Algorithmic Game Theory October 2, 2016, Week 2

Price of Anarchy and Hardness of Computing Equilibria

ETH Zürich Paolo Penna

We are going to talk about two things that *contradict with each other*:

1. We assume that selfish players will converge to (some) equilibrium, and therefore we study the quality of (pure Nash) equilibria and show that they are not too far from the optimum.

2. We then show that computing (pure Nash) equilibrium is computationally *difficult*, not just for the players, but for a computer too.

We shall resolve this 'contradiction' in the next lecture, essentially by looking at a more general (relaxed) equilibrium concept.

1 Price of Anarchy

Consider **cost-minimization games**, that is, the case in which each player i has a cost $c_i(s)$. Here it is natural to consider the **social cost** of a state s as the sum of all players' costs,[1]

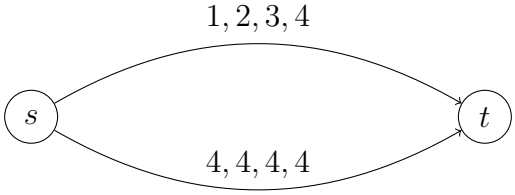
$$cost(s) = \sum_i c_i(s).$$

We want to study this question:

How bad is selfish behavior?

Selfish behavior results in some equilibrium. The next example shows that this may not be always the best in terms of social cost.

Example 1 (Pigou's Example – Discrete Version). *Consider the following symmetric network congestion game with four players.*



There are five kinds of states:

[1] Other definitions are also meaningful, for example, the *maximum* cost incurred by any player.

(4) all four players use the top edge, social cost: 16

(3) three players use the top edge, one player uses the bottom edge, social cost: 13

(2) two players use the top edge, two players use the bottom edge, social cost: 12

(1) one player uses the top edge, three players use the bottom edge, social cost: 13

(0) all players use the bottom edge, social cost: 16

Observe that only states of kind (4) and (3) can be pure Nash equilibria. The social cost, however, is minimized by states of kind (2). Therefore, when considering pure Nash equilibria, due to selfish behavior, we lose up to a factor of $\frac{16}{12}$ and at least a factor of $\frac{13}{12}$.

The Price of Anarchy compares the **worst equilibrium** with the **optimum**.

Definition 2 (Price of Anarchy). *For a (cost-minimization) game which admits pure Nash equilibria, the* price of anarchy for pure Nash equilibria *is defined as*

$$PoA = \frac{\max_{s \in \mathsf{PNE}} cost(s)}{\min_{s \in S} cost(s)} \ ,$$

where PNE *is the set of pure Nash equilibria of this game.*

1.1 Price of Anarchy in Smooth Games

A very helpful technique to derive upper bounds on the price of anarchy is *smoothness*.

Definition 3. *A game is called* (λ, μ)-smooth *for $\lambda > 0$ and $\mu < 1$ if, for every pair of states $s, s^* \in S$, we have*

$$\sum_i c_i(s_i^*, s_{-i}) \leq \lambda \cdot cost(s^*) + \mu \cdot cost(s) \ .$$

Observe that this condition needs to hold for *all* states $s, s^* \in S$, as opposed to only pure Nash equilibria or only social optima. We consider the cost that each player incurs when unilaterally deviating from s to his strategy in s^*. If the game is smooth, then we can upper-bound the sum of these costs in terms of the social cost of s and s^*.

Theorem 4. *In a (λ, μ)-smooth game, the PoA is at most*

$$\frac{\lambda}{1 - \mu} \ .$$

Proof. Let s be pure Nash equilibrium and s^* be an optimum solution, which minimizes social cost. Then:

$$\begin{aligned} cost(s) &= \sum_i c_i(s) && \text{(definition of social cost)} \\ &\leq \sum_i c_i(s_i^*, s_{-i}) && \text{(as } s \text{ is a pure Nash equilibrium)} \\ &\leq \lambda \cdot cost(s^*) + \mu \cdot cost(s) && \text{(by smoothness)} \end{aligned}$$

and by rearranging the terms we get

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

for any pure Nash equilibrium s and any social optimum s^*. That is, $PoA \leq \frac{\lambda}{1-\mu}$. □

1.2 Tight Bound for Affine Delay Functions

We next provide a tight bound on the price of anarchy for *affine delay functions*, that is, when all delay functions are (non-decreasing) of the form

$$d_r(x) = a_r \cdot x + b_r,$$

with $a_r, b_r \geq 0$.

Theorem 5. *Every congestion game with affine delay functions is* $\left(\frac{5}{3}, \frac{1}{3}\right)$*-smooth. Thus, the PoA is upper bounded by* $\frac{5}{2} = 2.5$.

We use the following lemma:

Lemma 6 (Christodoulou, Koutsoupias, 2005)**.** *For all nonnegative integers* $y, z \in \mathbb{Z}$ *we have*[2]

$$y(z+1) \leq \frac{5}{3} \cdot y^2 + \frac{1}{3} \cdot z^2 \ . \tag{1}$$

Proof. Consider the following subcases (assume now y and z nonnegative):

- ($y = 1$) In this case (1) can be rewritten as

$$z \leq \frac{2}{3} + \frac{1}{3}z^2 \tag{2}$$

  which follows from $(z-1)(z-2) \geq 0$ and from the hypothesis that $z \geq 0$ is integer.

- ($y \geq 2$) In this case we use $y \leq y^2/2$ and the following inequality:

$$yz \leq \frac{3}{4}y^2 + \frac{1}{3}z^2 \tag{3}$$

  which follows from $0 \leq \left(\sqrt{\frac{3}{4}}y - \sqrt{\frac{1}{3}}z\right)^2 = \frac{3}{4}y^2 + \frac{1}{3}z^2 - yz$. Putting these two inequalities together, we get

$$y(z+1) = yz + y \leq \frac{3}{4}y^2 + \frac{1}{3}z^2 + y^2/2 = \frac{5}{4} \cdot y^2 + \frac{1}{3} \cdot z^2$$

  which is even more than what we need to prove (1).

This completes the proof. □

[2]The lemma holds also for negative integers.

Remark 1. *Though we do not need it, Lemma 6 holds also for negative integers. Namely, for the case $y \leq 0$, observe that the claim is trivial for $y = 0$ or $z \geq -1$, because $y(z+1) \leq 0 \leq \frac{5}{3} \cdot y^2 + \frac{1}{3} \cdot z^2$. For the case $y < 0$ and $z < -1$, we use that $y(z+1) \leq |y|(|z|+1)$ and apply the bound for positive y and z shown above.*

Main Steps of Proof:

$$cost(s) = \sum_i c_i(s) = \sum_r n_r(s) d_r(s) = \sum_r a_r n_r(s)^2 + b_r n_r(s) \tag{4}$$

$$\sum_i c_i(s_i^*, s_{-i}) \leq \sum_r (a_r(n_r(s)+1) + b_r) n_r(s^*) \tag{5}$$

$$a_r(n_r(s)+1)n_r(s^*) \leq \frac{5}{3} a_r n_r(s^*)^2 + \frac{1}{3} a_r n_r(s)^2 \tag{6}$$

Proof of Theorem 5. Given two states s and s^*, we have to bound

$$\sum_i c_i(s_i^*, s_{-i}) \ .$$

We have

$$c_i(s_i^*, s_{-i}) = \sum_{r \in s_i^*} d_r(n_r(s_i^*, s_{-i})) \ .$$

Furthermore, as all d_r are non-decreasing, we have $d_r(n_r(s_i^*, s_{-i})) \leq d_r(n_r(s)+1)$. This way, we get

$$\sum_i c_i(s_i^*, s_{-i}) \leq \sum_i \sum_{r \in s_i^*} d_r(n_r(s)+1) \ .$$

By exchanging the sums, we have

$$\sum_i \sum_{r \in s_i^*} d_r(n_r(s)+1) = \sum_r \sum_{i : r \in s_i^*} d_r(n_r(s)+1) = \sum_r n_r(s^*) d_r(n_r(s)+1) \ .$$

To simplify notation, we write n_r for $n_r(s)$ and n_r^* for $n_r(s^*)$. Recall that delays are $d_r(n_r) = a_r n_r + b_r$. In combination, we get

$$\sum_i c_i(s_i^*, s_{-i}) \leq \sum_r (a_r(n_r+1) + b_r) n_r^* \ .$$

Let us consider the term for a fixed r. We have

$$(a_r(n_r+1) + b_r) n_r^* = a_r(n_r+1) n_r^* + b_r n_r^* \ .$$

Lemma 6 implies that

$$(n_r+1) n_r^* \leq \frac{1}{3} n_r^2 + \frac{5}{3} (n_r^*)^2 \ .$$

Thus, we get

$$\begin{aligned}
(a_r(n_r+1) + b_r) n_r^* &\leq \frac{1}{3} a_r n_r^2 + \frac{5}{3} a_r (n_r^*)^2 + b_r n_r^* \\
&\leq \frac{1}{3} a_r n_r^2 + \frac{1}{3} b_r n_r + \frac{5}{3} a_r (n_r^*)^2 + \frac{5}{3} b_r n_r^* \\
&= \frac{1}{3} \left(a_r n_r + b_r\right) n_r + \frac{5}{3} \left(a_r n_r^* + b_r\right) n_r^* \ ,
\end{aligned}$$

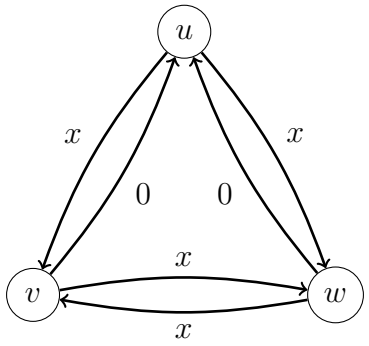
where in the second step we used that $b_r \geq 0$. Summing up these inequalities for all resources r, we get

$$\begin{aligned}\sum_r (a_r(n_r+1)+b_r)n_r^* &\leq \frac{5}{3}\sum_r (a_r n_r^* + b_r)n_r^* + \frac{1}{3}\sum_r (a_r n_r + b_r)n_r \\ &= \frac{5}{3}\cdot cost(s^*) + \frac{1}{3}\cdot cost(s) \ ,\end{aligned}$$

which shows $\left(\frac{5}{3}, \frac{1}{3}\right)$-smoothness. □

Theorem 7. *There are congestion games with affine delay functions whose price of anarchy for pure Nash equilibria is $\frac{5}{2}$.*

Proof sketch. We consider the following (asymmetric) network congestion game. Notation 0 or x on an edge means that $d_r(x) = 0$ or $d_r(x) = x$ for this edge.



There are four players with different source sink pairs. Refer to this table for a socially optimal state of social cost 4 and a pure Nash equilibrium of social cost 10.

player	source	sink	strategy in OPT	cost in OPT	strategy in PNE	cost in PNE
1	u	v	$u \to v$	1	$u \to w \to v$	3
2	u	w	$u \to w$	1	$u \to v \to w$	3
3	v	w	$v \to w$	1	$v \to u \to w$	2
4	w	v	$w \to v$	1	$w \to u \to v$	2

□

2 Hardness of Computing Equilibria

We have seen that congestion games always have a pure Nash equilibrium, and that in singleton congestion games pure Nash equilibria can be found in polynomially many steps using (best response) improvement steps. What about more general congestion games? Do best response sequences always converge in polynomial time? Or can we at least compute a pure Nash equilibrium using a different algorithm in polynomial time? If we cannot, how would we give evidence of computational intractability?

Why do we care? Positive results, such as quick convergence of best response dynamics, clearly speak in favor of an equilbrium concept. Negative results, in turn, cast a shadow on the predictive power of an equilibrium concept. A famous quote in this context is Kamal Jain's "If your laptop cannot find it, then neither can the market."

2.1 Polynomial-time Algorithm for Symmetric Network Games

An interesting mid-ground are symmetric network congestion games.

Definition 8. *In a* network congestion game*, the set of resources is the set of edges E of a directed graph $G = (V, E)$. Player i's strategies correspond to the set of paths between a fixed $s_i \in V$ and $t_i \in V$.*

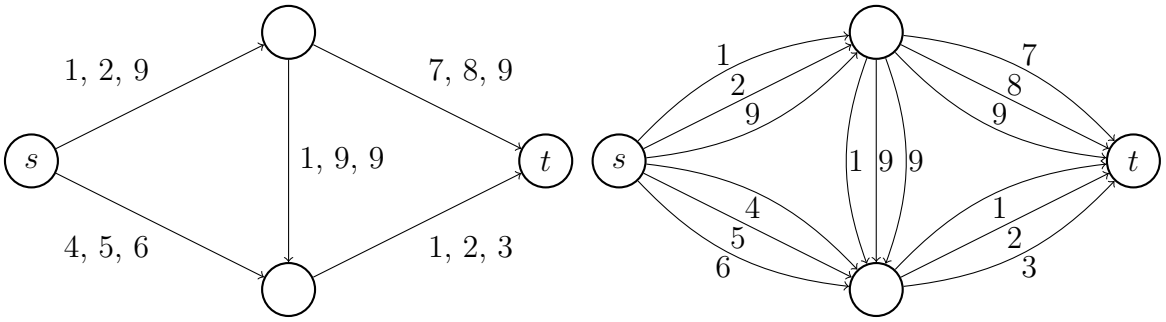
Definition 9. *A network congestion game is* symmetric *if all players share the same source $s \in V$ and have a common target $t \in V$.*

It is known that there are instances of symmetric congestion games in which there are states such that every improvement sequence from this state to a pure Nash equilibrium has exponential length. Hence, applying improvement steps is not an *efficient* (i.e., polynomial time) algorithm for computing Nash equilibria in these games. However, there is another algorithm which finds pure Nash equilibria in polynomial time.

Theorem 10 (Fabrikant, Papadimitriou, Talwar 2004)**.** *In symmetric network congestion games with non-decreasing delays on each edge, a pure Nash equilibrium can be computed in polynomial time.*

Proof. Find a min-cost flow in the following graph (and the flow to be sent equals to n):

- Each edge is replaced by n parallel edges of capacity 1 each.
- The i-th copy of edge e has cost $d_e(i)$, $1 \leq i \leq n$.



In particular, we want to send n units of flow from s to t, where n is the number of players. The optimal solution can be computed in polynomial time and, because of integer capacities, flow, and costs, it is guaranteed to be take integer values (check literature). Therefore, the optimal solution minimizes Rosenthal's potential function and, hence, is a pure Nash equilibrium (**Exercise 1**). □

Exercise 1. *Prove the last claim in the proof above, namely, that the optimum for the min-cost flow indeed minimizes the potential. Discuss why we need the assumption that delays are* non-decreasing*.*

2.2 The Complexity Class PLS

It turns out that for more general congestion games, computing a pure Nash equilibrium is a computationally hard problem. For this we will interpret the problem of finding a pure Nash equilibrium as the problem of finding a local optimum, and we will show that it is as hard as any other local search problem.

Definition 11. *A* local search problem Π *is given by its set of instances* $\mathcal{I}_\Pi$*. For every instance* $I \in \mathcal{I}_\Pi$*, we are given a finite set of* feasible solutions $\mathcal{F}(I)$*, an* objective function $c : \mathcal{F}(I) \to \mathbb{Z}$*, and for every feasible solution* $s \in \mathcal{F}(I)$*, a* neighborhood $N(s, I) \subseteq \mathcal{F}(I)$*. A feasible solution* $s \in \mathcal{F}(I)$ *is a* local optimum *if the objective value* $c(s)$ *is at least as good as the objective value* $c(s')$ *of every other feasible solution* $s' \in N(s, I)$*.*

How hard is it to compute a **local** optimum?

In order to have some 'hope' for efficient algorithms, we should be at least able to compute one 'starting solution', the cost of a current solution, and we should be able to 'recognize' a local optimum if for any reason we are given one.

Definition 12 (Johnson, Papadimitriou, Yannakakis 1988)**.** *A local search problem* Π *belongs to the class* PLS*, for* Polynomial Local Search*, if the following polynomial-time algorithms exist:*

Algorithm A: Given an instance $I \in \mathcal{I}_\Pi$*, return a feasible solution* $s \in \mathcal{F}(I)$*.*

Algorithm B: Given an instance $I \in \mathcal{I}_\Pi$ *and a feasible solution* $s \in \mathcal{F}(I)$*, return the objective value* $c(s)$*.*

Algorithm C: Given an instance $I \in \mathcal{I}_\Pi$ *and a feasible solution* $s \in \mathcal{F}(I)$*, either certify that* s *is a local optimum or return a solution* $s' \in N(s, I)$ *with better objective value.*

For every problem in PLS one can apply the following natural heuristic:

Local Search Algorithm

1. Use Algorithm A to find a feasible solution $s \in \mathcal{F}(I)$.
2. Iteratively, use Algorithm C to find a better feasible solution $s' \in N(s, I)$ until a locally optimal solution is found.

Note that this local search procedure is guaranteed to terminate because there are only finitely many candidate solutions, and in each iteration the objective function strictly improves. However, because there can be exponentially many feasible solutions, the local search algorithm need not run in polynomial time.

Question: For a given local search problem Π, is there a polynomial-time algorithm (not necessarily local search) for finding a local optimum?

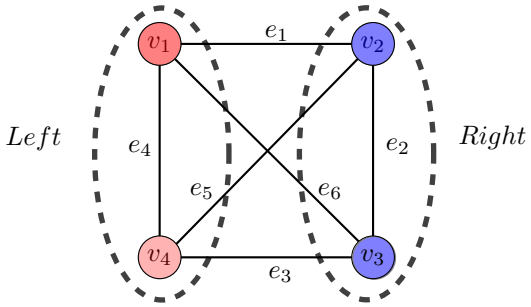
2.3 Max-Cut and PLS-Completeness

Definition 13 (**Max-Cut**)**.** *The search problem* Max-Cut *is defined as follows.*

Instances:	*Graph* $G = (V, E)$ *with edge weights* $w : E \to \mathbb{N}$*.*
Feasible solutions:	*A* cut*, which partitions* V *into two sets* Left *and* Right*.*
Objective function:	*The* value *of a cut is the weighted number of edges with one endpoint in* Left *and one endpoint in* Right*.*
Neighborhood:	*Two cuts are* neighboring *if one can obtain one from the other by moving only one node from* Left *to* Right *or vice versa.*

Observation 14 (**Membership in PLS**). *Max-Cut is a PLS problem.*

Example 15. *Consider the following instance of Max-Cut:*



Suppose the weights are

$$w_{e_1} = 1,\ w_{e_2} = 2,\ \ldots\ ,\ w_{e_6} = 6\ .$$

Then the cut that separates Left $= \{v_1, v_4\}$ *from Right* $= \{v_2, v_3\}$ *has a weight of* 15. *The four neighboring cuts each separate one vertex from the other three vertices. These cuts have weight* 11, 8, 11, *and* 12. *So the indicated cut is a* ***local optimum*** *but is* ***not a global optimum****: the cut that separates* v_1, v_2 *from* v_3, v_4 *has weight* 17.

Max-Cut unlike Min-Cut is an NP-hard problem, but we are not interested in a globally optimal solution. We only want a local optimum. Intuitively, computing a locally optimal solution should be easier. The next exercise provides a concrete example.

Exercise 2. *Consider Max-Cut in graphs* $G = (V, E)$ *with weights* $w_e = 1$ *for all* $e \in E$. *Computing a global optimum (a maximum cut) remains NP-hard also under this restriction. Show that we can find a local optimum with the local search algorithm in at most* $|E|$ *steps.*

Quite surprisingly, for Max-Cut in graphs with **general weights** no polynomial-time algorithm for computing a local optimum is known; and we can show that this problem is "as **hard** as" **any other local search** problem.

Definition 16 (PLS-reduction). *Given two PLS problems* Π_1 *and* Π_2, *there is a PLS-reduction (written* $\Pi_1 \leq_{PLS} \Pi_2$*) if there are two polynomial-algorithms* f *and* g*:*

- *Algorithm* f *maps every instance* $I \in \mathcal{I}_{\Pi_1}$ *to an instance* $f(I) \in \mathcal{I}_{\Pi_2}$.
- *Algorithm* g *maps every local optimum* s *of* $f(I) \in \mathcal{I}_{\Pi_2}$ *to a local optimum* $g(s)$ *of* $I \in \mathcal{I}_{\Pi_1}$.

As usual, one can read the symbol $\leq_{\text{PLS}}$ as "is not harder than". The reduction $\Pi_1 \leq_{\text{PLS}} \Pi_2$ gives us a way to derive an algorithm for Π_1 from an algorithm for Π_2.

Definition 17 (PLS-completeness). *A problem* Π^* *in PLS is called* PLS-complete *if, for every problem* Π *in PLS, it holds* $\Pi \leq_{PLS} \Pi^*$.

It is generally assumed that there are problems in PLS that cannot be solved in polynomial time. For this reason, showing PLS-completeness effectively shows that presumably there is no polynomial-time algorithm.

Theorem 18 (Schäffer and Yannakakis 1991)**.** *Max-Cut is PLS-complete.*

As in the theory of NP-completeness, showing PLS-completeness requires an "initial" PLS-complete problem. Such a problem was given by Johnson et al.; PLS-completeness of other problems such as Max-Cut can then be established through reduction. It is worth pointing out that in the original problem, local search takes (worst-case) exponential time and that all known reductions preserve these bad instances.

2.4 Pure Nash in General Congestion Games is PLS-Complete

The problem of computing a pure Nash equilibrium in potential games can be naturally regarded as a PLS problem:

Exercise 3 (**Membership in PLS**)**.** *Consider the problem of computing a pure Nash equilibrium in potential games. Explain how this problem can be formulated as a PLS problem (what is the set of instances, neighborhood, local optimum, etc.).*

Recall that congestion games are also potential games.

Theorem 19. *Max-Cut* $\leq_{PLS}$ *Pure Nash Equilibrium in Congestion Games*

Proof. For this reduction, we have to map instances of Max-Cut to congestion games. Given a graph $G = (V, E)$ with edge weights $w\colon E \to \mathbb{N}$, this game is defined as follows. Players correspond to the vertices V. For each edge $e \in E$, we add two resources r_e^{left} and r_e^{right} (see Figure 1(left)). The delays are defined by

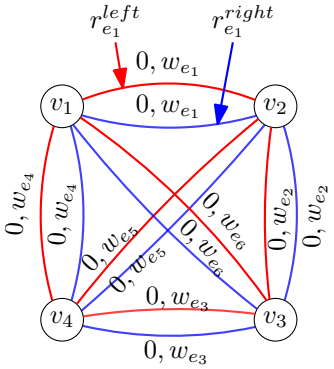
$$d_{r_e^{\text{left}}}(k) = d_{r_e^{\text{right}}}(k) = \begin{cases} 0 & \text{for } k = 1 \\ w_e & \text{for } k \geq 2 \end{cases}.$$

Each player $v \in V$ has two strategies, namely either to choose all the "left" resources for its incident edges $\{r_e^{\text{left}} \mid v \in e\}$ or all the "right" resources for its incident edges $\{r_e^{\text{right}} \mid v \in e\}$.

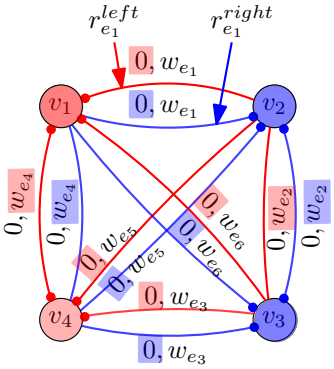
This way, cuts in the graphs are in one-to-one correspondence to strategy profiles of the game. A cut of weight W is mapped to a strategy profile of Rosenthal potential $\sum_{e \in E} w_e - W$ and vice versa. To see this, consider an edge $e \in E$. If its endpoints are in different sets of the cut, its resources contribute nothing to the potential; if its endpoints are in the same set, then the contribution is $0 + w_e = w_e$.

Consequently, local maxima of Max-Cut correspond to local minima of the Rosenthal potential, which are exactly the pure Nash equilibria. Therefore, the second part of the reduction is again trivial. □

Note that the above reduction was to a congestion game in which the players' strategy sets are not identical (so it is asymmetric) and required strategies to be arbitrary subsets of the resources (as opposed to, e.g., paths in a network). It turns out that either restriction can be dropped and the problem remains PLS-complete; but, as we have seen, dropping both turns the problem into one that we can solve in polynomial time.



(a) The instance derived from the Max-Cut instance in Example 15.



(b) The state corresponding to the cut in Example 15.

Figure 1: The main idea of the PLS-reduction from Max-Cut to congestion games. In the congestion game instance, a player can either choose (a) All All red resources incident to it – choose $Left$ in Max-Cut or (b) All blue resources incident to it – choose $Right$ in Max-Cut. The partition in Example 15 into two sets (cut) corresponds to the state in Figure 1b.

Recommended Literature

For the first part (PoA):

- B. Awerbuch, Y. Azar, A. Epstein. The Price of Routing Unsplittable Flow. STOC 2005. (PoA for pure NE in congestion games)
- G. Christodoulou, E. Koutsoupias. The Price of Anarchy of finite Congestion Games. STOC 2005. (PoA for pure NE in congestion games)
- T. Roughgarden. Intrinsic Robustness of the Price of Anarchy. STOC 2009. (Smoothness Framework and unification of previous results)
- T. Roughgarden. How bad is selfish routing? FOCS 2000. (PoA bound for non-atomic congestion games)

For the second part (computing pure Nash equilibria):

- D. S. Johnson, C. H. Papadimitriou, M. Yannakakis. How easy is local search? Journal of Computer and System Sciences, 37(1):79-100, 1988. (Class PLS, Cook-Like Theorem for CircuitFlip)
- A. Fabrikant, C. H. Papadimitriou, K. Talwar. The complexity of pure Nash equilibria. STOC 2004. (First Proof of PLS-Completeness of Pure Nash in Congestion Games)
- A. A. Schäffer and M. Yannakakis. Simple local search problems that are hard to solve. SIAM Journal on Computing, 20(1):56-87, 1991. (PLS-Completeness in Congestion Games via Max-Cut)

- H. Ackermann, H. Röglin, B. Vöcking. On the impact of combinatorial structure on congestion games. Journal of the ACM, 55(6), 2008. (Further Results on PLS-Completeness)

A significant part of this notes is from previous years' notes by Paul Dütting available here:

- https://www.cadmo.ethz.ch/education/lectures/HS15/agt_HS2015/index.html

Exercises
(during this exercise class - 2.10.2017)

We shall discuss and solve together these two exercises.

This exercise is about the upper bound on the time best response find a pure Nash equilibrium in singleton congestion games (see Lecture 1 notes).

Exercise 4. *I showed that Rosenthal's potential function with respect to the new delays $\bar{d}_r(k)$ can be upper-bounded as follows:*

$$\bar{\Phi}(s) = \sum_{r=1}^{m} \sum_{k=1}^{n_r(s)} \bar{d}_r(k) \leq \sum_{r=1}^{m} \sum_{k=1}^{n_r(s)} n\,m \leq (n\,m)^2 \ .$$

since we know that each $\bar{d}_r(k) \leq nm$.

I am not very clever in the upper bound above. Look at it again and show that $O(n^2\,m)$ is the correct bound.

We have seen an upper bound $\mathbf{5/2}$ *on the price of anarchy for* ***affine delays****. With this exercise we want to see if this assumption is needed (maybe the same holds for any congestion game/any delay function).*

Exercise 5. *For every $M \geq 1$, give an example of a two-player network congestion game whose price of anarchy for pure Nash equilibria is at least M.*