Applied Algorithms. Prof. Tami Tamir
Missing Proofs, lecture 9. July 11th.

Theorem: Every congestion game is a potential game.

Proof: For a given profile a, define the potential of a to be

$\Phi(a) = \sum_{j=1}^{m} \sum_{k=1}^{n_j(a)} c_j(k)$. Assume that a is not a NE and that player i performs an improving step. All other players do not change their strategies. Denote by b the new profile, and by Δu_i the change in i's cost.

$\Delta u_i = \sum_{j \in b_i - a_i} c_j(n_{j(a)} + 1) - \sum_{j \in a_i - b_i} c_j(n_{j(a)})$.

Note that the change in the potential is exactly equal to Δu_i.

Thus, given that i's cost decreases, we conclude that $\Phi(b) < \Phi(a)$. That is, the potential function strictly decreases along the BRD process.

Note: When $\Delta u_i = \Delta\Phi$ we say that Φ is an exact potential function.

In the example: A player that moves from the top path to the lower path will pay 8 instead of 9. $\Delta u_i = (5+3) - (4+5) = -1$.

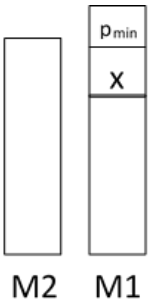
$\Phi(a) = c_{sx}(1) + c\_sx(2) + c\_sy(1) + c\_xy(1) + c\_xt(1) + c\_yt(1) + c\_yt(2)$
$\Phi(b) = c_{sx}(1) + c_{sx}(2) + c_{sx}(3) + c_{xy}(1) + c_{xt}(1) + c_{xt}(2) + c_{yt}(1)$

NE calculation in Symmetric Network Formation Congestion Games.

Let f be a max-flow in the network. The capacity of (s',s) implies that f has value n. Since all capacities are 1, it is possible to decompose the flow into n edge-disjoint s-t paths. Each such path is a possible strategy of some player. Denote by a(f) the profile induced by the flow.

Claim: Let f^* be a min-cost max-flow in the network. Consider $a(f^*)$. f^* has value n. For every pair of nodes u and v, if an edge (u,v) is used by $n_e(a)$ players, then, since f^* achieves min-cost, the $n_e(a)$ edges that are used in the flow are the cheapest ones, and the total flow cost is exactly $\Phi(a) = \sum_{j=1}^{m} \sum_{k=1}^{n_j(a(f*))} c_j(k)$.

We conclude that a min-cost flow corresponds to a profile with minimum potential – which must be a NE.



Theorem: LPT algorithm for job scheduling produces a NE schedule.

Proof:

Claim: If a job of length x is assigned on a machine M1 with load L1 (including x), then for every other machine, M2, with load L2, it holds that L1 ≤ L2+x.

Remark: L1 and L2 are the final loads of M1 and M2.

Proof: Case 1: L1≤ L2, then clearly L1 ≤ L2+x.

Case 2: L2<L1. Let $p_{min} \leq x$ be the length of the last job assigned on M1. Denote by L1t, L2t the loads on M1 and M2 before the assignment of p_{min}. By LPT rule, given that p_{min} was assigned on M1, we have L1t ≤ L2t. After the assignment of pmin, the load on M2 could

only increase. Therefore $L2 \geq L2t$. Also, by definition of p_{min}, $L1=L1t+p_{min}$. Combining the above inequalities, we get $L1 \leq L2t+p_{min} \leq L2t+x \leq L2+x$.

The claim implies that the gap in the loads between any two machines, is not larger than the length of the shortest job on the loaded machine.

In particular, a migration of a job of length x from M1 to M2 is not beneficial. Since the choice of x,M1 and M2 is arbitrary. Any schedule produced by LPT is a NE.

Theorem: Every job scheduling game has a strong NE.

Proof: We characterize a schedule by a vector of length m (L1,…,Lm) such that L1≥L2…≥Lm. That is, Li is the load on the i-th loaded machine.

Definition: Given two vectors X=(x1,…,xm) and Y=(y1,…,ym) we say that x is lexicographically smaller than y, (denoted $x \prec y$) if there exists an i such that $x_i<y_i$ and for all $j<I$, we have $x_j=y_j$. For example $(8,5,3) \prec (8,6,2)$.

Claim: Let p be a profile corresponding to a schedule achieving the minimal lexicographic load vector, then p is a SE (in particular, it achieves minimum makespan).

Remark: It is easy to see that this schedule is a NE – since every beneficial move corresponds to reducing the load vectors.

Proof: Assume by contradiction that some coalition exists. Let Γ be a coalition consisting of a minimal number of players. Let M(Γ) be the set of machines on which the jobs of Γ are assigned. We show that in every coordinated deviation of Γ, at least one job leaves and at least one job joins every machine in M(Γ). Consider a machine $M_a \in M(\Gamma)$.

-if no job leaves M_a, then p is not a NE – since every job migrating into M_a benefits.

-if no job joins M_a, then let j be a job that leaves M_a. Such a job exists since $M_a \in M(\Gamma)$. Note that Γ- {j} is also a coalition. Contradicting the minimality of Γ.

Let Mi be the most loaded machine in M(Γ). Its load in p is Li. By the above claim, some job joins Mi from some Mk. Therefore, the load on Mi after the deviation is less than Lk. Also $Lk \leq Li$ (Mi is most loaded). We conclude that the load on the most loaded machine reduces. Also, since the deviation is beneficial to all the coalition members, no other machine has load Li.

We conclude that the resulting load vector is lexicographically smaller – contradicting the definition of p as having the minimal lexicographic load vector.