Then, we write a number on each vertex $1, 2, 3$ in a cyclic order such that every elementary triangle has all three vertices with different numbers.

Then, at each vertex with coordinates $(x_1, x_2, ..., x_n)$ with number i, we ask player i that which piece of cake that he/she prefers if the sizes of pieces of cake is $x_1, x_2, ..., x_n$, respectively. We then color that vertex according to the answer.

Example

We ask player 1 that if the sizes of three pieces are $0, 0$, and 1, respectively, which piece does he/she prefer. We then color that vertex according to the answer.

Observe that, in each of the k-dimensional face of the polytope, one ore more of the coordinates must be zero.

Example

The second coordinate along the left edge of the triangle is always zero, hence, no one chooses that piece.

As, no people prefer the piece with size zero, so the color of vertices on each face must be the same as one of the corners of that face. Therefore, the color labeling of vertices in the simplex is a Sperner labeling.

By Sperners' lemma, there must be at least one elementary $(n-1)$ simplex that has vertices with different colors. We then divide the cake by the cut-set represented by any interior point of that $(n-1)$-simplex.

Since the size of each elementary simplex is less than ε, by the continuous preference assumption, all people will be satisfied with that cut-set. ■

The algorithm this solution leads to is named **Simmons' Algorithm** as it was developed by Forrest Simmons in 1980.

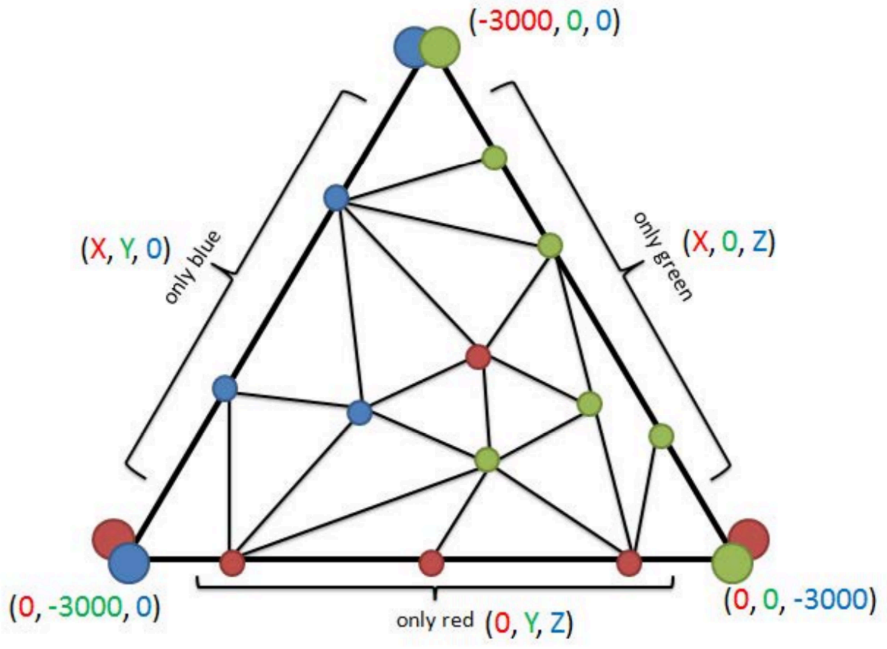
3.2.2. Harmonious Rent Problem

We will now look into solving Problem 3.3 using Sperner's lemma.

One possible attempt is to consider every possible assignment of price to each bedroom. i.e. set the price of room i to be x_i such that $x_1 + x_2 + \ldots + x_n = S$, when S is the total price of the apartment, and $x_i \geq 0$ for every $i = 1, 2, \ldots, n$.

Then, we consider the $(n-1)$ simplex obtained from the linear programming and color it like in the cake-cutting problem.

However, this problem is different from the cake-cutting problem in one aspect. In the cake-cutting problem, the bigger the piece of cake is, the higher chance people will want it; however, in this problem, the higher the price of a room is, the lower chance people will want that room. This would mean our simplex could look like:



Idea

If we could somehow turn every corner into a face and every face to a corner, we would be done.

Can that be done?

Yes. We transform each k face to $n-k-1$ face (for a n simplex).

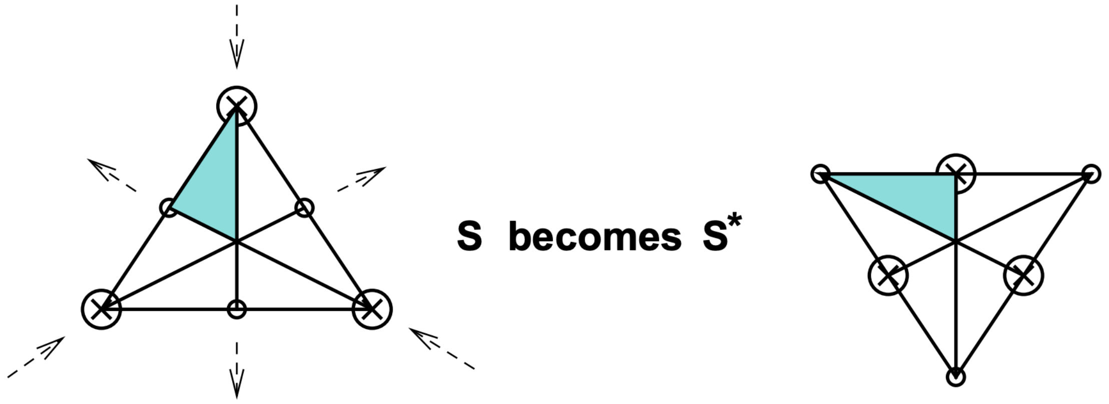


Figure 4: We have marked the vertices, points on line and colored an elementary cell to show how it transforms.

Let the triangulation of S^* inherit a labelling via this correspondence with S. One may now verify that the labelling of S^* is a Sperner labelling! Hence,there exists a fully labelled elementary simplex of S^*, which corresponds to a fully labelled elementary simplex of S, as desired. We can now present the solution.

Solution

A constructive algorithm is obtained by following "trap-doors" in Sperner's lemma. Choose an ε smaller than the rental difference for which housemates wouldn't care (a penny?). Following trap-doors corresponds to suggesting pricing schemes and then asking various players, "Which piece would you choose if the rooms were priced like this?" Once a fully labelled elementary simplex is found, any point inside it corresponds to an ε-approximate rent-partitioning.

However, we can also solve this without ever needing to go through this dualization process.

Solution

Each person first pays the full rent price and then assigns to each room a "rebate" price that a person who takes that room will get. Then, we can use Simmons' Algorithm to assign rooms to people.[5]

3.3. Fair Division of indivisible goods

Definition: Fair Division Setting

Given a set of agents $[n] = \{1...n\}$, set of items/resources $Z = \{z_1, z_2, ..., z_m\}$ and set of utility functions μ such that $v_i : \text{pow}(Z) \to \mathbb{R}$ for agent i.

A division is an allocation of items $(A_1, A_2, ..., A_n)$ such that $\bigcup_{i \in [n]} A_i = Z$ and $\forall i \neq j, A_i \cap A_j = \emptyset$.

We have types of valuations such that

Definition: Valuations

- **Additive** : $v_i(S) = \sum_{z \in S} v_i(z)$
- **Subadditve** : $v_i(S \cup T) < v_i(S) + v_i(T)$ where $S \cap T = \emptyset$
- **Submodular**: $v_i(S \cup) + v_i(S \cap T) \leq v_i(S) + v_i(T)$
- **Supramodular**: $v_i(S \cup) + v_i(S \cap T) \geq v_i(S) + v_i(T)$

We normally make a **normalization** assumption that is $\forall i, v_i(\emptyset) = 0$. A stronger assumption which is sometimes made is called **strong normalization**[6] where $v_i(Z) = 1$.

Another standardized assumption is **monotonicity** that is $\forall S \subseteq T, v_i(S) \leq v_i(T)$.

Assuming the valuations to be **Additive** is also a rather common assumption. We make it, unless stated otherwise.

We now we ask, what is fair?

3.3.1. Proportional Division

Definition: Probational

If for all agents $i \in [n], v_i(A_i) \geq \frac{1}{n}$, then the division satisfies **PROP**.

We are making the strong normalization assumption here.

As we will show, PROP is possible for the divisible case (recall that the continuity condition for valuations in divisible settings imply additivity).

However, it may not be possible for indivisible goods. Consider 2 agents and 1 item.

[5] The reason we didn't lead with this is as this fails in the case of where there is a room that nobody wants it even if it is free, someone may end up with that room with a negative price. We normally assume against it, that is called the **no "free closets" assumption** where free closets are rooms in which no one would live, even if free. This solution makes that assumption, while the other one doesn't.

[6] Non-standard term! Borrowed from Prof. Palavi Jain or Prof. Sushmita Gupta during COMSOC-2025 at IIT Jodhpur.

Remark

The 2 agent, 1 item case is a common counter example for indivisible fair division. It's generalization n agents, $n-1$ items is also a counter example in many cases.

Moving Knife Algorithm (PROP for divisible goods with continuous valuation)

```
1  We move a knife from left to right over a cake
2  As soon as the valuation is $\frac{1}{n}$ for an agent, they call stop.
3  We cut cake here.
4  The agent who calls stop is given the piece.
5  Continue with rest of cake and agents
```

Proof of correctness. All agents other than the last one call stop when they got a piece with proportional share.

For the last agent, we know that $v_n([0,1]) = 1$. Let $A = (A_1, ..., A_n)$ be the allocation vector. Then,

$$\sum_{i \in [n]} v_n(A_i) = v_n([0,1]) = 1$$

$$\text{Notice, } \forall i \in [n-1], v_n(A_i) < \frac{1}{n} \text{ as otherwise agent would have called stop}$$

$$\Rightarrow \sum_{i \in [n-1]} v_n(A_i) < \frac{n-1}{n}$$

$$\therefore \sum_{i \in [n-1]} v_n(A_i) + v_n(A_n) = 1$$

$$\Rightarrow v_n(A_n) > \frac{1}{n}$$

■

Would such an algorithm work for the indivisible case (given additive valuations)?

Bag Filling Algorithm

```
1  Add items to a bag till an agent says stop.
2  Give bag to agent and continue.
```

Why would this not work? Because unlike the above case where $v_i(A_i) = \frac{1}{n}$ for all but last agent(they can have better as well); here $v_i(A_i) \geq \frac{1}{n}$ and $\forall z \in A_i, v_i(A_i \setminus z) < \frac{1}{n}$.

So can we modify it to maybe work? Not really as PROP is not guaranteed to exist. We can instead modify it to work for **PROP1**.

Definition: PROP1

An allocation is **PROP1** if and only if

$$\forall i \in A_i, \exists z \in Z \setminus A_i \text{ s.t. } v_i(A_i) + v_{i(z)} \geq \frac{1}{n}$$

This can be generalized to **PROPc** where c is the number of goods we get to add.

Bag Filling Algorithm

```
1 Add items to a bag till an agent says stop.
2 Remove the last item and put it in a special bag
3 Give bag to agent and continue.
4 Once $n-1$ bags are given, give special bag to last agent.
```

The analysis is similer to the moving knife. We can also modify the algorithm a bit to get **PROPx** (if it exists).

Definition: PROPx

An allocation is **PROPx** if and only if

$$\forall i \in A_i, \forall z \in Z \setminus A_i \text{ s.t. } v_i(A_i) + v_{i(z)} \geq \frac{1}{n}$$

Counter Example to existence of PROPx. Consider an example with three agents with identical valuations; three large goods a, a, b; and 10 small goods c.

Each a brings utility 0.15 , b brings utility 0.4, and each c brings 0.03.

Check that, if Agent 1 gets only cs, and Agent 2 gets b, then PROPx fails.

However, PROPx also fails for Agent 1 who gets at most nine cs. If Agent 1 gets b, while Agents 2 and 3 get one a each, then one of Agents 2 and 3 gets at most five cs (otherwise a fails the EFX) and a total utility of 0.3, so PROPx fails again. ■

Remark

The counter example was though of with a with value $\frac{3}{2}$, b with 4 and c with $\frac{3}{10}$ and then scaled to the case.

This is another common counter example idea to make things almost equal but then divide it into parts.

Remark

Bag filling is a common approximation algorithm. It and it's modifications can be used to approximate a lot of bound based allocations. So keeping it in the back of mind is a good idea.

3.3.2. Envy Free Allocation

Recall, we discussed this for the divisible case and saw it always exists.

Definition: EF

An allocation is **EF** if and only if

$$\forall i, j \in [n] v_i(A_i) \geq v_i(A_j)$$

We don't make the strong normalization here.

It doesn't always exist (2 goods, 1 agent counter example).

Theorem 3.8. EF $\Rightarrow$ PROP *but* PROP $\nRightarrow$ EF

Proof. FTSOC, let there be some EF allocation which is not PROP. That means there is some agent i such that $v_i(A_i) < \frac{v_i(Z)}{n}$. But,

$$\begin{aligned} \sum_{j \in [n]} v_i(A_j) &= v_i(Z) \\ \Rightarrow \sum_{j \in [n], i \neq j} v_i(A_i) &> v_i(Z) \frac{n-1}{n} \\ \Rightarrow \exists j \in [n] \text{ s.t. } v_i(A_j) &> \frac{v_i(Z)}{n} \\ \Rightarrow \exists j \in [n] \text{ s.t. } v_i(A_i) &< v_i(A_j) \end{aligned}$$

which contradicts EF. Thus, our assumption was false and thus, EF $\Rightarrow$ PROP

As for showing the converse doesn't hold, consider 3 agents and 3 goods with the following allocation.

$$\begin{pmatrix} A : \boxed{1} \ 2 \ 2 \\ B : 2 \ \boxed{1} \ 2 \\ C : 2 \ 2 \ \boxed{1} \end{pmatrix}$$

■

Theorem 3.9. *Determining if EF exists is NP hard for even 2 agents.*

Proof. Consider the partition problem with set $S = \{s_1, s_2, ..., s_n\}$. Consider an EF division instance with

$$\begin{pmatrix} A : s_1 & s_2 & \dots & s_n \\ B : s_1 & s_2 & \dots & s_n \end{pmatrix}$$

The equivalence in solutions is trivial. ■

Can we weaken the case?

Definition: EF1

An allocation is EF1 if for any agents $i, j \in [n]$

$$\exists g \in A_j \text{ s.t. } v_i(A_i) \geq v_i(A_j - g)$$

that is agents don't envy each other upto removal of one good.

Theorem 3.10. *EF1 always exists*

Proof.

Round Robin Algorithm

```
Order the agents arbitrarily
for $i \in [n]$:
    if $Z = \emptyset$:
        break
    sort $Z$ with respect to $v_i$
    $g$ = highest item
    add $g$ to $A_i$
    remove $g$ from $Z$
return $(A_1, A_2, \dots, A_n)$
```

We claim this algorithm gives an EF1 allocation.

If $i < j$, then i got to pick before j every round and can't envy j.

If $i > j$, then i can envy j only over the first item as she passed on the other items and doesn't envy wrt to them. ■

Remark

Round Robin is also a class of algorithms which do work in a lot of cases.

Definition: Chore

A chore is something to be allocated which all agents value negatively.

While it is clear that round robin works in only chore case, we can show it works to give an EF1 in a mix of goods and chore case.

Definition: EF1

An allocation of goods and chores is EF1 if for any agents $i, j \in [n]$

$$\exists g \in A_j \text{ s.t. } v_i(A_i) \geq v_i(A_j - g)$$

or

$$\exists c \in A_i \text{ s.t. } v_i(A_i - c) \geq v_i(A_j)$$

that is agents don't envy each other upto removal of one good.

Double Round Robin

```
Order agents arbitrarily
divide $Z = Z_g \cup Z_c$ where $Z_g$ is goods and $Z_c$ is chores
Forever:
    for $i \in [n]$:
        if $Z_c = \emptyset$:
            break
        sort $Z$ with respect to $v_i$
        $c$ = highest item
        add $c$ to $A_i$
        remove $c$ from $Z_c$
Forever:
    for $i$ in $\{n, n-1, ..., 1\}$:
        if $Z_g = \emptyset$:
            break
        sort $Z$ with respect to $v_i$
        $g$ = highest item
        add $g$ to $A_i$
        remove $g$ from $Z$
```

Showing this works is not very hard and the proof is almost identical to the goods only case.

We will now also discuss another algorithm which works when valuations are not additive, only monotonicity is given. We first show round robin doesn't work in this case.

Consider 2 agents with valuations over A, B, C, D, E as $5, 4, 3, 2, 1$. By round robin, 1 ends up with A, C, E and 2 with B, D. Setting valuations of $\{A, C\}, \{C, E\}$ and $\{A, E\}$ to be higher than $\{B, D\}$ is easy.

Someone may suggest choosing the best thing to add. Constructing a counterexample for that using the same idea is not hard.

So what do we do now?

Definition: Envy Graph

An envy graph is a graph in which each vertex represents an agent along with its partial allocations. There is a directed edge from vertex i to vertex j if agent i envies agent j under the current partial allocation.

Definition: Envy Cycle

An envy cycle is a directed cycle in the envy graph

Definition: Source

Source node is a node in the envy graph that does not have any incoming edges.

Envy Cycle Elimination

```
while there is an unallocated object $X$:
    there is an unallocated object $X$
    if the envy graph has a source vertex $v$ then
        Allocate $X$ to $v$
    else
        Resolve cycles until a source vertex shows up
```

Resolving a cycle here is just moving a good from a person to another person till the cycle is not there. As we give out one good at a time, the envy at the end is all upto 1 and hence, EF1 is achieved.

It is easy to show that the number of edges strictly decrease after reduction. This implies we make atmost $O(n^2)$ cycle reductions. Thus, the algorithm is polytime.

3.3.3. Pareto Optimality

Definition: Pareto

An allocation $A \succ B$ if

$$\begin{aligned} \forall i \in [n] \quad & v_i(A_i) \geq v_i(B_i) \\ \exists j \in [n] \quad & v_j(A_i) \gneqq v_i(B_i) \end{aligned}$$

Definition: Pareto Optimal (PO)

A is Pareto optimal if there is no B such that $B \succ A$.

Basically, A is PO if you cannot make some agents better off without making anyone worse off.

Note, PO always exists and the proof if by the fact that giving everything to the same agent does achieve that.

Definition: Social Welfare

Social welfare of allocation A is

$$\mathrm{SW}(A) = \sum_{i \in [n]} v_i(A_i)$$

Theorem 3.11. *If*

$$A \in \arg \max_{B \in \Pi_m(m)} \mathrm{SW}(B)$$

then A is PO.

Proof. If A is not PO, we can just switch the goods around and end up with higher valuation for some agents and hence, a higher social welfare which contradicts the maximality of social welfare. ■

So can we get a **EF1 + PO**? Trying to do pareto switches to EF1 doesn't work as:

$$\begin{pmatrix} A: & 1 & 1 & \ldots & 1 \\ B: & 1 & 1 & \ldots & 1 \\ C: & 0 & 0 & \ldots & 0 \end{pmatrix}$$

here the allocation $A_1 = \{1, 2, ..., 10\}, A_2 = \{11, 12, ..., 20\}, A_3 = \{21, 22, ..., 30\}$ is EF but making pareto switches wull make it extremely envy prone.

Does the round robin algorithm work?

$$\begin{pmatrix} A: & \boxed{2} & 2 & \boxed{2} & 2 \\ B: & 10 & \boxed{1} & 1 & \boxed{1} \end{pmatrix}$$

This is not pareto optimal as switching 10 ↔ 2 and 2 ↔ 1 can be done.

Does Envy cycle agent elimination work?

$$\begin{pmatrix} A: & \boxed{10} & 5 & \boxed{1} & 1 \\ B: & 1 & \boxed{1} & 5 & \boxed{10} \end{pmatrix}$$

but this is not PO as switching 5 ↔ 1 and 1 ↔ 5 can be done.

So does a EF1 + PO allocation even exist?

Definition: Nash Social Welfare

$$\mathrm{NSW}(A) = \left(\prod_{i\in[n]} v_i(A_i)\right)^{\frac{1}{n}}$$

Definition: Nash Optimal

We say A is Nash Optimal if it is the largest set of agents S who can simultaneously get positive values and take such a division which maximizes the geometric mean among S.

Working in an instance where $m \geq n$ where $\forall i \in [n], \forall g, v_i(g) > 0$.

Lemma 3.12. *If A is Nash optimal then:*
1. A is PO
2. A is EF1

Proof of (1).. Suppose not. Say $B \succ A$ then the set of people with non-zero utility in B, say $S' \supseteq S$ which is the set of people with non zero utility in A. Thus,

$$\mathrm{NSW}(B) = \prod_{i\in S'} v_i(B_i) > \prod_{i\in S} v_i(B_i) \geq \prod_{i\in S} v_i(A_i)$$

which is a contradiction. ■

Proof of (2).. Suppose A is Nash optimal but not EF1.

That is $\exists i, j$ such that

$$\forall g \in A_j v_i(A_i) < v_i(A_k - g)$$

Let

$$g^* = \arg \min_{\substack{g\in A_k \\ v_i(g)>0}} \frac{v_k(g)}{v_i(g)}$$

Construct a new allocation which everything same but $v_i(B_i) = v_i(A_i) + v_i(g^*)$ and $v_j(B_j) = v_j(A_j) - v_j(g^*)$.

We claim

$$
\begin{gathered}
\frac{\mathrm{NSW}(B)}{\mathrm{NSW}(A)} > 1 \\
\iff v_i(B_i)v_j(B_j) > v_i(A_i)v_j(A_j) \\
\iff [v_i(A_i) + v_i(g^*)][v_j(A_j) - v_j(g^*)] > v_i(A_i)v_j(A_j) \\
\iff v_i(A_i)v_j(A_j) + v_i(g^*)v_j(A_j) - v_i(A_i)v_j(g^*) - v_i(g^*)v_j(g^*) > v_i(A_i)v_j(A_j) \\
\iff 1 + \frac{v_i(g^*)}{v_i(A_i)} - \frac{v_j(g^*)}{v_j(A_j)} - \frac{v_i(g^*)v_j(g^*)}{v_i(A_i)v_j(A_j)} > 1 \\
\iff v_j(A_j) > \frac{v_j(g^*)}{v_i(g^*)}[v_i(A_i) + v_i(g^*)] \quad \text{(by Vishwa bhaiya claims this to be true)}
\end{gathered}
$$

From the choice of g^*,

$$
\begin{gathered}
\frac{v_j(g^*)}{v_i(g^*)} = \theta \\
\Rightarrow \frac{v_j(g)}{v_i(g)} \geq \theta \\
\Rightarrow v_j(g) \geq \theta v_i(g)
\end{gathered}
$$

Using this,

$$
\frac{v_j(A_j)}{v_i(A_j)} \geq \theta = \frac{v_j(g^*)}{v_i(g^*)}
$$

Using envy freeness,

$$
\begin{gathered}
v_i(A_i) < v_i(A_j) - v_i(g^*) \\
v_i(A_i) + v_i(g^*) < v_i(A_j)
\end{gathered}
$$

We can multiply these to get

$$
v_j(A_j) > \frac{v_j(g^*)}{v_i(g^*)}[v_i(A_i) + v_i(g^*)]
$$

leading to a contradiction and completing the proof. ■

While NSWO is guaranteed to exist, it is NP hard to find and even hard to approximate. The question if EF1+PO can be done in poly time is open.

4. Interlude: Mechanism Design

Definition: Mechanism Design

Till now we have been given players, game and strategies and we want to find outcome via the equilibrium.

We will now do mechanism design where we are given the players and our ideal outcome and we want to come up with a game where the equilibrium is in line with our outcome.

Example

In making a cricket tournament, we don't want players to lose on purpose to get an easier opponent later and have better medal odds.

A failure of this was the 2012 London Olympics where the women's badminton doubles was structured so badly that both teams were trying to lose in an extremely hard to watch match. Search "Wang Xiaoli / Yu Yang (CHN) vs Ha Jung-eun / Kim Min-jung (KOR)" if you want to see the match, the referee being tense and the crowd booing.

They were later disqualified for unsportsperson like behavior. Although, trying to win by all legal means is sportsperson like in my book. Maybe they should have set the tournament better to never let this be incentivized in the first place.

Mechanism design is used to design tournaments, voting schemes and schemes to divide stuff.

5. Auction Theory

5.1. Single Item Auction

Definition: Single Item Auction

One seller wants to sell a single item, n interested buyers (agents). Each agent has a value v_i for the item.

Goal: To give the item to an agent and get payment from that agent.

To decide: Who gets the item? At what price?

We normally make the assumption that the agents are unaware of each other's valuations.

5.2. Sealed Bid Auctions

Definition: Single Item, Sealed Bid

(1) Each bidder i privately communicates a bid b_i to the auctioneer.

(2) The auctioneer decides who gets the good (if anyone).

(3) The auctioneer decides on a selling price.

There is an obvious choice of (2) being letting the highest bidder have the good. This is not always optimal from a revenue standpoint, but that is conversation for later.

We can implement (3) as:

- **First Price Auction** Winner pays their bid
- **Second Price Auction** Winner pays the next highest bid

Reasoning About Auctions

The following is taken from Pollak's notes on Game Theory.

We claim that agents bid as if they know they will win. This is to avoid the so called 'winner's curse' where one has to pay more than their valuation.

The proof is obvious as if your bid is such that you regret winning, you should not bid that amount. Furthermore, we make the assumption that an agent would like to have the object than not.

Bidding below this amount is not optimal as say if you bid $y - \varepsilon$ where y is the bid as if you know you win bid; then if an agent bids $y - \frac{\varepsilon}{2}$ and wins; the agent could be better off bidding $y - \frac{\varepsilon}{2} + \tilde{\varepsilon}$.

This implies that it is weakly dominant to bid one's valuation in a second price auction as the payoff function is 0 if you are not the highest bidder and $v_i - b_{(2)}$ otherwise.

Similarly, bidding your value is weakly dominated in First Price Auction as the valuation function is 0 if you are not the highest bidder and $v_i - b_i$ otherwise.

Definition: Awesome Auctions

We want an auction to have the following properties

- **Dominant Strategy Incentive Compatible (DSIC)**: Truthful bidding must be a dominant stratergy.
- **Strong Performance Guarantee**: Maximize $\sum x_i v_i$ or social surplus where x_i is 1 if i wins and 0 if i loses, subject to the obvious feasibility constraint that $\sum_{i=1}^{n} x_i \leq 1$ as we have only one item.
- **Polyline Computability**

An awesome auction is clearly desirable as truthful bidding is a dominant strategy, the auction guarantees that the item will go to the agents who value it the most and the auction can be decided before the heat-death of the universe.

Claim 5.1. Second Price Auction is awesome.

5.3. Sponsored Search Auction

Perhaps the most profitable auction of all time. In 2014, it made around 98% of google's revenue[7].

Similarly, not having a good sponsored search auction was one of the reasons behind the downfall of Yahoo.

Definition: Sponsored Search Auction

Given k add slots and and agents $j \in [n]$ with per click value of v_j of the agents. Assuming the click through rate of these slots is $\alpha_i, i \in k$. Thus, if agent i gets slot j then the value obtained is $\alpha_j v_{i_j}$.

An auction to allot thee k slots to n agent is called an sponsored search auction.

Problem 5.2. Can we design an awesome sponsored search auction?

Solution

Step 1 Assume that all agents bid their true values and choose an allocation rule that maximizes the social surplus and runs in polytime.

Step 2 Design a payment rule that makes truthfulness dominant.

Step 3 Profit!

Claim 5.3. Greedy Assignment maximizes the social surplus.

Proof. The proof is an follows from Rearrangement inequality. ■

The step 2 is easy and is done by something called 'Myerson's Lemma' which will be the focus of a lot of what is to come.

For step 3, well, that is left to the reader.

Remark: A detour to Olympiad Inequalities

The statement and proof for Rearrangement Inequality and it's somewhat stronger cousin, Chebyshev's Inequality are given below. They are quite common in Olympiad circles.

My proofs are taken from Doorway to Math Olympiad[8].

[7] You know, back when it was search engine and advertisement company and not an AI obsessed mega corporation burning money left and right

[8] Which is an incomplete and unpublished textbook I was writing at some point. Hopefully, I'll complete it at some point and will need to edit this footnote.

Rearrangement Inequality 5.4. *If $a_1 \leq ... \leq a_n$ and $b_1 \leq ... \leq b_n$ then for any permutation (rearrangement) $c_1, ..., c_n$ of $b_1, ..., b_n$, *

$$a_1 b_n + ... + a_n b_1 \leq a_1 c_1 + ... + a_n c_n \leq a_1 b_1 + ... + a_n b_n$$

Proof. The proof of the Rearrangement Inequality can be handled with proof by contradiction. We will prove the maximization first, the minimization will follow from that.

Let us first consider the case where $n = 2$. We can take $a_1 \leq a_2$ and $b_1 \leq b_2$.

$$\begin{gathered} \therefore (a_1 - a_2)(b_1 - b_2) \leq 0 \\ \iff a_1 b_1 + a_2 b_2 - a_1 b_2 - a_2 b_1 \leq 0 \\ \iff a_1 b_1 + a_2 b_2 \leq a_1 b_2 + a_2 b_1 \end{gathered}$$

Now for the general case. Let $a_1 \leq a_2 \leq ... \leq a_n$ and $b_1 \leq b_2 \leq ... \leq b_n$; and let's to the contrary assume that in the grouping maximizing the sum, a_m is not paired with b_m. We'll instead assume that a_m is paired with b_l and b_m is paired with a_l.

Hence we are claiming, $a_m b_l + a_l b_m \leq a_m b_m + a_l b_l$ which is untrue as we showed above.

The minimization equality can be very easily proved by noting that if we have the set $\{-b_1, -b_2, ..., -b_n\}$, ordered in increasing order(which makes $b_1 \leq b_2 \leq b_3 ... \leq b_n$) and the set $\{a_1, a_2, ...\}$, ordered in decreasing order, then the maximum sum is just $-a_1 b_n - a_2 b_{n-1} + ...$. Whose negative is $a_1 b_n + ... + a_n b_1$ which will be the minimum possible value. ■

A more refined form of the rearrangement inequality is

Chebyshev's Inequality 5.5. *If $a_1 \leq a_2 \leq ... \leq a_n$ and $b_1 \leq b_2 \leq ... \leq b_n$ then the following inequality holds:*

$$n\left(\sum_{i=1}^{n} a_i b_i\right) \geq \left(\sum_{i=1}^{n} a_i\right)\left(\sum_{i=1}^{n} b_i\right)$$

On the other hand, if $a_1 \leq a_2 \leq ... \leq a_n$ and $b_n \leq b_{n-1} \leq ... \leq b_1$ then:

$$n\left(\sum_{i=1}^{n} a_i b_i\right) \leq \left(\sum_{i=1}^{n} a_i\right)\left(\sum_{i=1}^{n} b_i\right)$$

Proof. The proof is simple. We know that $\sum_{i=1}^{n} a_i b_i$ is maximal.

$$\therefore \sum_{i=1}^{n} a_i b_i \geq a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$$
$$\sum_{i=1}^{n} a_i b_i \geq a_1 b_2 + a_2 b_3 + \ldots + a_n b_1$$
$$\vdots$$
$$\sum_{i=1}^{n} a_i b_i \geq a_1 b_n + a_2 b_1 + \ldots + a_n b_{n-1}$$

Adding them will give us the inequality. ■

5.4. Myerson's Lemma

Let's define everything formally

Definition: Auction

n agents, one seller. Private valuations $v_i, i \in [n]$. X is the set of feasible allocations of the items.

Allocation and payment rules are:
1. Collect bids $b_i, i \in [n]$ as input
2. Allocation rule $x(B) \in X$ that is $x : \mathbb{R}^n \to X$
3. Payment rule $p(B) \in \mathbb{R}^n$ that is $p : \mathbb{R}^m \to \mathbb{R}^n$

Utility is $u_i(b) = v_i x_i(B) - p_i(B)$

Where $p_i(B) \in [0, b_i x_i(B)]$

Definition: Implementable Allocation Rules

An allocation rule x is implementable if $\exists$ a payment rule p such that the sealed bid auction (x, p) is DSIC.

Definition: Monotone Allocation Rule

An allocation rule x is monotone if

$$\forall i \in [n], \forall z > y, \quad x_i(z, b_{-i}) > x_i(y, b_{-i})$$

Definition: Awesome Auctions?

We want an auction to have the following propoerties
- **Dominant Strategy Incentive Compatible (DSIC)**: Truthful bidding must be a dominant stratergy.

- **Strong Performance Guarantee**: Maximize $\sum x_i v_i$ or social surplus.
- Polytime Computability

[9]

Definition: Sponsored Search Auctions

n agents, k slots, i^{th} slot has click through rate α_i where $\alpha_1 \geq ... \geq \alpha_k$ and valuations v_j. Agent j derives value $\alpha_i v_j$ if i^{th} slot is allocated to j.

We want to choose allocation and pricing rules.

Myerson's Lemma 5.6.

1. *An allocation rule is implementable if and only if, it is monotone.*
2. *For any implementable allocation rule* $\exists$, *unique pricing rule* p *such that* (x, p) *is DSIC.*
3. p *is given by an explicit formula.*

Proof. ($\Longrightarrow$) Assume x is implementable with DSIC pricing p. Suppose $0 \leq y \leq z$

(i) Suppose $b_i = z$ and $v_i = y$ and other agents have bids fixed at b_{-i}. This is the case of **overbidding**.

As we want the DSIC property, by the power of abuse of notation, we shall now read $x_i(z) = z_i(z, b_{-i})$.

$$yx_i(z) - p_i(z) \leq yx_i(y) - p_i(y)$$

(ii) Suppose $b_i = y$, $v_i = z$. This is the case of **underbidding**.

$$zx_i(y) - p_i(y) \leq zx_i(z) - p_i(z)$$

We can write $z(x_i(y) - x_i(z)) \leq p_i(y) - p_i(z) \leq y(x_i(y) - x_i(z))$
$\Rightarrow (z - y)(x_i(z) - x_i(y)) \geq 0$
$\Rightarrow z \geq y \Rightarrow x_i(z) \geq x_i(y)$.

($\Longleftarrow$) Assume x is monotone.

$$y(x_i(z) - x_i(y)) \leq p_i(z) - p_i(y) \leq z(x_i - x_i(y))$$

(i) x is flat at y

$$\lim_{z \to y^+} y(x_i(z) - x_i(y)) \leq \lim_{z \to y^+} p_i(z) - p_i(y) \leq \lim_{z \to y^+} z(x_i - x_i(y))$$
$$\Rightarrow p_i(z) = p_i(y)$$

for z in neighborhood of y.

[9] I am not sure if this is the real term for this. Will check at some point.

(ii) x has a step jump at y of height h

$$\begin{aligned}\lim_{z\to y^+} y(x_i(z)-x_i(y)) \leq \lim_{z\to y^+} p_i(z)-p_i(y) \leq \lim_{z\to y^+} z(x_i-x_i(y))\\ \Rightarrow yh \leq p_i(z)-p_i(y) \leq yh\\ \Rightarrow p_i(z)-p_i(y) = yh\end{aligned}$$

jump i x at y implies a jump in p at y.

$$\therefore \text{Price at } y_l \text{ when there are } l \text{ jumps } y_1, y_2, ..., y_l$$
$$= \sum_{i=1}^{l} y_i(\text{jump height at } y_i \text{ in } x_i)$$

(iii) When x is differentiable

$$\begin{aligned}\lim_{z\to y} \frac{y(x_i(z)-x_i(y))}{z-y} \leq \lim_{z\to y} \frac{p_i(z)-p_i(y)}{z-y} \leq \lim_{z\to y} \frac{z(x_i-x_i(y))}{z-y}\\ \Rightarrow yx_i'(y) \leq p_i'(y) \leq yx_i'(y)\\ \Rightarrow p_i'(y) = yx_i'(y)\\ \Rightarrow p_i(z) = \int_0^z p_i'(y) = \int_0^z yx_i'(y)\end{aligned}$$

TODO. *Will copy from somewhere else!!!!*

■

5.5. Knapsack Auctions

Definition

Seller's Capacity: W

Buyer i has requirement w_i.

Goal: Allocate x_i amount to buyer $i \in [n]$ such that $\sum_{i=1}^{n} w_i x_i \leq W$ where $x_i \in \{0,1\}$.

The issue with using our normal strategy is welfare maximization (assuming truthful bidding) maximize $\sum_{i=1}^{n} x_i b_i$.

This is NP Hard as this is literally the Knapsack problem which is NP Hard!

Can we modify the existing approximation algorithm to be monotone, retaining the approximation gurentee? From [Chawla, Immorlica, Lucier 2012], it is not true in general. As in we can't do a black box reduction that is we need to know about the instance and can't do so in a general way.

Can there be a dominant strategy that is different from truthful bidding?

Definition: Revelation Principle

For any mechanism M which always has a dominant strategy, then there is a mechanism M' such that truthful bidding is dominant strategy for M'.

Example : 'Silly' example

Single item auction where seller runs a second price auction of bids $2b_i$ when agents submit $b_i, i \in [n]$. (Note, the agents bid say $b_1 > b_2 > ...$ then agent i pays $2b_2$).

Here dominant strategy is $b_i = \frac{v_i}{2}$.

Proof. Notice b_i is always a function of v_i.

For M, let the dominant strategy be $b_i = f_i(v_i)$.

Let M' be a mechanism which takes $\{b_i\}$ as inputs and runs mechanism M on $f_i(B_i)$. Thus, dominant strategy for M' is $b_i = v_i$. ■

This implies DSIC is free if we can design a mechanism with dominant strategy.

5.6. Revenue Maximization

Example

One Agent, one item. The agent values it v which is private.

This general case is not solvable. So we assume v is drawn from a known probability distribution.

Exercise

Let's say seller's price is drawn from $v \sim \text{Uniform}[0, 1]$ and seller sets the price to r. What r maximizes revenue?

Solution. Expected revenue is $r\mathbb{P}(v > r) = r(1 - r)$ which is maximized at $r = \frac{1}{2}$ and hence, expected revenue is $\frac{1}{4}$. ■

Exercise

Let $v_1, v_2 \sim \text{Uniform}[0, 1]$. What is expected revenue in a second price auction?

Solution. Revenue is clearly $\mathbb{E}(\min(v_1, v_2))$. We can compute it by:

$$\begin{aligned}
\mathbb{P}(x < \min(v_1, v_2)) &= 1 - (1-x)^2 = F_X(x) \\
\because \mathbb{E}(X) &= \int_0^1 1 - F_X(x)\,\mathrm{d}x \\
\Rightarrow \mathbb{E}(X) &= \int_0^1 (1-x)^2\,\mathrm{d}x \\
&= \frac{x^3}{3}]_{-1}^{0} \\
&= \frac{1}{3}
\end{aligned}$$

■

Exercise

Let $v_1, v_2 \sim \text{Uniform}[0,1]$. What is expected revenue in a second price auction with reserve price R (we don't sell below R)?

Solution. Revenue is 0 if $v_1, v_2 < R$; R if $v_i > R, v_{-i} < R$ and $\min(v_1, v_2)$ if $v_1, v_2 > R$.

Thus,

$$\begin{aligned}
\mathbb{E}(\text{revenue}) &= 0 * R * R + R * 2 * (R * (1-R)) + (1-R) * (1-R) * \mathbb{E}(\min(v_1, v_2) \mid v_1, v_2 > R) \\
&= 0R^2 + 2R^2(1-R) + (1-R)^2\mathbb{E}(\min(v_1, v_2) \mid v_1, v_2 > R) \\
&= 2R^2(1-R) + (1-R)^2(R + \mathbb{E}(\min(u_1, u_2)) \quad \text{where } u_1, u_2 \sim \text{uniform}[0, 1-R] \\
&= 2R^2(1-R) + (1-R)^2\left(R + \frac{1-R}{3}\right) \\
&= 2R^2(1-R) + \frac{(1-R)^2(1+2R)}{3}
\end{aligned}$$

This achieves maxima at $R = \frac{1}{2}$. ■

Exercise

Let $v_1, v_2, ..., v_n \sim \text{Uniform}[0,1]$. What is expected revenue in a second price auction with reserve price R (we don't sell below R)?

Solution. We use the order statistics notation.

Revenue is 0 if $v_{(1)} < R$, R if $v_{(1)} > R; v_{(2)} < R$ and $v_{(2)}$ if $v_{(1)}, v_{(2)} > R$.

Consider

$$\begin{aligned}
\mathbb{P}(x < v_2 < x + h) &= nh(n-1)(1-x-h)x^{n-2} \\
\Rightarrow \mathbb{P}(v_2 = x) = \lim_{h \to 0} \frac{\mathbb{P}(x < v_2 < x+h)}{h} &= n(n-1)(1-x)x^{n-2}
\end{aligned}$$

.

Thus,

$$\begin{aligned}\mathbb{E}(\text{revenue}) &= \underbrace{\int_R^1 n(n-1)(1-x)x^{n-2}x \,\mathrm{d}x}_{\text{selling at second price}} + \underbrace{RnR^{n-1}(1-R)}_{\text{selling at reserve price}} + \underbrace{0}_{\text{not selling}} \\ &= n(n-1)\int_R^1 x^{n-1} - x^n \,\mathrm{d}x + nR^n(1-R) \\ &= n(n-1)\left[\frac{1}{n} - \frac{1}{n+1} - \frac{R^n}{n} + \frac{R^{n+1}}{n+1}\right] + nR^n(1-R)\end{aligned}$$

For $R = \frac{1}{2}$

$$\mathbb{E}(\text{revenue}) = n(n-1)\left[\frac{1}{n} - \frac{1}{n+1} - \frac{1}{2^n n} + \frac{1}{2^{n+1}(n+1)}\right] + \frac{n}{2^{n+1}}$$

And maximizing expected revenue, we differentiate and set to 0

$$\begin{gathered} n(n-1)[-R^{n-1} + R^n] + n^2R^{n-1}(1-R) - nR^n = 0 \\ \Rightarrow n(n-1)[R-1] + n^2(1-R) - nR = 0 \\ \Rightarrow (n-1)[R-1] + n(1-R) - R = 0 \\ \Rightarrow (1-R)(n-1-n) - R = 0 \\ \Rightarrow 1 - 2R = 0 \\ \Rightarrow R = \frac{1}{2}\end{gathered}$$

Thus, the revenue is maximized at $R = \frac{1}{2}$. ∎

5.7. Revenue Maximizing Auctions

Single parameter environments where each agent has a private valuation v_i and X is a set of feasible allocations.

Our model Single parameter environment where v_i's are drawn form distributions F_i, f_i which are independent and supported on $[0, v_{\max}]$.

The goal is to design a DSIC auction (X, p) that maximizes the expected revenue.

Seller knows $F_i \forall i \in [n]$, v_i's are private to the agents.

$$\text{Welfare} = \sum_{i=1}^{n} x_i v_i$$

$$\begin{aligned}\mathbb{E}(\text{Welfare}) &= \mathbb{E}\left[\sum_{i=1}^{n} x_i(v) v_i\right] \\ &= \sum_{i=1}^{n} \mathbb{E}[x_i(v) v_i] \\ &= \sum_{i=1}^{n} \int_0^{v_{\max}} x_i(v) v_i f_i(v_i)\,\mathrm{d}v_i\end{aligned}$$

$$\text{Revenue} = \sum_{i=1}^{n} P_i(v)$$

$$\begin{aligned}\mathbb{E}(\text{Revenue}) &= \mathbb{E}\left[\sum_{i=1}^{n} P_i(v)\right] \\ &= \sum_{i=1}^{n} \mathbb{E}[P_i(v)]\end{aligned}$$

By Myerson's lemma,

$$P_i(v_i, v_{-i}) = \int_0^{v_i} z x_i'(z, v_{-i})\,\mathrm{d}z$$

$$\begin{aligned}\Rightarrow \mathbb{E}[P_i(v)] &= \int_0^{v_{\max}} P_i(v) f_i(v_i) dv_i \\ &= \int_0^{v_{\max}} \left[\int_0^{v_i} z x_i'(z, v_{-i})\,\mathrm{d}z\right] f_i(v_i)\,\mathrm{d}v_i\end{aligned}$$

Notice, the region of integration is

TODO. *Nice drawing!*

Interchanging the integral

$$\begin{aligned}&= \int_0^{v_{\max}} \left[\int_z^{v_{\max}} f_i(v_i)\,\mathrm{d}v_i\right] z x_i'(z)\,\mathrm{d}z \\ &= \int_0^{v_{\max}} [F_i(v_{\max}) - F_i(z)] z x_i'(z)\,\mathrm{d}z \\ &= \int_0^{v_{\max}} [1 - F_i(z)] z x_i'(z)\,\mathrm{d}z\end{aligned}$$

Integrating by parts

$$\begin{aligned} &= [(1 - F_i(z))zx_i(z)]_0^{v_{\max}} - \int_0^{v_{\max}} x_i(z)[1 - F_i(z) - zf_i(z)]\,\mathrm{d}z \\ &= (0-0) - \int_0^{v_{\max}} x_i(z)f_i(z)\left[\frac{1-F_i(z)}{f_i(z)} - z\right]\mathrm{d}z \\ &= \int_0^{v_{\max}} x_i(z)f_i(z)\left[z - \frac{1-F_i(z)}{f_i(z)}\right]\mathrm{d}z \end{aligned}$$

Now compare this to the welfare function:

$$\int_0^{v_{\max}} x_i(v)v_i f_i(v_i)\,\mathrm{d}v_i$$

The only difference we get is the valuation. We call this the virtual valuation of the agent.

$$\phi_i(z) := Z - \frac{1-F_i(z)}{f_i(z)}$$

$$\begin{aligned} \Rightarrow \text{Expected Revenue} &= \sum_{i=1}^{n} \int_0^{v_{\max}} \phi_i(v)v_i f_i(v_i)\,\mathrm{d}v_i \\ &= \sum_{i=1}^{n} \mathbb{E}[\phi_i(z)x_i(z)] \\ &= \mathbb{E}\left[\sum_{i=1}^{n} \phi_i(z)x_i(z)\right] \\ &= \text{expected virtual welfare} \end{aligned}$$

Example : Virtual Valuation

Let's consider $F_i \sim$ uniform $[0,1]$.

Then $F_i(z) = Z, f_i(z) = 1$. This implies

$$\phi_i(z) = z - \frac{1-z}{1} = 2z - 1$$

Notice, $\varphi_i(z) \leq 0$ for $z \leq \frac{1}{2}$. This implies that if all the virtual prices are below 0, the seller should not sell/allocate item to anyone.

Example : Single Item Auction

Let $F_1, F_2 ...$ are iid. How to design (x, p)?

Goal $\forall v$ maximize $\sum_{i=1}^{n} \phi_i(v)x_i(v)$. Is such an x monotone?

We need $\varphi_i(v)$ to be increasing in v. A distribution F_i such that $\varphi_i(v) = v - \frac{1-F_i(v)}{f_i(v)}$ is increasing is called "regular".

Allocation rule: Allot the item to the agent with max virtual valuation ⇒ second price auction with reserve price $\phi^{-1}(0)$ where $\phi = \phi_i \forall i$.

Remark: A digression on Regular Distribution

$$\phi(v) = v - \frac{1 - F(v)}{f(v)} = v - \frac{1}{h(v)}$$

where h is called the hazard rate of the distribution.

Regular functions are characterized by the strict convexity of $\frac{1}{1-F(x)}$ for the distribution (given we can differentiate, this is easy to prove given someone, somehow makes this hypothesis). Details on the characterization if smoothness assumption is not made can be found in Ewerhart, 2012.

It uses weird probability theory things like Dini Derivative and Prekopa-Borel theorem etc which is way beyond the amount of math I know.

6. Matching Theory

Definition: Stable Matching

Sets A, B where:

- $|A| = |B|$
- each $a \in A$ has an ordering on B say Π_a
- each $b \in B$ has an ordering on a say Π_b

Goal To find a stable matching ie each $a \in A$ is paired with $b \in B$ and vice versa and $\nexists$ a blocking pair.

Definition: Blocking Pair

(a, b) is a blocking pair for matching M if a prefers b over $M(a)$ and b prefers a over $M(b)$ as per Π_a and Π_b.

A simple greedy algorithm for this is the deferred acceptance algorithm.

Deferred Acceptance or Gale-Sharply 6.1. Each round:

1. Boys who are without a provisional match propose to the their top girl who is yet to reject them
2. Each Girl rejects all but the favorite offer received that round + her provisional match.
3. Each Boy crosses off the rejecting girls from their list.
4. Repeat till everyone has a provisional match. These are final matches.

Deferred Acceptance or Gale-Shapley

```
$M = \emptyset$
while ($\exists$ an unmatched $a \in A$ who has not yet proposed to some $b \in B$)“
    $a$ proposes to their most preferred $b \in B$ as per $\Pi_a$
        if $b$ is not matched:
            $b$ accepts $a$, $M$.insert($(a, b)$)
        else:
            if $M(b) = a'$ and $b$ prefers $a$ over $a'$:
                $M$.remove($(a', b)$)
                $M$.insert($(a, b)$)
            else:
                $b$ rejects $a$
```

Proving the termination of the algorithm is trivial as all agents prefer being matched then unmatched and no proposal is made twice. This also guarantees execution in $O(n^2)$ time. The proof of correctness follows as:

Proof of Correctness:.

Lemma 6.2. *For all women, their provisional match in round $t+1$ is better or equal to match in round t.*

Proof of Lemma. Obvious by induction. ■

FTSOC, let DAA scheme match (m, w) and (m', w'), which is unstable. As m prefers w' more then w, it must have proposed w'. But in a later round, w' has a worse match. This contradicts the above lemma, thus, the proof of correctness follows from contradiction. ■

Definition: Stable Partner

b is a stable partner of a if there exists a stable matching M such that $(a, b) \in M$.

Theorem 6.3. *The output M of DA gives every $a \in A$ hus best possible stable partner.*

Proof.

Lemma 6.4. *No stable partner rejects a*

Proof. We proceed the contradiction.

Let $(a, b) \in M$ and $(a, b') \in M'$ such that

$$a : ...b'...b...$$

and rejection of a by b' be the first rejection of a stable partner.

For M to be stable, b' must prefer $M(b) = a'$ over $M'(b) = a$.

$$b : ...a'...a...$$

For M' ot be stable, a' must prefer $M'(a')$ over $M(a')$. But then a' must have been rejected by $M'(a')$ before a was rejected by b' in DA.

This contradicts the fact that rejection of a by b' be the first rejection of a stable partner.

Thus, by infinite descent, we have a contradiction. ■

Bu the lemma, no stable partner rejects in DA. Thus, output of DA is such that every $a \in A$ hus best possible stable partner. ■

A similar proof will show:

Theorem 6.5. *The output M of DA simultaneously is such that $b \in B$ has the worst possible stable partner.*

Remark

Hannah Fry in "The Mathematics of Love" remarks on the above two theorems:

This result does make some intuitive sense. If you put yourself out there, start at the top of the list, and work your way down, you'll always end up with the best possible person who'll have you. If you sit around and wait for people to talk to you, you'll end up with the least bad person who approaches you. Regardless of the type of relationship you're after, it pays to take the initiative.

The difference in outcomes between those who do the asking and those who wait to be asked is particularly important when the stable marriage problem is applied beyond imaginary couples at a party: something the US government found out the hard way. Through the National Resident Matching Program, the US government has been using the Gale Shapley algorithm to match doctors to hospitals since the 1950s. Initially, the hospitals did the "proposing." This gave the hospitals the students they wanted, but didn't work well for doctors who had to move halfway across the country to accept their least bad offer. It meant the system ended up full of unhappy doctors and, hence, unhappy hospitals. The organizers gave doctors the role of proposer when they found that out.

But it's not just hospitals and Friday-night action. The Gale-Shapley matching algorithm has been exploited in a host of real-world scenarios: dental residencies, placement of Canadian lawyers, assignment of students to high schools, and the sorority rush. It's so useful that there is a huge amount of academic literature

> dedicated to investigating a range of extensions and special cases—many of which still apply to the original dating problem.
>
> Mathematicians have adapted the method to allow both men and women to approach either gender simultaneously, and changed the rules to include ties in preference lists, or scenarios where you'd rather go home alone than hook up with the weird guy in the corner. Academics have even explored what happens when you have cheating men (not cheating women, though, strangely).
>
> The math in these special cases can get quite heavy in places (although there are lots of lovely references at the end of this book if you're interested in finding out more). But for all the extensions and examples, the message remains the same: If you can handle the occasional cringe-inducing rejection, ultimately, taking the initiative will see you rewarded. It is always better to do the approaching than to sit back and wait for people to come to you. So aim high, and aim frequently: The math says so.

Theorem 6.6. *Let M_1, M_2 be two stable matchings. Form a set S by assigning each $a \in A$ his more preferred partner from M_1, M_2.*

S is a stable matching.

Let's now talk about the mechanism design aspects of this algorithm

Theorem 6.7. *DA is truthful for the proposing side by not for the receiving side.*

Proof. Suppose $a \in A$ submits a false list and get's a better partner.

Let M be the DA matching with true lists and M' be the DA output with A falsifying the list and $M'(a) = b' \underset{a}{\succ} b = M(a)$.

Let $M(b') = a'$. This means $M'(a') \underset{a'}{\succ} M(a')$ as otherwise (a', b') block M'.

Similarly now, $M(M'(a'))$ will get a better partner and so on. But, we can't repeat b at any point as that would contradict M's stability by the no stable partner rejection lemma.

But as the sets are finite, this leads to a contradiction.

We can just make a counter example to show the second part of the statement.

TODO. *The counterexample*

■

Remark

The socially maximal matching is NP complete to find via a reduction from Quadratic Programming.

However, Quadratic programming is solvable by Quantum Computers in poly-time; hence, we can solve socially maximal matching on a Quantum computer. It's implementation (named MIT-Qute, to be pronounced Meet Cute) in Quiskit came in second at MIT's Quantum Hackethon. You can see it here.

Remark

This generalizes to Stable Roommates problem. An $O(n^2)$ algorithm for it exists and was given by Robert Irving. I had implemented it in Haskell (in $O(n^2 \log(n))$ to prevent use of Hash maps. With hash maps, it is $O(n^2)$) and the code can be found

For more data in this regard, check out Grusfield and Irving's "Stable Matching: Structure and Algorithms".

6.1. House Allocation

Problem 6.8. There are N agents and each agent has a house. Each agent also has a preference ordering over all houses.

Goal: Design an exchange mechanism so that the resulting matching allocation Π is stable.

Example

$$\begin{pmatrix} a: & h_b & h_c & h_a \\ b: & h_c & h_a & h_b \\ c: & h_a & h_b & h_c \end{pmatrix}$$

In this case the permutation $\Pi = h_b, h_c, h_a$ makes everyone happy.

But

$$\begin{pmatrix} a: & h_b & h_c & h_a \\ b: & h_a & h_c & h_b \\ c: & h_a & h_b & h_c \end{pmatrix}$$

Here, as $\Pi = h_b, h_c, h_a$ is not gonna work as a, b can make a blocking coalition switching their houses.

Definition: Stable Allocation

Π is stable if there is no blocking coalition with respect to Π

Definition: Blocking Coalition

Given Π, if there i a subset of agents $A \subseteq N$ such that $\exists \sigma : A \to A$ such that $\forall i \in A, \sigma_i \succeq \Pi_i$ and $\exists i \in A, \sigma_i \succ \Pi_i$.

Top Trading Cycle Algorithm

```
N' = N
while $N' \neq \emptyset$
    Construct a graph where $(i, j) \in E \iff h_j$ is $i's$ most favorite house among $j \in N'$.
    If $\exists$ a cycle $C = \langle i_1, ..., i_k \rangle$:
        perform an exchange $\Pi_{i_1} = h_{i_2}, ..., \Pi_{i_k} = h_{i_1}$
        Delete $i_1, ..., i_k$ from $N$.
return the resulting allocation $\Pi$
```

Detecting the cycle takes $O(V + E)$ time, here, every agent(vertex) only has one edge, hence $O(V + V) = O(V)$. Every turn, atleast 1 agent is removed from the list. This gives a worst case

$$O(N) + O(N - 1) + ... + O(N) = O(N^2)$$

complexity.

Theorem 6.9. Π *is a stable allocation*

Proof. FTSOC let Π, outputted by TTC is unstable. Thus, $\exists A \subseteq N$ and $\sigma : A \to A$ such that $\forall i \in A, \sigma_i \geq \Pi_i$ and $\exists i \in A, \sigma_i > \Pi_i$.

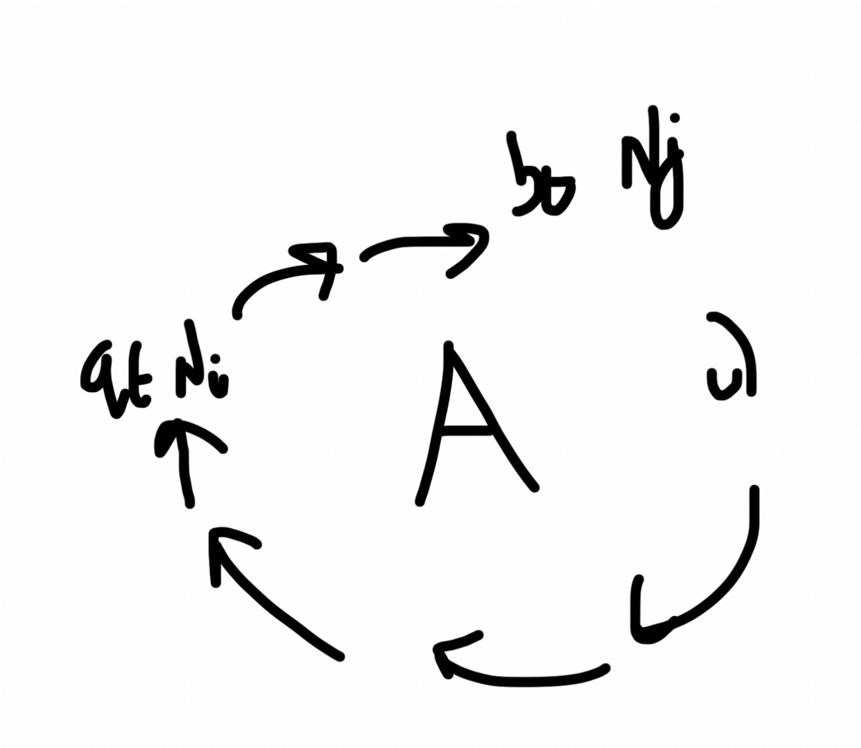
We begin by partitioning the agents into iterations based on the round they were alloted their house.

$$\begin{gathered} N_1 : \text{Agents that get house in round 1} \\ N_2 : \text{Agents that get house in round 2} \\ \vdots \\ N_t : \text{Agents that get house in round } t \\ \vdots \end{gathered}$$

Notice, $A \subseteq N_t$ for some t is not possible as all the agents got their best house in $N \setminus (N_1 \cup ... \cup N_{t-1})$ by TTC. Thus, no coalition is formed.

Thus, A has to be over two of the parts of the partition. Let $a, b \in A$ such that $a \in N_i$ and $b \in N_j$, $i < j$.

$$i = \min k \text{ s.t. } A \cap N_k \neq \emptyset$$



TODO. *Make bad image → good image*

In Π, a gets top choice from $N \setminus (N_1 \cup \ldots \cup N_{i-1})$

$$\therefore \Pi_a > \sigma_a = h_b \text{ for } a$$
$$\Rightarrow a \notin A$$

■

Theorem 6.10. Π *is unique stable allocation.*

Proof.

Idea

Let Π' be another stable allocation. We can get this by inducting on the number of iterations. If Π, Π' are same upto to t iterations, it will be same in $t + 1$.

■

Definition: Strategyproof

By Strategyproofness, here we mean that no agent can get a better house in Π by submitting a false list.

Theorem 6.11. *The TTC algorithm is strategyproofness.*

Proof. An agent can put a false preference, putting up a false outgoing edge but not a false incoming edge. Hence, they can't get anything better in an earlier round and thus, can't get better by lying. ■

This makes the algorithm quite robust, despite being simple.

Remark

The problem has been studied in a number of settings.

- Envyfreeness : Defining envy as $i \underset{\text{envy}}{\to} j$ if $\Pi_i \underset{i}{\prec} \Pi_j$, can we allocate in an envy free fashion?
- Popular: An allocation Π is popular, if for any other allocation σ, atleast 50% agents prefer Π over σ.

6.2. Kidney Exchange

Problem 6.12. There are patients who need kidney transplant and there are donors willing to donate a kidney.

However, patient-donor pairs may be incompatible. In that case, sometimes we can exchange donors to get compatible donations. We also don't want huge cycles as some donor once their patient receives a donation, they can walk out.

The last condition is there as a hospital can do only a number of transplants at once. The threshold we will see is 2 as taking 3 makes the problem NP hard (although approximation algorithms exist).

Possible Solution 1

Treat patients as agents, donors as houses and use TTC.

This solution fails due to the walk out risk, making large cycles impractical. Second, preferences are binary and not a total order[10].

Notice, we can't just do the maximal bipartite matching as $d_1 \to p_2 \Rightarrow d_2 \to p_1$ so that algorithm doesn't work as is.

[10] In biology, we actually could have a total order. We can use the tissue match percentage or edit distance on the DNA etc.

Solution

We make a graph $G = (V, E); V = \{(p_i, d_i) \mid i \in [n]\}, E = \left\{(i, j) \mid d_i \to p_j, d_j \to p_i\right\}$.

We want to find a maximum matching in G. We define a priority order on patients $1, …, n$.[11]

Kidney Matching

```
$M_0$ = set of maximum matching in $G$
for $i \in [n]$ :
    $z_i$ = matchings in $M_{i-1}$ that match $i$
    if $Z_i \neq \emptyset$
        $M_i = z_i$
    else
        $M_i = M_{i-1}$
output an arbitrary matching from $M_n$
```

Claim 6.13. The mechanism is DSIC

The proof follows from the above mechanism being truthful that is no agent has incentive to not declare a matching.

Remark

Three way exchanges are practical but finding maximum number of disjoint 3-cycles is NP-complete.

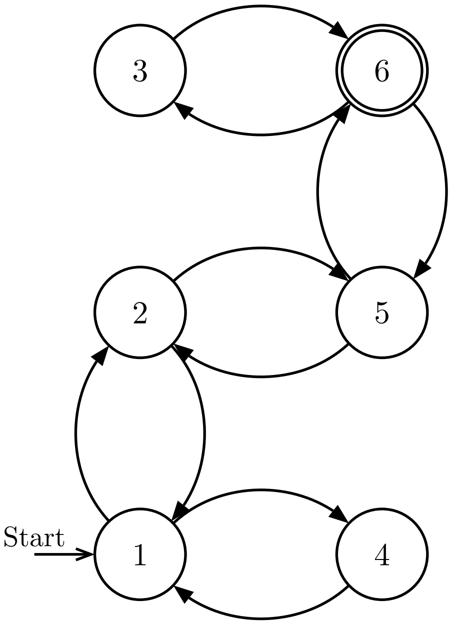
Empirical results show that 4 and larger cycles don't drastically improve the number of exchanges or people getting kidneys.

6.2.1. Truthfulness by Hospitals

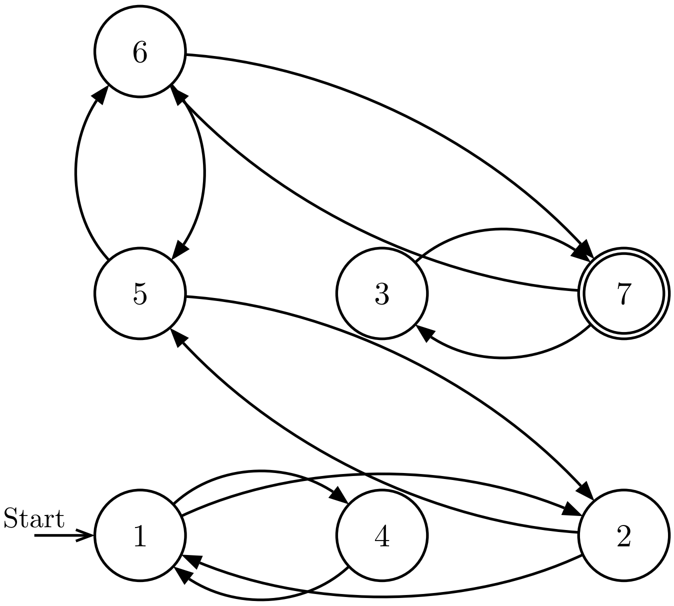
Consider the case

[11] This may sound unethical but

1. Some patients have less survival odds if they don't get kidney fast enough.
2. We might have multiple maximum matchings in G and have to choose 1 in an unbiased/truthful fashion.



In this case, if hospital 1 has patient $1, 2, 3$ and hospital 2 has patients $4, 5, 6$; then there is incentive to perform $1 \leftrightarrow 2$ and $5 \leftrightarrow 6$ which will severely undermine the system!



Similarly in this case, Hospital 1 has an incentive to not report $(1, 2)$ so that 3 always gets matched.

6.3. Matching Markets

Definition: Popular Matching

Given agents and items where agents have an ordering on items (possibly incomplete). Given matching M_1 and M_2, M_1 is more popular than M_2 if more agents prefer M_1 over M_2.

> A popular matching is M such that $M \succeq M'$ for $M' \in \mathcal{M}$ where $\mathcal{M}$ is the set of all matchings.

Input: Given $G = (A \cup B, E)$ where each $a \in A$ has a preference order over it's neighbors in B.

We want to find a matching M such that $\nexists n \in \mathcal{M}$ more popular than M.

Notice, this is not guaranteed to exist as

$$\begin{pmatrix} a_1 : b_1 & b_2 & b_3 \\ a_2 : b_1 & b_2 & b_3 \\ a_3 : b_1 & b_2 & b_3 \end{pmatrix}$$

Notice, $(b_1, b_2, b_3) \prec (b_3, b_1, b_2) \prec (b_2, b_3, b_1) \prec (b_1, b_2, b_3)$ and we have a cycle. In this case, we don't have a popular matching.

The following set of claims to some extent will provide a characterization of popular matching.

Claim 6.14. A popular matching has to maximum on top-choice edges

Proof. FTSOC, let there be a popular matching that is not maximum on the top-choice edges.

That means $\exists a_i, a_j, g_i, g_j$ such that a_i's first choice is g_i and a_j's first choice is g_j but $a_i \to g_j$ and $a_j \to g_k$ and some $a_k \to g_i$.

In this case, we can match $a_i \to g_i$ and $a_j \to g_j$ and $a_k \to g_k$ which atmost degrades 1 agent while making 2 agents better.

Thus, this can't be popular and hence, we have a contradiction! ■

Notice, the converse is not necessarily true.

Claim 6.15. For every a, adding $l(a)$ the last choice of a for consistency (is last is a's preference but is on the list. Is on nobody else's list);

If we let $f(a)$ be the first choice of a and $s(a)$ be the first choice among those items which are nobody's first choice then:

A popular matching must matches each agent to either $f(a)$ and $s(a)$.

Proof. FTSOC, let there be a popular matching that has some agent a matched to neither $f(a)$ or $s(a)$.

This means there exists a_i, a_j, g_i, g_j such that $g_i = s(a_i)$ and $g_j = f(a_j)$ but $a_i \to g_j$ and $a_j \to g_k$ and some $a_k \to g_i$.

We can match $a_i \to g_i$, $a_j \to g_j$ and $a_k \to g_k$ which is better as a_i is matched to $s(a_i)$, a_j is matched to $f(a_j)$.

Thus, we have a contradiction. ■

Claim 6.16. Any matching that has a maximum number of top choice edges and matches each a to $f(a)$ or $s(a)$ is a popular matching.

Popular Matchings

```
Construct $G'$
    f-edges: for each $(a, b)$ where $b$ is first choice of $a$ in $G$:
    s-edges: $(a, b)$ where $b$ is not the first choice of nobody else in $G$. Let $b = b_a$, a dummy vertex if no such vertex exists.
Find a max matching on $f$-edges.
Add $s$-edges and augment M_1.
```

TODO. *Copy from Kelly for GT definitions!*

Notice, the algorithm runs in $O(m + n)$ time.

6.4. Strategic Concerns

Notice, our algo is not strategyproof!

Example : Perfect Matching is not Strategyproof

Consider the true preferences

$$\begin{pmatrix} a_1 : p_1 \;\; p_2 \\ a_2 : p_1 \;\; p_2 \;\; p_5 \\ a_3 : p_2 \;\; p_3 \\ a_4 : p_2 \;\; p_4 \end{pmatrix}$$

We get the matching $(a_1, p_1), (a_2, p_5), (a_3, p_2), (a_4, p_4)$.

However, by lying

$$\begin{pmatrix} a_1 : p_1 \;\; p_2 \\ a_2 : p_2 \\ a_3 : p_2 \;\; p_3 \\ a_4 : p_2 \;\; p_4 \end{pmatrix}$$

We get the matching $(a_1, p_1), (a_2, p_2), (a_3, p_3), (a_4, p_4)$.

The first half of this material comes from Abraham, Irving, Kavitha, Melhorn 2003. The strategic part comes from Popular Matchings : Structure and Strategic issues, Narse 2013. The latter shows a linear time algo that given everyone's ordering, gives the best falsification to make.

The problems of manipulation by a coalition is open[12]

Remark

In case of tied preferences, we can modify the above algorithm with 'galois' decomposition.

6.5. Perfect Matching with double sided preferences

Definition

Consider $G = (A \cup B, E)$, each $v \in A \cup B$ has a preference ordering over its neighbors.

We want to find a popular matching in G

Theorem 6.17. *A popular matching always exists and is computable in $O(m+n)$ time where $n = |V|$ and $m = |E|$.*

Theorem 6.18.

- *Every stable matching is popular.*
- *A stable matching is min-size popular*
- *All stable matchings have the same size*

Computation of max size popular matchings

```
1  Execute the Gale-Shapley algorithm
2  Unmatched vertices on the proposing side are promoted to a $*$ status
3  The $*$-vertices propose again (from top of their lists) similar to Gale Shapley over a "normal" one
```

Example

Consider

$$\begin{pmatrix} a_1 : b_1 \;\; b_2 \\ a_2 : b_1 \qquad \\ b_1 : a_1 \;\; a_2 \\ b_2 : a_1 \qquad \end{pmatrix}$$

Running GS directly gives (a_1, b_1). But if we run our star algorithm, a_2^* proposes to b_1 and then a_1 proposes to b_1 to get$(a_1, b_2), (a_2, b_1)$.

[12] Was in the 2013 paper, Prof. Prajakta is not sure if it still is.

We will not get into the proof but here are a few interesting things.

Theorem 6.19. *The algo gives a max size popular matching and it is* $\geq \frac{2}{3}$ *size of maximum matching.*

Interestingly, giving 2 or more stars doesn't lead to popular matchings but leads to larger matchings who are popular in their size. This is a paper by Kavitha titled "A size popularity trade off in the stable marriage setting" in 2015.

This algorithm is quite general and hence, can be extended to a lot of cases. For example, we can use it in matching kids to collages.

THE END