Algorithmic Game Theory HS2010 [1]

Stefan Heule

January 23, 2011

Contents

1 Introduction

1.1 Examples

In the prisoner's dilemma, there is only one pure Nash equilibrium, namely if both players confess.

		Prisoner 2: confess	Prisoner 2: silent
Prisoner 1	confess	$4, 4$	$1, 5$
	silent	$5, 1$	$2, 2$

The prisoner's dilemma can be extended to many players. One extension is known as the pollution game, where n countries try to control pollution. Every country can either control their pollution at a cost of 3, or ignore the problem (without any costs). However, if the problem is ignored by one country, this adds 1 to the costs of all countries (including the one ignoring the problem).

The only stable solution is when no country controls pollution, having a cost of n for every country. In contrast, if they would collaborate, everyone would only pay 3 (social optimum).

1.2 Games, Strategies, Costs and Payoffs

Definition 1.1 (Strategic game (simultaneous game))**.** *A simultaneous move game consists of a set of n players, usually* $\{1, \dots, n\}$*. Each player* i *has his own set of strategies* S_i*. We denote by* $s = (s_1, \dots, s_n)$ *the vector of all strategies picked by the players, and by* $S = S_1 \times \cdots \times S_n$ *the set of possible outcomes.*

The players have a preference ordering *over* S*. The simplest possibility is to assign a number to each outcome for every player:*

$$\begin{aligned} \textit{cost:} \quad & c_i : S \to \mathbb{R} \\ \textit{payoff/utility:} \quad & u_i : S \to \mathbb{R} \end{aligned}$$

Clearly, the two notions can be used interchangeably, as $u_i(s) = c_i(s)$*.*

We make two assumptions about players in such games:

- Players are *rational*: they peruse well-defined goals (e.g. optimizing their payoff).
- Players are *strategical*: their actions take into account the knowledge about the other players.

1.3 Solution Concepts

1.3.1 Dominance

Definition 1.2. *A strategy* $s_i \in S_i$ strictly dominates $s_i' \in S_i$ *if*

$$\forall s_{-i} \in S_{-i} : \; u_i(s_i, s_{-i}) > u_i(s_i', s_{-i})$$

A strategy $s_i \in S_i$ weakly dominates $s_i' \in S_i$ *if*

$$\forall s_{-i} \in S_{-i} : \; u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i})$$

A strategy $s_i \in S_i$ dominates $s_i' \in S_i$ *if*

- s_i *weakly dominates* s_i'
- $\exists s_{-i} \in S_{-i} : \; u_i(s_i, s_{-i}) > u_i(s_i', s_{-i})$

Note the use of $S_{-i} = S_1 \times S_{i-1} \times S_{i+1} \times S_n$, and s_{-i} as elements from this set as an abbreviation. Also, we used (x, s_{-i}) as $(s_1, \dots, s_{i-1}, x, s_{i+1}, \dots, s_n)$.

Definition 1.3 (dominance principle)**.** *Don't play [strictly/weakly/-] dominated strategies.*

For some games, the dominance principle leads to a solution by iteratively eliminating dominated strategies.

1.3.2 Pure strategy Nash Equilibrium

Definition 1.4 (Pure strategy Nash equilibrium)**.** *A strategy profile $s \in S$ with $s = (s_1, \ldots, s_i, \ldots, s_n)$ is called a* (pure strategy) Nash equilibrium *if*

$$\forall i : \forall s_i' \in S_i : u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i})$$

This definition is very intuitive: s is a Nash equilibrium if no player can unilaterally improve its outcome. That is, no player gets a higher payoff by switching its strategy (while all other players stay). Note that s_i is the solution of the following maximization problem:

$$\max_{s_i' \in S_i} u_i(s_i', s_{-i})$$

However, there are games which do not have such a pure strategy Nash equilibrium.

Definition 1.5 (Best response)**.** *For $s_{-i} \in S_{-i}$ the strategy $s_i \in S_i$ is called a* best response against s_{-i} *if*

$$s_i = \arg\max_{s_i' \in S_i} u_i(s_i', s_{-i})$$

Lemma 1.1. *In a pure strategy NE (Nash equilibrium) $s \in S$, every player plays best response in s.*

1.3.3 Mixed-strategy Nash Equilibrium

Definition 1.6 (Mixed strategy)**.** *A* mixed strategy *of player i in a strategic game is a probability distribution over S_i. Understanding:*

- *Each player i chooses a strategy $s_i \in S_i$ randomly and independently using the probability distribution.*
- *Every player wants to maximize her payoff.*
- *Risk-neutrality: Expected payoff is maximized regardless of risk. For instance, 5 for sure is equally preferred as 10 or 0, each with probability $\frac{1}{2}$.*

Definition 1.7 (Mixed-strategy Nash equilibrium)**.** *A vector of mixed strategies (one for each players) is a mixed NE if no player can unilaterally change her mixed strategy and improve the payoff.*

Definition 1.8 (Best response with mixed strategies)**.** *The* best response *of player i against mixed strategies of other players is a mixed strategy of player i that maximizes her expected payoff, given what the others do.*

A game with two players can be described by two matrices A and B, where A denotes the payoffs of player 1, and B for player 2. Such a game is also called *bi-matrix game*. A mixed strategy of a player can be written as $p = (p_1, p_2, \ldots, p_n)$, where n is the number of strategies available to the player. Clearly, we require $0 \leq p_i \leq 1$ for all i and $\sum_i p_i = 1$.
Using this notation, we easily can compute the expected payoff for the two players:

- Expected payoff of player 1: $p \cdot A \cdot q^T$
- Expected payoff of player 2: $p \cdot B \cdot q^T$

Definition 1.9 (Support of a mixed strategy)**.** *The* support of a mixed strategy *is a set of pure strategies that are played with positive probability.*

Proposition 1.1. *A mixed strategy is a best response if and only if every pure strategy in its support is a best response.*

The following general rule can be applied to find a mixed NE:

- Fix supports S_A and S_B (there are exponentially many)
- Solve a system of linear equations
  - $(A \cdot q)_i = (A \cdot q)_j \quad \forall i, j \in S_A$
  - $(A \cdot q)_i \leq (A \cdot q)_j \quad \forall i \notin S_A, j \in S_A$

- the same equations for S_B and $(p \cdot B)$
- $\sum p_i = \sum q_i = 1$
- $\forall i: \quad (i \in S_A) \implies p_i > 0$
- $\forall i: \quad (i \notin S_A) \implies p_i = 0$
- the same equations for S_B and q_i.

Theorem 1.1 (Nash 51'). *Every finite n-player strategic game has a mixed NE. Finite in this context means that n is finite, and that every player only has a finite number of strategies.*

Theorem 1.2 (Brouwer's fixed point theorem). *Let $S \subseteq \mathbb{R}^n$. Every continuous function $f : S \to S$ where S is convex, closed and bounded, has a fixed point, i.e.*

$$\exists s \in S: \ f(s) = s$$

We now consider triangulations (i.e. splittings into multiple triangles), and Sperner colorings. A Sperner coloring is a coloring of a triangulation of a triangle with edges A, B and C. This coloring has the following properties:

- A has color 1, B color 2 and C color 3.
- Any vertex on AB have color 1 or 2, on BC colors 2 or 3, and colors 1 or 3 for AC.
- Any other vertex has an arbitrary coloring.

Lemma 1.2 (Sperner's lemma). *Any Sperner coloring of a triangulation of a triangle has a small triangle of all colors.*

Proof sketch. We consider an edge between two vertices of color 1 and 2 as a door. On AB there is at least one door, so we enter it. Now either the triangle has all colors, or it has a second door which we can follow.

Note that if we get to the outside again, we can get in through an unused door, as the number of doors on AB is odd (start with no vertices on AB (there is only one door), and inductively add vertices. We increase the number of doors by 2 or 0).

It is not possible to loop inside the big triangle. Otherwise, consider the first triangle we visit twice. Since a triangle only has two doors, we cannot enter through an unused door, i.e. we enter through a used door. But now, the triangle behind this used door has also been visited twice, i.e. our triangle was not the first.

As there are only finitely many triangles, we eventually must find a triangle with all colors. □

Proof of Brouwer's theorem. We proof the theorem for $S \subseteq \mathbb{R}^n$ where S is a triangulation. For higher dimensions, everything is analogous, and S can be homomorphically mapped to any set S'.

Rather than Cartesian coordinates, we use baigcentrice coordinates where a point p is represented as triple (a, b, c) representing the Cartesian point $aA + bB + cC$, with $a + b + c = 1$ and $a, b, c \geq 0$. A, B and C are the edges of the triangle S.

We start with the full triangle S as a triangulation and color any point p which is not a fixpoint as follows ($\bar{p} = f(p)$, $p = (a, b, c)$, $\bar{p} = (\bar{a}, \bar{b}, \bar{c})$):

- if p moves way from A, color it 1, i.e. if $\bar{a} < a$,
- else if p moves away from B, color it 2, i.e. if $\bar{b} < b$,
- otherwise color it 3.

We consider any triangulation (without a fixed point), then the coloring introduced gives a Sperner's coloring:

- $A = (1, 0, 0)$ is not a fixpoint (there is none), thus it gets color 1, as we can only move away from A. The same for B and C.
- points on AB have coordinates $(a, b, 0)$, and can only get color 1 or 2, as we cannot move away from C. Analogously for BC and AC.

Using this result, we consider sequences of triangulations $\{T_i\}_{i=1}^{\infty}$, where the triangles get strictly smaller in each step.

The rest of the proof is left as an exercise. □

Using Brouwer's theorem, it is now fairly easy to proof for two-player games the existence of a mixed NE.

Proof. Let P be the space of mixed strategies for player 1, and Q the same set for player 2. We set $S = P \times Q$, and define $f : S \to S$ as $f(p, q) \to (\bar{\bar{p}}, \bar{\bar{q}})$.
If (p, q) is not a NE, then there is a strategy mixed that does not achieve the maximum of pure strategies, i.e.

$$\exists i : \ (A \cdot q^T)_i < p \cdot A \cdot q^T$$

Thus, we set $\bar{p}_i = p_i + \mathcal{X}_i(p, q)$, where

$$\mathcal{X}_i(p, q) = \min\{0, (A \cdot q^T)_i - p \cdot A \cdot q^T\}$$

and analogously for $\bar{q}_i$. Now we can normalize such that the values sum up to 1, i.e.

$$\bar{\bar{p}}_i = \frac{\bar{p}_i}{\sum_j \bar{p}_j} \qquad \text{and} \qquad \bar{\bar{q}}_i = \frac{\bar{q}_i}{\sum_j \bar{q}_j}$$

Now, using Brouwer's theorem, we know that f has a fixed point (p^*, q^*), and (p^*, q^*) is a NE, since no player can improve. If there would be a player i that can improve, $\mathcal{X}_i(p^*, q^*)$ would not be zero, and thus (p^*, q^*) not a fixed point. □

1.4 Quantifying the Inefficiency of Equilibria

We now consider two measures of the inefficiency of equilibria, the price of anarchy as a worst-case approach (PoA), and the price of stability (PoS). The price of stability can be useful in cases where a global entity (e.g. a government) can prescribe a starting position. If this position is a NE, then nobody would want to change.

Definition 1.10 (Price of anarchy)**.** *The* price of anarchy *is defined as the ratio between the worst objective function value of an equilibrium, and that of an optimal outcome. That is*

$$\text{PoA} := \frac{\textit{worst Nash value}}{\textit{optimal value}}$$

Definition 1.11 (Price of stability)**.** *The* price of stability *is defined as the ratio between the best objective function value of an equilibrium, and that of an optimal outcome. That is*

$$\text{PoS} := \frac{\textit{best Nash value}}{\textit{optimal value}}$$

1.5 Potential games

Definition 1.12 (Potential function)**.** *A function $\Phi : S \to X$ is called a potential function, if*

- *The set X is totally ordered, and*
- *if $s \in S$ is not a NE, then there exists $s' \in S$ such that $\Phi(s') < \Phi(s)$.*

If the strategic space S is finite, there is a $s \in S$ that minimizes Φ, and this s is a NE.

1.5.1 Load balancing game

There are m machines, all identical and n jobs with weights W_i. The jobs are selfish agents and have strategies $S_i = \{1, \ldots, m\}$, that is they can choose a machine. The latency of a machine j is defined as

$$l_j(s) = \sum_{\{i : s_i = j\}} W_i$$

and the objective of the agents is given by

$$f_i(s) = \text{fct}(l_{s_i}(s))$$

where fct is a non-decreasing, monotone function. The social goal is to optimize the makespan, that is to minimize

$$f(s) = \max_j l_j(s) \quad \to \quad \min$$

This is not utilitarian (minimize a sum) anymore, but *egalitarian*.

Theorem 1.3. *There exists a pure strategy NE in the load balancing game.*

Proof (using a potential function). We define $\Phi : S \to X$, where X is the set of ordered load vectors. That is, we consider vectors $(l_1, \ldots, l_m)$ where $l_1 \geq \cdots \geq l_m$. The total order is given by the lexicographic ordering. Note that Φ actually is a potential function, since if s is not a NE, then there is an outcome s' where someone can improve his utility by switching (i.e. he decreases one value in the vector, and increases another. But the one he increases is smaller (otherwise he would not improve), and thus $\Phi(s') < \Phi(s)$. □

This potential function can also be used to show other properties of this game. We can observe that $\Phi(s') < \Phi(s) \implies f(s') \leq f(s)$. This can easily be seen, as the makespan (i.e. $f(s)$) is just the first entry of $\Phi(s)$, namely the largest load.

Claim 1.1. *The price of stability for this load balancing game is 1.*

Proof. Let s be the optimum OPT. If s is a NE, we are done. If not, the potential function guarantees us a s' with $\Phi(s') < \Phi(s)$, and thus $f(s') \leq f(s) = f(\text{OPT})$ (since the social optimum minimizes f, we really have $f(s') = f(s)$. Now we set s to s', and start again. If s is a NE, we're done, otherwise we again find a s', and so on. As there are only finitely many strategic profiles, we must reach at some point a NE that has the same value as the social optimum. □

If s is a NE, then we have for all jobs i

$$\forall j : \; l_{s_i}(s) \leq l_j(s) + w_i$$

This is no different than saying that job i cannot improve by switching to a different machine.

To analyses the price of anarchy, we first give two trivial lower bound for the optimal solution OPT:

$$\text{OPT} \geq \max_i w_i \qquad \text{and} \qquad \text{OPT} \geq \frac{1}{m} \sum_i w_i$$

We rename the machines and jobs such that machine 1 defines the makespan (i.e. runs the longest), and job 1 is the lightest job on machine 1. We know

$$\begin{aligned} l_1(s) &\leq l_2(s) + w_1 \\ l_1(s) &\leq l_3(s) + w_1 \\ &\vdots \\ l_1(s) &\leq l_m(s) + w_1 \\ (m-1) \cdot l_1(s) &\leq \sum_{i=2}^{m} l_i(s) + (m-1) \cdot w_1 \end{aligned}$$

Adding $l_1(s)$ on both sides results in

$$\begin{aligned} m \cdot l_1(s) &\leq \sum_{i=1}^{m} l_i(s) + (m-1) \cdot w_1 \\ m \cdot l_1(s) &\leq \sum_{i=1}^{n} w_i + (m-1) \cdot w_1 \\ l_1(s) &\leq \underbrace{\frac{1}{m} \sum_{i=1}^{n} w_i}_{\leq \text{OPT}} + \frac{m-1}{m} \cdot w_1 \end{aligned}$$

Therefore, we know $l_1(s) \leq \text{OPT} + \frac{m-1}{m} \cdot w_1$. By using a case distinction, we can get a bound on the price of anarchy.

- Job 1 is alone on machine 1, i.e. job 1 is the longest job, and thus we have the optimal solution. This would mean PoS = PoA = 1.

- Job 1 is not alone, and is the lightest on machine 1 (as defined). Clearly, $w_1 \leq l_1(s)/2$. Therefore, we have

$$l_1(s) \leq \mathrm{OPT} + \frac{m-1}{m}\frac{l_1(s)}{2} \quad \Longrightarrow \quad \frac{m+1}{2m} l_1(s) \leq \mathrm{OPT}$$

  Further simplifying and using the results from above, we get

$$f(s) = l_1(s) \leq \frac{2m}{m+1}\,\mathrm{OPT} = \left(2 - \frac{2}{m+1}\right)\mathrm{OPT} < 2 \cdot \mathrm{OPT}$$

This means, that the price of anarchy is bounded by $(2 - 2/(m+1))$. Note that this bound is tights, i.e. there exists examples where it is reached.

1.6 Network Design Games

1.6.1 Atomic Flow Example (Shapely)

In this example we have k players, and the payments are according to Shapley, that is the cost for a link are shared among all players using it. Every player can choose a path from his start node to t, and wants to minimize its cost. Figure 1 shows an example of such a network.

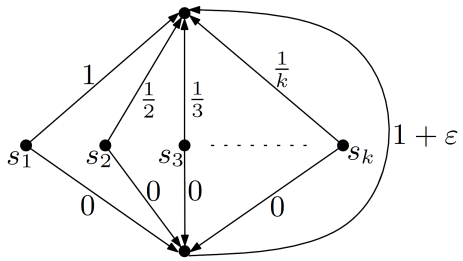

Figure 1: Atomic flow with bad PoA.

The optimal solution is to route all traffic through the $1+\varepsilon$ path, resulting in an average cost of $\frac{1+\varepsilon}{k}$ per player. However, this is not a NE, as player k can improve by choosing his alternative path of cost $\frac{1}{k}$. But then, also player $k-1$ wants to switch, and so on, resulting in the situation where every player chooses the direct path. This leads to a price of stability of essentially $\ln k$, which is bad.

1.6.2 Non-Atomic Traffic Routing (Pigou's Example)

We have the network shown in figure 2, and want to achieve a rate of 1. That is, we would like to route one unit of traffic, and we an split the traffic in any way (non-atomic traffic).

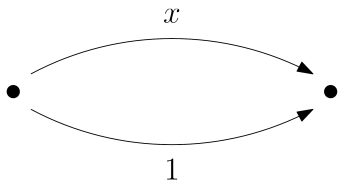

Figure 2: Pigou's example.

The only NE is when all the traffic is routed on the top arch, inducing a cost of 1 per player. However, it is easy to see that in the optimal solution, the traffic is split into equal parts, resulting in an average cost per player of $\frac{3}{4}$. Thus, the price of anarchy/stability is $\frac{4}{3}$.

1.6.3 Variant of Pigou's Example

A simple variant of Pigou's Example is the same network topology, with one edge having again cost 1, but the other having x^n for a large n. The best solution in this case would be to route ε on the edge with cost 1, and the rest on the second edge. This results in a total cost of $\varepsilon + (1-\varepsilon)^{n+1}$ which tends to 0 for n to infinity and ε to zero.

However, the worst (and only) NE still has a total cost of 1, which means that the price of anarchy is unbounded.

1.6.4 Braess's Paradox

Braess's paradox states, that in some networks, where the moving entities selfishly choose their path, adding network resources can reduce the overall performance. An example for such a network is given in figure 3.

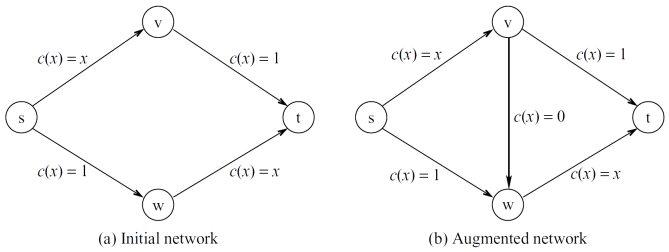

Figure 3: Instance of Braess's paradox.

In the initial network (again with one unit of traffic) the traffic splits evenly resulting in all traffic experiencing 3/2 units of cost. In the augmented network however, the unique equilibrium flow uses the new edge, resulting in cost 2 for all traffic.

1.6.5 General Non-Atomic Selfish Routing

A selfish routing game occurs in a *multicommodity flow network*, or simply a network. A network is given by a directed graph $G = (V, E)$, with vertices V and edges E, together with a set $(s_1, t_1), \ldots, (s_k, t_k)$ of source-sink pairs. We also call these pairs *commodities*. Each player is identified with one commodity. We use $\mathcal{P}_i$ to denote the s_i-t_i paths of a network and only consider networks where $\mathcal{P}_i \neq \emptyset$.

We describe the routes chosen by players using a *flow*, which is simply a non-negative vector indexed by the set $\mathcal{P}$ of source-sink paths. For a flow f and a path $P \in P_i$, we interpret f_P as the amount of traffic of commodity i that chooses the path P to travel from s_i to t_i. For every commodity there is a prescribed amount r_i of traffic. A flow is *feasible* for a vector r if it routes all of the traffic: for each $i \in \{1, 2, \ldots, k\}$, $\sum_{P \in P_i} f_P = r_i$. In particular, we do not impose explicit edge capacities.

Finally, every edge e has a *cost function* $c_e : \mathbb{R}^+ \to \mathbb{R}^+$. We are interested in a specific type of network, where we have affine latencies. That is, the latency of edge e can be written as $l_e(f_e) = q_e \cdot f_e + b_e$, where q_e and b_e are constants. Thus, the cost of a flow can be written as

$$C(f) = \sum_i \sum_{P \in \mathcal{P}_i} l_P \cdot f_P = \sum_e l_e(f_e) \cdot f_e \qquad \text{where} \qquad l_P := \sum_{e \in P} l_e$$

Definition 1.13. *Let f be a feasible flow. Then f is an* equilibrium flow *(also called* Nash flow*) if, for every commodity i and every pair $P, Q \in \mathcal{P}_i$ of s_i-t_i paths with $f_P > 0$*

$$c_P(f) \leq c_Q(f)$$

In other words, all paths in use by an equilibrium flow f have minimum-possible cost (given their source, sink, and the congestion caused by f). In particular, all paths used have equal cost.

In the case of affine latencies, we can reformulate the definition of a Nash flow as follows:

$$\forall P, Q \in \mathcal{P}_i : \; f_P > 0, f_Q > 0 : \; \sum_{e \in P} a_e \cdot f_e + b_e = \sum_{e \in Q} a_e \cdot f_e + b_e$$

Furthermore, for an optimal flow f^* it holds (for slightly unknown reasons):

$$\forall P, Q \in \mathcal{P}_i : \; f^*_P > 0, f^*_Q > 0 : \; \sum_{e \in P} 2 \cdot a_e \cdot f^*_e + b_e = \sum_{e \in Q} 2 \cdot a_e \cdot f^*_e + b_e$$

1.7 Extensive Games

An extensive game is a dynamic game where players play in turns. The game can be described as a tree, where internal nodes are decision nodes (or special random nodes), where players make a move, and leaves represent the end of the game. At the leaves there is a vector of utilities.

Such games can be solve by a backwards induction by starting at the leaves of the tree.

2 Congestions Games

Definition 2.1 (Congestion game)**.** *A* congestion game *consists of n players $\{1, \dots, n\}$ and m resources $M = \{1, \dots, m\}$. The player can choose any subset of M as a strategy. However, the concrete instance of the game might further restrict the set of valid strategies. The costs per resource are given by a function $c_j(k)$ which specifies the cost if k players are using it.*
For a strategic profile $s = (s_1, \dots, s_n)$ the cost of player i can be calculated as

$$u_i(s) = \sum_{j \in s_i} c_j(n_j(s)) \qquad n_j(s) = \#\text{ of players using resourse } j \text{ in } s$$

We can analyze this class of games using the potential function method. But first, we introduce Δu_i as the change that happens if i switches from strategy s_i to s_i':

$$\Delta u_i = u_i(s_i', s_{-i}) - u_i(s_i, s_{-i}) = \sum_{j \in s_i' \setminus s_i} c_j(n_j(s) + 1) - \sum_{j \in s_i \setminus s_i'} c_j(n_j(s))$$

If $\Delta u_i < 0$, we call this an improvement step. We now define a potential function Φ with $\Delta\Phi = \Delta u_i$ as follows

$$\Phi(s) := \sum_{j=1}^{m} \sum_{i=1}^{n_j(s)} c_j(i)$$

This potential function is called *Rosenthal's potential function.* We arrive at this potential function by looking at the game from a historic perspective, where we start with zero players, and they join the game one after another. For every resource j we have $c_j(1)$ for the first player using it, $c_j(2)$ for the second, and so on, up to $c_j(n_j(s))$. This is precisely the potential function.

Theorem 2.1 (Rosenthal, 73')**.** *Every congestion game is a potential game.*

Theorem 2.2 (Monderer/Shapley, 96')**.** *Every potential game is a congestion game.*

Such potentials have a lower and upper bound, thus we can decrease only a finite number of times.

Corollary 2.1. *Finite potential games have a pure NE. This NE can be reached by strategy changes in a finite number of steps.*

We can identify three types of potential functions:

- exact potential function: $\Delta u_i = \Delta\Phi$,
- weighted potential function: $\Delta u_i \cdot w_i = \Delta\Phi$, and
- ordinal potential function: $\Delta u_i < 0 \implies \Delta\Phi < 0$.

Theorem 2.3. *If after a finite sequence of improvement steps we reach a NE, we have a potential game.*

Proof. Define Φ to be the length of the longest improvement steps sequence. This is in fact a potential function, as the length of the longest sequence must strictly decrease after having made one improvement step. □

2.1 Local Search Problems

Definition 2.2 (Local search problem Π)**.** *For a local search problem Π, $\mathcal{I}_\Pi$ is the set of instances, and for $I \in \mathcal{I}_\Pi$ we define $\mathcal{F}(I)$ to be the set of feasible solutions. Each solution S has an integer cost assigned, i.e.*

$$\forall S \in \mathcal{F}(I) : \; c(S) \in \mathbb{Z}$$

A paradigm to solve local search problems is to start with any $S \in \mathcal{F}(I)$, and then check whether S is a local optimum. If yes, we are done, and otherwise we go to a better neighbor and continue. The fact that we use integers guarantees termination of this approach.

Definition 2.3 (Polynomial local search, PLS)**.** *PLS is a class of local search problems with a polynomial time algorithm for*

- *find an initial* $S \in \mathcal{F}(I)$*,*
- *evaluate* $c(S)$*, and*
- *check whether* s *is a local optimum, and if not, return a better neighbor.*

For instance, MaxCut and finding a NE in congestion games are in PLS.

Definition 2.4. *A problem* Π *is PLS-complete if and only if every problem* $\Pi' \in$ *PLS can be transformed to* Π *by a PLS-reduction.*

As known from the literature, MaxCut is PLS-complete.

Theorem 2.4. *Finding a NE in a congestion game is PLS-complete.*

Proof. Given an instance of MaxCut, we construct a congestion game as follows. For each edge e with weight w, we have two resources r_e^A and r_e^B, with cost 0 if used by only one player, and cost w if used by more players (though it is clear that at most two player can use it at the same time, as we will see). The player correspond to the nodes, and every player v has two strategies: one strategy contains all r_e^A's for edges e incident to v, and another that contains all r_e^B's for the same edges.

The first strategy corresponds to assigning v to the set A and the latter strategy corresponds to assigning v to B. This one-to-one correspondence between the assignment of the nodes in the MaxCut-instance and the strategies of the players in the congestion game has the property that the local optima of the MaxCut-instance coincide with the Nash equilibria of the congestion game. Hence, our construction is a PLS-reduction from MaxCut to congestion games. □

3 Network Formation Games

3.1 The Local Connection Game

We have n players modeled as nodes in a graph G. Every player can choose any subset of nodes to which he will build edges at a price of α. Given the strategies of all players, the union of all edges will form the resulting (undirected) network. The players want to minimize their cost, which consists of the cost for building the network, and the quality of the network. In particular, players would like to be close (in terms of number of edges) to all other nodes. The cost of player i is defined as follows

$$\text{cost}_v(s) = n_v \cdot \alpha + \sum_{u \neq v} \text{dist}(u, v)$$

where n_v is the number of edges build by player v, and $\text{dist}(u, v)$ denotes the distance (number of edges) from u to v. Note that this term is infinity if u and v are not connected.

The social optimum is a globally good network. That is, the social cost $SC(G)$ is defined as

$$SC(G) = |E| \cdot \alpha + \sum_{i=1}^{n} \sum_{j=1}^{n} \text{dist}_G(i, j) = \sum_{i=1}^{n} \text{cost}_i(s)$$

The last equality holds as in any NE (or optimum), every edge is bought exactly once.

Lemma 3.1. *Any NE is a connected graph, thus has at least $n - 1$ edges.*

Proof. A graph that is not connected has cost infinity, thus regardless of the value of α, this cannot be a stable situation. □

Lemma 3.2. *If $\alpha \geq 2$, then any star is an optimal solution, and if $\alpha \leq 2$, the complete graph is an optimal solution.*

Proof. Consider an optimal solution with m edges. We know that $m \geq n - 1$. All ordered pairs not directly connected by an edge must have a distance of at least 2, and there are $n(n-1) - 2m$ such pairs. Adding the remaining $2m$ pairs with distance of 1 yields

$$\alpha \cdot m + 2n(n-1) - 4m + 2m = (\alpha - 2)m + 2n(n-1)$$

This is a lower bound on the social cost of G. Both a star and the complete graph meet this bound, and the social cost is minimized by making m as small ($\alpha > 2$, a star) or large ($\alpha < 2$, complete graph) as possible. □

Lemma 3.3. *If $\alpha \geq 1$, then any star is a Nash equilibrium, and if $\alpha \leq 1$, the complete graph is a Nash equilibrium.*

Proof. If $\alpha \leq 1$, a player who does not buy all edges, but leaves away k edges saves αk, but increases his total distance by k, i.e. he cannot improve.

Similar for the star. Lets assume player one buys all edges. He has no incentive to stop buying some of the edges, as the distance would then go to infinity. The other players however do not want to buy any edges, as by any edge they buy, they can only decrease the distance by 1, but pay α. Again, no one can improve. □

Theorem 3.1. *If $\alpha \geq 2$ or $\alpha \leq 1$ the price of stability is 1. Otherwise, that is for $1 < \alpha < 2$, the price of stability is at most 4/3.*

3.2 Global Connection Game

In a global connection game, we have k players associated with pairs (s_i, t_i) in a directed graph G. The players want to connect s_i to t_i and the cost for edges are shared by all players using it (Shapely cost sharing game):

$$\text{cost}_i = \sum_{e \in P_i} \frac{c_e}{k_e}$$

where P_i is the path used by player i, c_e are the total edge cost for edge e, and k_e the number of players using edge e.

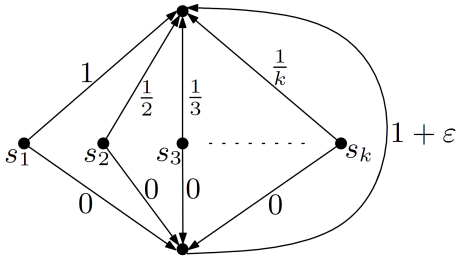

Figure 4: Example from figure 1 revisited.

We can look at a previous example, as shown in figure 4. This came can be modeled as a congestion game, by having the edges as resources, and the costs are just

$$c_e(n_e(s)) = \frac{c_e}{k_e}$$

This means in particular that we can use Rosenthal's potential function:

$$\Phi(s) = \sum_{e \in E} \sum_{i=1}^{k_e} \frac{c_e}{i}$$

Let s^* be a strategic profile that minimizes Φ. We already know that such a s^* exists, and that it is a NE. By comparing $\Phi(s)$ with $\text{cost}(s)$, we can bound the price of stability. First, we can trivially see that $\text{cost}(s) \leq \Phi(s)$ as

$$\sum_{e \text{ used}} c_e = \text{cost}(s) \leq \Phi(s) = \sum_{e \in E} \sum_{i=1}^{k_e} \frac{c_e}{i}$$

Furthermore, we have

$$\Phi(s) = \sum_{e \in E} \sum_{i=1}^{k_e} \frac{c_e}{i} = \sum_{e \in E} c_e \sum_{i=1}^{k_e} \frac{1}{i} = \sum_{e \in E} c_e \cdot H_{k_e} \leq H_k \cdot \sum_{e \in E} c_e = H_k \cdot \text{cost}(s)$$

Therefore, $\text{cost}(s) \leq \Phi(s) \leq H_k \cdot \text{cost}(s)$, and in particular

$$\text{cost}(\text{best NE}) \leq \text{cost}(s^*) \leq \Phi(s^*) \leq \Phi(\text{OPT}) \leq \text{cost}(\text{OPT}) \cdot H_k$$

The price of stability is therefore bounded by H_k, that is $\text{PoS} \leq H_k$.

3.3 Cost-Sharing Games

We consider an undirected graph G with $t_1 = t_2 = \ldots = t_k = t$. The costs for edges do not need to be shared equally, but rather according to a cost-sharing method φ, where $\varphi_i(e, \text{PL}_e)$ tells us the cost of player i of edge e, if the set PL_e use the edge e.

We require two properties for φ:

- *Budged balance.* That is, $c_e = \sum_i \varphi_i(e, \text{PL}_e)$.
- *Fairness.* If i is not using e, he does not have to pay anything.

As there is only one sink t for all s_i, the result is a connected subgraph, or more precisely, a tree. This tree is known as *Steiner tree*.

4 Mechanism Design without Money

4.1 Voting Schemes

We look at voting schemes and try to achieve a social choice.

Definition 4.1 (Voting scheme)**.** *A* voting scheme *consists of candidates, also known as alternatives A, and voters (as players).*

The preferences of the voters can be expressed by a linear order $\succ_i$ over A, where these linear ordering is often given as a list. L is the set of all possible preferences. We would like to have

- a social welfare function $F : L^n \to L$,
$$F(\succ_1, \succ_2, \ldots, \succ_n) = \succ$$
- and a social choice function $f : L^n \to A$
$$f(\succ_1, \succ_2, \ldots, \succ_n) = a \quad a \in A$$

Definition 4.2 (Unanimity)**.** *F satisfies unanimity if every voter prefers a to b, then also $a \succ b$ where $\succ = F(\succ_1, \ldots, \succ_n)$.*

Definition 4.3 (Dictatorship)**.** *F is a dictatorship if there is a voter who is a dictator. A voter i is called a dictator, if*
$$\forall \succ_1, \succ_2, \ldots, \succ_n \colon\ F(\succ_1, \succ_2, \ldots, \succ_n) = \succ_i$$

Definition 4.4 (Consistency (independence of irrelevant alternatives))**.** *Given two sets of preferences for all voters with*
$$\forall i :\ a \succ_i b \iff a \succ'_i b \qquad \textit{(all voters agree on } a \textit{ and } b \textit{ in the two profiles}$$
then
$$a \succ b \iff a \succ' b \qquad \textit{where} \quad \succ = F(\succ_1, \ldots, \succ_n) \textit{ and } \succ' = F(\succ'_1, \ldots, \succ'_n)$$

Theorem 4.1 (Arrow)**.** *Every social welfare function F over more than two candidates that satisfies unanimity and consistency is a dictatorship.*

Lemma 4.1 (Pairwise neutrality)**.** *Let F be as in Arrow's theorem. If*
$$\forall i :\ a \succ_i b \iff c \succ'_i d$$
then
$$a \succ b \iff c \succ' d$$

Definition 4.5 (Strategic manipulation)**.** *A social choice function f can be* strategically manipulated *if*
$$\exists \succ_1, \ldots, \succ_n,\ \succ'_i \in L,\ a, a' \in A : \quad f(\succ_1, \ldots, \succ_i, \ldots, \succ_n) = a \neq a' = f(\succ_1, \ldots, \succ'_i, \ldots, \succ_n) \quad \textit{and} \quad a' \succ_i a$$

Definition 4.6 (Truthfulness)**.** *f is* truthful*, if it cannot be strategically manipulated.*

Note that f in this case is also called *incentive compatible.*

Definition 4.7 (Monotonicity)**.** *f is called* monotone *if*
$$f(\succ_1, \ldots, \succ_i, \ldots, \succ_n) = a \neq a' = f(\succ_1, \ldots, \succ'_i, \ldots, \succ_n)$$
implies
$$a' \succ'_i a \quad \textit{and} \quad a \succ_i a'$$
That is, the change from a to a' can only happen if i changed his mind about a and a'.

Lemma 4.2. *f is truthful if and only if f is monotone.*

Definition 4.8 (Dictatorship)**.** *f is called a dictatorship if there is a dictator. Player i is called a dictator if*
$$(\forall b \neq a :\ a \succ_i b) \implies f(\succ_1, \ldots, \succ_n) = a$$

Definition 4.9 (Onto)**.** *f is a social choice function* onto *A, if every candidate has a chance to win.*

Theorem 4.2 (Gibbard-Satterthwaite)**.** *Let f be a social choice function onto A, $|A| \geq 3$, f is truthful. Then, f is a dictatorship.*

4.2 House Allocation

In this game we have n agents (players), each owning a house. Every agent has a strict order $\succ_i$ of the n houses, and we would like to reallocate the houses such that everyone is "happy". We would like a mechanism that satisfies:

- Voluntary participation. No agent gets a house that is worse than his own house.
- No subset of agents can improve (compared to the mechanism) when they trade among themselves. We will call this situation absence of blocking coalitions.

The outcome of the mechanism is a reallocation. Players only care about the house they get assigned. That is, two outcomes where player i gets the same house are equally good for player i. On the other hand, player i prefers outcome x to y if the house he gets in x is strictly better than the house in outcome y (according to $\succ_i$).

Since we don't have a strict ordering over the outcomes, we cannot apply Arrow's theorem.

We let A be the set of all outcomes, where we understand $a = (a_1, \dots, a_n) \in A$ as the outcome where player i gets house a_i. The initial setting is therefore $a_0 = (1, \dots, n)$.

Definition 4.10. *Let $S \subseteq \{1, \dots, n\}$. We denote by $A(S)$ all allocations where the players from S only trade among themselves:*

$$A(S) = \{a \mid \forall i \in S : \ a_i \in S\}$$

Definition 4.11. *Given $a \in A$, $S \subseteq \{1, \dots, n\}$, we call S a* blocking coalition *of allocation a if there exists $\overline{a} \in A(S)$ s.t.*

- $\forall j \in S : \ (\overline{a_j} \succ_j a_j \vee \overline{a_j} = a_j)$ *(no player is worse off), and*
- $\exists j \in S : \ \overline{a_j} \succ_j a_j$ *(one player is better off).*

Definition 4.12 (Core)**.** *The* core *is the set of allocations with no blocking coalition.*

We can represent this game as a labeled graph $G = (V, E)$, where V is the set of houses, and a vertex from v_i to u with label c indicates that u is the cth most preferred house of player i.

If we only consider the edges with label 1, we also get a graph. Note that this graph always has a cycle (possibly a loop), as every vertex has exactly one outgoing edge, and we can start at an arbitrary vertex and just follow the edges. At some point we will see a vertex twice, as there are only finitely many.

4.2.1 The Trading Cycle Algorithm (TTCA)

Algorithm 1 The Trading Cycle Algorithm (TTCA)

```
start with V = {1, ..., n}
loop
  for all vertices v_i do
    add edge to most preferred house, that is still available
  end for
  identify all cycles, and trade accordingly
  remove all vertices of such cycles
  remove all edges
end loop
```

Theorem 4.3. *TTCA delivers an allocation with no blocking coalition. Moreover, this is the only such allocation.*

Theorem 4.4. *The mechanism using TTCA is truthful.*

4.3 Stable Matchings

We consider a marriage game with a set M of men, and women W, where $|M| = |W|$. Every man $m \in M$ has a strict preference $\succ_m$ over all women, and vice versa.

Definition 4.13 (Blocking pair)**.** *The set $\{m, w\}$ is called a blocking pair if m is matched with w' and w is matched with m', but they would prefer each other, that is*

$$w \succ_m w' \qquad \text{and} \qquad m \succ_w m'$$

We would like to avoid such blocking pairs.

Definition 4.14 (Deferred acceptance, male version)**.** *First, each man proposes to his top-ranked choice. Next, each woman who has received at least two proposals keeps (tentatively) her top-ranked proposal and rejects the rest. Then, each men who has been rejected proposes to his top-ranked choice among the women who have not yet rejected him. Again, each woman who has at least two proposals (including ones from the previous round) keeps her top-ranked proposal. The process repeats until no man has a woman to propose to or each woman has at most one proposal. This results in a matching.*

Definition 4.15 (Stable matching)**.** *A matching is called* stable *if it has no blocking coalition.*

Theorem 4.5. *The male version of the algorithm deferred acceptance returns a stable matching.*

Proof. Consider a man m with preferences $w_1 \succ_m w_2 \succ_m \cdots \succ_m w_i \succ_m \cdots \succ_m w_n$, and w.l.o.g. let us assume he gets matched to w_i. A blocking pair with m would involve a woman w_j with $j < i$ that likes m better than what she got. However, all woman w_j with $j < i$ rejected m, and this only happens if w_j had a better proposal at that time. Since the men for woman only improve over time (either they keep their proposal, or they get a better one), there can't be a blocking pair. □

Theorem 4.6. *The mechanism using the male version of the algorithm of deferred acceptance is truthful for men.*

5 Mechanism Design with Money

Recall that we defined the utility u_i of player i in terms of the valuation v_i and the payment p_i. For *voluntary participation* (also called *individual rationality*), we require

$$u_i = v_i - p_i \geq 0$$

A mechanism then is social choice function $f : V_1 \times \cdots \times V_n \to A$, and a vector of payment functions $p_1, \ldots, p_n$, where $p_i : V_1 \times \cdots \times V_n \to \mathbb{R}$ determines how much player i pays. V_i is the set of possible valuations, and A the set of outcomes.

For a mechanism to be truthful, we require for true valuations $v_1, \ldots, v_n$ and bids $b_1, \ldots, b_n$ that

$$\forall i, v_i' \in V_i : \; v_i(a) - p_i(a) \geq v_i(a') - p_i(a') \qquad \text{where } a = f(v_i, v_{-i}) \text{ and } a' = f(v_i', v_{-i})$$

5.1 Vickrey-Clarke-Groves Mechanism

Definition 5.1 (VCG mechanism)**.** *A mechanism $(f, p_1, \ldots, p_n)$ called VCG mechanism, if two conditions are satisfied:*

1. *The objective is utilitarian: $f(v_1, \ldots, v_n) \in \arg\max_a \sum_i v_i(a)$*
2. *Payments are of the following form:*

$$p_i(a) = h_i(b_{-i}) - \sum_{j \neq i} v_j(a) \qquad \textit{where } a = f(v_1, \ldots, v_n)$$

Note that the payment of i is not influenced by what i says, only the outcome is.

The main idea lies in the term $-\sum_{j \neq i} v_j(a)$. When this term is added to his own value $v_i(f(v_1, \ldots, v_n))$, the sum becomes exactly the total social welfare of $f(v_1, \ldots, v_n)$. Thus the mechanism aligns all player's incentives with the social goal of maximizing social welfare, which is exactly achieved by telling the truth. The other term is (from i's point of view) only a constant and can be used to control the amount that is paid.

Theorem 5.1. *Every VCG mechanism is truthful.*

Proof. Fix i, v_{-i}, v_i and v_i'. We need to show that for every player i with valuation v_i, the utility when declaring v_i is not less than the utility when declaring v_i'. Denote $a = f(v_i, v_{-i})$ and $a' = f(v_i', v_{-i})$. The utility of i when declaring v_i is

$$v_i(a) + \sum_{j \neq i} v_j(a) - h_i(v_{-i}) = \sum_j v_j(a) - h_i(v_{-i})$$

but when declaring v_i' it is

$$v_i(a') + \sum_{j \neq i} v_j(a') - h_i(v_{-i})$$

Since $a = f(v_i, v_{-i})$ maximizes social welfare over all alternatives, we have

$$\sum_j v_j(a) \;=\; v_i(a) + \sum_{j \neq i} v_j(a) \;\geq\; v_i(a') + \sum_{j \neq i} v_j(a')$$

□

5.1.1 Clarke Pivot Rule

The question of choosing the "right" h_i's is solved by the Clarke pivot rule. We would like the following properties:

- A mechanism is *individually rational* if players always get a non-negative utility. We say the mechanism fulfills *voluntary participation*
- A mechanism has no positive transfers if no player is every paid money. Formally, if for every $v_1, \ldots, v_n$ and every i, $p_i(v_1, \ldots, v_n) \geq 0$.

Definition 5.2 (Clarke pivot rule)**.** *The choice*

$$h_i(v_{-i}) = \max_{b \in A} \sum_{j \neq i} v_i(b)$$

is called the Clarke pivot rule.

Theorem 5.2. *A VCG mechanism with Clarke pivot payments makes no positive transfers. If $v_i(a) \geq 0$ for every $v_i \in V_i$ and $a \in A$ then voluntary participation does also hold.*

5.1.2 Choosing Payment Schemes

A VCG-mechanism with the Clarke pivot rule intuitively chooses payments as follows:

$$p_i = \sum_{j \neq i} v_j \,(\text{opt. allocation without } i \text{ present}) - \sum_{j \neq i} v_j \,(\text{current situation with } i \text{ present})$$

5.2 Non-Utilitarian Problems

5.2.1 Related Machines

As in an earlier example we consider m machines and n jobs, but now the machines rather than the jobs are the players. Every machine i has a speed s_i and every job j has a processing requirement p_j. The time required by machine i for job j is then p_j/s_i.

Clearly, the machines could lie about their speed in order to get less work. We want to minimize the make-span, that is our goal can be written as:

$$\max_i \frac{1}{s_i} \sum_{j \in X_i} p_j \;\to\; \min$$

where X_i is the set of jobs assigned to machine i. We define the bids to have the form $b_i = 1/s_i$ rather than directly announce the speed.

We can define the work curve as the curve given by the function that determines the amount of work w for bid b_i. Note that every machine has a work curve.

Theorem 5.3 (Aascher/Tardos)**.** *The following two statements hold.*

1. *A truthful mechanism for the above problem exists if and only if each work curve is decreasing.*
2. *In this case, the payments are of the form*

$$p_i(b) = h_i(b_{-i}) + b_i \cdot w_i(b) - \int_0^{b_i} w_i(b_{-i}, u)\, \mathrm{d}u$$

Theorem 5.4. *There is a mechanism with voluntary participation if*

$$\int_0^{\infty} w_i(b_{-i}, u)\, \mathrm{d}u$$

is finite. Furthermore, the payment scheme is

$$-p_i(b) = b_i \cdot w_i(b) + \int_0^{b_i} w_i(b_{-i}, u)\, \mathrm{d}u$$

6 Best-Response Mechanisms

6.1 Introduction

Definition 6.1 (Synchronous best-response dynamics)**.** *Players play their best response infinitely many times, one by one in a fixed order (round robin).*

Definition 6.2 (Asynchronous best-response dynamics)**.** *At each step an adversary activates an arbitrary subset of players who best respond to the current profile (the adversary also chooses a starting profile). The adversary must activate each player an infinite number of times.*

The choice of the adversary and the "response strategies" of each player determine an infinite sequence

$$s^0 \implies s^1 \implies \cdots \implies s^t \implies \cdots$$

We are interested what games converge. We can consider the base game G with strategies $s_i \in S_i$ and utilities $u_i(s)$. But there is also a repeated game G^* with response strategies $R_i() \in S_i$ and a total utility

$$\Gamma_i := \limsup_{t\to\infty} u_i(s^t)$$

Definition 6.3. *Best-response are* incentive compatible *for G if repeated best-responding is a Nash equilibrium for the repeated game G^*, that is for every i*

$$\Gamma_i \geq \Gamma_i'$$

where Γ_i is the total utility when all players best respond and Γ_i' is the total utility when all but i best respond (starting from the same initial profile s^0 and applying the same activation sequence).

We consider two intuitive ideas:

1. Introduce tie breaking rule.
2. Eliminate "useless" strategies.

Definition 6.4 (Never best response (NBR))**.** *A strategy $s_i \in S_i$ is a* never best response *(for tie breaking rule $\prec$) if there is always another strategy that gives a better payoff or that gives the same payoff but is better with regard to the tie breaking rule.*
More formally, for all s_{-i} there exists $s_i' \in S_i$ such that one of these holds:

1. $u_i(s_i, s_{-i}) < u_i(s_i', s_{-i})$, *or*
2. $u_i(s_i, s_{-i}) = u_i(s_i', s_{-i})$ *and* $s_i \prec_i s_i'$.

This condition is enough to guarantee convergence (of best response):

Definition 6.5 (NBR-solvable)**.** *A game G is* NBR-solvable *if iteratively eliminating NBR strategies results in a game with one strategy per player. That is, there exists a tie breaking rule $\prec$, sequence $p_1, \ldots, p_l$ of players, and a corresponding sequence of subsets of strategies $E_1, \ldots, E_l$ such that:*

1. *Initially $G_0 = G$, and G_{i+1} is the game obtained from G_i by removing the strategies E_i of player p_i. That is, $G_{i+1} = G_i \setminus E_i$.*
2. *Strategies E_i are NBR for $\prec$ in the game G_i.*
3. *The final game G_{l+1} has one strategy for each player (this unique profile is thus a pure NE for G).*

A sequence of players and of strategies as above is called an elimination sequence *for the game G.*

Definition 6.6 (Round)**.** *A* round *is the first elements of the sequence such that all players have been activated at least once.*

Lemma 6.1 (Rounds vs subgames)**.** *Let $p_1, \ldots, p_l$ be the players of any elimination sequence for the game under consideration. Suppose that players $p_1, \ldots, p_k$ always best respond (according to the prescribed tie breaking rule $\prec$). Then, for any initial profile and for any activation sequence, every profile after the k-th round is a profile in the subgame G_{k+1}.*

Theorem 6.1. *For NBR-solvable games best response (according to the prescribed tie breaking rule $\prec$) converge even in the asynchronous case.*

Proof. Take $k = l$ and observe that G_{l+1} only contains one profile. □

Definition 6.7 (NBR-solvable with clear outcome)**.** *A NBR-solvable game G has a clear outcome if the following holds. Let s^* be the unique profile of the game G_{l+1} (a pure NE of G). For each player i there is a (player specific) elimination sequence consisting of players $p_1, \ldots, p_a, \ldots, p_l$ and strategies $E_1, \ldots, E_a, \ldots, E_l$ (according to definition 6.5) such that the following holds. If a is the first occurrence of i in the sequence (i.e. $p_a = i$ and $p_1 \neq i, \ldots, p_{a-1} \neq i$) then, in the corresponding game G_{a_i-1} the pure NE s^* is globally optimal for i, that is, for every other profile s' in the game G_{a_i-1} it holds that $u_i(s') \leq u_i(s^*)$.*

Theorem 6.2 (Incentive compatibility)**.** *For games that are NBR-solvable with clear outcome best response (according to the prescribed tie breaking rule $\prec$) is also incentive compatible.*

6.2 Border Gateway Protocol as an Example

The border gateway protocol (BGP) can be modeled as game, where the nodes are players, the strategies are the neighbors ("where do I send my traffic?“) and the strategy profiles are a set of paths (or loops). The utility is defined by a order over the paths connecting player i to destination d:

$$P_1 \prec_i P_2 \prec_i \cdots \prec_i P_k$$

and any path $\emptyset$ which does not connect i to d is strictly worse, i.e. $\emptyset \prec_i P_1$.

Definition 6.8 (Dispute wheel)**.** *A* dispute wheel *is a situation where every node prefers routing its traffic over the next node in the "wheel“.*

This model of the BGP game does in general not converge, nor is it incentive compatible. However, in practice such non-converging examples do not occur. Thus, we refine our model.
There are two types of commercial relationship between autonomous systems (AS), as shown in figure 5.

Figure 5: Commercial relationships between ASs.

Each node i classifies paths according to its commercial relationship with neighbor in the path (first hop):

1. customer paths (neighbor is a customer),
2. peer paths (neighbor is a peer), or
3. provider paths (neighbor is a provider).

The preferences of all nodes respect the Gao-Rexford model:

Definition 6.9 (Gao-Rexford model)**.** *There are three conditions that are satisfied by all nodes:*

1. *$\emptyset \prec$ provider paths $\prec$ peer paths $\prec$ customer paths, and*
2. *transit traffic to/from my customers only, and*
3. *no customer-provider cycles.*

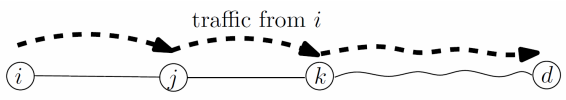

Figure 6: Commercial relationships between ASs.

Note that the situation shown in figure 6 can only occur in one of these two situations (or both):

- i is a customer of j, or
- k is a customer of j.

Lemma 6.2. *The Gao-Rexford model does not allow dispute wheels.*

Lemma 6.3. *If there is no dispute wheel, the BGP game is NBR-solvable with clear outcome.*

Definition 6.10 (Happy path)**.** *A path $h_1 \to \cdots \to h_l \to d$ is called a* happy path *if this path gives the highest possible payoff to all of the nodes. That is, h_a's top ranked path is*

$$h_a \to h_{a+1} \to \dots h_l \to d$$

Lemma 6.4. *If there is no dispute wheel, a happy path exists.*

Proof. Assume there is no happy path and show that there is no dispute wheel. □

Proof of lemma 6.3. We prove lemma 6.3 by explicitly stating the elimination sequence. Consider a happy path $h_1 \to \cdots \to h_l \to d$.

1. h_l eliminates all strategies other than $h_l \to d$ from the current subgame G_i, and this gives us game G_{i+1}. In G_{i+1} the path is still happy, and thus we eliminate all strategies from h_{l-1} other than $h_{l-1} \to h_l$. We continue, until h_1.
2. In the resulting subgame we find another happy path and repeat the previous step until there are no happy paths that start with a node with at least two strategies.

The final pure NE is the one that give the nodes the highest payoff and thus the game is NBR-solvable with clear outcome. □

7 Combinatorial Auctions

We consider an auction of many goods, where the bidders have valuations for subsets of items. All bids are collected by the auctioneer, and he then allocates the items.

If the auctioneer can accept multiple bids per player, we call this setting an OR-bid. For instance, if player 1 bids 3 for goods $\{a, b\}$ and 2 for $\{c\}$, the auctioneer is allowed to charge $5 = 3 + 2$ for items $\{a, b, c\}$. If we do not want to allow such behavior, but rather have the auctioneer to choose at most one bid per player, we call it an XOR-bid.

Lemma 7.1. *Finding an optimal allocation in a combinatorial auction is NP-hard.*

Proof. Reduction to the maximum weight independent set problem. □

If an exact solution cannot be found fast, maybe we can approximate the problem. We have a bound on the ratio of $\sqrt{m}$ for m goods.

A solution by Lehman, O'Calhagan and Shaham (LOS) from 2001 achieves this bound. We sort the bids b_i for items S_i according to $k_i = b_i/\sqrt{|S_i|}$, and grant the first bid (i.e. with highest k_i) first. The question is now how we chose the payment scheme. We use a similar idea as with the second-price auction by Vickery, as we shall see.

In LOS we sort the bids according to k_i, and grant the first bid. Then, all bids below with conflicting item sets drop out, and we repeat the procedure. Now, the payment is determined by the first bid that we kick out. However, since a bid can be blocked by multiple other bids, we choose the highest uniquely blocked bid (HUBB), that is the highest bid that drops out if I am there, but does not drop out without me. Then, we pay the per-item price of the highest uniquely blocked bid (or zero, if there is no HUBB).

Lemma 7.2. *The mechanism according to LOS satisfies the following properties:*

1. *Exactness: if we bid (S, b), we either get all items from S, or none at all.*
2. *Monotonicity: if bid (S, b) is grated, then (S', b') with $S' \subseteq S$ and $b' \geq b$ will be grated as well.*
3. *If we don't get anything, we don't have to pay anything.*
4. *There is a critical value c for bid $(S, \cdot)$:*
   - *Bid (S, b) with $c < b$ will be grated.*
   - *Bid (S, b) with $c > b$ will not be grated.*
   - *Bid (S, c) is either granted or not granted.*
   - *If the bid is grated, the payment is c.*

   In particular, the critical value is for bid $(S_i, \cdot)$ is

$$c = \frac{b_j}{\sqrt{S_j}}\sqrt{S_i} \qquad \text{where } j = \mathrm{HUBB}_i$$

Theorem 7.1. *LOS is truthful.*

8 Matching Markets

8.1 Bipartite Matching

Let us first consider the simple case of bipartite matching. We are given a bipartite graph with sets N and M of nodes, and a set E of edges. We would like to find a perfect matching in this graph. This example can be motivated by the following scenario: Suppose at a collage students can express what rooms in the dorm would be acceptable (0/1-preference), and then we somehow would like to match rooms with students. Note that in this case N is the set of students, and M stands for the rooms.

If a bipartite graph has a perfect matching, it easy to prove this: one can just show the matching. What about the inverse case, i.e. proving that there is no perfect matching?

Definition 8.1 (Constricted set)**.** *A set $S \subset V$ in a bipartite graph is called constrictive if*

$$|N(S)| < |S|$$

where $N(\cdot)$ stands for the neighborhood.

Theorem 8.1 (König '31, Hall '35)**.** *If a bipartite graph doesn't have a perfect matching, then there is a constricted set.*

8.2 Prices and the Market-Clearing Property

Slightly more complicated is the setting with general valuations: A set of sellers have a house each, and the buyers would like to buy a house. They have a valuation for every house, i.e. for buyer i we have a valuation vector $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$. Every house has a price, and the utility of player i if he is assigned house j is

$$u_i = v_{ij} - p_j$$

This utilities give rise to new preferences over the houses.

Definition 8.2 (Marked Clearing Prices)**.** *A set of prices in the above setting are called* market-clearing prices *(MCP) if the most preferred graph (only edges to the most preferred seller) contains a perfect matching.*

Theorem 8.2. *Market-clearing prices do always exist.*

Theorem 8.3 (Optimality of market-clearing prices)**.** *For any set of market-clearing prices, a perfect matching in the resulting preferred-seller graph has the maximum total valuation of any assignment of sellers to buyers.*

Proof. Consider a set of market-clearing prices, and let M be a perfect matching in the preferred-seller graph. Now, consider the total payoff of this matching, defined simply as the sum of each buyer's payoff for what she gets. Since each buyer is grabbing a house that maximizes her payoff individually, M has the maximum total payoff of any assignment of houses to buyers. Now how does total payoff relate to total valuation, which is what we're hoping that M maximizes? If buyer j chooses house i, then her valuation is v_{ij} and her payoff $v_{ij} - p_i$. Thus, the total payoff to all buyers is simple the total valuation, minus the sum of all prices:

$$\text{total payoff of } M = \text{total valuation of } M - \text{sum of all prices}$$

But the sum of all prices is something that does not depend on which matching we choose. □

But how can we construct market-clearing prices? The idea is to increase the price for houses with high demand.

8.3 Sponsored Search

We consider advertising in Internet searches, where advertisers (buyers) are interested in slots on a web page (sellers). The available slots are numbered starting with 1, and users are more likely to click on higher slots. We assume that every slot has a specific *click-through rate* r_i associated with it. This is the number of clicks per hour that an ad placed in that slot will receive.

Algorithm 2 Finding market-clearing prices

start with $p_1 = \cdots = p_m = 0$
loop
  construct most-preferred seller graph
  if there is a perfect matching **then**
    stop
  end if
  find constricted set $S \subseteq N$
  increase prices in $N(S)$ by 1
  if all prices are non-zero **then**
    decrease all prices by smallest price
  end if
end loop

8.3.1 VCG-Auction

The prices for ad slots are determined by a VCG-auction. The buyers place their bids as their valuation per click, and we assign the slots to advertiser in a way that maximizes the total valuations. That is, we simple sort by the bids b_i. The price that player i is charged is then the harm he causes to all others by his presence. More formally, let us define

$$V_N^M = \text{maximum value of matching of slots } M \text{ to advertisers } N$$

Now we can express the prices as

$$p_i = V_{N-i}^M - V_{N-i}^{M-j}$$

However, this maximizes the advertisers happiness, and thus is not used in practice.

8.3.2 Generalized Second-Prize (GSP) Auction

The assignment is the same as with the VCG-auction, but the pricing scheme differs. We charge buyer j the price of the price-per-click of the $(j+1)$-th bid. This is not truthful in general, but the solution is always a NE.