Algorithmic Game Theory

Britta Peis

RWTH Aachen, Lehrstuhl für Management Science

Edited by Daniel Schmand, and Andreas Tönnis

Summer Term 2016

Contents

Chapter 1

Introduction

In this course, we will investigate models, concepts and algorithms for mechanism design, equilibrium computation in strategic games, and fair cost-sharing rules in cooperative games. This chapter gives an overview about the different topics covered throughout the course.

1.1 Mechanism Design: The Science of Rule-Making

One of the central question in the area of mechanism design is the following.

How should systems with strategic participants be designed so that they have good performance guarantees?

1.1.1 Example: Badminton and the science of rule-making.

Here is an example that shows that a failed tournament design for women's badminton at the Olympics 2012 in London led to one of the biggest scandals of the event [7]. The tournament design failed since it did not carefully consider the incentives of the participants. The rules of the tournament are similar to the one from World Cup soccer: There are four groups (A,B,C,D) of four teams each. The tournament consists of two phases. In the first "round-robin" phase, each team plays the other three teams in its group, and does not play teams in other groups. The top two teams from each group advance to the second phase, the bottom two teams from each group are eliminated. In the second phase, the remaining eight teams play a standard "knockout" tournament: there are four quarterfinals (with the losers eliminated), then the two semi-finals (with the losers playing an extra match to decide the bronze medal), and then the final (the winner gets the gold, the loser the silver).

The incentives of the participants and of the Olympics committee (and the fans) are not necessarily aligned in such a tournament. Certainly, the

teams want to get as good a medal as possible, of course. But what does the Olympics committee want? They didn't seem to think carefully about this question. However, they should want every team to try their best to win every match. Why, one could ask, should a team ever want to lose a match?

To understand the incentive issues, we need to explain how the eight winners of the round-robin phase were paired up in the quarterfinals: The team with the best record from group A plays the second best team from group B in the first quarterfinal; similarly with the best team from group B and the second best from group A in the third quarterfinal; and similar with the top two teams from groups B and D in the second and fourth quarterfinals.

Here is what happened: On the last day of round-robin competition, there was a shocking upset: the Danish team of Pederson and Juhl (PJ) beat the Chinese team of Qing and Wunley (QW), and as a result PJ won group D with QW coming in second (so both teams advanced in the knockout phase). The first controversial match involved another team from China, Xiaoli and Yang (XY), and the South Korean team of Kyung-eun and Hana (KH). Both teams had a 2-0 record an group A play. Thus, both were headed for the knockout stage, with the winner and loser of this match the top and second-best team from the group, respectively. Here was the issue: the group-A winner would meet the fearsome QW team (which came in only second in group D) in the semifinals of the knockout tournament (where a loss means a bronze medal at the most), while the second-best team in group-A would not have to face QW until the final (where a silver medal is guaranteed). Both, the XY and KH teams found the difference between these two scenarios significant enough to try to *deliberately lose the match*[1]. This unappealing spectacle led to scandal, derision, and, ultimately, the disqualification ot the XY and KH teams, as well as two group C teams (from Indonesia and another team from South Korea) that used the same strategy, for the same reason[2].

1.1.2 The design of rules matters.

The point is that, in systems with strategic participants, the *rules matter*. Poorly designed systems suffer from unexpected and undesirable results. But, can we blame the badminton players for acting to maximize their medal placements? Instead, we should blame the rule designers to not have taken the incentives of the strategic participants carefully enough into

[1] That the teams feared the Chinese team QW far more than the Danish team PJ seems justified in hindsight: PJ were knocked out in the quarterfinals, while QW won the gold medal.

[2] If you want to see how such a game where both teams prefer to loose looks like, I encourage you to track down the video from YouTube.

considerations!

There is a well-developed science of rule-making, the field of *mechanism design*. The goal in this field is to design rules so that strategic behavior of the participants leads to a desirable outcome. Applications of mechanism design, which we will discuss later in class, include Internet search auctions, wireless spectrum auctions, matching medical residents to hospitals, and kidney exchange markets.

We will study some basics of the traditional economic approach to mechanism design, along with several complementary contributions from computer science that focus on computational efficiency, approximate optimality, and robust guarantees.

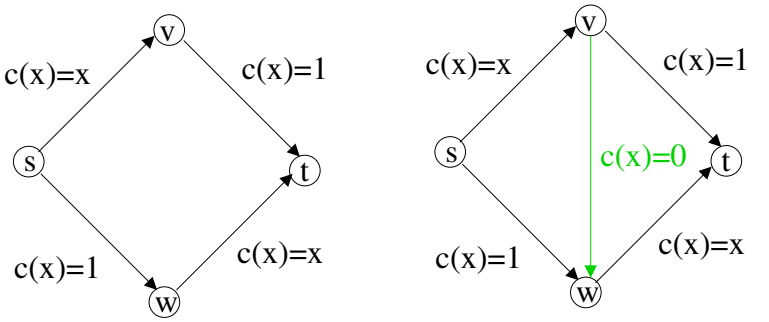
1.2 Equilibria in Strategic Games

We don't always have the luxury of designing the rules of a game from scratch. Instead, we would like to understand a game that occurs "in the wild" – the Internet or a road network for example. Some of the central questions in strategic game theory are

When is selfish behavior essentially benign? How do selfish players reach an equilibrium (and do they)?

1.2.1 Braess's Paradox

For a motivating example, consider Braess's paradox (Figure 1.1)



(a) Initial network

(b) Augmented network

Figure 1.1: Braess's Paradox

There is a suburb s, a train station t, and a fixed number of drivers who wish to commute from s to t. For the moment, assume two non-interfering

routes from s to t, as shown in Figure 1.1 (a). Each of the two routes consists of two roads, one of which is a long and wide road with travel time of one hour, no matter how much traffic uses it, and the other one is a short narrow road on which the travel time in hours is equal to the fraction of traffic using it. The combined travel time in hours of the two roads (edges) on one of these routes is $1+x$, where x is the fraction of traffic that uses the route. The routes are therefore identical and traffic should split evenly between them. In this case, all drivers arrive at their destination 90 minutes after their departure from s. Now, suppose we install a teleportation device allowing drivers to travel instantly from v to w. The new, augmented network is shown in Figure 1.1 (b), with the teleporter represented by edge (v, w) with constant cost $c(x) = 0$, independent of the road congestion. How will the drivers react?

We cannot expect the previous traffic pattern to persist in the new network. The travel time along the new route $s \to v \to w \to t$ is never worse than that along the two original paths, and it is strictly less whenever some traffic fails to use it. We therefore expect all drivers to deviate to the new route. Because of the now heavy congestion on the edges (s, v) and (w, t), all the drivers now experience two hours of travel time when driving from s to t. Braess's paradox thus shows that improvement of the infrastructure may in fact lead to higher travel time for all drivers!

Braess's paradox shows that selfish routing does not minimize the commute time of the drivers – in the network in Figure 1.1 (b), a central authority ("dictator") could dictate routes to drivers and improve everyone's commute time by 25%.

1.2.2 The price of anarchy (POA)

We define the *price of anarchy (POA)* to be the ratio between the worst system performance with selfish players and the best possible system performance. For the network in Figure 1.1 (b), the ratio between 2 and $\frac{3}{2}$, i.e. $\frac{4}{3}$.

The POA was first defined and studied by computer scientists. Every economist and game theorist knows that selfish behavior in general leads to bad, inefficient outcomes, but until the 21st century there had been almost no attempts to quantify such inefficiency in different application domains. We study applications in which the POA is bounded (sometimes close to one) so that selfish behavior leads to near-optimal outcomes. Applications include network routing [12], scheduling, resource allocation, and simple auction design.

1.2.3 Complexity of Strategic Games

Some games are easy to play. For example, in the augmented network of Braess's Paradox (Figure 1.1 (b)), it is clear that every individual uses the teleporter– it is the best route, no matter what the other drivers do. In many other games, like the first-price auctions mentioned in the next lecture, it is much harder to figure out how to play [3].

How do strategic players reach an equilibrium? Or does an equilibrium exist at all?

Informally, an equilibrium is a "steady state" of a system where each participant, assuming everything else stays the same, wants to remain as-is. (Please do not use the definition of a Nash equilibrium as explained in the movie *A Beautiful Mind*. Instead, use the definition below).

In most games, the best action to play depends on what the other players are doing. Rock-Paper-Scissors, rendered below in "bimatrix" form, is a canonical example.

Example: Rock-Paper-Scissors

In the Rock-Paper-Scissors game, one player chooses a row and, simultaneously, the other player chooses a column of the following matrix:

	Rock	Paper	Scissors
Rock	0,0	-1,1	1,-1
Paper	1,-1	0,0	-1,1
Scissors	-1,1	1,-1	0,0

The numbers in the corresponding matrix entry are the payoffs for the row and column player, respectively. There is certainly no "pure equilibrium" in the Rock-Paper-Scissors game: whatever the current state, at least one player would benefit from a unilateral move.

This bimatrix game belongs to the class of *strategic games in standard form* which are more precisely described as follows:

Strategic games (description). A *strategic game (in standard form)* is given by a finite set $N = \{1, \ldots, n\}$ of players, and, for each player $i \in N$, a finite set X_i of strategies and a utility function u_i. After each player i has chosen one of his strategies $x_i \in X_i$, we arrive at a situation (also called "outcome", "profile", or "action") $x = (x_1, \ldots, x_n)$. It is assumed that the players choose their strategies simultaneously, i.e., without knowing about the strategy choices of the other players. The utility function u_i is defined on the set $X := X_1 \times \ldots \times X_n$ of all possible outcomes and reflects player i's preferences among the different outcomes of the game. Thus, a strategic game is given by a tuple of the form $(N, \{X_i\}_{i \in N}, \{u_i\}_{i \in N})$.

Depending on the situation to be modeled, the utility-functions u_i are called "payoff-" or "cost functions". For example, in the Braess's paradox, each player chooses a route from s to t as strategy. The cost of the chosen route is the travel time which highly depends on the strategy (route) choices of the remaining players. Certainly, each player wants to minimize this travel time. In contrast, in the Rock-Paper-Scissors game, the matrix entries reflect the payoffs which each player aims to maximize. It should be clear from the context whether the game under consideration is a cost- or benefit game.

Definition 1.1 (Pure Nash equilibrium (PNE))**.** *Given a strategic game* $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$*, an outcome* $x \in X$ *is called a* pure Nash equilibrium (PNE) *if for each player* $i \in N$ *the value* $u_i(x)$ *is not worse than any value* $u_i(x_{-i}, x_i')$ *where* $x_i' \in X_i$ *is any alternative strategy choice for player* i*, and* $x_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ *is the vector consisting of the strategies in* x *chosen by the other players* $j \in N \setminus \{i\}$*.*

Game theory asks for which classes of games equilibria can be guaranteed. The Rock-Paper-Scissors example shows that even very simple games do not necessarily admit PNE.

1.2.4 Randomized Games

A crucial idea, developed largely by von Neumann, is to allow *randomized* (a.k.a. *mixed*) strategies. In fact, in a game like Rock-Paper-Scissors, your opponent is effectively randomizing. If both players randomize uniformly in Rock-Paper-Scissors, then neither player can increase his expected payoff via a unilateral deviation. A pair of probability distributions with this property is a *(mixed-strategy) Nash equilibrium*. Remarkably, with randomization, *every* strategic game in standard form has a Nash equilibrium (see Nash's theorem, which will be covered later in class).

We will furthermore see, that in the special case where the game is *zero-sum*– meaning that the pair of payoffs in each entry sums to zero, like in Rock-Paper-Scissors– then a mixed-strategy Nash equilibrium can even be calculated very efficiently using linear programming.

A far more recent result (mid-last decade) is a negative one: computer scientists have shown that (under suitable complexity assumptions), there is *no* polynomial time algorithm for computing a Nash equilibrium in general (non-zero-sum) games. While the problem is not NP-hard (unless NP=coNP), it is sometimes called "PPAD-hard", which we will explain and interpret later in the course (at least for CS-students).

1.3 Fair Cost Sharing Methods in Cooperative Games

In strategic games, as described above, the players play against each other, trying to maximize their individual payoff/benefit, resp. to reduce their individual costs.

In contrast, in cooperative games the players in N are willing to cooperate in order to achieve a better payoff in total. The central question in cooperative game theory is

How should the total benefit/cost be shared among the players in a fair and stable manner?

Definition 1.2. *A **cooperative game** consists of a finite set N and some function $v : 2^N \to \mathcal{R}$ which assigns every subset of players $S \subseteq N$ a value $v(S) \in \mathbb{R}$.*

Throughout, we let $\mathbb{R}$ denote the set of real numbers. The function $v : 2^N \to \mathcal{R}$ is called *characteristic function* of the game (N, v). Depending on the interpretation, function v is also called *benefit- or cost-function* of game (N, v).

In a cooperative benefit game (N, v) the value $v(S)$ can be interpreted as the *benefit achieved by coalition $S \subseteq N$*, while in a cooperative cost game, $v(S)$ stands for the *cost caused by coalition $S \subseteq N$*.

Remark: We assume $v(S)$ to be given by an oracle, i.e. we do not worry (here) about the computability of $v(S)$.

1.3.1 Example: Owen's Production Game.

A set N of n firms (the players) produce k different types $T_1, \ldots, T_k$ of a certain product (e.g., cars, houses, cakes, etc.) The basic ingredients to produce these types are $G_1, \ldots, G_m$ (e.g., wheels, windows, eggs, etc.). Firm $p \in N$ owns b_{ip} units of ingredient G_i. A quantity of a_{ij} units of G_i are necessary in order to produce one unit of type $T_j, j = 1, \ldots, k$. The market price of one unit T_j is $c_j \geq 0$. The players/firms are discussing whether they should collaborate in order to increase the total benefit. Together, the players in coalition $S \subseteq N$ can achieve a total value of

$$v(S) := \max\{\sum_{j=1}^{k} c_j x_j \mid \sum_{j=1}^{k} a_{ij} x_j \leq \sum_{p \in S} b_{ip} \forall i = 1, \ldots, m,\ x \geq 0.\}$$

(Depending on the product types, it might be necessary to require integrality of x.) How should the total benefit $v(N)$ be allocated among the players in N in a fair and stable manner?

1.3.2 Example: Facility Location Game

In the *facility location game* we are given a set $N = \{1, \ldots, n\}$ of clients ("the players"), and a set $\mathcal{F}$ of facilities. Each facility $j \in \mathcal{F}$ has an opening cost f_j. Furthermore, for each pair $\{i, j\}$ consisting of a client $i \in N$ and a facility $j \in \mathcal{F}$, there is a connection cost of c_{ij} that needs to be paid in order to connect i to j. For each coalition $S \subseteq N$ of clients, the value $v(S)$ denotes the minimum cost to connect all clients in S to open facilities, i.e.,

$$v(S) := \min_{F \subseteq \mathcal{F}} \{ \sum_{j \in F} f_j + \sum_{i \in S} \min_{j \in F} c_{ij} \} \quad \forall S \subseteq N.$$

How should the total cost $v(N)$ that needs to be paid in order to connect *all* clients to open facilities be shared among the clients?

Throughout the course, we will study various solution concepts to share the total costs or benefit in a fair and stable way. It turns out, that linear programming plays a crucial role in the design and analyse of these solution concepts.

Chapter 2

Mechanism Design: The Science of Rule Making

2.1 Single Item Auctions

Consider the following situation. A seller has a single good, e.g. an antiquated playmobile figure, or a worthful painting. This is the setup in a typical eBay auction, for example. There is a set $N = [n]$ of bidders who are potentially interested in buying the item. How do the bidders behave? Which bid should they give?

The answers to these questions depend on

a) what the bidders want, i.e., how valuable the good is for the single bidder, and

b) how the auction is designed, i.e., what are the rules that decide which bidder will win the item and for which price.

We make the following assumptions.

(A1) each bidder $i \in N$ has a *valuation* $v_i \in \mathbb{R}_+$ which can be interpreted as his *willingness-to-pay* for the item being sold,

(A2) the valuation v_i is *private*, i.e., only known to bidder i.

(A3) the *utility* of bidder i is 0 if he loses the auction, and $v_i - p$ if he wins the auction for a price of p.

The utility model under assumption $(A3)$ is called *quasi-linear utility model*, and we will focus on this in this course.

Here, we consider so-called *sealed-bid auctions* in which

(1) each bidder i privately communicates a bid b_i to the auctioneer (like in a sealed envelope),

(2) the auctioneer decides who gets the good (if anyone gets it at all), and

(3) the auctioneer decides on a selling price p.

Note that the bids b_i do not necessarily correspond to the true valuation v_i. A strategic bidder might lie about his bid in order to increase his utility.

An obvious way to implement (2) is to give the item to the highest bidder. For now, this will be the only winner selection rule which we consider. However, for revenue maximization, for example, there are other winner selection rules that are important. But how should we implement step (3)? There are multiple reasonable ways to implement step (3) and the implementation significantly affects bidder behavior. Let's consider a few examples of charging methods:

For example, we could charge nothing. But then the auction develops into a game of "who can name the highest number?" which is not what the auctioneer and the bidders desire.

2.1.1 First-Price Auctions

In a first-price auction, the winner is charged its bid. Such first-price auctions are commonly used in practice. There are, however, a couple of disadvantages: As a bidder, it is hard to figure out how to bid. As a seller, it is hard to predict what happens. A first price auction may lead to strategical manipulation as soon as one knows the valuations (at least partially) of the remaining bidders.

2.1.2 Second-Price Auctions (or Vickrey Auctions)

In a *second-price auction* (also called *Vickrey auction*), the highest bidder wins the item and pays a price that is equal to the second-highest bid. This auction is also commonly used in practice. For example, consider an eBay auction: if you bid 50 Euro and win, do you really pay 50 Euro? Not necessarily–eBay uses a "proxy bidder" that increases your bid on your behalf until your bid is reached, or until you are the highest bidder (whichever comes first). Hence, if you win an eBay auction, the sale price equals the highest other bid, plus a small increment.

Lemma 2.1. *In a second-price auction, every bidder has the dominant strategy to bid truthfully, i.e., to set the bid $b_i = v_i$. That is, the strategy choice $b_i = v_i$ maximizes the utility of i, no matter what the other players do.*

Note that an auction with the property that truthful bidding is a dominant strategy is easy to participate in. As a bidder, you don't need to reason about the other players.

Proof. Fix an arbitrary bidder i, his valuation v_i, and the bids b_{-i} of the remaining bidders. We assume that ties are broken in favor of bidder i. (Exercise: Why does the proof also work without that property?) We need to show that

$$u_i(v_i, b_{-i}) \geq u_i(b_i, b_{-i}) \quad \forall b_i \in \mathbb{R}.$$

Let $B := \max_{j \in N \setminus \{i\}} b_j$ denote the highest bid of the other bidders.

- If $b_i < B$, then i loses and receives utility 0.
- If $b_i \geq B$, then i wins at price B, so the utility is $v_i - B$.

Now, consider the case where $v_i < B$. Then the highest utility i can get is $\max\{0, v_i - B\} = 0$, and this can be achieved by bidding truthfully (and losing). In case $v_i \geq B$ the highest utility that bidder i can get is $\max\{0, v_i - B\} = v_i - B$, and he achieves this by bidding truthfully (and winning). □

A second important property of second-price auctions is that a truthful bidder will never regret participation in the auction.

Lemma 2.2. *In a second-price auction, every truth-telling bidder is guaranteed non-negative utility.*

Proof. In a second-price auction, a loser will get utility 0, while a winner will get utility $v_i - p$, where p is the second-highest bid. Since i has the highest bid, and since the bid equals v_i, it follows that $v_i - p \geq 0$. □

Definition 2.1. *An auction for which Lemma 2.1 and Lemma 2.2 hold, is called* dominant-strategy incentive compatible (DSIC)

2.1.3 Awesome Auctions

Following Tim Roughgarden (Professor in Stanford), an auction is called *awesome* if it enjoys the following three properties

(P1) The auction is *dominant-strategy incentive compatible (DSIC).*

(P2) If bidders report truthfully, then the auction maximizes the *social surplus* $\sum_{i=1}^{n} v_i x_i$, where, in the case of single-item auctions, $x_i = 1$, if i wins the item, and $x_i = 0$, if i loses the item.

(P3) The auction can be implemented in polynomial time.

Theorem 2.1 (Vickrey'61)**.** *The second-price auction is awesome. It can even be implemented in linear time.*

Proof. Follows from the definition and Lemmata 2.1 and 2.2. □

Note that, even though the valuations are not known to the auctioneer, the auction nevertheless identifies the bidder with the highest valuation!

2.2 Sponsored Search Auctions

A Web search result page comprises a list of search results – deemed by some underlying algorithm such as PageRank – and a list of sponsored links which have been paid for by advertisers. For example, if you type in a key word like "Damenschuhe" into Google, you will see a page like the one in Figure 2.1 with a list of advertisements on the right.

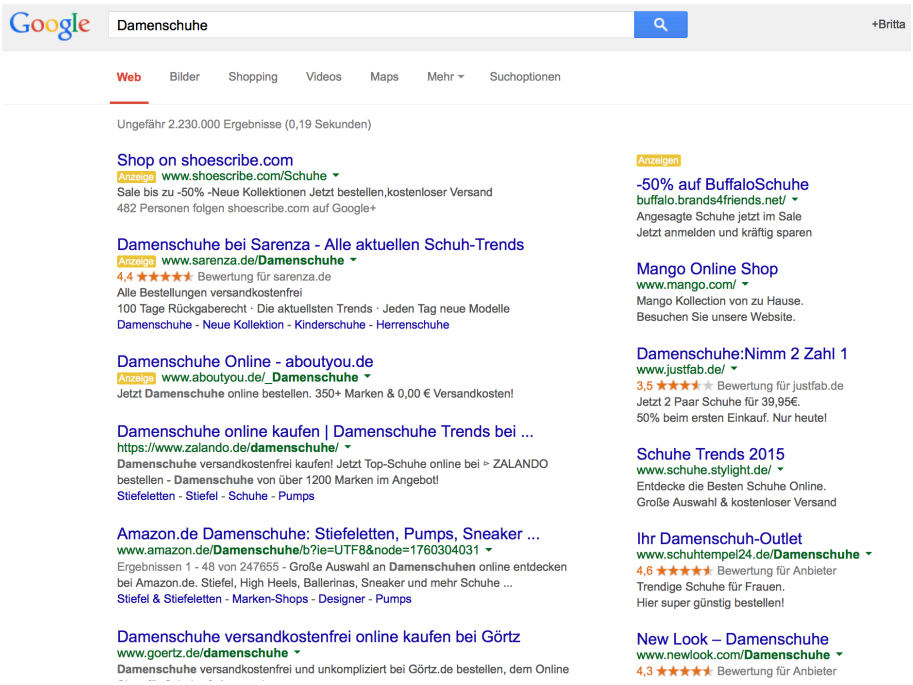


Figure 2.1: The result of searching "Damenschuhe" in Google.

Every time you type a search query into a search engine, an auction is run to decide which advertisers links are shown, in what order, and how they are charged. Due to a statistics from 2005[4],sponsored search auctions generate roughly 98% of Google's revenue.

The model. We discuss a simplistic but useful and influential model of sponsored search auctions due independently to [4] and [14]: the "goods"

are the k slots for sponsored links on a search results web page, and the "bidders" are the advertisers who have a standing bid on the keyword that was searched on. For example, *Buffalo* and *Mango* belong to the bidders for keyword "Damenschuhe", *WMF* and *Kochform.de* belong to the bidders for keyword "Töpfe".

Such auctions are in two ways more general than single-item auctions:

1. there are multiple goods for sale (i.e., $k > 1$), and
2. the items are not identical: higher slots are more valuable.

We quantify the difference between different slots using so-called *click-through-rates (CTRs)*: The CTR α_j denotes the probability that the user clicks on slot j, for $j = 1, \ldots, k$. Ordering the slots from top to bottom, we make the reasonable assumption that

$$\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_k.$$

For simplicity, we also make the unreasonable assumption that the CTR of a slot is independent of its occupant. However, the following results can be extended to the more general and more realistic model in which each advertiser i has a "quality score" β_i (the higher the better) and the CTR of advertiser i in slot j is the product $\alpha_j \beta_i$. To do that, define new valuations for the players by $\tilde{v_i} := \beta_i v_i$. (see exercises.)

We assume that the advertiser is not interested in being displayed on a page per se. Instead, we assume that each advertiser i has a private valuation v_i for each *click* on his link. Hence, the value derived by i from slot j is $v_i \alpha_j$.

Is there an awesome auction?

That is, does our model of the sponsored search auction satisfy

(1) DSIC. That is, truthful bidding should be a dominant strategy, and never lead to negative utility.

(2) Social surplus maximization. That is, the assignment of bidders to slots should maximize $\sum_{i=1}^{n} v_i x_i$, where x_i now denotes the CTR of the slot to which bidder i is assigned (or 0 if i is not assigned to a slot). Each slot can only be assigned to one bidder, and each bidder gets at most one slot.

(3) Polynomial running time. Remember how many auctions need to be run every day!

In mechanism design problems, like this one, we have to jointly design two things: the choice of **who wins what**, and the choice of **who pays what**. The good news is that in many applications including sponsored search auctions, we can tackle the two things one at a time.

2.2.1 Two-step design approach.

Step 1: Assume, without justification, that bidders bid truthfully. Then, how should we assign bidders to slots so that the above properties (2) and (3) hold?

Step 2: Given the answer to Step 1, how should we set prices so that property (1) holds?

If we have successfully answered both questions, then we have designed an awesome auction. Step (1) can be executed for sponsored search auctions as follows:

For $j = 1, 2, \ldots, k$, assign the jth highest bidder to the jth highest slot.

This natural greedy algorithm assigns bidders to slots such that the surplus $\sum_{i=1}^{n} v_i x_i$ is maximized (assuming truthful bidding), and runs in polynomial time. The proof is left as an exercise.

In the following section, we show that Myerson's Lemma, one of the most powerful tools in mechanism design, can be used to implement step (2).

2.3 Myerson's Lemma

To state Myerson's Lemma, we need a couple of definitions.

Definition 2.2. *In a* single-parameter environment*, there is a set of n bidders. Each bidder i has a private valuation v_i which denotes the value "per unit of stuff" he gets. Moreover, there is a* feasible set $X \subseteq \mathbb{R}^n$*, s.t., for each $x = (x_1, \ldots, x_n) \in X$ the component x_i denotes the "amount of stuff" given to bidder i.*

Examples:

- In a single-item auction, X is the set of $0-1$ vectors that have at most one 1, i.e., $X = \{x \in \{0,1\}^n \mid \sum_{i=1}^{n} x_i \leq 1\}$.
- In an auction with k identical goods and the constraint that each customer gets at most one, the feasible set X consists of $0-1$ vectors satisfying $\sum_{i=1}^{n} x_i \leq k$, i.e., $X = \{x \in \{0,1\}^n \mid \sum_{i=1}^{n} x_i \leq k\}$.
- In sponsored search auctions, X is the set of n-vectors corresponding to assignments of bidders to slots, where each slot is assigned to at most one bidder, and each bidder is assigned to at most one slot. If bidder i is assigned to slot j, then the component x_i equals the CTR α_j of its slot.

2.3.1 Allocation and payment rules

In general, any sealed-bid auction in a single-parameter environment consists of three steps:

1. Collect bids $b = (b_1, \ldots, b_n)$.
2. Choose a feasible allocation $x(b) \in X \subseteq \mathbb{R}^n$ as a function of the bids. ("Allocation rule").
3. Choose payments $p(b) \in \mathbb{R}^n$ as a function of the bids ("Payment rule").

In a quasi-linear utility model, in an auction with allocation rules x and p, respectively, bidder i has utility

$$u_i(b) = v_i x_i(b) - p_i(b)$$

on the bid profile $b \in \mathbb{R}^n$.

We focus on payment rules satisfying

$$p_i(b) \in [0, b_i x_i(b)] \quad \forall i \in [n],\ \forall b \in \mathbb{R}^n.$$

Note that $p_i(b) \geq 0$ is equivalent to prohibiting the seller from paying the bidders. The constraint $p_i(b) \leq b_i x_i(b)$ ensures that a truthtelling bidder receives non-negative utility.

Definition 2.3. *An allocation rule x for a single-parameter environment is called* implementable *if there is a payment rule p such that the sealed-bid auction is DSIC.*

That is, the implementable allocation rules are those that extend to DSIC mechanisms.

Example: In a single-item auction, the assignment rule which assigns the item to the highest bidder is implementable (since, for example, the payment rule that assigns the winner the second-highest bid renders DSIC).

Definition 2.4. *An allocation rule x for a single-parameter environment is called* monotone *if for every bidder i and bids b_{-i} of the remaining bidders holds*

$$x_i(z, b_{-i}) \leq x_i(z', b_{-i}) \quad \forall z, z' \in \mathbb{R} \text{ with } z \leq z'.$$

That is, in a monotone allocation rule, bidding higher can only get you more stuff. For example, assigning the good to the highest bidder is monotone. By contrast, assigning the good to the second-highest bidder is not monotone.

Theorem 2.2 (Myerson's Lemma '81 [8])**.** *For a single-parameter environment the following properties (a)-(c) hold.*

(a) An allocation rule x is implementable if and only if x is monotone.

(b) If x is monotone, then there is a unique payment rule such that the sealed-bid mechanism (x, p) is DSIC (assuming the normalization that $b_i = 0$ implies $p_i(b) = 0$).

(c) The payment rule in (b) is given by an explicit formula (see Equations (2.5) and (2.7) below).

Myerson's Lemma builds the foundation for most of the mechanism design theory we will see throughout this lecture. Part (a) states that Definitions 2.3 and 2.4 define exactly the same class of allocation rules. This equivalence is very powerful. Definition 2.3 describes our design goal. But, how should we verify that some rule x is implementable? That's why statement (a) is so important: in most applications, verifying that some rule x is monotone, is easy to check. Part (b) states that, when an allocation rule x is implementable, there is a unique way how to assign the payments to achieve the DSIC property. Moreover, there is an explicit formula for this payment rule (see below).

2.3.2 Proof of Myerson's Lemma

Part 1: DSIC $\Rightarrow$ monotone Consider an allocation rule x, which may or may not be monotone. Suppose there is a payment rule p such that (x, p) is a DSIC mechanism –what should p look like?

Recall the DSIC condition: for each bidder i, every possible private valuation b_i, and every set of bids b_{-i} by the other players, it must be that i's utility is maximized by bidding truthfully. Now, fix i and b_{-i} arbitrarily. To shorten notation, let us write $x(z)$ for allocation $x_i(z, b_{-i})$, and $p(z)$ for payment $p_i(z, b_{-i})$ of bidder i, when i bids z. Figure 2.2 gives two examples of what the function x might look like.

Suppose (x, p) is DSIC, and consider any two bids y and z with $0 \leq z \leq y$. Because bidder i might well have private valuation z and can submit the false bid y, if he wants, DSIC demands that

$$\underbrace{z \cdot x(z) - p(z)}_{\textbf{utility of bidding } z} \geq \underbrace{z \cdot x(y) - p(y)}_{\textbf{utility of bidding } y} . \tag{2.1}$$

Similarly, since bidder i might well have private valuation y and can submit the false bid z, if he wants, DSIC demands that

$$\underbrace{y \cdot x(y) - p(y)}_{\textbf{utility of bidding } y} \geq \underbrace{y \cdot x(z) - p(z)}_{\textbf{utility of bidding } z} . \tag{2.2}$$

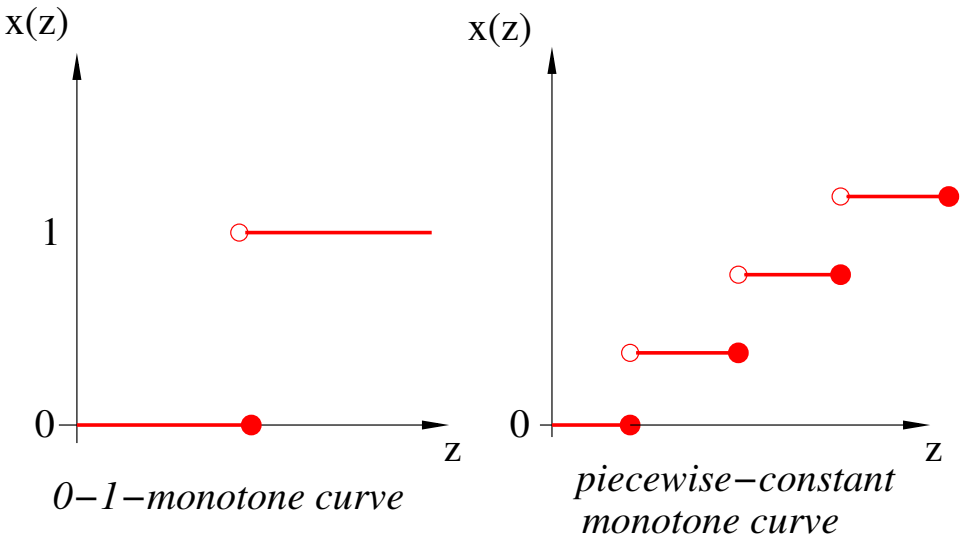


Figure 2.2: Two piecewise constant monotone curves

Rearranging inequalities (2.1) and (2.2) yields the following "payment difference sandwich", bounding $p(y) - p(z)$ from below and above

$$z \cdot [x(y) - x(z)] \leq p(y) - p(z) \leq y \cdot [x(y) - x(z)]. \tag{2.3}$$

Note that the payment difference sandwich, i.e., inequality (2.3), implies that $x(y) - x(z)$ is non-negative whenever $z \leq y$. Hence, statement (a) of Myerson's Lemma is proved: x is monotone.

Part 2: The uniqueness of the payment formula For the rest of the proof, we can therefore assume that x is monotone. In order to show statements (b) and (c) of Myerson's Lemma, consider again inequality (2.3), fix z, and let y tend to z from above. Since a detailed proof is very technical, we will be slightly informal in the following arguments, but cover the main ideas of the proof. We focus on the case where x is piecewise constant, as in Figure 2.2. In this case, x is flat except for a finite number of "jumps". Taking the limit in (2.3) where y tends to z from above, the left- and right-hand sides become 0 if there is no jump in x at z. If there is a jump of magnitude h at z, then the left- and right-hand sides both tend to $z \cdot h$. This implies the following constraint on p, for every z:

"magnitude of jump in p at z" $= z \cdot$ "magnitude of jump in x at z". (2.4)

Thus, assuming the normalization $p(0) = 0$, we derived the following *payment formula*, for every bidder i, bids b_{-i} by other bidders, and bid b_i of

bidder i:

$$p_i(b_i, b_{-i}) = \sum_{j=1}^{l} z_j \cdot \text{magnitude of jump in } x_i(\cdot, b_{-i}) \text{ at } z_j, \tag{2.5}$$

where $z_1, \ldots, z_l$ are the breakpoints of the allocation function $x_i(\cdot, b_{-i})$ in the range $[0, b_i]$.

A similar argument applies when x is a monotone function that is not necessarily piecewise constant. For instance, suppose that x is differentiable. Dividing the payment difference sandwich (2.3) by $y - z$ and taking the limit as y tends to z from above yields the constraint

$$p'(z) = z \cdot x'(z) \tag{2.6}$$

and, assuming $p(0) = 0$, the payment formula

$$p_i(b_i, b_{-i}) = \int_0^{b_i} z \cdot \frac{d}{dz} x_i(z, b_{-i}) dz \tag{2.7}$$

for every bidder i, bids b_{-i} by other bidders, and bid b_i of bidder i.

It follows that Equation (2.5) is the *only* payment rule with a chance of extending the given piecewise constant allocation rule x into a DSIC mechanism. Thus, for every allocation rule x, there is at most one payment rule p such that mechanism (x, p) is DSIC (hence, we have already shown part (b) of Myerson's Lemma). But the proof is not complete – we still have to show that the payment rule works provided x is monotone.

Part 3: x monotone ⇒ Payment formula yields DSIC mechanism We focus again on the case of a piecewise constant allocation function and show that the payment formula given by Equation (2.5) has the property that truthful bidding yields a non-negative utility and is a dominant strategy. We are again a little bit informal and show this by Figure 2.3. Note that the red curve is the allocation function and the blue area is the surplus, price and utility, where the utility is equal to $v \cdot x(b) - p(b)$. One can see that the utility is maximized for truthful bidding.

2.4 Knapsack Auctions

The model of knapsack auctions belongs to single-parameter environments. So, Myerson's Lemma will apply.

Model. In a *knapsack auction* each bidder $i \in N$ has a publicly known size $w_i \in \mathbb{R}_+$ and a private valuation $v_i \in \mathbb{R}_+$. The seller has a capacity

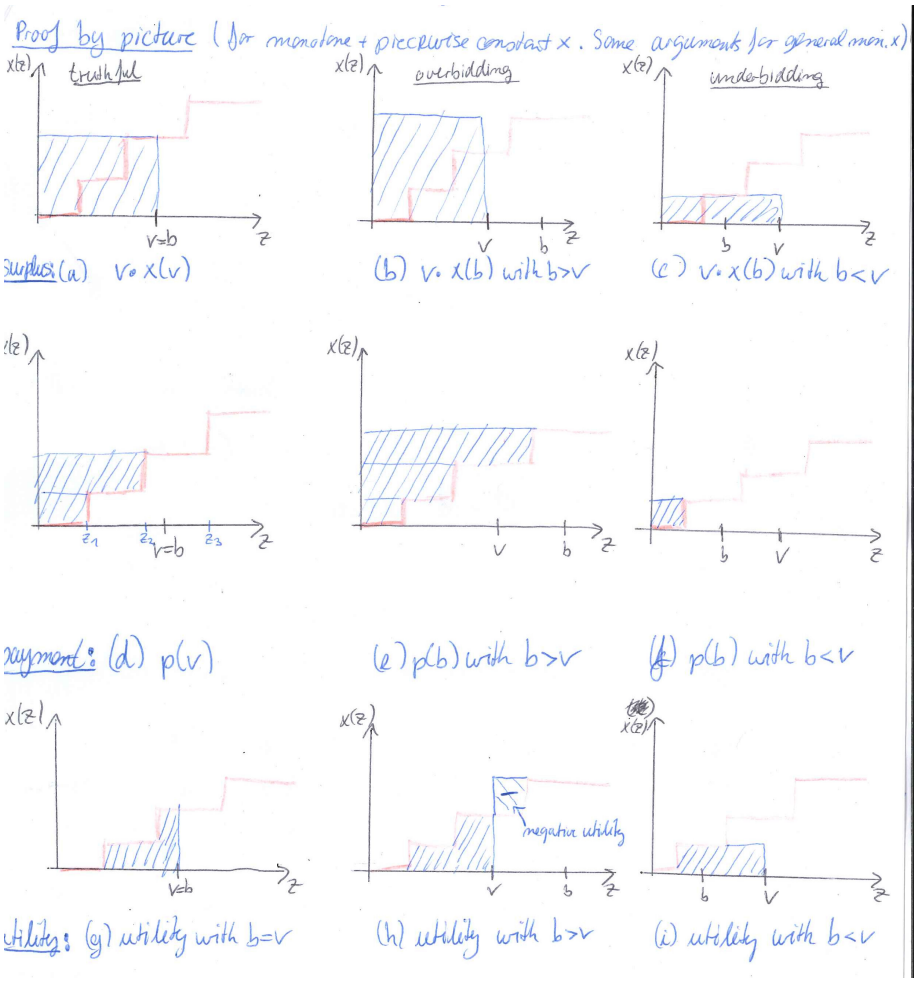


Figure 2.3: proving Part 3 by picture

B. The feasible set X corresponds to all subsets $S \subseteq [n]$ that "fit into the knapsack of capacity B", i.e.,

$$X := \{x \in \{0,1\}^n \mid \sum_{i=1}^{n} w_i x_i \leq B\}.$$

The mechanism selects an allocation $x \in X$ corresponding to the winners of the auction, and decides how much each winner has to pay for being "packed into the knapsack".

Note that in the special case where $w_i = 1$ for all $i \in [n]$ and $B = k \in \mathbb{Z}_+$, we are in the situation of a k-item auction with k identical goods.

2.4.1 Examples

Consider the situations where, for example,

- bidders want their TV-ads (each of a certain length w_i) being shown at a commercial break at the World Cup soccer final. The commercial break has a predefined length (the capacity B),
- bidders want their files being stored on a shared server,
- processes want to be executed on a shared supercomputer,
- ...

In general, whenever we have a shared resource with limited capacity, we are in the situation of a *knapsack problem*.

Knapsack problem. Recall the classical *knapsack problem*: We are given n items of values $v_1, \ldots, v_n$ and sizes $w_1, \ldots, w_n$, as well as a capacity B. The task is to select among those subsets $S \subseteq [n]$ that fit into the knapsack (i.e. with $\sum_{i \in S} w_i \leq B$), one of maximal value $\sum_{i \in S} v_i$. The knapsack problem can be formulated as the following integer linear program

$$\max\{\sum_{i=1}^{n} v_i x_i \mid \sum_{i=1}^{n} w_i x_i \leq B,\ x_i \in \{0,1\} \forall i \in N\}.$$

2.4.2 Mechanism for knapsack auctions.

Since knapsack auctions are single-parameter, Myerson's Lemma applies and we can thus follow the two-step approach:

Step 1: Assuming truthful bids $b = (b_1, \ldots, b_n)$, compute an optimal assignment $x(b)$ of

$$\max\{\sum_{i=1}^{n} b_i x_i \mid \sum_{i=1}^{n} w_i x_i \leq B,\ x_i \in \{0,1\} \forall i \in N\}.$$

Step 2: Given the optimal solution $x(b)$ from Step 1, compute prices $p_i(b)$ for all $i \in [n]$ via

$$p_i(b) = \sum_{j=1}^{l} z_j \cdot (\text{jump in } x_i(\cdot, b_{-i}) \text{ in range } [0, b_i]),$$

where $z_1, \ldots, z_l$ are the breakpoints of function $x_i(\cdot, b_{-i})$ in range $[0, b_i]$.

How does the function $x_i(\cdot, b_{-i}))$ look like? Since x is monotone and only attains values 0 and 1 the function needs to look like the left function shown in Figure 2.2. Hence, there is exactly one breakpoint z which can be interpreted as the "critical bid" for bidder i. If he bids less than z, he will not packed into the knapsack, otherwise, he will belong to the winners of the knapsack auction. Following the payment formula, bidder i has to pay

$$p_i(b) = \begin{cases} z(1-0) = z & \text{if } b_i > z \\ 0 & \text{otherwise.} \end{cases}$$

That is, each bidder pays his critical bid.

But, is the knapsack auction awesome? The answer is "No" for the following reason: the knapsack problem is well-known to be NP-hard, meaning that there is no polynomial-time algorithm to find the optimal allocation $x \in X$ (unless P=NP).

Nevertheless, the knapsack problem turns out to be tractable in practice. Using dynamic programming, it can be solved in so-called *pseudo-polynomial time*.

Still, for several other examples, social surplus maximization cannot be implemented efficiently.

2.4.3 Relationship to theory of approximation algorithms

Mechanism design theory deals with the problem to relax the second constraint (maximal social surplus) as little as possible subject to the first (DSIC) and third (polynomial-time running time) constraint.

For single-parameter environments, this task is equivalent to design a polynomial time and monotone allocation rule that comes as close as possible to maximal social surplus.

Thus, there is a very close relationship to the field of *approximation algorithms* where the primary goal is to design polynomial-time algorithms for NP-hard problems that yield "almost optimal" solutions.

2.4.4 Greedy algorithm for the Knapsack problem

The following greedy algorithm turns out to be a $\frac{1}{2}$-approximation for the Knapsack problem.[1]

[1] An α-approximation for a maximization problem is an algorithm that returns a feasible solution of objective value at least α times the optimal objective value.

Knapsack Greedy Algorithm.
Input: Bids $b_1, \ldots, b_n$; weights $w_1, \ldots, w_n$, capacity $B > 0$.
Output: Set $S \subseteq [n]$ of winners.

1. Sort and re-index bidders so that
$$\frac{b_1}{w_1} \geq \frac{b_2}{w_2} \geq \ldots \geq \frac{b_n}{w_n}.$$
2. Pick winners (add them to the initially empty set S) in this order until one doesn't fit, and the halt.
3. Return either the step-2 solution or the highest bidder, whichever creates more surplus.

Remark: The performance of the above greedy algorithm can only improve if you continue to follow the sorted order, packing further bidders that happen to fit.

Theorem 2.3. *Assuming truthful bids, the surplus of the greedy algorithm is at least* 50% *of the maximal possible surplus.*

Proof. Suppose, as a thought experiment, we relax the problem so that a bidder can be chosen *fractionally*. That is, we relax the knapsack problem to the *fractional knapsack problem*

$$(FKP) \quad \max\{\sum_{i=1}^{n} b_i x_i \mid \sum_{i=1}^{n} w_i x_i \leq B,\ x_i \in [0,1] \forall i \in N\}.$$

For example, if 60% of a bidder with value 10 is chosen, then it contributes 6 to the surplus. Let us modify the above greedy to this setting as follows: leave the ordering proposed in step (1), pick winners in this order until the entire capacity of the knapsack is fully used (picking the final winner fractionally, as needed), omit step (3). A straightforward exchange argument proves that this algorithm returns an optimal solution to (FKP).

Suppose in the optimal fractional solution, the first k bidders win and the $(k+1)$th bidder wins fractionally. The surplus achieved by steps (1) and (2) of the original greedy algorithm above is exactly the total value of the first k. The surplus achieved in step (3) of the original greedy is at least the total value of the $(k+1)$th bidder. Hence, the better of these two values is at least half of the surplus of the optimal fractional solution, which is at least the surplus of an optimal (non-fractional) solution of the original knapsack problem. □

Remark: In the special case where $w_i \leq \alpha B$ for all $i \in [n]$ and some $\alpha \in (0, \frac{1}{2}]$, the approximation guarantee improves from $\frac{1}{2}$ to $1 - \alpha$, even if step (3) is omitted.

Warning: Natural allocation rules are not always monotone!!

For example, for every $\epsilon > 0$, there is a $(1-\epsilon)$-approximation algorithm for the Knapsack problem which runs in time polynomial in the input size and $\frac{1}{\epsilon}$. (This is called FPTAS= fully polynomial-time approximation scheme.) However, the induced allocation rule is, in general, not monotone. Nevertheless, the Knapsack-FPTAS can be modified so that it yields a monotone allocation rule without reducing the performance guarantee.

2.4.5 Black box reduction

This raises the question whether there exists a generic way of taking a possible non-monotone approximation algorithm and turn it into a monotone poly-time algorithm without degrading the performance guarantee. Such a "black-box reduction" would show that it is always possible to design monotone allocation rules whose performance guarantee is never worse than the best known performance guarantee for the corresponding optimization problem without monotonicity constraint. However, as has been shown recently by Chawla et al. [1], the answer is "no", there is no such general black-box reduction.

2.4.6 The revelation principle

The *revelation principle* states the following: if you design a mechanism to have dominant strategies, you might as well design one that is DSIC.

Here is a silly example of a mechanism with dominant strategies that is not DSIC: Consider a single-item auction in which the seller, given bids b, runs a Vickrey-auction on bids $2b$. Then bidding half of the true valuation is each bidder's dominant strategy.

Theorem 2.4. *For every mechanism M in which every bidder has a dominant strategy (no matter of what is his private valuation), there is an equivalent DSIC mechanism M'.*

Proof. By assumption, for every bidder i and private valuation v_i that i might have, bidder i has a dominant strategy, which we denote by $s_i(v_i)$, in the given mechanism M. We construct mechanism M' as follows:

(i) Collect bids $b_1, \ldots, b_n$;

(ii) Submit bids $s_1(b_1), \ldots, s_n(b_n)$ to mechanism M;

(iii) Return the same outcome as M does (e.g., the allocation x and the prices p returned by M).

Mechanism M' is DSIC for the following reason: submitting another bid than the true valuation v_i can only decrease i's utility. □

2.5 Multi-Parameter Mechanism Design and the VCG Mechanism

2.5.1 Multi-Parameter Mechanism Design

Recall that in a single-parameter environment each bidder i has just one piece of information, its value $v_i \in \mathbb{R}$ per unit of stuff.

In contrast, in multi-parameter mechanism design, each bidder i can have different valuations $v_i(\omega) \in \mathbb{R}$ for each possible outcome ω of the game.

Definition 2.5. *A* multi-parameter mechanism design problem *consists of*

(a) n strategic participants,[2]

(b) a finite set Ω of outcomes,

(c) a private valuation $v_i(\omega) \in \mathbb{R}$ for each possible outcome $\omega \in \Omega$.

Note that the outcome set Ω is abstract and can be very large. So, we don't assume that Ω is given explicitly to the mechanism.

Example: In a single-item auction with n bidders, the set of possible outcomes Ω has $n+1$ Elements, corresponding to the winner of the auction (if any). If we consider such an auction in a single-parameter environment, we assume that bidder i has a valuation of v_i if he wins the auction, and a valuation of 0 if he loses. In a multi-parameter environment of such a single-item auction, a bidder can have different valuations for each possible winner.

2.5.2 VCG Mechanism

The Vickrey-Clarke-Groves (VCG) mechanism is one of the cornerstones in the theory of mechanism design theory.

Theorem 2.5 (Vickrey-Clarke-Groves (VCG) mechanism [15, 2, 6])**.** *For every multi-parameter mechanism design problem, there is a DSIC-welfare-maximizing*[3] *mechanism.*

That is, the theorem promises properties (1) and (2) of the definition of awesome auctions, but not property (3), the polynomial-time running time.

[2] you could also say "players" or "bidders" or "agents". All these synonyms are used in the lecture and in the literature.

[3] "welfare" is a synonym for "social surplus"

Proof of Theorem 2.5

In order to prove Theorem 2.5, we apply a similar approach as for Myerson's Lemma:

Step 1: We assume, without justification, that agents truthfully report their private valuation, i.e., $b_i = v_i$, where $v_i : \Omega \to \mathbb{R}$ is a function assigning a real value to each possible outcome in Ω.

That is, given bids $b_1, \ldots, b_n$ we define the allocation rule x such that it maximizes the social welfare $\sum_{i=1}^{n} b_i(\omega)$ over all allocations $\omega \in \Omega$, i.e.,

$$x(b) = \text{argmax}_{\omega \in \Omega} \sum_{i=1}^{n} b_i(\omega). \tag{2.8}$$

Step 2: Given the allocation-rule x as defined in step 1, we know define the payment rule p, such that mechanism (x, p) is DSIC. The key idea is to let each agent i pay the welfare-loss it causes to the remaining agents. This idea corresponds to the payment rule

$$p_i(b) = \underbrace{\max_{\omega \in \Omega} \sum_{j \in N \setminus \{i\}} b_j(\omega)}_{\text{without } i} - \underbrace{\sum_{j \in N \setminus \{i\}} b_j(\omega^*)}_{\text{with } i} \quad \forall i \in [n], \tag{2.9}$$

where $\omega^* = x(b)$ denotes the welfare-maximizing allocation computed in step 1. Note that $p_i(b)$ is always non-negative.

It remains to show:

Claim 2.1. *The resulting mechanism (x, p) is DSIC.*

Proof. (of claim.) Fix a bidder i and the bids b_{-i} of the other bidders. When the chosen outcome computed in step 1 is $\omega^* = x(b)$, then i's utility is

$$v_i(\omega^*) - p_i(b) = \underbrace{v_i(\omega^*) + \sum_{j \in N \setminus \{i\}} b_j(\omega^*)}_{(A)} - \underbrace{\max_{\omega \in \Omega} \sum_{j \in N \setminus \{i\}} b_j(\omega)}_{(B)}. \tag{2.10}$$

Note that term (B) is constant, i.e., independent of bidder i's choice. Thus, i maximizes his utility if and only if he maximizes term (A). Now, if bidder i bids his true valuation $b_i = v_i$, then the mechanism choses an allocation that maximizes i's utility. □

The welfare-maximization together with the statement of Claim 2.1 proves the theorem.

Summarizing, the VCG-theorem states that DSIC welfare-maximization is, in principle, always possible. Still, the VCG-mechanism is often infeasible to implement in practice.

2.5.3 Combinatorial Auctions

Combinatorial auctions are very important in practice. For example, they are used for government spectrum auctions, or for allocating take-off and landing slots at airports.

Model: A *combinatorial auction* consists of

- a set $N = [n]$ of n bidders,
- a set $M = [m]$ of m (not necessarily identical) items.
- The outcome set Ω corresponds to n-vectors $(S_1, \ldots, S_n)$, where S_i denotes the set of items allocated to bidder i (its "bundle") and with no item allocated twice. (Hence, there are $(n+1)^m$ possible outcomes.)
- Each bidder i has a private valuation $v_i(S)$ for each bundle $S \subseteq M$.
- The welfare of outcome $(S_1, \ldots, S_n)$ is $\sum_{i \in [n]} v_i(S_i)$.

We make the general assumption that the valuation functions v_i are *normalized* in the sense $v_i(\emptyset) = 0$, and *monotone* in the sense that $S \subseteq T \subseteq M$ implies $v_i(S) \leq v_i(T)$.

Note that combinatorial auctions belong to the multi-parameter environment, so VCG provides a DSIC solution for maximizing the welfare.

In general, however, there are several challenges in applying VCG. We mention only two major challenges here:

- each bidder has $2^m - 1$ private parameters (a huge number unless m is small).
- the problem to maximize welfare cannot be solved efficiently, unless the valuation functions are very simple.

We will later, in the second part of the course (AGT II), learn about so-called *iterative auctions* in which the mechanism learns about the valuations on a "need-to-know"-basis.

Chapter 3

Strategic Games (Basics)

3.1 Equilibria in strategic games

Strategic games are used to model and analyse situations in which several selfish decision makers (individuals, representatives of countries, drivers, animal populations, ...) interact and influence each others outcome.

The (mathematical) model of a *strategic game* can be described as follows:

- We are given a finite set $N = \{1, \ldots, n\} = [n]$ of players (also called "participants", or "agents"), and
- a strategy set X_i (which may or may not be finite) for each $i \in N$.
- After all players $i \in N$ simultaneously[1] choose one of their strategies $x_i \in X_i$, we arrive at a situation (also called "outcome" or "profile")

  $$x = (x_1, \ldots, x_i, \ldots, x_n) \in X,$$

  where $X := X_1 \times \ldots \times X_n$ denotes the set of all possible outcomes.
- Each $i \in N$ is endowed with a preference relation $\prec_i$ defined on the set X with the interpretation

  $$x \prec_i y \quad \text{if and only if } i \text{ prefers outcome } x \text{ to outcome } y.$$

Thus, a strategic game is defined by a tuple $(N, \{X_i\}_{i\in N}, \{\preceq_i\}_{i\in N})$.

The probably most prominent solution concept for strategic games is the so-called *(pure) Nash equilibrium* which is referred to Nash's Dissertation (1950), but in fact goes already back to *Cournot's "Recherche sur les principes mathématiques de la théorie de la richesse"*, Paris 1838).

[1] choosing strategies "simultaneously" models situations where players do not know about the strategy choices of the other players.

Definition 3.1. *A profile $x \in X$ is a* (pure Nash) equilibrium (PNE) *if for none of the players $i \in N$ there is an alternative strategy $x_i' \in X_i$ such that*

$$(x_{-i}, x_i') \prec_i x,$$

where (x_{-i}, x_i') denotes the profile obtained from x by replacing player i's strategy choice x_i by x_i'.

Example: Players are drivers through a congested network. The strategy sets X_i correspond to the set of possible routes from player i's origin node s_i to its destination node t_i. Player i prefers faster routes. The travel times on the various routes in X_i highly depend on the other players' route choices.

3.1.1 Utility functions

Throughout this lecture, we assume that the players' preferences are modeled via *utility functions* $u_i : X \to \mathbb{R}$ for all $i \in N$.

The preference relation induced by utility function u_i is defined via

$$x \prec_i y \quad \Longleftrightarrow \quad u_i(x) > u_i(y) \quad \forall x, y \in X.$$

Thus, we assume that each player chooses his strategy with the goal to maximize his utility.

Remark: We might as well assume that each player aims at minimizing his utility, for example, if the utility function represents the cost instead of the benefit of the various outcomes. In this case, the interpretation would be $x \prec_i y \quad \Longleftrightarrow \quad u_i(x) < u_i(y)$. However, if not stated otherwise, we assume throughout that a higher utility is better. In case the utilities model the costs instead of the benefits, we can simply multiply the utilities by -1.

Remark: Not all preference relations can be modeled via utility functions. Consider, for example, a game with four outcomes $X = \{a, b, c, d\}$, and a player i with preferences $a \prec_i b, c \prec_i b, c \prec_i d$, and indifferences between all other pairs of outcomes.

3.1.2 Strategic games in standard form

Definition 3.2. *A strategic game is* in standard form*, also called a* matrix game*, if the preferences are modeled via utility functions $u_i : X \to \mathbb{R}$, and all strategies and utilities are given explicitly.*

Games in standard form are in particular comfortable if $n = 2$: they can be represented easily by a matrix (called "payoff matrix") where the rows correspond the the strategies of player 1 (the "row player"), and the columns correspond to the strategies of player 2 (the "column player"). Each matrix

entry consists of a tuple whose first entry contains the utility of the row player, and the second entry contains the utility of the second player, for the profile corresponding to the respective row and column.

3.1.3 Examples

The following three examples (BoS), (PD) and (MD) belong to the most prominent, classical examples of game theory. The matrix entries are chosen to reflect the players' preferences.

Battle of the Sexes (BoS): The story is (roughly) that a boy and a girl want to spend the evening together. The boy prefers to go to the cinema (C), while the girl prefers to watch the soccer game (S). Both prefer to spend the evening together than separately. However, they are too shy to simply talk to each other. The following matrix reflects the situation (girl = row player, boy = column player). The strategies are $X_1 = X_2 = \{C, S\}$, i.e., go to the cinema, or the soccer game.

	S	C
S	$(4,3)$	$(1,1)$
C	$(0,0)$	$(3,4)$

Prisoners' Dilemma (PD): Two suspects of a crime are put in separate cells. Both face the two choices to either confess (C), or remain silent (S), i.e., the strategies are $X_1 = X_2 = \{C, S\}$. The utilities for the four different outcomes $\{(S,S),(S,C),(C,S),(C.C)\}$ correspond to the expected years in prison (multiplied by (-1)).

	S	C
S	$(-2,-2)$	$(-5,-1)$
C	$(-1,-5)$	$(-4,-4)$

Matching Pennies (MP): Two players own a coin whose one side shows a number (N), and whose other side shows a head (H). Both players simultaneously lie their coin on the table hidden by their hand.

If both coins show the same after they lifted their hands, i.e., if the two coins show either both head or both number, the first player wins the coin of the second player. Otherwise, the second player wins the coin of the first player. The strategies are therefore $X_1 = X_2 = \{H, N\}$. Matching Pennies admits no PNE.

	H	N
H	$(1,-1)$	$(-1,1)$
N	$(-1,1)$	$(1,-1)$

3.2 Equilibria and fixpoints

The question for an equilibrium can be formulated as a question for a fixpoint of some associated function:

Given a profile x^* of a strategic game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$, we denote by

$$\mathcal{B}_i(x^*_{-i}) := \{x_i \in X_i \mid u_i(x_i, x^*_{-i}) \geq u_i(x'_i, x^*_{-i}) \ \forall x'_i \in X_i\}$$

the set of *best responses* of player i towards the strategy choice x^*_{-i} of the remaining players. Note that

$$x^* \text{ is a PNE if and only if } x^*_i \in \mathcal{B}_i(x^*_{-i}) \quad \forall i \in N.$$

Therefore, if we denote by f the function that maps each profile $x \in X$ to the set of vectors whose ith component is a best response of player i to x_{-i}, then

$$x^* \text{ is a PNE if and only if } x^* \in f(x^*),$$

i.e., iff x^* is a fixpoint of f.

This connection between equilibria and fixpoints lead to the following theorem by Nikaido-Isoda which provides sufficient conditions for the existence of equilibria:

Theorem 3.1 (Nikaidô, Isoda [9])**.**
The strategic game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$ admits an equilibrium if

1. *X is a non-empty, compact and convex set in $\mathbb{R}^m$.*
2. *The functions u_i are component-wise continuous and concave.*

Proof. The proof is omitted (it heavily relies on Brouwer's fix point theorem). □

3.3 Randomized Games

Note that Theorem 3.1 does not apply to strategic games in standard form for the following reason:

In strategic games in standard form, we assume that the strategy sets X_i are given explicitly. Hence, the sets are finite and (unless they are empty) cannot be convex sets[2].

However, if we apply randomization, the situation becomes a lot better.

[2]Recall that a set $X \subseteq \mathbb{R}^m$ is *convex* if for any two points $x, y \in X$ every (of the infinitely many) convex combinations $\lambda x + (1-\lambda)y$ for $\lambda \in [0,1]$ belongs to X as well.

Randomized matrix games. In the randomized version of a matrix game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$, each player $i \in N$ chooses a probability distribution

$$p^i \in \Delta_i := \{p \in \mathbb{R}_+^{|X_i|} \mid \sum_{k=1}^{|X_i|} p_k = 1\},$$

and selects a concrete (pure) strategy $x_i \in X_i$ with the corresponding probability $p^i(x_i)$. For example, a player in Matching Pennies could decide to toss the coin. In case the coin is fair, this corresponds to the probability distribution ("mixed strategy") $p = (\frac{1}{2}, \frac{1}{2})$.

The *expected gain* of player $i \in N$, given profile $(p^1, \ldots, p^n)$ is

$$U_i(p^1, \ldots, p^n) := \sum_{x=(x_1,\ldots,x_n)\in X} u_i(x) \cdot \underbrace{p^1(x_1) \cdot \ldots \cdot p^n(x_n)}_{\substack{\text{probability that player } i \\ \text{chooses } x_i \text{ for each } i}}.$$

Terminologie: In the randomized version of a matrix game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$, the original strategies $x_i \in X_i$ are called *pure strategies*, and the probability distributions $p^i \in \Delta_i$ are called *mixed strategies* of player $i \in N$.

The famous theorem of Nash follows as a consequence of Theorem 3.1.

Corollary 3.3.1. *[Theorem of Nash '51] Every randomized matrix game admits a mixed equilibrium.*

Proof. Note that each set $\Delta_i := \{p \in \mathbb{R}_+^{m_i} \mid \sum_{k=1}^{m_i} p_k = 1\}$ is a compact and convex set in $\mathbb{R}_+^{m_i}$ for $m_i = |X_i|$. This implies that also $\Delta := \Delta_1 \times \ldots \times \Delta_n$, i.e., the set of all possible profiles of the randomized game, is a compact and convex set. Hence, property (NI1) of Theorem 3.1 is satisfied for $X = \Delta$. Moreover, each function U_i is the sum and product of continuous functions on Δ, and therefore continuous. Furthermore, it can easily be verified that each U_i is concave (even linear) in each component. □

3.3.1 Zero-sum games

A matrix game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$ is called *zero-sum game* if

$$\sum_{i\in N} u_i(x) = 0$$

holds for all $x \in X$.

Two-player zero-sum matrix games. In the special case of a two-player zero-sum game (i.e., $n = 2$), the first player wins exactly the amount which the second player looses, and vice versa. For example, Matching-Pennies is such a two-player zero-sum game. Note that for two-player zero-sum matrix

games, it suffices to write down the payoff matrix of one player (usually, the row player), since $u_1(x) = -u_2(x)$ for all $x \in X$.

The following theorem has been shown even prior to Nash's Theorem by van Neumann:

Theorem 3.2. *An equilibrium of a randomized two-player zero-sum matrix game can be computed efficiently via linear programming techniques.*

Proof. To be filled in later. □

Chapter 4

Cooperative Games (Basics)

Recall the introduction of cooperative games in chapter 1: Cooperative games model situations in which the participants ("players") are, in principle, willing to cooperate in order to achieve a better payoff in total. The central question studied in the theory of cooperative games is how the total benefit/cost should be shared among the players in a fair and stable manner.

Recall that a **cooperative game** is given by a tuple (N, v), consisting of some finite set N of players, and some function $v : 2^N \to \mathbb{R}$ that assigns a value $v(S) \in \mathbb{R}$ to each subset (called "coalition") of players in N.

Function $v : 2^N \to \mathbb{R}$ is called *characteristic function*, or (depending on the interpretation) *benefit- or cost-function* of game (N, v). We may interpret the value $v(S)$ as the benefit achieved [resp. the cost caused] by coalition $S \subseteq N$.

Remark. We assume $v(S)$ to be given by an oracle, i.e. we do not worry (here) about the computability of $v(S)$.

In the Introduction (Chapter 1), we already saw two examples of cooperative games: Owen's production game as an example of a cooperative benefit game, as well as the facility location game as an example of a cooperative cost game. The following *connection game* can be regarded as a special class of a facility location game in which there is just one facility with zero opening costs.

Connection game. In this model, we are given a set $N = \{v_1, \ldots, v_n\}$ of clients (the players) that want to be connected to a common source node (facility) v_0, either directly or indirectly (via other nodes). In order to connect nodes i and j, the cost of c_{ij} needs to be paid. Hence, the cost $c(S)$ to connect $S \subseteq N$ to v_0 is the minimum cost of a spanning tree on

the subgraph induced by $S \cup \{v_0\}$. How should $c(N)$ be shared among the clients?

Assumptions. We assume that v is **normalized** (i.e., $v(\emptyset) = 0$), **monotone** (i.e. $S \subseteq T$ implies $v(S) \leq v(T)$), and **superadditive** (i.e., $S \cap T = \emptyset$ implies $v(S) + v(T) \leq v(S \cup T)$) in benefit games, and **subadditive** (i.e. $S \cap T = \emptyset$ implies $v(S) + v(T) \geq v(S \cup T)$) in cost games.

Remark. The assumptions above are not "really" restrictive. For benefit games (N, v), one usually assumes that a coalition $S \cup T$ would be smart enough to realize when they do better by playing separately. Hence, we can replace function v by $\tilde{v}$ with values

$$\tilde{v}(B) := \max\{\sum_{A \subseteq B} v(A)y_A \mid \sum_{A \subseteq B: i \in A} y_A \leq 1 \ \forall i \in B,\ y_A \in \{0,1\}\ \forall A \subseteq B\}.$$

That is, $\tilde{v}(B)$ denotes the maximum value the players in B can achieve by partition into disjoint sets. It is easy to check that $\tilde{v}$ is monotone and superadditive.

For cost games (N, c) that are not monotone or not subadditive, one can similarly replace the cost of a coalition $c(B)$ by

$$\tilde{c}(B) := \min\{\sum_{A \subseteq N} v(A)y_A \mid \sum_{A \subseteq N: i \in A} y_A \geq 1 \ \forall i \in B,\ y_A \in \{0,1\}\ \forall A \subseteq N\}.$$

That is, a minimum cost of a collection of subsets whose union contains B. Function $\tilde{c}$ is monotone and subadditive.

Solution concepts. In cooperative games, we are searching for **allocations** $x \in \mathbb{R}^n$, which assign to each player $i \in N$ of a given cooperative game (N, v) a "fair" value $x_i \in \mathbb{R}$. However, the meaning of "fair" is not a priori clear, it needs to be defined (and there are various ways to define "fairness" meaningful.) A specified allocation modus is called **solution concept**.

4.1 Core of cooperative games.

The solution concept **core** goes back to John von Neumann (1903-57):

Idea: Given a proposed allocation $x \in \mathbb{R}^N$, each coalition $S \subseteq N$ might compare the total amount $x(S) := \sum_{i \in S} x_i$ assigned to players in S with value $v(S)$.

- In a fair allocation of a cooperative **benefit game**, none of the coalitions should get less than the value gained by the coalition, i.e. $x(S) \geq v(S)$ for all $S \subseteq N$.

- In a fair allocation of a cooperative **cost game**, none of the coalitions should pay more than the cost it causes, i.e. $x(S) \leq v(S)$ for all $S \subseteq N$.
- Moreover, the allocation should allocate the total value $v(N)$ among the players, i.e. $x(N) = v(N)$.

The set of "fair" allocations in this sense are the vectors in polyhedron

$$\text{core}(v) := \{x \in \mathbb{R}^n \mid x(N) = v(N) \text{ and } x(S) \geq v(S)\ \forall S \subseteq N\},$$

for a benefit game (N, v), and

$$\text{core}(c) := \{x \in \mathbb{R}^n \mid x(N) = c(N) \text{ and } x(S) \leq c(S)\ \forall S \subseteq N\},$$

for a cost game (N, c).

Observation: There is no structurell difference between cost- and benefit games. Multiplying by -1 transforms one into the other.

The core of a cooperative game is defined by a set of linear inequalities: Let $A \in \{0,1\}^{2^n \times n}$ be a matrix with entries

$$a_{S,i} = \begin{cases} 1 & i \in S \\ 0 & i \notin S \end{cases}$$

for row S and column i. That is, a row of A (with index S) is the incidence vector of set $S \subseteq N$. Given vector $v \in \mathbb{R}^{2^N}$ with components $v(S)$ for $S \subseteq N$, we have that

$$x(S) \geq v(S) \quad \forall S \subseteq N \quad \iff \quad Ax \geq v.$$

4.1.1 Theorem of Bondareva

In general, the core of a cooperative game might well be empty.

Example 4.1. *Consider the benefit game (N, v) with $N = \{1, 2, 3\}$ and characteristic function $v(\{1, 2\} = v(\{1, 3\} = v(\{2, 3\} = 2, v(\{1, 2, 3\} = 2, 5$ and $v(\emptyset) = v(\{1\} = v(\{2\} = v(\{3\} = 0$. Then, summing the three core inequalities $x_1+x_2 \geq 2, x_2+x_3 \geq 2, x_1+x_3 \geq 2$ implies $x(N) = x_1+x_2+x_3 \geq 3 > v(N) = 2, 5$. Hence, the core of this game is empty.*

The following application of the **strong duality theorem of linear programming** in known under the name "Theorem of Bondareva":

Theorem 4.1 (Bondareva '63, Shapley '67)**.** *Given a payoff game $\Gamma = (N, v)$ the core is non-empty if and only if for all vectors $y \in \mathbb{R}^{2^N}$ satisfying*

- $\sum_{S:i\in S} y_S = 1$ *for all* $i \in N$, *and*
- $y_S \geq 0$ *for all* $S \subseteq N$

holds $v^T y = \sum_{S\subseteq N} v(S) y_S \leq v(N)$.

Proof. Consider the linear program

$$(P) \quad \max\{v^T y \mid \sum_{S\subseteq N. i\in S} y_S = 1 \ \forall i \in N, \ y \geq 0\}$$

and it's dual linear program

$$(D) \quad \min\{\sum_{i\in N} x_i \mid x(S) \geq v(S) \ \forall S \subseteq N\}.$$

Both LPs have feasible solutions (e.g. $y(N) = 1$ and $y(S) = 0$ for all $S \subset N$ is feasible for problem (P), and $x_i = \max_{S\subseteq N} v(S)$ for all $i \in N$ is feasible for (D).)

By the definition of the core, $\text{core}(v) \neq \emptyset$ if and only if (D) has an optimal solution value equal to $v(N)$. Note that, by weak LP-duality, for any pair of a feasible primal and dual solutions y and x, we have $v^T y \leq x(N)$. By strong LP-duality, we even know that $v^T y^* = x^*(N)$ holds for any pair $\{y^*, x^*\}$ of optimal primal and dual solutions. Since $x^*(N) \geq v(N)$ (by feasibility of x^*) the statement of the theorem follows. □

4.1.2 Core of Owen's production game

Recall Owen's production game from above. Let $b_i(S) := \sum_{p\in S} b_{ip}$. denote the total amount of basic ingredients of type G_i owned by players in coalition $S \subseteq N$. The maximum benefit achievable by coalition $S \subseteq N$ is

$$(P_S) \quad v(S) = \max_{x\geq 0}\{\sum_{j=1}^{k} c_j x_j \mid \sum_{j=1}^{k} a_{ij} x_j \leq b_i(S) \ \forall i = 1, \ldots, m\}.$$

Consider the dual LP

$$(D_S) \quad \min_{y\geq 0}\{\sum_{i=1}^{m} b_i(S) y_i \mid \sum_{i=1}^{m} a_{ij} y_i \geq c_j \ \forall j = 1, \ldots, k\}.$$

Recall that, by strong duality, we $v(S) = \sum_{i=1}^{m} b_i(N)\bar{y}_i$ for any optimal solution $\bar{y}$ of (D_S).

Theorem 4.2 (Owen 1975)**.** *Let* y^* *be an optimal solution of* (D_N). *Then* $z^* \in \mathbb{R}^N$ *with* $z^*_p = \sum_{i=1}^{m} b_{ip} y^*_i$ *is an allocation in the core of the production game* (N, v).

Proof. Consider the LP to compute the total value $v(N)$:

$$v(N) = \max_{x \geq 0}\{\sum_{j=1}^{k} c_j x_j \mid \sum_{j=1}^{k} a_{ij} xj \leq b_i(N) \ \forall i = 1, \ldots, n\}.$$

Compute an optimal solution $y^* = (y_1^*, \ldots, y_m^*)$ of the dual LP

$$(D_N) \quad \min_{y \geq 0}\{\sum_{i=1}^{m} b_i(N) y_i \mid \sum_{i=1}^{m} a_{ij} y_i \geq c_j \ \forall j = 1, \ldots, k\}.$$

By strong duality, we know that $v(N) = \sum_{i=1}^{m} b_i(N) y_i^*$. Given y^*, we define an allocation $z^* \in \mathbb{R}_+^N$ via

$$z_p^* := \sum_{i=1}^{m} b_{ip} y_i^* \quad \forall p \in N.$$

Note that $z^*(N) = \sum_{p \in N} z_p^* = \sum_{p \in N} \sum_{i=1}^{m} b_{ip} y_i^* = \sum_{i=1}^{m} b_i(N) y_i^* = v(N)$, where the last equality follows by strong duality. Furthermore, for each coalition $S \subseteq N$ we observe

$$\begin{aligned} z^*(S) &= \sum_{p \in S} z_p^* = \sum_{p \in S} \sum_{i=1}^{m} b_{ip} y_i^* \\ &\geq \min_{y \geq 0}\{\sum_{i=1}^{m} b_i(S) y_i \mid \sum_{i=1}^{m} a_{ij} y_i \geq c_j \ \forall j = 1, \ldots, k\} = v(S), \end{aligned}$$

where the last inequality follows as y^* is a feasible solution for (D_S). □

Remark 4.1. *The values y_i^* are called "shadow prices" for the ingredients G_i, $i = 1, \ldots, m$.*

4.2 Assignment Games

In assignment games, a special class of cooperative benefit games, the player set is divided into two disjoint sets $N = J \cup M$. We might identify the players in J with *jobs*, and the players in M with *machines*. Processing $j \in J$ on $m \in M$ produces a benefit of $a_{jm} \in \mathbb{R}$.

Definition 4.1. *A **(fractional) matching** is an assigment $x : J \times M \to \mathbb{R}_+$ such that for all $j \in J$ and $m \in M$ the following holds: $\sum_{j \in J} x_{jm} \leq 1$ and $\sum_{m \in M} x_{jm} \leq 1$.*

Consider the **assignment game** (N, v) with values

$$v(S \cup T) := \max\{\sum_{j \in S} \sum_{m \in T} a_{jm} x_{jm} \mid x \text{ fractional matching}\} \quad \forall S \subseteq J, T \subseteq M.$$

Remark 4.2. *One can show that there always exists an optimal* ***integral*** *solution of the linear program above.*

Theorem 4.3. *The Core-allocations of the assignment game are* ***exactly*** *the optimal solutions of the LP*

$$(*) \quad \min_{y,z\geq 0}\{\sum_{j\in J} y_j + \sum_{m\in M} z_m \mid y_j + z_m \geq a_{jm} \ (j \in J, m \in M)\}.$$

Proof. Let (y^*, z^*) be an optimal solution of $(*)$. Note that $\sum_{j\in J} y_j^* + \sum_{m\in M} z_m^* = v(J \cup M)$ by strong duality, since $(*)$ is the dual to the linear program corresponding to $v(J\cup M)$. Since y^* and z^* are feasible for problem

$$\min_{y,z\geq 0}\{\sum_{j\in S} y_j + \sum_{m\in T} z_m \mid y_j + z_m \geq a_{jm} \ (j \in S, m \in T)\}$$

it follows that $\sum_{j\in S} y_j^* + \sum_{m\in T} z_m^* \geq v(S \cup T)$. Thus, any optimal solution y^*, z^* of LP $(*)$ lies in the core of the game.

Conversely, take any core-solution (y, z). Then (y, z) is feasible for $(*)$ since

$$a_{jm} \leq v(\{j\} \cup \{m\}) \leq y_j + z_m,$$

where the last inequality follows since (y, z) is in the core(v), and the first inequality follows from the definition of $v(\{j\} \cup \{m\})$. Moreover, $\sum_{j\in J} y_j + \sum_{m\in M} z_m = v(J \cup M)$ guarantees that (y, z) is an optimal solution of $(*)$. □

4.3 Convex games

In this section, we study the special class of *convex cooperative games*. We will see that, for convex games, the core is always non-empty, and a core-allocation can be computed efficiently.

Let (N, v) be a cooperative game. Given $S \subseteq N$ and some player $p \in S$, we denote by

$$v(S) - v(S \setminus \{p\})$$

the **marginal value of** p **w.r.t.** S.

Let $\pi = \{p_1, \ldots, p_n\}$ be an arbitrary permutation of the player set N. Then $x^\pi \in \mathbb{R}^n$ with values

$$x_i^\pi := v(\{p_1, \ldots, p_j\}) - v(\{p_1, \ldots, p_{j-1}\}) \quad \forall p_j = i \in \{1, ..., n\}$$

is called **marginal vector w.r.t.** π. That is, the ith component of x^π is the marginal value of player i w.r.t. coalition $\{p_1, \ldots, p_j = i\}$.

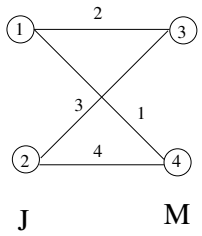


Figure 4.1: Graph for assignment game

Example 4.2. *Consider the assignment game corresponding to the graph in Figure 4.2 and the two permutations* $\pi = \{1, 2, 3, 4\}$ *and* $\pi' = \{2, 4, 1, 3\}$.

The associated marginal vectors for π *and* π' *are* $x^\pi = (0, 0, 3, 3)^T$ *and* $x^{\pi'} = (0, 0, 2, 4)^T$. *Observe that both marginal vectors are not in the core of the assignment game.*

Observation 4.1. *Given permutation* $\pi = \{p_1, \ldots, p_n\}$ *of* N, *let* $N_i := \{p_1, \ldots, p_j = i\}$ *for each* $i \in [n]$. *Then*

$$\sum_{p \in N_i} x_p^\pi = v(N_i)$$

holds for each $i \in [n]$. *In particular,* $\sum_{p \in N} x_p^\pi = v(N)$ *holds.*

Proof. $\sum_{p \in N_i} x_p^\pi = v(N_i) - v(N_{i-1}) + v(N_{i-1}) - v(N_{i-2}) + \ldots + v(N_2) - v(N_1) + v(N_1) - v(N_0) = v(N_i) - v(N_0) = v(N_i)$. □

Hence, x^π is **budget-balanced** in the sense $\sum_{p \in N} x_p^\pi = v(N)$. Under which conditions is x^π in the core of the game (N, v)? That is, when is $\sum_{p \in S} x_p^\pi \geq v(S)$ for all $S \subseteq N$ (provided that (N, v) is a benefit game)?

Definition 4.2. *Function* $v : 2^N \to \mathbb{R}$ *is called* ***supermodular*** *[resp.* ***submodular****] if*

$$v(S \cup \{p\}) - v(S) \leq \ [\geq] \ v(T \cup \{p\}) - v(T) \quad \forall S \subseteq T \subseteq N \text{ and } p \in N \setminus T.$$

Alternatively, sub- and supermodularity can be defined as follows:

Definition 4.3. *Function* $v : 2^N \to \mathbb{R}$ *is called* ***supermodular*** *[resp.* ***submodular****] if*

$$v(S) + v(T) \leq \ [\geq] \ v(S \cap T) + v(S \cup T) \quad \forall S, T \subseteq N.$$

Definition 4.4. *A cooperative benefit game (N, v) [resp. cost game (N, c)] is **convex** if and only if v is supermodular [resp. submodular].*

Theorem 4.4. *(N, v) is a convex cooperative game if and only if for every permutation π of the player set N, the marginal vector x^π is a core-allocation.*

Proof. We consider a cooperative benefit game (N, v). The proof for cooperative cost games is analogue.

"$\Longrightarrow$:" We show that any marginal vector x^π lies in the core of game (N, v) provided that v is supermodular. Suppose this is not true, i.e., suppose there is a marginal vector x^π that is not a core-allocation. Up to renaming assume that p_i is the i-th player in the permutation π. Let $S \subset N$ be a coalition of minimal cardinality among those for which $\sum_{p\in S} x_p^\pi < v(S)$. Then $S \neq \emptyset$ (since we assume v to be normalized). Choose the unique set $N_i = \{p_1, \ldots, p_i\}$ such that $S \subseteq N_i$ and $S \not\subseteq N_{i-1}$. By the choice of S we have $|S \cap N_{i-1}| < |S|$ and $\sum_{p\in S\cap N_{i-1}} x_p^\pi \geq v(S \cap N_{i-1})$. Since $S \cup N_{i-1} = N_i$, Observation 4.1 implies $\sum_{p\in S\cup N_{i-1}} x_p^\pi = v(S \cup N_{i-1})$, and therefore

$$\begin{aligned} v(S \cap N_{i-1}) + v(S \cup N_{i-1}) &\leq \sum_{p\in S\cap N_{i-1}} x_p^\pi + \sum_{p\in S\cup N_{i-1}} x_p^\pi && (4.1)\\ &= \sum_{p\in S} x_p^\pi + \sum_{p\in N_{i-1}} x_p^\pi && (4.2)\\ &< v(S) + v(N_{i-1}) && (4.3) \end{aligned}$$

in contradiction to the supermodularity of v.

"$\Longleftarrow$:" We need to show that v is supermodular if $x^\pi \in \text{core}(v)$ for all permutations π. Choose two coalitions $S, T \subseteq N$ arbitrarily. Consider a permutation π in which the players are ordered in the following way: the first players in π are players from $S \cap T$, followed by the players in $S \setminus T$, followed by the players in $(S \cup T) \setminus S$, and finally the remaining players in $N \setminus (S \cup T)$. By Observation 4.1, the induced marginal vector x^π satisfies

$$\sum_{p\in S\cap T} x_p^\pi = v(S \cap T),\ \sum_{p\in S} x_p^\pi = v(S),\ \text{and}\ \sum_{p\in S\cup T} x_p^\pi = v(S \cup T).$$

Since, by our assumption, x^π is a core-allocation, we observe:

$$\begin{aligned} v(S \cap T) + v(S \cup T) &= \sum_{p\in S\cap T} x_p^\pi + \sum_{p\in S\cup T} x_p^\pi && (4.4)\\ &= \sum_{p\in S} x_p^\pi + \sum_{p\in T} x_p^\pi && (4.5)\\ &\geq v(S) + v(T). && (4.6) \end{aligned}$$

□

Chapter 5

Cooperative Games (cont.)

Remark 5.1. *Convex benefit [resp. cost] games are, in particular, super-additive [resp. subadditive], in the sense* $v(S) + v(T) \leq v(S \cup T)$ *[resp.* $v(S) + v(T) \geq v(S \cup T)$*] whenever* $S \cap T = \emptyset$.

5.1 Core of connection games

Recall the connection game (N, v) described above in which the players in N correspond to vertices $\{v_1, \ldots, v_n\}$ of some undirected complete graph $G = (V, E)$ with $V = N \cup \{v_0\}$ and edge-costs $c : E \to \mathbb{R}_+$. The cost $v(S)$ of coalition $S \subseteq N$ is the minimal cost to connect all players in S (directly or indirectly) to the source node ("facility") v_0, i.e., $v(S)$ denotes the cost of a min-cost spanning tree on the subgraph $G[S \cup \{v_0\}]$.

Since connection games are in general not convex games, we cannot apply Theorem 4.4 to show that the core is non-empty. Still, for carefully selected permutations π of the players in N, the corresponding marginal vector x^π turns out to be a core-allocation.

Theorem 5.1. *The core of a connection game is always non-empty. Moreover, a core-allocation can be computed efficiently.*

Proof. Consider a cost-minimal connection of all players to v_0. Since $c_e \geq 0$ for all $e \in E$, we might assume that this optimal connection does not contain any cycle. Thus, the connection forms a spanning tree T of minimal cost in G. Given this tree, we construct a permutation π as follows:

1. Initialize $T_n = T$;
2. For $k = n$ down to $k = 1$ do
   (a) select a leave-vertex (i.e., vertex of degree one) v_k from T_k;

(b) delete vertex v_k and the (unique) incident edge $\{u_k, v_k\}$ from T_k;

(c) Iterate with resulting tree T_{k-1};

3. Return permutation $\pi = \{v_1, \ldots, v_n\}$;

It remains to show that the calculated vector is in fact in the core. Note that each tree T_k (for $k = 1, \ldots, n$) is a min-cost spanning tree in the subgraph induced by vertex-set $\{v_0, v_1, \ldots, v_k\}$. Moreover, note that the induced marginal vector x^π has values $x^\pi(v_k) = v(N_k) - v(N_{k-1}) = c(\{u_k, v_k\})$ for $k = 1, \ldots, n$.

For the sake of contradiction, assume that x^π is not a core-allocation, i.e., we assume that there is at least one coalition $S \subseteq N$ with $x^\pi(S) > v(S)$. Let $T_S = \{\{u_k, v_k\} \mid v_k \in S\}$ be the set of edges removed in iterations k for which $v_k \in S$. Let T_S^* be a min-cost spanning tree in $G[S \cup \{v_0\}]$ Note that $\tilde{T} := (T \setminus T_S) \cup T_S^*$ is also a feasible connection (since every player is connected to v_0). However,

$$\begin{aligned} v(\tilde{T}) &= v(T) - v(T_S) + v(S) && (5.1)\\ &= x^\pi(N) - x^\pi(S) + v(S) && (5.2)\\ &< x^\pi(N) = v(T), && (5.3) \end{aligned}$$

in contradiction to the optimality of T. □

5.2 Shapley value

Shapley proposed to allocate $v(N)$ among the players in N by the following allocation modus.

Definition 5.1. *The **Shapley value** of the cooperative game (N, v) is the allocation $\Phi \in \mathbb{R}^N$ with values*

$$\Phi_i := \frac{1}{n!} \sum \{x_i^\pi \mid \pi \text{ permutation of } N\} \quad \forall i \in N.$$

Example 5.1. *Consider the game (N, v) with $N = \{1, 2, 3\}$ and characteristic function $v(\emptyset) = v(\{1\}) = 0, v(\{2\}) = v(\{3\}) = v(\{1, 2\}) = 2, v(\{2, 3\}) = v(\{1, 3\}) = v(N) = 3$. The Shapley value is $\Phi = \frac{1}{6}(1, 7, 10)^T$.*

End of AGT I

Definition 5.2. *A cooperative benefit game (N, v) is **simple** if there exists a coalition $T \neq \emptyset$ and some value $v_T \in \mathbb{R}$ such that for all $S \subseteq N$*

$$v(S) = \begin{cases} v_T & T \subseteq S \\ 0 & else. \end{cases}$$

Observation 5.1. *A marginal vector x^π w.r.t. a simple game (N, v_T) assigns*

$$x_i^\pi = \begin{cases} v_T & \text{if } i \text{ is the last player of } T \text{ w.r.t. } \pi \\ 0 & else. \end{cases}$$

Since the number of permutations π in which a player i occurs as the last member of T is the same for each player in T, we observe

Observation 5.2. *The Shapley value Φ of a simple game (N, v_T) assigns*

$$\Phi_i = \begin{cases} \frac{v_T}{|T|} & \text{if } i \in T \\ 0 & \text{if } i \notin T. \end{cases}$$

Remark 5.2. *It turns out that the Shapley value is the unique budget-balanced allocation modus Θ that*

- *shares the value $v(N) = v_T$ of a simple game (N, v_T) evenly among the players in T, and*
- *is additive in the sense that $\Theta(v) + \Theta(w) = \Theta(v + w)$ for any two cooperative games (N, v) and (N, w) with sum $(N, v + w)$.*

5.3 Nucleolus of cooperative games

We consider further solution concepts besides the core. W.l.o.g. we restrict to cooperative benefit games (N, v). Cooperative cost games can be analyzed similarly.

Definition 5.3. *An allocation $x \in \mathbb{R}^N$ is called **budget-balanced** if $x(N) = v(N)$.*

Definition 5.4. *Given a budget-balanced allocation $x \in \mathbb{R}^N$ and a coalition $S \subseteq N$, we call $e(S, x) := v(S) - x(S)$ the **excess** of S w.r.t. x.*

Note: If $x \in \text{core}(v)$, then $e(S, x) \leq 0$ for all $S \subseteq N$. We can interpret $e(S, x)$ as **concession** of S to the grand coalition under allocation x.

Definition 5.5. *Given an allocation $x \in \mathbb{R}^N$, the **excess-vector** $l(x) \in \mathbb{R}^{2^N}$ is the vector with components*

$$l(x)_S = e(S, x) \quad \forall S \subseteq N,$$

ordered by non-increasing values.

Example 5.2. *Consider, for example, the cooperative game (N, v) with $N = \{1, 2, 3\}$, $v(S) = |S|$ for all $S \subseteq N$, and the two allocations $x = (2, 1, 0)^T$ and $y = (1, 1, 1)^T$. The corresponding excess vectors are*

$$l(x) = \begin{pmatrix} e(\{3\}, x) & = & 1 \\ e(\{2, 3\}, x) & = & 1 \\ e(\{2\}, x) & = & 0 \\ e(\{1, 3\}, x) & = & 0 \\ e(N, x) & = & 0 \\ e(\{\}, x) & = & 0 \\ e(\{1\}, x) & = & -1 \\ e(\{1, 2\}, x) & = & -1 \end{pmatrix}$$

and $l(y) = (0, \ldots, 0)^T$.

Definition 5.6. *Let $u, v \in \mathbb{R}^m$ be two vectors. Then, u is **lexicographically smaller** than v (for short: $u \prec_{lex} v$) if there exists some index $k \in [m] = \{1, \ldots, m\}$ such that $u_i = v_i$ for all $i \leq k - 1$ and $u_k < v_k$.*

Definition 5.7. *The **nucleolus** of game (N, v) is the budget-balanced allocation that lexicographically minimizes $l(x)$.*

Remark 5.3. *The nucleolus exists and can be computed by a series of linear programs (with possible exponential number of constraints). If the core of the game is non-empty, the nucleolus is a specific core-allocation that might be seen to be more fair than other core-allocations. If the core is empty, the nucleolus still tries to allocate the total benefit in a fair manner.*

Chapter 6

Network Games

In this chapter, we consider a special class of strategic games called *network games*.

Network games, also called *routing games*, are used to model and analyze situations where players send traffic (e.g. cars, messages, bits, water, electricity, etc.) through a commonly used network. The cost / travel time on an arc depends on the congestion / load caused by the players using that particular arc.

6.1 Excurs: Inefficiency of equilibria

We have seen in Chapter 3, for example in the Prisoner's Dilemma that rational, selfish behavior may lead to equilibria that are far away from "socially optimal" solutions.

Question: How efficient is an equilibrium? How much better could a solution be that is proposed by a "central authority"?

The *social value* or *social cost* of a profile $x \in X$of a strategic game $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$ can be measured in different ways. As we are mainly talking about routing games, we assume throughout this chapter that the utility functions u_i of a strategic game describe the costs of the players $i \in N$. That is, each player $i \in N$ aims at *minimizing* his/her cost $u_i(x)$. The most popular functions to measure the social value/cost of a profile $x \in X$ are the

- *utilitarian* function $\sum_{i\in N} u_i(x)$, and
- the *egalitarian* function $\max_{i\in N} u_i(x)$.

A profile $x \in X$ is called *socially optimal* if it minimizes the social cost function under consideration.

Remark 6.1. *The unique equilibrium in the prisoner's dilemma minimizes none of the two objective functions. Moreover, the example can be modified*

such that the equilibrium is arbitrarily inefficient (compared to the social optimum).

Proof. Let

$$U = \begin{array}{c|cc} & S & C \\ \hline S & (1,1) & (k-1,0) \\ C & (0,k+1) & (k,k) \end{array}$$

be the cost matrix of the modified prisoner's dilemma. Then the outcome of the only Nash-eqilibrium is (C, C) of costs (k, k) which is arbitrarily bad compared to the social optimum (by using the utilitarian or egalitarian function). □

6.1.1 Price of Anarchy/Stability

Definition 6.1. *The* price of anarchy [stability] *(for short:* PoA *resp.* PoS*) is defined as the worst [best] social value of an equilibrium divided by the value of a socially optimal solution.*

Remark 6.2. *The PoA and PoS depends on the chosen social value function. If not stated otherwise, we consider the social value function* $C(x) = \sum_{i \in N} u_i(x)$.

For example, the PoA and PoS in Pigou's example is $\frac{4}{3}$. The costs in the Prisoner's Dilemma can be chosen so that the PoA and PoS is arbitrarily bad.

6.2 Network games

Network games are a special class of strategic games in which we are given

- a directed graph $G = (V, E)$ ("one-way-streets"),
- a set $N = [n]$ of populations, also called commodities,
- source-sink pairs $\{s_i, t_i\} \subseteq V$ with demands $d_i \geq 0$ for each $i \in N$,
- cost/travel times $c_e : \mathbb{R} \to \mathbb{R}$ on each arc $e \in E$.

Interpretation: Each $i \in N$ needs to send d_i units of flow/traffic from s_i to t_i. The cost c_e on arc $e \in E$ depends on the load (total flow) on e.

Assumption: The players are selfish and rational: they want to route as fast as possible and do not care about the time the other players need.

6.3 Wardrop model

One of the most famous models used to analyze traffic in logists or communication technology is the model of Wardrop (1952), also called *nonatomic selfish routing model*.

Let $\mathcal{P}_i \subseteq \{0,1\}^{|E|}$ denote the set of (incidence vectors of) directed $s_i - t_i$-paths in $G = (V, E)$, and $\mathcal{P} = \bigcup_{i \in N} \mathcal{P}_i$.

Definition 6.2. *An assignment $f^{(i)} : \mathcal{P}_i \to \mathbb{R}_+$ is called $s_i - t_i$-*flow*. An assignment $f : \mathcal{P} \to \mathbb{R}_+$ is called* (total) flow*. A total flow f is called* feasible *if $\sum_{P \in \mathcal{P}_i} f_P = d_i$.*

In the Wardrop model, the strategy set of $i \in N$ is

$$X_i = \{f^{(i)} : \mathcal{P}_i \to \mathbb{R}_+ \mid \sum_{P \in \mathcal{P}_i} f_P^{(i)} = d_i\}.$$

For the Wardrop model it is necessary to assume c_e to be continuous and monotone.

Suppose each $i \in N$ chooses a flow $f^{(i)} \in X_i = \{f^{(i)} : \mathcal{P}_i \to \mathbb{R}_+ \mid \sum_{P \in \mathcal{P}_i} f_P^{(i)} = d_i\}$.

Notation:

- Flow $f^{(i)}$ causes a *load/congestion* of $f_e^{(i)} = \sum_{P \in \mathcal{P}_i : e \in P} f_P^{(i)}$ on each arc $e \in E$.
- The load or congestion on $e \in E$ under total flow $f = (f^{(1)}, \ldots, f^{(n)})$ is $f_e = \sum_{P \in \mathcal{P} : e \in P} f_P = \sum_{i \in N} f_e^{(i)}$.
- Flow f causes a *cost on* e of $c_e(f_e)$ per unit of flow that is sent along e.
- The *total cost* caused by flow f is $C(f) = \sum_{e \in E} c_e(f_e) f_e$.
- The cost for player i caused by flow f is $C^{(i)}(f) = \sum_{e \in E} c_e(f_e) f_e^{(i)}$.

6.3.1 Equilibria in the Wardrop model

A feasible flow f is in an equilibrium state (Wardrop equilibrium) if for all $i \in N$, $s_i - t_i$ paths $P_1, P_2 \in \mathcal{P}_i$ with $f_{P_1} > 0$ and all amounts $\delta \in (0, f_{P_1}]$ we have that

$$\sum_{e \in P_1} c_e(f_e) \leq \sum_{e \in P_2} c_e(\tilde{f}_e),$$

where $\tilde{f}$ is obtained from f by moving δ units of flow from P_1 to P_2:

$$\tilde{f}_e = \begin{cases} f_e - \delta & \text{if } e \in P_1 \setminus P_2, \\ f_e + \delta & \text{if } e \in P_2 \setminus P_1, \\ f_e & \text{else.} \end{cases}$$

Observation 6.1. *f is an equilibrium flow if and only if for all commodities $i \in N$, and all paths $P_1, P_2 \in \mathcal{P}_i$ with $f_{P_1}^{(i)} > 0$ holds*

$$\sum_{e \in P_1} c_e(f_e) \leq \sum_{e \in P_2} c_e(f_e).$$

Proof. "$\Longrightarrow$:" Let $\sum_{e \in P_1} c_e(f_e) > \sum_{e \in P_2} c_e(f_e)$, i.e. there is $k \in \mathbb{R}$ with $k > 0$ and

$$\begin{aligned}
\sum_{e \in P_1} c_e(f_e) &= \sum_{e \in P_2} c_e(f_e) + k \\
&> \sum_{e \in P_2 \setminus P_1} \left(c_e(f_e) + \frac{k}{|P_2 \setminus P_1| + 1} \right) + \sum_{e \in P_1 \cap P_2} c_e(f_e) \\
&\geq \sum_{e \in P_2 \setminus P_1} c_e(f_e + \delta_e) + \sum_{e \in P_1 \cap P_2} c_e(f_e) \\
&\geq \sum_{e \in P_2 \setminus P_1} c_e(f_e + \min\{\delta_e\}) + \sum_{e \in P_1 \cap P_2} c_e(f_e) = \sum_{e \in P_2} c_e(\tilde{f}_e).
\end{aligned}$$

"$\Longleftarrow$:" Let $\sum_{e \in P_1} c_e(f_e) > \sum_{e \in P_2} c_e(\tilde{f}_e)$. It holds

$$\begin{aligned}
\sum_{e \in P_1} c_e(f_e) > \sum_{e \in P_2} c_e(\tilde{f}_e) &= \sum_{e \in P_2 \setminus P_1} c_e(f_e + \delta) + \sum_{e \in P_1 \cap P_2} c_e(f_e) \\
&\geq \sum_{e \in P_2 \setminus P_1} c_e(f_e) + \sum_{e \in P_1 \cap P_2} c_e(f_e) = \sum_{e \in P_2} c_e(f_e)
\end{aligned}$$

□

6.3.2 Existence of equilibria in the Wardrop model

Theorem 6.1. *Under the assumption that all cost functions $c_e, e \in E$ are monotone non-decreasing and continuous, a Wardrop equilibrium is guaranteed to exist.*

Proof. Consider the following program:

$$\begin{aligned}
\min & \sum_{e \in E} \int_0^{f_e} c_e(t)dt & \\
\text{s.t.} & \sum_{P \in \mathcal{P}_i} f_P = d_i & \forall i \in N \\
& f_e = \sum_{P \in \mathcal{P}: e \in P} f_P & \forall e \in E \\
& f_P \geq 0 & \forall P \in \mathcal{P}.
\end{aligned}$$

Note that $c_e(t)$ is continuous and non-decreasing. Therefore $\int_0^x c_e(t)dt$ is convex. So we are minimizing the sum of convex functions over a closed and bounded region, thus the program admits an optimal solution.

Now, consider an optimal solution f^*. Consider an $s_i - t_i$ path $P_1 \in \mathcal{P}_i$ with $f^*_{P_1} > 0$ and another $s_i - t_i$ path $P_2 \in P_i$. Since every $\int_0^x c_e(t)dt$ is continuously differentiable, transferring a small $\lambda \in (0, f^*_{P_1}]$ amount of flow from P_1 to P_2 yields a feasible flow with objective function value equal to

$$\sum_{e \in E} \int_0^{f_e} c_e(t)dt + \lambda \left[\sum_{e \in P_2} c_e(f^*_e) - \sum_{e \in P_1} c_e(f^*_e) \right]$$

plus an error term that vanishes for $\lambda \downarrow 0$. This can not be smaller than the optimal flow f^*. Thus

$$\sum_{e \in P_1} c_e(f^*_e) \leq \sum_{e \in P_2} c_e(f^*_e),$$

which means that f^* is an equilibrium flow. □

Remark 6.3. *If the cost functions c_e are of type $c_e(x) = a_e x + b_e$ for $a_e, x, b_e \in \mathbb{R}$, it can be shown that the PoA is at most $\frac{4}{3}$, i.e. the Pigou example is a worst case example!*

Remark 6.4. *For non-linear cost functions, the PoA in the Pigou example can be arbitrarily bad (Exercise.)*

6.3.3 Braess Paradox

Under the assumption that traffic always converges to an equilibrium, Braess discovered in 1968 the following phenomen:

"It is possible that reducing the cost of an arc might in fact increase the total cost incurred by traffic in an equilibrium."

Braess paradox: ... see Section 1.2.1.

Hence, it might well be that building new streets has negative consequences for the overall traffic.

6.4 Atomic selfish routing model

The *atomic selfish routing model* is almost similar to the non-atomic selfish routing (Wardrop) model:

As before, an instance of the game consists of a directed graph $G = (V, E)$, with arc costs $c_e \geq 0$ together with a set N of commodities, each $i \in N$

associated with a source $s_i \in V$, a sink $t_i \in V$, and a certain demand d_i that needs to be send from s_i to t_i.

In contrast to the non-atomic model, in the atomic model each commodity $i \in N$ needs to *send the entire demand d_i along a single path $P_i \in \mathcal{P}_i$*, i.e., the strategies are now

$$X_i = \{f^{(i)} : \mathcal{P}_i \to \{0, d_i\} \mid \sum_{P \in \mathcal{P}_i} f_P^{(i)} = d_i\}.$$

Intuitive difference between non-atomic and atomic selfish routing model: in the former, each commodity represents a large population of individuals, each of whom controls a negligible amount of traffic; in the latter, each commodity i represents a single player who must route a significant amount of traffic on a single path.

6.4.1 Equilibria in the atomic selfish routing model

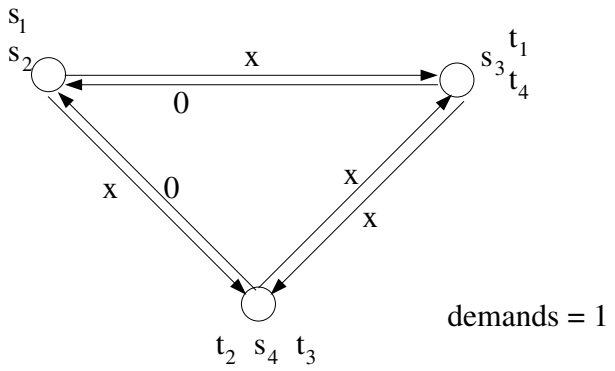
Definition 6.3. *A feasible flow $\hat{f}$ is an* atomic equilibrium flow *if for each $i \in N$ holds: if $f_P^{(i)} > 0$, then for all $\tilde{P} \in \mathcal{P}_i$*

$$\sum_{e \in P} c_e(\hat{f}_e) \leq \sum_{e \in \tilde{P}} c_e(\tilde{f}_e),$$

where $\tilde{f}$ is the feasible flow obtained from $\hat{f}$ by shifting the demand of player i from P to $\tilde{P}$.

6.4.2 Price of anarchy in atomic routing model

Example 6.1. *The following example shows that the PoA in atomic routing games might be larger than $\frac{4}{3}$ even for linear cost functions:* ... (see board).



6.4.3 Existence of equilibria for unique demands

Remark 6.5. *There are examples showing that atomic selfish routing games do not necessarily admit an equilibrium, if the players have different demands. (even for $|N| = 2$).*

Theorem 6.2. *Each atomic selfish routing game with unique demands, i.e., $d_i = d \; \forall i \in N$, admits at least one equilibrium flow.*

Proof. We might assume that $d = 1$. Set

$$\Phi_d(f) = \sum_{e \in E} \sum_{k=1}^{f_e} c_e(k) \quad \forall f \in X.$$

Take any flow f^* minimizing $\Phi_d(f)$ over all $f \in X$. Note that a minimum exists since $|X|$ is finite. Claim: f^* is an equilibrium flow. □

Proof of Claim. Suppose, for contradiction, that one player, say i, could strictly decrease her cost by deviating from $P \in \mathcal{P}_i$ to $\tilde{P} \in \mathcal{P}_i$. This yields a flow $\tilde{f} = f - P + \tilde{P}$. That is, we assume

$$0 > c_{\tilde{P}}(\tilde{f}) - c_P(f) = \sum_{e \in \tilde{P} \setminus P} c_e\left(f_e + 1\right) - \sum_{e \in P \setminus \tilde{P}} c_e\left(f_e\right), \tag{6.1}$$

where $c_P(f) \coloneqq \sum_{e \in P} c_e(f_e)$.

On the other hand, the impact of player i's deviation on the "potential" function Φ_d is as follows. There is an extra term $c_e(f_e + 1)$ for edges in $\tilde{P} \setminus P$ where we lose a term $c_e(f_e)$ for edges in $P \setminus \tilde{P}$. That is,

$$\Phi_d\left(\tilde{f}\right) = \Phi_d(f) + \sum_{e \in \tilde{P} \setminus P} c_e(f_e + 1) - \sum_{e \in P \setminus \tilde{P}} c_e(f_e) \overset{\text{Eq.(6.1)}}{<} \Phi_d(f).$$

□

Remark 6.6. *The proof did not need any assumptions on the cost functions!*

6.5 Network design games

In *network design games*, we are (again) given a directed graph $G = (V, E)$, as well as commodities $i \in N$, each of whom associated with a source s_i and a sink t_i.

In contrast to the routing models seen before, each player i's goal is now to establish connectivity between s_i and t_i at minimum cost.

Thus, the strategy set of player $i \in N$ is the set $\mathcal{P}_i$ of all $s_i - t_i$-paths in G. Each arc $e \in E$ is endowed with a (fix) cost $c_e \geq 0$ that needs to be paid by the players before it becomes available. The cost of an arc is shared among the players using it by a pre-described cost-sharing rule.

Definition 6.4. *A* Shapley network design game *is a network design game in which the costs c_e are shared evenly among the players using e.*

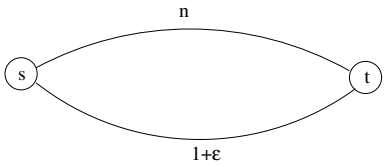
Given a profil $f = (P_1, \ldots, P_n) \in \mathcal{P}_1 \times \ldots \times \mathcal{P}_n$, we denote by f_e the number of players using $e \in E$.

In a Shapley network design game, player i would therefore need to pay $C^{(i)}(f) = \sum_{e \in P_i} \frac{c_e}{f_e}$.

The social cost of f is $C(f) = \sum_{e \in E(f)} c_e$, where $E(f) = \{e \in E \mid e \in P_i \text{ for some } i \in N\}$ are the arcs that are bought under profile f.

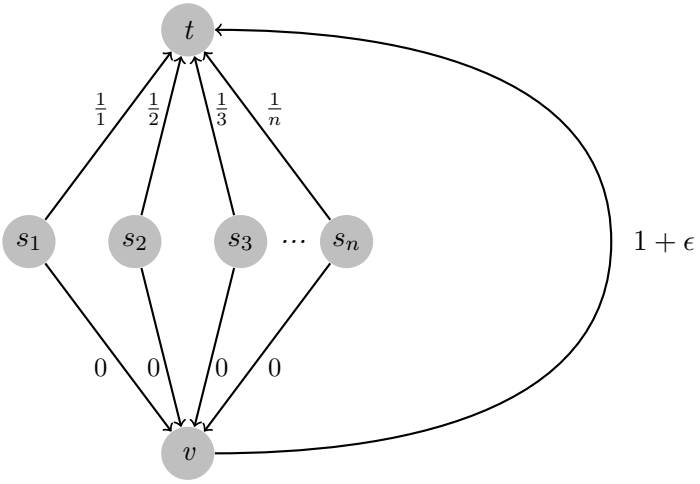
6.5.1 PoS in network design games

Example 6.2. *Consider the Shapley network design game illustrated below with n players and $\epsilon > 0$ arbitrarily small. What is the price of anarchy/stability?*



It is easy to see that all players choosing the upper route is an equilibrium. Note that all players choosing the lower route is a second equilibrium and the social optimum. Therefore we have a Price of Anarchy of $n/1+\epsilon$ and a Price of Stability of 1.

Example 6.3. *For an arbitrarily small $\epsilon > 0$, consider the Shapley network design game given by the graph below with n players, costs c_e on the edges, start nodes s_i for $i \in \{1, ..., n\}$ and the common target node t.*



It is easy to see that the price of stability is arbitrarily close to the n-th harmonic number $\mathcal{H}_n$, where

$$\mathcal{H}_n = \sum_{i=1}^{n} \frac{1}{i}.$$

6.6 Rosenthal's congestion games

Congestion games generalize atomic selfish routing games with $d_i = 1, i \in N$ by going over from paths in networks to arbitrary families of subsets of some common resource set E (not necessarily edges).

Definition 6.5. *A* congestion game $(N, \{X_i\}_{i\in N}, \{c_e\}_{e\in E})$ *consists of a finite set of players N, a finite set of resources E, a set-family $X_i \subseteq 2^E$ for each $i \in N$, and load-dependent cost functions $c_e : \mathbb{N} \to \mathbb{R}$ for each $e \in E$.*

After each player $i \in N$ has chosen one of his sets $S_i \in X_i$, the profil $S = (S_1, \ldots, S_n) \in X$ causes a load of $l_e(S) = |\{i \in N \mid e \in S_i\}|$ on resource $e \in E$. Each player $i \in N$ aims at minimizing the cost $C^{(i)}(S) := \sum_{e\in S_i} c_e(l_e(S))$.

Example 6.4. *$X_i = \mathcal{P}_i$ is the set of all $s_i - t_i$-paths in $G = (V, E)$.*

Note that a congestion game is a strategic games with finitely many strategies.

Example 6.5. *Scheduling games.*
Each player $i \in N$ is associated with a unit-sized job that needs to be processed on one of the machines in $E_i \subseteq E$. The processing time c_e of machine $e \in E$ depends on the load (= number of jobs processed) on e.

In a pioneering paper [10], Rosenthal proved:

Theorem 6.3. *Each congestion game* $(N, \{X_i\}_{i\in N}, \{c_e\}_{e\in E})$ *admits a PNE.*

Proof. Define the potential function

$$\Phi(S) = \sum_{e\in E} \sum_{t=1}^{l_e(S)} c_e(t).$$

If S is not a Nash equilibrium, at least one player i can perform an improvement step by switching from S_i to $\tilde{S}_i$. Since

$$C^{(i)}\left(S_{-i}, \tilde{S}_i\right) = \sum_{e\in\tilde{S}_i} c_e\left(l_e\left(S_{-i}, \tilde{S}_i\right)\right) < \sum_{e\in S_i} c_e\left(l_e(S)\right) = C^{(i)}(S),$$

it follows that

$$\Phi\left(S_{-i}, \tilde{S}_i\right) < \Phi(S).$$

Hence, a minimizer of Φ is an equilibrium. A minimum exists since there are only finitely many different profiles. □

6.7 Potential Games

Definition 6.6. *Given a strategic game* $(N, \{X_i\}_{i\in N}, \{u_i\}_{i\in N})$*, a function* $\Phi : X \to \mathbb{R}$ *is called* (exact) potential function *if*

$$\Phi(x) - \Phi(x_{-i}, x_i') = u_i(x_{-i}, x_i') - u_i(x) \quad \forall x \in X,\ i \in N,\ x_i' \in X_i.$$

In other words, if $x \in X$ is the current state of the game, and player i switches from strategy x_i to $x_i' \in X_i$, then the resulting benefit i incurs exactly matches the decrease in the value of the potential function Φ.

Observation 6.2. *Every game has at most one exact potential function, modulo addition by a constant.*

Definition 6.7. *Every game that admits an exact potential function is called* (exact) potential game.

For the remainder of this section, we will omit the term "exact" when talking about potential games.

Remark 6.7. *A surprising number of interesting games turn out to be potential games, and this structure has a number of strong implications for the existence of and convergence to equilibria.*

Theorem 6.4. *Every potential game has at least one PNE, namely each strategy-profile $x^* \in X$ minimizing $\Phi(x)$.*

Proof. Let $\Phi : X \to \mathbb{R}$ be a potential function for the potential game under consideration, and let $x^* \in X$ be a pure strategy-profile minimizing $\Phi(x)$. Consider any move from x^* to (x^*_{-i}, x'_i) by a player i from x^*_i to $x'_i \in X_i$.

By assumption, $\Phi(x^*_{-i}, x'_i) \geq \Phi(x^*)$, and by definition of a potential function, $u_i(x^*_{-i}, x'_i) - u_i(x^*) = \Phi(x^*) - \Phi(x^*_{-i}, x'_i)$. Thus, i's utility cannot increase from this move, and hence x^* is a PNE. □

We call a potential game finite, if there are only finitely many different strategy-profiles, i.e., if $|X| < \infty$.

A *best response dynamics* for strategic games is defined as follows. Starting at any strategy profile, as long as this strategy profile is not a Nash equilibrium, choose any player whose current strategy is not a best response and let this player deviate to her best response.

Observation 6.3. *In any finite potential game, best response dynamics always converge to a PNE.*

Proof. It follows from the definition of a potential function that any profile $x \in X$ with the property that Φ cannot decreased further by altering any one strategy in x is a PNE.

Furthermore, best response dynamics simulate local search[1] on Φ: improving moves for players decrease the value of the potential function Φ. □

Example 6.6. *Rosenthal's congestion games are potential games with potential function $\Phi : X \to \mathbb{R}$ that assigns to profile $S = (S_1, \ldots, S_n) \in X$ the value*

$$\Phi(S) = \sum_{e \in E} \sum_{t=1}^{l_e(S)} c_e(t).$$

Example 6.7. *Shapley network design games (also called "global connection games" in the literature) are congestion games with load-dependent costs $c_e(S) := \frac{c_e}{l_e(S)}$ on each edge $e \in E$. Hence, every such game is a potential game.*

Interestingly, potential games and congestion games are structurally the same, as the following theorem shows (the proof is omitted for the purpose of this lecture):

Theorem 6.5 (Monderer/Shapley 1996)**.** *For every potential game, there is a congestion game with the same potential function.*

[1] Recall that local search is a method that always searches for local improvement steps as long as this is possible.

6.7.1 Potential Games and the Price of Stability

We saw already for the special class of Shapley network design games, that the price of anarchy (PoA) can be arbitrarily bad. Hence, studying the price of stability (PoS) for potential games is, in some sense, more interestingly. As before, let $C(x) := \sum_{i \in N} u_i(x)$ denote the *social cost* of profile $x \in X$. We assume for the remainder of this section, that the potential game under consideration is a cost game, i.e., a game where the social optimum is a profile $\tilde{x}$ minimizing $C(x)$.

Theorem 6.6. *Suppose we have a potential game with potential function $\Phi : X \to \mathbb{R}$, and assume that for every profile $x \in X$, we have*

$$\frac{C(x)}{A} \leq \Phi(x) \leq B \cdot C(x),$$

for some constants $A, B \in \mathbb{R}$. The the price of stability is at most $A \cdot B$.

Proof. Let x^* be a profile minimizing the potential function $\Phi(x)$. We know that x^* is a PNE by the theorem above. It suffices to show that its social cost $C(x^*)$ is not much larger than the optimal social cost $C(\tilde{x})$, where $\tilde{x} \in X$ is some socially optimal profile. By assumption, we have

$$\frac{C(x^*)}{A} \leq \Phi(x^*). \tag{6.2}$$

Furthermore, since x^* is a minimizer of Φ, we know that

$$\Phi(x^*) \leq \Phi(\tilde{x}) \leq B \cdot C(\tilde{x}). \tag{6.3}$$

Plugging the two inequalities 6.2 and 6.3 together yields $\frac{C(x^*)}{A} \leq B \cdot C(\tilde{x})$, implying the desired bound on the PoS

$$\text{PoS} \leq \frac{C(x^*)}{C(\tilde{x})} \leq A \cdot B.$$

□

Remark 6.8. *This technique for bounding the PoS is called* potential function method.

6.7.2 Application to Shapley network design games

Recall that in Shapley network design games each player $i \in N$ aims to connect his two vertices $\{s_i, t_i\} \subseteq V$. The strategy set of i consists therefore of the collection $\mathcal{P}_i$ of all $s_i - t_i$-paths in $G = (V, E)$. The cost of an edge $e \in E$ is $c_e \geq 0$ which is shared evenly among the players using it.

We have seen that there exist instances of the game where the unique PNE has a social cost of $\mathcal{H}_n := \sum_{i=1}^{n} \frac{1}{i}$ (called the "n-th harmonic number"), whereas the social optimum has a cost of only $1 + \epsilon$ for some arbitrarily small value $\epsilon > 0$. Hence, the PoS of this instance is roughly $\mathcal{H}_n$.

Observation 6.4. *The Shapley network design game is a potential game with potential function Φ which assigns to each profile $S = (P_1, \ldots, P_n) \in X$ the value*

$$\Phi(S) = \sum_{e \in E : l_e(S) > 0} c_e \cdot \mathcal{H}_{l_e(S)},$$

where $l_e(S) := |\{i \in N : e \in P_i\}|$ is, as before, the load on e induced by S.

Proof. Let $S = (P_1, \ldots, P_n) \in X$ be a profile. Now consider the change in the potential function under the move of player i from P_i to P_i^*.

$$\begin{aligned}
&\Phi(S) - \Phi(S_{-i}, P_i^*) \\
&= \sum_{e \in P_i \setminus P_i^*} \left(c_e \mathcal{H}_{l_e(S)} - c_e \mathcal{H}_{l_e(S_{-i}, P_i^*)} \right) + \sum_{e \in P_i^* \setminus P_i} \left(c_e \mathcal{H}_{l_e(S)} - c_e \mathcal{H}_{l_e(S_{-i}, P_i^*)} \right) \\
&= \sum_{e \in P_i \setminus P_i^*} c_e \frac{1}{l_e(S)} + \sum_{e \in P_i^* \setminus P_i} -c_e \frac{1}{l_e(S) + 1} \\
&= C_i(S) - C_i(S_{-i}, P_i^*).
\end{aligned}$$

I.e. the change of the potential function under a move of player i is exactly the change of the cost of player i. Note that these potential functions are called *exact potential function.* □

Lemma 6.1. *For every profile $S = (P_1, \ldots, P_n) \in X$ we have*

$$C(S) \leq \Phi(S) \leq \mathcal{H}_n \cdot C(S).$$

Proof. Consider any edge $e \in E$ used under S. Therefore, $l_e(S) \geq 1$, implying $c_e \cdot \mathcal{H}_{l_e(S)} \geq c_e$. On the other hand, $c_e \cdot \mathcal{H}_{l_e(S)} \leq c_e \cdot \mathcal{H}_n$, since $l_e(S) \leq n$. It follows that

$$C(S) = \sum_{e \in E(S)} c_e \leq \sum_{e \in E(S)} c_e \cdot \mathcal{H}_{l_e(S)} = \Phi(S) \leq \mathcal{H}_n \cdot C(S).$$

□

Hence, we can apply the potential function method with constants $A = 1$ and $B = \mathcal{H}_n$ to conclude:

Corollary 6.7.1. *The price of stability in Shapley network design games is at most $\mathcal{H}_n$.*

Chapter 7

Selfish Load Balancing

Load balancing is a fundamental optimization problem: whenever a set of tasks should be executed on a set of resources (e.g. processors, machines, human workers, ...), one needs to balance out the load among the resources. One of the most fundamental load balancing problems is:

7.1 Makespan Scheduling on Uniformly Related Machines

Makespan scheduling on uniformly related machines is the problem to determine an assignment of tasks to machines that is a balanced as possible. The model can be described as follows:

Model: We are given

- a set $J = \{1, \ldots, n\}$ of tasks ("jobs"), where each job $i \in J$ has a certain weight $w_i \in \mathbb{R}_+$.
- a set $M = \{1, \ldots, m\}$ of machines ("resources"), where each machine $j \in M$ has a certain speed $s_j \in \mathbb{R}_+$.

Every assignment $A : J \to M$ of tasks to machines induces a *load* $l_j(A)$ on each machine $j \in M$ defined by

$$l_j(A) := \sum_{i \in J\colon A(i)=j} \frac{w_i}{s_j}.$$

The *makespan* of assignment A is the maximum load over all machines. Hence, the objective in load balancing is to find an assignment that minimizes the makespan, which we throughout call the *cost of* A, i.e.,

$$\text{cost}(A) = \max_{j \in M} l_j(A).$$

Makespan scheduling on identical machines. Makespan scheduling on identical machines is the special case where all machines have identical speed. Here, we can assume that $s_1 = \ldots = s_n = 1$.

Load balancing is a very typical algorithmic question: a "global authority" tries to compute an assignment of tasks to machines that minimizes the makespan.

Remark 7.1. *Makespan scheduling is (weakly) NP-hard, even on two identical machines. Hence, there exists no (strongly) polynomial-time algorithm to compute an optimal assignment.*

In this chapter, we are concerned with the following question:

"*What happens to the makespan it there is no global authority that can enforce a good assignment of tasks to machines, but instead, there are selfish users that decide how to assign the tasks to machines?*"

A typical Internet application is the setting where tasks correspond to requests for downloading large files. These requests are send by the users to servers (the "machines"). Certainly, each user tries to send his request to the least-loaded server.

7.2 Load Balancing Games

Load balancing games are a special class of strategic games. The model can be described as follows:

- the tasks $J = [n]$ correspond to the players,
- each task $i \in J$ needs to be assigned to one machine in $M = [m]$. Thus the strategy sets are
$$X_i = M \quad \forall i \in J.$$
- Therefore, every profile yields an assignment $A : J \to M$, where $A(i) \in M$ denotes the machine to which task i is assigned to.
- The *utility/ individual cost* of player $i \in J$ under assignment/profile A is the load of machine $A(i)$ to which i is assigned, i.e.,
$$u_i(A) = l_{A(i)}(A) \quad \forall i \in J.$$
- The *social cost* of assignment A is its makespan, denoted by cost(A).

Remark 7.2. *Load balancing games are strategic games where the social cost is* not *the utilitarian objective!*

7.3 Equilibria in load balancing games

Recall that a profile is a PNE if and only if none of the players can decrease his cost by deviating his strategy, while all remaining players stick to their strategies. Therefore:

Definition 7.1. *An assignment $A : J \to M$ is a PNE if and only if*

$$u_i(A) \leq \frac{w_i}{s_r} + \sum_{k \in J \setminus \{i\}:\ A(k)=r} \frac{w_k}{s_r} \quad \forall i \in J,\ r \in M.$$

Note that the right-hand-side of the inequality above is exactly the load of machine r after i switched from $A(i)$ to r.

Theorem 7.1. *Every instance of the load balancing game admits at least one PNE.*

Proof. An assignment A induces a *sorted load vector* $(\lambda_1, \ldots, \lambda_m)$, where λ_j denotes the load on the machine with j-highest load (ties broken abitrarily). If assignment A is *not* a PNE, then there is a player $i \in J$ that can perform an improvement step, i.e., this player can decrease his cost by moving to another machine. We show that the sorted load vector after performing this improvement step is lexicographically smaller than the preceding one. Hence, a PNE is reached after a finite number of improvement steps.

Suppose, given the sorted load vector $(\lambda_1, \ldots, \lambda_m)$ induced by assignment A, player i performs an improvement step by moving his task from machine $A(i) = j$ to machine k. Then, clearly, $k > j$. The improvement step decreases the load on j and increases the load on k. However, the increased load on k is smaller than λ_j, as otherwise, i would not have decreased his cost. Hence, the number of machines with load $\geq \lambda_j$ is decreasing. Furthermore, the load on all other machines with load $\geq \lambda_j$ remains unchanged. Consequently, the improvement step yields a sorted load vector that is lexicographically smaller than $(\lambda_1, \ldots, \lambda_m)$. □

7.3.1 Example of a bad equilibrium

Consider an instance with two machines of speed 1 and four jobs $J = \{1, \ldots, 4\}$ of weights $w_1 = w_2 = 1$ and $w_3 = w_4 = 2$. A social optimum (and also an equilibrium) is an assignment with exactly one large and one small job on each machine. This assignment has a social cost/makespan of 3. However, the assignment with both large jobs on one machine, and both small jobs on the other machine, is an equilibrium as well, but has cost 4. Hence, the PoA is at least $\frac{4}{3}$.

7.3.2 PoS and PoA in load balancing games

Note that in the previous example, the price of stability (PoS) is 1. It turns out, that this is true in every load balancing game:

Lemma 7.1. *The PoS in load balancing games is always* 1*.*

Proof. Take a socially optimal assignment, and let the players iteratively perform improvement steps until a PNE is reached. Since an improvement step never increases the makespan, the resulting PNE is socially optimal. □

Load balancing games furthermore belong to the class of strategic games where the worst equilibrium is not too bad: the following theorem states that the PoA can be bounded from above by $2 - \frac{2}{m+1}$. As we can see by the previous example, there exists an instance for which this bound is tight.

Theorem 7.2. *For load balancing games on m identical machines, the PoA is at most* $2 - \frac{2}{m+1}$*.*

Proof. Consider any assignment $A : J \to M$ which is a PNE. We need to show that $\text{cost}(A) \leq (2 - \frac{2}{m+1}) \cdot \text{OPT}$, where OPT denotes the makespan of a socially optimal assignment. Let j^* be the machine with highest load under A (i.e., $\text{cost}(A) = l_{j^*}(A)$), and let i^* be a task of smallest weight assigned to this machine j^*. We can assume that there are at least two tasks assigned to j^*, since otherwise, $\text{cost}(A) = \text{OPT}$, and the statement of the theorem holds trivially. Thus

$$w_{i^*} \leq \frac{1}{2} \cdot \text{cost}(A). \tag{7.1}$$

Suppose there is a machine $j \in M \setminus \{j^*\}$ with load less than $l_{j^*} - w_{i^*}$. Then A could not be a PNE as i^* could improve by switching to j. Hence, together with Equation (7.1), we observe that the following inequality holds for each $j \in M \setminus \{j^*\}$.

$$l_j \geq l_{j^*} - w_{i^*} \geq \text{cost}(A) - \frac{1}{2} \cdot \text{cost}(A) = \frac{1}{2} \cdot \text{cost}(A). \tag{7.2}$$

Now, observe that the cost of an optimal assignment cannot be smaller than the sum of all weights divided by the number of machines, i.e.,

$$\text{OPT} \geq \frac{1}{m} \sum_{i \in J} w_i. \tag{7.3}$$

Hence, we derive the following chain of inequalities:

$$\begin{aligned}
\mathrm{OPT} &\overbrace{\geq}^{(7.3)} \frac{1}{m}\sum_{i\in J} w_i \\
&= \frac{1}{m}\sum_{j\in M} l_j \\
&\overbrace{\geq}^{(7.2)} \frac{1}{m}[\mathrm{cost}(A) + (m-1)\cdot\frac{1}{2}\cdot\mathrm{cost}(A)] \\
&= \frac{1}{2m}(m+1)\cdot\mathrm{cost}(A).
\end{aligned}$$

As a consequence, the desired result follows, i.e.,

$$\mathrm{cost}(A) \leq \frac{2m}{m+1}\cdot\mathrm{OPT} = (2 - \frac{2}{m+1})\cdot\mathrm{OPT}.$$

□

Chapter 8

Mechanism Design without Money

Mechanism design without money has several applications in situations where there are significant incentive issues, but where money is infeasible or illegal. Examples include voting, organ donation, school choice, etc.

We start with a little example, the *house allocation problem* introduced by Shapley and Scarf [13]:

8.1 House allocation problem

In the house allocation problem, we are given n players, each initially owning a house, together with a total order $\prec_i$ over the n houses for each player i (with the interpretation that $j \prec_i k$ if and only if player i likes the house of player j more than the one of player k.) Note that player i need not necessarily like his own house the most. The task is to reallocate the houses to make the players better off.

Consider the following algorithm who Shapley and Scarf in [13] credited to Gale:

Top Trading Cycle Algorithm (TTCA):

WHILE players remain DO:

- let each remaining player point to his favorite house among the remaining houses. This induces a directed graph G on the remaining players in which every vertex has out-degree 1. Note that graph G has at least one directed cycle (why?). Self-loops are also counted as cycles.
- Reallocate as suggested by the directed cycles, with each player on a cycle giving his house to the player pointing to it.

- Drop all players and their house on the cycles from the player-list.

Observation 8.1. *The TTCA terminates with each player owning exactly one house. Moreover, each player can only be better off.*

Proof. The algorithm maintains the invariant that the remaining players still own their original house. When a player is dropped from the list, he gets a house that he likes better or he is on a self-loop. □

The following theorem states that under the TTCA assignment rule, no player has an incentive to misreport his preferences:

Theorem 8.1. *The TTCA induces a DSIC mechanism.*

Proof. Let N_j denote the players allocated (and dropped) in the j-th iteration of the TTCA when all players report truthfully. Each player of N_1 gets his first choice and hence has no incentive to misreport. Any player i of N_2 is not pointed to by a player of N_1 (otherwise, i would belong to N_1 rather than to N_2). Thus, no misreport of a player in N_2 nets a house originally owned by a player in N_1. Since each player in N_2 gets his first choice outside the houses owned by the players in N_1, he has no incentive to misreport.

In general, a player i in N_j is never pointed to in the first $j-1$ iterations by any player in $N_1 \cup \ldots \cup N_{j-1}$. Thus, whatever i reports, i will not receive a house in $N_1 \cup \ldots \mathbb{N}_{j-1}$. Since the TTCA gives i his favorite house outside this set, i has no incentive to misreport. □

Even better, the TTCA has the property that no coalition of players can find a re-allocation among its members in which all of them are better off. (Such a property is often called "group-strategy-proof" in the literature.)

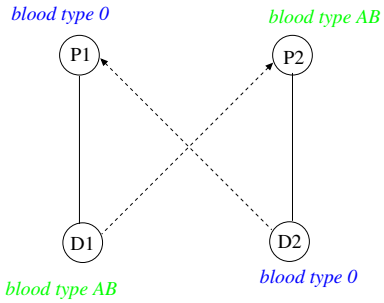
Theorem 8.2. *The TTCA has the property that no coalition $S \subseteq N$ can find an internal allocation in which all members of S are better off.*

Proof. Consider an arbitrary coalition S of the players. Define N_j as in the previous proof. Let l be the first iteration with $N_l \cap S \neq \emptyset$, and consider some player $i \in N_l \cap S$. That is, player i receives his house in the l-th iteration of TTCA. Recall that TTCA assigns i his favorite house outside the ones owned by players in $N_1 \cup \ldots \cup N_{l-1}$. Since no player of S belong to $N_1 \cup \ldots \cup N_{l-1}$, no re-allocation of houses among the players in S can make i strictly better off. □

8.2 Case Study: Kidney Exchange

Many people suffer from kidney failure and need a kidney transplant. Currently, only in the US, there are more than 100.000 people on the waiting list. One possibility which is also used for other organs is *deceased donors* – when a registered organ donor dies, his or her kidney can be transplanted into a patient in need of a new kidney. A special feature of kidneys is that a healthy person has two and can survive with only one of them. This gives the additional possibility of *living kidney donors*, like a family member or friend of a patient in need that is willing to donate his/her kidney.

However, sometimes a patient-donor pair is incompatible, mainly due to different blood types. For example, a patient with blood type 0 can only receive a kidney from a donor of blood type 0. Similiar, an AB donor can only donate to an AB patient. Suppose now, patient P1 has blood type 0, but his donor D1 has blood type AB. Now, suppose there is another pair where it is just the other way around: patient P2 has blood type AB, and donor D2 has blood type 0. Obviously, *kidney exchange* is a great idea: donor D2 gives his kidney to patient P1, and donor D1 gives his kidney to patient P2.



A few kidney exchanges were done, on an ad hoc basis, around the beginning of this century. These isolated successes make clear the need for a nation-wide, or even global-wide, kidney exchange where incompatible patient-donor-pairs can register and be matched with others.

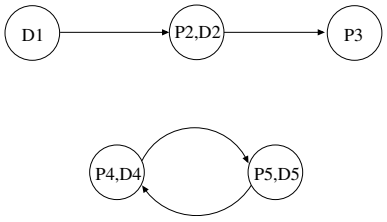
How should such an exchange be designed?

The overall goal is certainly to enable as many matches as possible. We will discuss a few early ideas, as well as current challenges.

Remark 8.1. *It should be noted that currently financial compensation for organ donation is illegal in all countries except for the Iran. It is a great discussion whether financial compensation should be allowed or not.*

8.2.1 First idea: Use the Top Trading Cycle Algorithm

This idea has been described in [11], before the authors extensively talked to doctors: The kidney exchange problem in its most basic form is similar to the house-allocation problem discussed earlier in the following sense: the incompatible living donors correspond to the houses, and the owners to the corresponding patients. Recall that in the house allocation problem, each player has a total ordering (or preference list) on the set of all houses. In kidney exchange, such a total ordering might be drawn from the estimated probability that a certain kidney can be successfully be transplanted into a patient. Whenever the Top Trading Cycle Algorithm (TTCA) detects a cycle, all patients within this cycle can "in theory" profit from the suggested exchange. In fact, the actual kidney exchange is in several ways more complex and needs for extensions. For example, a patient without living donors can be interpreted as a player without initial house, and a deceased donor can be interpreted as a house without initial owner. An extension of the TTCA, as described in [11], allocates along cycles and along suitable chains (starting with a "homeless" player, and ending in an empty house). The authors show that this extension of the TTCA remains DSIC – so no patient (or his/her doctor) can misreport information to increase the estimated probability of a successful transplant.



Big challenges/ draw-backs: The cycles computed throughout the Top Trading Cycle Algorithm might be way too long! The issue is that already cycles of length two correspond to four surgeries. Moreover, those surgeries must happen *simultaneously* for incentive reasons. For example, if the surgeries for P3 and D3 happen first, there is a risk that D1 will renege from his offer to donate his kidney to P2. However, too many simultaneous surgeries may exceed the capacities of operating rooms, surgical teams, etc. Hence, we need small cycles!

Another critique is that total ordering over the donors is overkill: empirically, patients don't really care which kidney they get as long as it is compatible. Therefore, binary preferences (either compatible or not) are more appropriate.

8.2.2 Second idea: use a matching algorithm

Recall that a *matching* of an undirected graph $G = (V, E)$ is a subset of edges $F \subseteq E$ such that no two edges in F share an endpoint.

The relevant graph for kidney exchange has a vertex set V corresponding to incompatible patient-donor pairs (one vertex per pair), and an undirected edge between vertices $(P1, D1)$ and $(P2, D2)$ is drawn if and only if $P1$ and $D2$ are compatible, and $P2$ and $D1$ are compatible.

Hence, our problem is to determine a **matching of maximum cardinality**.

Remark 8.2. *Note, that this way we are restricting to pairwise kidney exchanges.*

In our model, each vertex $i \in V$ has a *true* edge set E_i of incident edges, and can report any subset $F_i \subseteq E_i$ to the mechanism. Our mechanism design goal is to compute a max-cardinality matching, and to be DSIC, meaning that for every player reporting the full edge set is a dominant strategy. The mechanism therefore looks as follows in its basic version:

Mechanism (basic):

1. Collect a report F_i from each $i \in V$;
2. Form the edge set $E = \{\{i, j\} \mid \{i, j\} \in F_i \cap F_j\}$, i.e., include egde $\{i, j\}$ iff both endpoints agree to echange;
3. Return a matching of maximum cardinality in $G = (V, E)$;

The right tie-breaking rule (there might be several max-cardinality matchings) turns out to be crucial for the mechanism to be DSIC. Note that different max-cardinality matchings can match different subsets of vertices (cf. a star-graph). So the question is, which max-cardinality matching the mechanism should return given that the desired goal is to be DSIC?

Possible solution: Prioritize the vertices before the mechanism begins. (Note that such a prioritization is quite natural: most hospitals already rely on similar priority schemes, e.g., by waiting time or difficulty of finding a compatible kidney.) Now, suppose the vertices $1, 2, \ldots, n$ are ordered from highest to lowest priority. Then we implement step (3.) of the basic mechanism above as follows:

(3.a) Let $\mathcal{M}_0$ denote the set of max-cardinality matchings in G;

(3.b) For $i = 1, 2, \ldots, n$ do

- Let $\mathcal{Z}_i$ denote the matchings in $\mathcal{M}_{i-1}$ that cover vertex i;

- If $\mathcal{Z}_i \neq \emptyset$, set $\mathcal{M}_i = \mathcal{Z}_i$;
- Otherwise, set $\mathcal{M}_i = \mathcal{M}_{i-1}$;

(3.c) Return an arbitrary matching of $\mathcal{M}_n$;

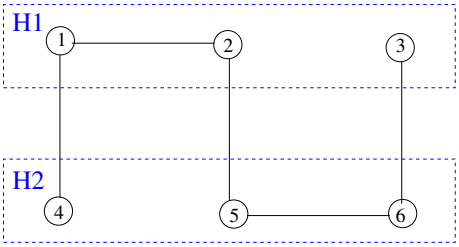
Theorem 8.3. *The mechanism is DSIC: no patient can go from unmatched to matched by reporting a strict subset of E_i.*

Proof. The proof is omitted for the purpose of this lecture. A nice explanation which relies on the Gallai-Edmonds-Decomposition of graphs is presented in [11]. □

Current research on kidney exchange is focused on incentive problems at the *hospital level*, rather than at the level of individual patient-donor pairs. This perspective is well motivated, since many patient-donor pairs are reported to national kidney exchanges by hospitals.

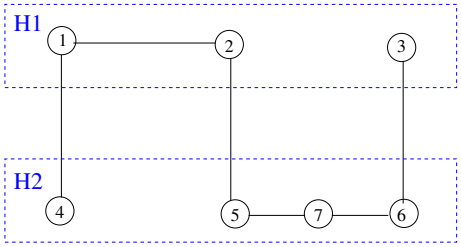
The issue is that the objective of the hospitals (to match as many of its patients as possible), and of society (to match as many patients overall as possible) are not perfectly aligned. The key incentive issues are best explained through examples:

Example 8.1. *Consider an instance with two hospitals with three incompatible patient-donor pairs each with exchange possibilities as illustrated by the graph in Figure 8.1 In this example, full reporting by hospitals leads to*



more matches than with only internal matches. If the hospitals act "greedily", execute internal matches first and report only pairs they cannot match, both hospitals are worse off.

In general, to save as many lives as possible, the mechanism design goal should be to incentive hospitals to report all of their patient-donor pairs. However, the following example shows that there is an irreconcilable tension between social and hospital incentives: there cannot be a DSIC mechanism that always computes a maximum cardinality matching:



Example 8.2. *In the example as depicted in Figure 8.2, hospital H1 has an incentive to report only patient-donor pair 3 and not the whole set.*

Hence, the revised goal should be to compute an approximate maximum cardinality matching (i.e., a matching that is not "too far away" from a maximum cardinality matching) so that, for each participating hospital, the number of its patients that get matched is approximately optimal.

8.3 Stable Matchings

Stable matchings is the canonical example for mechanism design without money. This concept has several applications, including the assignment of medical school graduates to hospitals, students to universities, students/teachers to classrooms and time slots, etc.

In our model, the player set consists of two disjoint sets U and V with $|U| = |V| = n$. These two sets are represented by two vertex sets U and V. (It might be helpful to think of U and V as the "men" and "women" between which we would like to arrange marriages. For this reason/interpretation, the stable matching problem is also well-known under the name "stable marriage problem".)

Each vertex has a total ordering of the vertices on the other side. That is, each men $u \in U$ has a total ordering $\prec_u$ on the set V of women, and each woman $v \in V$ has a total ordering $\prec_v$ on the set U of men, such that

$$v \prec_u v' \quad \text{if and only if} \quad \text{"man } u \text{ likes woman } v \text{ more than woman } v'.$$

Definition 8.1. *A* stable matching *is a perfect matching (i.e., a matching where each vertex is matched to exactly one vertex on the other side) if there is no* blocking pair*, meaning*

(*) if $u \in U$ and $v \in V$ are not *matched, then either u prefers his matching neighbor v' to v, or v prefers his matching neighbor u' to u.*

Remark 8.3. *A perfect matching that fails condition* $(*)$ *would run into trouble: the blocking pair* $\{u, v\}$ *would be tempted to run off with each other.*

The following, famous *proposal algorithm* goes back into the early 60's to Gale and Shapley [5]. Interestingly, as it has been later discovered, this algorithm is essentially the same algorithm that has been used (and still is used) since the 1950s, to assign medical residents to hospitals.

Proposal algorithm:

- While there is an unattached man $u \in U$
  - u proposes to its favorite woman who has not rejected him yet;
  - each woman entertains only her best offer (from her perspective) thus far;

Theorem 8.4. *The Proposal Algorithm terminates after at most* n^2 *iterations with a stable matching.*

Proof. The proof is omitted. The interested reader is referred to [5], or any standard text book covering stable matchings (e.g. the book "Algorithmic Game Theory"). □

Bibliography

[1] Shuchi Chawla, Nicole Immorlica, and Brendan Lucier. On the limits of black-box reductions in mechanism design. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 435–448. ACM, 2012.

[2] Edward H Clarke. Multipart pricing of public goods. *Public choice*, 11(1):17–33, 1971.

[3] Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.

[4] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. Technical report, National Bureau of Economic Research, 2005.

[5] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *American mathematical monthly*, pages 9–15, 1962.

[6] Theodore Groves. Incentives in teams. *Econometrica: Journal of the Econometric Society*, pages 617–631, 1973.

[7] Jason Hartline and Robert Kleinberg. The badminton and the science of rule making. *The Huffington Post*, 2012. `http://www.huffingtonpost.com/jason-hartline/badminton-and-the-science-of-rule-making_b_1773988.html`.

[8] Roger B Myerson. Optimal auction design. *Mathematics of operations research*, 6(1):58–73, 1981.

[9] Hukukane Nikaidô and Kazuo Isoda. Note on non-cooperative convex games. *Pacific Journal of Mathematics*, 5(Suppl. 1):807–815, 1955.

[10] Robert W. Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973.

[11] Alvin E Roth, Tayfun Sonmez, and M Utku Unver. Kidney exchange. Technical report, National Bureau of Economic Research, 2003.

[12] Tim Roughgarden. The price of anarchy is independent of the network topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003.

[13] Lloyd Shapley and Herbert Scarf. On cores and indivisibility. *Journal of mathematical economics*, 1(1):23–37, 1974.

[14] Hal R Varian. Position auctions. *international Journal of industrial Organization*, 25(6):1163–1178, 2007.

[15] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37, 1961.