Algorithmic Game Theory, Summer 2018 Lecture 2 (4 pages)

Computing Equilibria in Congestion Games

Instructor: Thomas Kesselheim

In the last lecture, we have got to know Congestion Games and we proved that every Congestion Game has a pure Nash equilibrium. Today, we will discuss how to compute them. This is an important question: If equilibria are easy to find, then it is reasonable to assume that players actually end up in one. If, however, equilibria are hard to compute, then assuming that players are in equilibrium is a strong assumption.

Today's results will mainly be positive – later on we will see that it is actually hard to compute an equilibrium in general games.

1 Improvement Dynamics

One way of computing a pure Nash equilibrium was already implicit in our existence proof. We start from an arbitrary state. As long as there is a player who is not playing a best response, we let one of them switch to a strategy that is a better response. Rosenthal's theorem says that after finitely many steps, we reach an equilibrium. Note that this holds true without any assumptions on which player changes the strategy or how she changes it.

One particularly meaningful assumption is that players always switch to best responses. In this case, we speak of *best-response dynamics*.

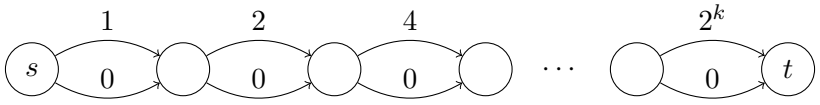
1.1 Length of Improvement Sequences

We would like to better understand the length of improvement sequences. If they are guaranteed to be short, we immediately have a good algorithm to compute equilibria. It is also a reasonable one: This is indeed a way how players could find an equilibrium.

We already know that improvement sequences are no longer than $2\sum_{r\in\mathcal{R}}\sum_{k=1}^{n}|d_r(k)| \leq 2mn\max_{r,k}|d_r(k)|$. Another upper bound is the number of states. There are at most $|\Sigma_1\times\ldots\times\Sigma_n| \leq 2^{mn}$ states.

Indeed, there can be sequences this long. To get a sense of what can happen, let us consider the following example.

Example 2.1. *The following example is a simple illustration of what can go wrong. We have only one player that wants to reach t from s in the following graph.*



There are $m = 2(k+1)$ many resources here but 2^{k+1} many strategies. Starting from the state in which the player uses all top edges, we can construct an improvement sequence of length exponential in m. The only best response would be to use all bottom edges instead. However, it is also an improvement to only switch the first edge. From this state, it is even an improvement to switch the first edge back to the top while switching the second edge to the bottom simultaneously.

Writing 1 for the top edge, 0 for the bottom edge, we get the sequence $111\ldots1 \to 011\ldots1 \to 101\ldots1 \to 001\ldots1 \to \ldots$. So, we have effectively implemented a binary counter. The initial cost is $2^{k+1}-1$ and it is reduced by 1 in every step, reaching 0 eventually. So the number of steps is $2^{k+1}-1$.

In this example, there is a much shorter improvement sequence (of length 1). However, there are (much more complex) examples in which *all* sequences are long. This means that there are n players, each of which only has a constant number of strategies, and states, such that exponentially many (in n) improvement steps are necessary before reaching a pure Nash equilibrium.

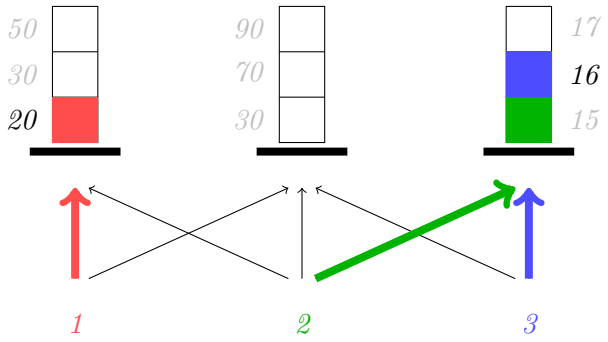
1.2 Singleton Congestion Games

We will show a significantly better, namely polynomial, guarantee for *singleton congestion games*. In this subclass of congestion games every player wants to allocate only a single resource at a time from a subset of allowed resources. Formally:

Definition 2.2 (Singleton Games)**.** *A congestion game is called* singleton *if, for every $i \in \mathcal{N}$ and every $R \in \Sigma_i$, it holds that $|R| = 1$.*

Although this constraint on the strategy sets is quite restrictive, there are still up to m^n different states.

Example 2.3 (Singleton Congestion Game)**.** *Consider a "server farm" with three servers a, b, c (resources) and three players 1,2,3 each of which wants to access a single server.*



The colored arrows indicate a pure Nash equilibrium.

Theorem 2.4. *In a singleton congestion game with n players and m resources, all improvement sequences have length $O(n^2 m)$.*

Proof. We will replace the original delays so that they get bounded by nm but the preferences do not change. To this end, sort the set of delay values $V = \{d_r(k) \mid r \in \mathcal{R}, 1 \leq k \leq n\}$ in increasing order. Define alternative, new delay functions:

$$\bar{d}_r(k) := \text{position of } d_r(k) \text{ in sorted list.}$$

The new delay of a player i using resource r in state S is now set to $\bar{\delta}_i(S) = \bar{d}_r(n_r(S))$. New cost functions $\bar{c}_i$ are defined based on these new delays.

We observe that if (S, S') is an improvement step for some player i with respect to the original delays, then (S, S') is an improvement step for i with respect to the new delays, as well. The reason is as follows. Using the singleton property, we can write $S_i = \{r\}$, $S'_i = \{r'\}$ and therefore $c_i(S) = d_r(n_r(S))$ and $c_i(S') = d_{r'}(n_{r'}(S'))$. As (S, S') is an improvement step for player i, we have $d_{r'}(n_{r'}(S')) < d_r(n_r(S))$ and also $\bar{d}_{r'}(n_{r'}(S')) < \bar{d}_r(n_r(S))$. This means that $\bar{c}_i(S') < \bar{c}_i(S)$.

Furthermore, observe that $\bar{d}_r(k) \leq nm$ for all $r \in \mathcal{R}$ and $k \in [n]$ because there are at most nm elements in V. Therefore, Rosenthal's potential function with respect to the new delays $\bar{d}_r(k)$ can be upper-bounded as follows:

$$\bar{\Phi}(S) = \sum_{r \in \mathcal{R}} \sum_{k=1}^{n_r(S)} \bar{d}_r(k) \leq \sum_{r \in \mathcal{R}} \sum_{k=1}^{n_r(S)} n\,m \leq n^2\,m \ ,$$

where in the last step we use $\sum_{r \in \mathcal{R}} n_r(S) = n$ because every player uses exactly one resource.

It holds that $\bar{\Phi} \geq 1$. Also, $\bar{\Phi}$ decreases by at least 1 in every step. Therefore, the length of every improvement sequence is upper-bounded by $n^2 m$. □

Example 2.5. *The sorted list of delay values in Example 2.3 is*

$$15, 16, 17, 20, 30, 50, 70, 90.$$

Hence, the old and new delay functions are

$$d_a(1,2,3) = (20,30,50) \qquad \bar{d}_a(1,2,3) = (4,5,6)$$
$$d_b(1,2,3) = (30,70,90) \qquad \bar{d}_b(1,2,3) = (5,7,8)$$
$$d_c(1,2,3) = (15,16,17) \qquad \bar{d}_c(1,2,3) = (1,2,3)$$

2 Pure Nash Equilibria as (Local) Minima

Following improvement steps is only one way to find pure Nash equilibria. In some cases, improvement sequences may be long but Nash equilibria can still be computed in polynomial time. Let us first show the following characterization of pure Nash equilibria in congestion games.

Lemma 2.6. *In a congestion game, a state S is a pure Nash equilibrium if and only if there is no state S' such that $S_i = S'_i$ for all but one i and $\Phi(S') < \Phi(S)$.*

In other words: Pure Nash equilibria are exactly the *local* minima of the Rosenthal potential. In this case, "local" refers to the fact that no state of smaller potential can be reached by unilateral deviation.

Proof. We will show that S is *not* a pure Nash equilibrium if and only if there is a state S' such that $S_i = S'_i$ for all but one i and $\Phi(S') < \Phi(S)$.

Let us first show the "if" part. That is, there is some S' that fulfills the condition and we have to use it to show that S is not a pure Nash equilibrium. Because S and S' only differ in the ith component, Rosenthal's theorem implies

$$\Phi(S') - \Phi(S) = c_i(S') - c_i(S) \ .$$

So, in combination, $c_i(S') < c_i(S)$, which means that S_i cannot be a best response against S_{-i}.

Coming to the "only if" part, if S is not a pure Nash equilibrium, there has to be a player i not playing a best response. That is, there is a state S' such that $S'_{-i} = S_{-i}$ and $c_i(S') < c_i(S)$. By Rosenthal's theorem

$$\Phi(S') - \Phi(S) = c_i(S') - c_i(S) \ ,$$

which means that $\Phi(S') < \Phi(S)$. □

Besides being very useful later, the lemma paves the way to a very simple algorithm to compute a pure Nash equilibrium that does not work via improvement sequences: Find a state S that minimizes $\Phi(S)$. Such a global minimum is clearly also a local minimum and therefore a pure Nash equilibrium.

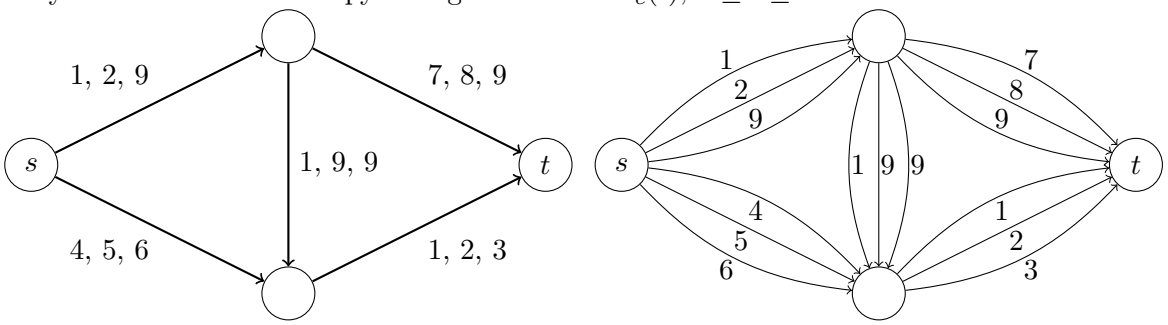
2.1 Symmetric Network Congestion Games

We will now consider symmetric network congestion games: There is a directed graph $G = (V, E)$ with delay functions $d_e \colon \{1, \dots, n\} \to \mathbb{Z}$, $e \in E$, with a source $s \in V$ and a target $t \in V$, such that each player wants to allocate a path of minimal delay between s and t.

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 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, it is possible to find a global optimum of the Rosenthal potential in polynomial time.

Theorem 2.7. *In symmetric network congestion games with non-decreasing delay functions, there is a polynomial-time algorithm that computes a state S that minimizes $\Phi(S)$. Therefore a pure Nash equilibrium can be computed in polynomial time.*

Proof. Based on the graph of the network congestion game, we define a graph, on which the algorithm computes a min-cost flow. To this end, each edge is replaced by n parallel edges of capacity 1 each and the ith copy of edge e has cost $d_e(i)$, $1 \le i \le n$.



In this graph, we compute a min-cost flow of n units from s to t. It is well known that this computation can be done in polynomial time and that the min-cost flow is integral. That is, it can be decomposed into the sum of n path flows each carrying 1 unit of flow, which we interpret as the players' strategies. The flow cost corresponds to the Rosenthal potential. This is only true because of monotonicity of the delay functions: For a resource r, if an flow-graph edge of cost $d_r(k+1) > d_r(k)$ is used by the flow, then also the edge of cost $d_r(k)$ is used. □

In general, finding a pure Nash equilibrium is not that easy. Indeed, under some complexity theoretic assumption, it is impossible in polynomial time.