Lecture 6:Congestion and potential games

Lecturer:Yishay Mansour *Scribe:Nir Yosef, Ami Koren*

6.1 Lecture overview

So far we've seen that not every strategic game has a deterministic *Nash* equilibrium.
In this lecture we discuss a certain class of games: congestion and potential games, for which we prove the existence of a deterministic *Nash* equilibrium.
In the coming sections we define the above classes, show the relation between them and estimate the complexity of finding a deterministic *Nash* equilibrium for a potential game.

6.2 Congestion game

6.2.1 Example

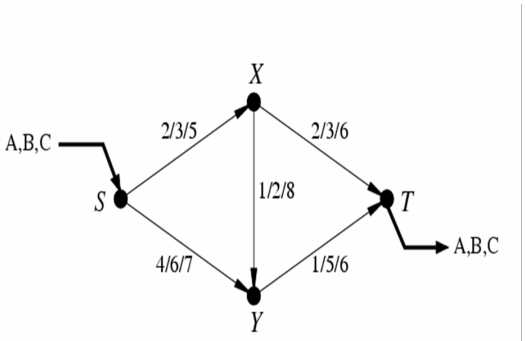

Fig.1 Example of a congestion game

Let us start with an illustrative example: In the model described above, Players A,B and C have to go from point S to T using road segments SX,XY,...etc. Numbers on edges denote the cost for a single user for using the corresponding road segment, where the actual cost is a function of the actual number of players using that road segment(i.e. a *descrete delay* function). For example: if segment SX is used by a 1,2, or 3 users, the cost on that segment would be 2,3, or 5, respectively. The total cost of each player is the sum of all segments he uses. Note that the players are therefore engaged in a game which can be represented in a strategic form(as a cost matrix).

6.2.2 Congestion game - Definition

A congestion model (N,M,$(A_i)_{i\in N}$,$(c_j)_{j\in M}$) is defined as follows:

- $N = \{1..n\}$ denotes the set of players
- $M = \{1..m\}$ denotes the set of facilities
- For $i \in N$, A_i denotes the set of strategies of player i, where each $a_i \in A_i$ is a non empty subset of the facilities.
- For $j \in M$, $c_j \in R^n$ denotes the vector of costs, where $c_j(k)$ is the cost related to each user of facility j, if there are exactly k players using that facility.

The *congestion game* associated with a congestion model is a game in strategic form with the set of N players, with sets of strategies $(A_i)_{i\in N}$ and with cost function defined as follows: Let $A = \times_{i\in N} A_i$ be the set of all possible deterministic profiles(players strategy vectors). For any $\vec{a} \in A$ and for any $j \in M$, let $n_j(\vec{a})$ be the number of players using facility j, assuming $\vec{a}$ to be the current profile.
Now, define the overall cost function for player i: $u_i(\vec{a}) = \sum_{j\in a_i} c_j(n_j(\vec{a}))$

Remark 6.1 *All players are equal in a sense that they have the same 'weight'(it doesn't matter* which *players are using a facility, only* how many *players are using it).*

6.2.3 Deterministic equilibrium

Theorem 6.2 *Every finite congestion game has a pure strategy(deterministic) equilibrium.*

Proof: Let $\vec{a} \in A$ be a deterministic strategy vector as defined above,
let Φ: $A \to R$ be a potential function defined as follows: $\Phi(\vec{a}) = \sum_{j=1}^{m} \sum_{k=1}^{n_j(\vec{a})} c_j(k)$
Consider the case where a single player changes its strategy from a_i to b_i(where $a_i, b_i \in A_i$).
Let Δu_i be the change in its cost caused by the the change in strategy:
$\Delta u_i = u_i(b_i, \vec{a_{-i}}) - u_i(a_i, \vec{a_{-i}}) = \sum_{j\in b_i - a_i} c_j(n_j(\vec{a}) + 1) - \sum_{j\in a_i - b_i} c_j(n_j(\vec{a}))$
(explanation: change in cost = cost related to the use of new facilities minus cost related to use of those facilities which are not in use anymore due to strategy change)
Let $\Delta\Phi$ be the change in the potential caused by the change in strategy:
$\Delta\Phi = \Phi(b_i, \vec{a_{-i}}) - \Phi(a_i, \vec{a_{-i}}) = \sum_{j\in b_i - a_i} c_j((n_j(\vec{a}) + 1) - \sum_{j\in a_i - b_i} c_j(n_j(\vec{a}))$
(explanation: immediate from potential function's definition)
Thus we can conclude that for a single player's strategy change we get
$\Delta\Phi = \Delta u_i$.

That's an interesting result: We can start from an arbitrary deterministic strategy vector $\vec{a}$, and at each step one player reduces it's cost. That means, that at each step Φ is reduced identically. Since Φ can accept a finite amount of values, it will eventually reach a local minima. At this point, no player can achieve any improvement, and we reach a *NE*. □

6.3 Potential games

6.3.1 Potential functions

Let $G =< N, (A_i), (u_i) >$ be a game in strategic form and let $A = \times_{i \in N} A_i$ be the collection of all deterministic strategy vectors in G.

Definition A function Φ: $A \to R$ is an *exact potential* for game G if
$\forall_{\vec{a} \in A} \forall_{a_i, b_i \in A_i}\ \Phi(b_i, \vec{a_{-i}}) - \Phi(a_i, \vec{a_{-i}}) = u_i(b_i, \vec{a_{-i}}) - u_i(a_i, \vec{a_{-i}})$

Definition A function Φ: $A \to R$ is a *weighted potential* for game G if
$\forall_{\vec{a} \in A} \forall_{a_i, b_i \in A_i}\ \Phi(b_i, \vec{a_{-i}}) - \Phi(a_i, \vec{a_{-i}}) = \omega_i(u_i(b_i, \vec{a_{-i}}) - u_i(a_i, \vec{a_{-i}})) = \omega_i \Delta u_i$
Where $(\omega_i)_{i \in N}$ is a vector of positive numbers (weight vector).

Definition A function Φ: $A \to R$ is an *ordinal potential* for a minimum game G if
$\forall_{\vec{a} \in A} \forall_{a_i, b_i \in A_i}\ (\Phi(b_i, \vec{a_{-i}}) - \Phi(a_i, \vec{a_{-i}}) < 0) \Leftarrow (u_i(b_i, \vec{a_{-i}}) - u_i(a_i, \vec{a_{-i}}) < 0)$ (the opposite takes place for a maximum game).

Remark 6.3 *Considering the above definitions, it can be seen that the first two definitions are private cases of the third.*

6.3.2 Potential games

Definition A game G is called an *ordinal potential game* if it admits an ordinal potential.

Theorem 6.4 *Every finite ordinal potential game has a pure strategy(deterministic) equilibrium.*

Proof: Similarly to the previous proof, starting from an arbitrary deterministic strategy vector,after a finite number of steps of single player improvement, we will reach a local minima which is, as was explained above, a deterministic equilibrium. □

6.3.3 Examples

Exact potential game

Consider an undirected graph $G = (V, E)$ with a weight function $\vec{\omega}$ on its edges. The goal is to partition the vertices set V into two distinct subsets D_1, D_2(where $D_1 \cup D_2 = V$):
for every player i, choose $s_i \in \{-1, 1\}$ where choosing $s_i = 1$ means that $i \in D_1$ and the opposite for D_2. The weight on each edge denotes how much the corresponding vertices 'want' to be on the same set. Thus, define the value function of player i as $u_i(\vec{s}) = \sum_{j \neq i} \omega_{i,j} s_i s_j$

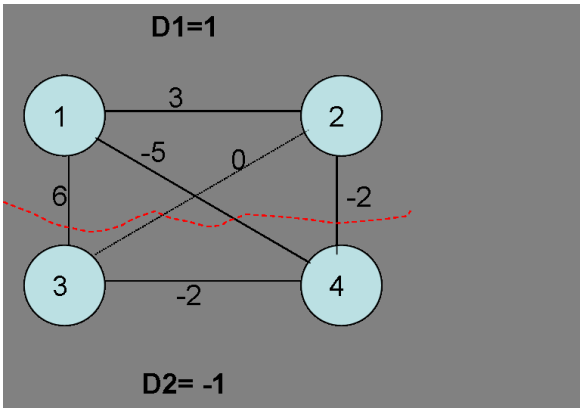

Fig.2 Example for an exact potential game

On the example given in figure 2 it can be seen that players 1,2,4 have no interest in changing their strategies, However, player 3 is not satisfied, it can increase his profit by changing his chosen set to D_1.
Using $\Phi(\vec{s}) = \sum_{j<i} \omega_{i,j} s_i s_j$ as our potential function, let us consider the case where a single player i changes its strategy(shifts from one set to another):
$\Delta u_i = \sum_{j \neq i} \omega_{i,j} s_i s_j - \sum_{j \neq i} \omega_{i,j}(-s_i) s_j = 2 \sum_{j \neq i} \omega_{i,j} s_i s_j =$
$2 \sum_{j:\ j<i} \omega_{i,j} s_i s_j + 2 \sum_{j:\ i<j} \omega_{i,j} s_i s_j = \Delta(\Phi)$
Which means that Φ is an exact potential function, therefore we conclude that the above game is an exact potential game.

Weighted potential game

Consider the following load balancing congestion model(N,M,$(\omega_i)_{i \in N}$) with M identical machines, N jobs and $(\omega_i)_{i \in N}$ weight vector($\omega_i \in R^+$). The load on a machine is defined as the sum of weights of the jobs which use it: $L_j(\vec{a}) = \sum_{i:\ a_i = j} \omega_i$ where $\vec{a} \in [1..M]^N$ is a deterministic strategy vector.
Let $u_i(\vec{a}) = L_{a_i}(\vec{a})$ denote the cost function of player i. We would like to define a potential function whose reaction to a single player's strategy change will be correlated with the reaction on the player's cost function.
The potential function is defined as follows: $\Phi(\vec{a}) = \sum_{j=1}^{M} \frac{1}{2} L_j^2$, consider the case where a single job shifts from its selected machine M_1 to another machine M_2 (where M_1 and M_2 are two arbitrary machines):
Let Δu_i be the change in its cost caused by the strategy change:
$\Delta u_i = u_i(M_2, \vec{a_{-i}}) - u_i(M_1, \vec{a_{-i}}) = L_2(\vec{a}) + \omega_i - L_1(\vec{a})$
(explanation: change in job's load = load on new machine minus load on old machine)
Let $\Delta\Phi$ be the change in the potential caused by the strategy change:
$\Delta\Phi = \Phi(M_2, \vec{a_{-i}}) - \Phi(M_1, \vec{a_{-i}}) = \frac{1}{2}[(L_1(\vec{a}) - \omega_i)^2 + (L_2(\vec{a}) + \omega_i)^2 - L_1^2(\vec{a}) - L_2^2(\vec{a})] =$
$= \omega_i(L_2(\vec{a}) - L_1(\vec{a})) + \omega_i^2 = \omega_i(L_2(\vec{a}) + \omega_i - L_1(\vec{a})) = \omega_i \Delta u_i$

Therefore, we can conclude that the model at hand is a weighted potential game.

General(ordinal) potential game

Consider the following load balancing congestion model$(N,M,(\omega_{i,j})_{i\in N,j\in M})$ with M related machines, N jobs and $\vec{\omega}$ a machine dependent weight vector(where $\omega_{i,j} \in N^+$ is the weight of job i having been assigned to machine j).
Here we have similar definitions to those we have presented in the above example:
Load on a machine j: $L_j(\vec{a}) = \sum_{i:\ a_i=j} \omega_{i,j}$
Cost function for player i: $u_i(\vec{a}) = L_{a_i}(\vec{a})$
The potential function will now be defined as $\Phi(\vec{a}) = \sum_{j=1}^{M} 4^{L_j(\vec{a})}$
Consider the case where a single job shifts from its selected machine M_1 to another machine M_2. Change in players' cost is calculated in a similar fashion to the above example:
$\Delta u_i = L_2(\vec{a}) + \omega_{i,2} - L_1(\vec{a})$
Change in the potential caused by the strategy change will now be:
$\Delta\Phi = 4^{L_1(\vec{a})-\omega_{i,1}} + 4^{L_2(\vec{a})+\omega_{i,2}} - 4^{L_1(\vec{a})} - 4^{L_2(\vec{a})}$
If $\Delta u_i < 0 \Rightarrow L_2(\vec{a}) + \omega_{i,2} < L_1(\vec{a}) \Rightarrow L_2(\vec{a}) + \omega_{i,2} + 1 \leq L_1(\vec{a})$.
In addition $L_1(\vec{a}) - \omega_{i,1} + 1 \leq L_1(\vec{a})$ (both conclusions under the assumption that$(\omega_{i,j} \in N^+)$). From the two inequalities we conclude:
$\{4^{L_1(\vec{a})-\omega_{i,1}} \leq 4^{L_1(\vec{a})-1}, 4^{L_2(\vec{a})+\omega_{i,2}} \leq 4^{L_1(\vec{a})-1}\} \Rightarrow \Delta\Phi \leq 2\cdot 4^{L_1(\vec{a})-1} - 4^{L_1(\vec{a})} = -2\cdot 4^{L_1(\vec{a})-1} < 0.$
Therefore, we can conclude that the model at hand is a general potential game.

Another example of a general potential game is given by taking the same model we have described as an example for an exact potential game, along with a slightly different value function $u_i(\vec{s}) = SIGN(\sum_{j\neq i} \omega_{i,j} s_i s_j)$ and the same potential function.
Following a similar proof it can be seen that this time we get an ordinal potential.

6.3.4 Finite improvement path

We use the concept of a Finite improvement path in order to define an abstract ordinal potential function for a given strategic game. A finite improvement path is defined as follows:
For $G =< N, (A_i), (u_i) >$ minimum game in strategic form, and $A = \times_{i\in N} A_i$ collection of all deterministic strategy vectors let $\Pi = (V, E)$ be a graph such that
$V = A$ and $E = \{< \vec{a}, \vec{b} >\in A^2$: $\exists i\ [(b_i, \vec{a_{-i}}) = \vec{b}\ ] \wedge [u_i(\vec{b}) < u_i(\vec{a})]\}$

Lemma 6.5 *If* Π *is acyclic then the corresponding game* G *possesses a deterministic equilibrium.*

Proof: Every acyclic graph has a sink(vertex without outgoing edges). Every sink on Π is a deterministic equilibrium(follows immediately from the definition of E). □

Now let us define the potential function $\Phi(\vec{a})$ as the length of the longest route on Π starting from $\vec{a}$. Note that going from one vertex to another on Π(which is equivalent to a step where a single user changes its strategy, thus reducing its cost) will reduce the value of Φ (immediate from the definition of Φ). In addition - the number of such steps is final(because G is final).
Having said that and from the definition at **8.3.1** we can conclude that Φ is an ordinal potential function.
Every path on that graph is an improvement path with respect to the above potential function. The last vertex of every such path corresponds to an equilibrium point, as was explained in the proof above.

6.4 Computing equilibrium in congestion games

We have seen on proof of theorem **8.2** that every general congestion game has an exact potential function. We can use that potential function in order to find an equilibrium point(by following the same scheme as described on the proof of theorem **8.2**). The problem is that the number of steps might be exponential in the size of the game.

6.4.1 Symmetric network's game

(An example for computing equilibrium using reduction)
A symmetric network's game NG is defined as follows: given a graph $G = (V, E)$ with source and destination vertices (S, T), the players have to choose a route on G leading from S to T. Each edge has a delay value which is a function of number of players using it.
Now, let us look at the full definition of NG as a congestion game $(\mathrm{N},\mathrm{E},(A_i)_{i\in N},(c_e)_{e\in E})$:

- N denotes the set of players
- E denotes the set of edges of G
- A_i is the set of strategies of player i, where each $a_i \in A_i$ is a route on G leading from S to T
- For $e \in E$, $c_e \in R^n$ denotes the vector of delays, where $c_e(k)$ is the delay related to edge e, if there are exactly k players using that edge.
- Player's cost function is $u_i(\vec{a}) = \sum_{e\in a_i} c_e(n_e(\vec{a}))$ (where n_e as before denotes the number of players using edge e)

Remark 6.6 *On proof of theorem 8.2 we saw that for this kind of game the potential function* $\Phi(\vec{a}) = \sum_{e=1}^{m}\sum_{k=1}^{n_e(\vec{a})} c_e(k)$ *is exact.*

reduction to min-cost flow

considering the graph $G = (V, E)$ and the delay functions $\{c_e\}_{e \in E}$, we replace in G each edge e with n parallel edges between the same nodes, each with capacity 1, and with costs $c_e(1), .., c_e(n)$.

Lemma 6.7 *The cost of a full min-cost flow of* n *units on the new network is equal to* $\Phi(\vec{a})$ *where* $\vec{a}$ *is a strategy vector corresponding to that flow.*

Proof: Let f be a full flow of n units.f can be divided into n distinct routes(because the capacity of each edge is exactly 1). Consider every route as the strategy of a single player of the game *NG*. We define the corresponding strategy vector $\vec{a}$ as some ordered collection of all these routes(it is not important in which order we define $\vec{a}$ because all players are equal). Since f is minimal then it will first use the cheaper edges, therefore the contribution of a collection of edges $e_1, .., e_n$ on the new network which corresponds to single edge e on G will be $\sum_{k=1}^{n_e(\vec{a})} c_e(k)$ and the total cost of f is thus $\sum_e \sum_{k=1}^{n_e(\vec{a})} c_e(k) = \Phi(\vec{a})$

□

Remark 6.8

- *Any minima on* Φ *is an equilibrium point(Immediate from the definition of* Φ*).*
- *It is easy to see that for any integer min-cost flow in the new network, the strategy vector corresponding to that flow minimizes* Φ*.*

Corollary 6.9 *For every min-cost flow on the network defined above, the corresponding strategy vector is an equilibrium point.*

Corollary 6.10 *There is a polynomial algorithm for finding a pure equilibrium in symmetric network congestion games.*

6.4.2 PLS class

We saw that we can always find deterministic equilibrium in general congestion game. We also saw that in some cases, we have polynomial algorithm for doing that. How hard it is to find the equilibrium? We will now show, that in some cases, the problem becomes exponential. In order to do this, we will define a class for local-optima search.

Definition PLS class(Polynomial-time Local Search)
Terminology:

- I - Collection of possible inputs(graphs, for example)

- F_x - For each instance $x \in I$ we have a finite set F_x of the possible solutions, all with the same polynomially bounded length(For example, all *TSP* paths at each graph).
- $c(x, s)$ - Cost function of a solution $s \in F_x$ given an instance $x \in I$. For example- for each graph $x \in I$ and path $s \in F_x$, $c(x, s)$ marks the cost of the path(For $s \notin F_x$ $c()$ returns "illegal")
- $N_x(s) \subseteq F_x$ - Neighborhood function. This function defines the environment of each possible solution.
- both $N_x(s)$ and F_x are recognizable in polynomial time

The problem: finding a local optimum solution. That is, to find $x \in I$ and $s \in F_x$, such that $\forall \grave{s} \in N_x(s) : \ c(x, \grave{s}) \geq c(x, s)$

For complete definition of PLS, see [3].

Sample for generic problem

- $I = \{0, 1\}^n$
- $F_x = I$ -(has no meaning for solution)
- $N_x = \{y | H(x, y) = 1\}$ - The set of neighbors of each vector, defined as the set of all vectors which differ from it in exactly one bit(Hamming distance of 1)
- $c(x)$ - Some generic cost function

This problem can be thought of as seeking for a local minimal-cost vector among the set $\{0, 1\}^n$ where locality is defined by hamming distance.
It can be shown that there are cost functions for which there exists an improvement path at length $\frac{1}{4}2^n \leq l \leq 2^n$(*Snake in a box*). This means that the improvement path has to go through a significant number of the edges of the box in order to reach a local optima.

The complexity of the problem is cost-function depended. For example, if we define cost function c(x) : $I \rightarrow \{0, 1\}$, we have an easy 1-step algorithm: Check all your neighbors, if they're all 1 - exit, else - go to the 0-neighbor and exit. The problem is getting hard when there are a lot of different costs, since the improvement at each step can be much smaller then the range of the costs.

PLS-Complete problems
Definition $L \in PLS$ is **PLS-complete** when every $\Pi \in PLS$ is reducible to L

Here are some of the problem which are known to be PLS-Complete:

- Local minima in a cube(See above)
- MAX-CUT - Find the edges that give maximal weight cut of graph vertices - In the local version, we consider two cuts as neighbors, if they differ in exactly one vertex
- W2SAT - Try to satisfy as much clauses as possible(in cases that the expression is not satisfiable). In the local-version, we consider two solutions as neighbors if they differ in the value of exactly one variable(i.e. our goal is to reach a solution where a single switch of variable will not improve the cost)
- TSP_2 - Local version of Traveling Sales Person, where 2 routes are considered as neighbors if they differ in at most 2 edges

Theorem 6.11 *A general congestion game, symmetric congestion game, and asymmetric network game are all PLS-complete*

Proof:
We will show that a generic congestion game is PLS-complete. We will do it by a reduction from a known PLS-complete problem:

Weighted not-all-equal 3SAT is a PLS-complete problem described as follows:

- **Input** - Monotonic 3-CNF. A clause is considered satisfied if it is not all-1 and not all-0. There is a weight for each clause
- **Output** - An assignment that satisfies maximal weight of clauses
- **Cost** - Sum of weights on all unsatisfied clauses
- **Locality** - In the local version, we consider two assignments as neighbors if they have different values for exactly one variable

The Reduction
Given an instance of the 3SAT problem above, we build a corresponding congestion game. We want to show that the *Nash* equilibrium at the congestion game is equivalent to local-minima at the above 3SAT problem. Therefore, had we known to find a *Nash* equilibrium at the congestion game, we would have been able to solve our 3SAT problem.

For any given *3-CNF*, we build the corresponding congestion game as follows:

- variable $\longrightarrow$ player

- clause $T_j \longrightarrow$ 2 resources: m_j^0, m_j^1
- action $\longrightarrow$ For a player we have 2 actions:
  - $I = \{m_j^1 |\ x_i \in T_j\}$
  - $II = \{m_j^0 |\ x_i \in T_j\}$

  Explanation: Each player can choose whether to play all on 0, or all on 1
- 3SAT cost function: For $a \in \{0,1\}$, $C_{m_j^a}(1) = C_{m_j^a}(2) = 0, C_{m_j^a}(3) = w_i$ - This cost function punishes clauses with all-equal values. $C_{m_j^a}(0) = 0$, because we already punish it at $C_{m_j^{1-a}}(3)$.
- assignment of $x_i = 1$ $(3SAT) \Longleftrightarrow$ Player's action A_i=I(congestion game) , which means that for every vector of deterministic strategy on the congestion game there exists a corresponding assignment on(3SAT) and vice versa.

After defining the game, we show the equivalence between *Nash* equilibrium at the congestion game to local minima at the our 3SAT problem.

- Change of player's action at game: Assuming a player changed it's action from I to II:
  - D_1 - The clauses that became satisfied as result of the change
  - D_2 - The clauses that became unsatisfied as result of the change

  The gain from the change:

$$\triangle u^i = \sum_{j \in D_2} w_j - \sum_{j \in D_1} w_j$$

- Switching a variable's value at 3SAT will have the same affect:

$$\triangle u^i = \sum_{j \in D_2} w_j - \sum_{j \in D_1} w_j$$

Therefore, *Nash* equilibrium at the congestion game is equivalent to a local-minima at the 3SAT problem

□

6.4.3 ε-*Nash* of congestion game

Finding deterministic *Nash* equilibrium for a general congestion game is hard, but it might be easier in some cases. For instance, for small number of players, or for finding ε-*Nash*.

We now present an algorithm for finding an ε-*Nash* of a congestion game with the following potential function:

$$\phi(\overrightarrow{a}) = \sum_{j=1}^{m} \sum_{k=1}^{n_j} c_j(n_k)$$

We start from an arbitrary deterministic strategy vector $\vec{a_0}$. At each step we decrease ϕ with at least ε. If we can't, we reached the ε-*Nash*. Since for each $\vec{a}$ we have

$$\phi(\overrightarrow{a}) \leq \sum_{j=1}^{m} n_j c_j(n_j) \leq n \cdot m \cdot c_{max}$$

Then the number of steps is at most $\frac{\phi(\overrightarrow{a_0})}{\varepsilon}$, which is limited by $\frac{n \cdot m \cdot c_{max}}{\varepsilon}$

6.5 Equivalence of potential and congestion game

At the beginning of this lecture we saw that each congestion game admits a potential function and therefore it is a potential game. We now show the other direction: For each exact potential game there exists a matching congestion game.

Definition Equivalence of games

Assuming there are 2 games: $< N, (A_1^i)_{i \in N}, (u_1^i)_{i \in N} >$, $< N, (A_2^i)_{i \in N}, (u_2^i)_{i \in N} >$

If there is a 1-1 mapping between game 1 and game 2: $g^i : A_1^i \longrightarrow A_2^i$ such that:

$$u_1^i(a_1, \ldots, a_n) = u_2^i(g^i(a_1), \ldots, g^i(a_n))$$

We say that the games are equivalent.

Theorem 6.12 *Each game with exact potential function has equivalent congestion game*

Proof:

Given a game $< N, (A^i)_{i \in N}, (u_1^i)_{i \in N} >$ with an exact potential function ϕ. For simplicity let us assume that $\forall i \in N : \ l = |A^i|$.

We'll start building an equivalent congestion game $< N, V, (c_j)_{j \in M} >$.

Players: Stay the same

Resources: $V = \{0,1\}^{l \cdot n}$ - Each vector represent a single resource, which gives us a total of $2^{l \cdot n}$ resources.
We treat each of these vectors as n vectors, each of size l.

Holding resources: Each player i that play action j, holds the resources with the appropriate bit set: $B_j^i = \{v : \ v_{ij} = 1\}$ (therefore, $|B_j^i| = \frac{2^{ln}}{2}$).

We now seek for a cost function on the resources of the new congestion game, such that the cost u_2^i for each player will satisfy: $u_2^i(a_1, \ldots, a_n) = u_1^i(a_1, \ldots, a_n)$ (according to the definition above).
We define the cost function c for specific resources that will sum to the desired cost for each player. All other resources will have a cost=0.

Given strategy vector $\vec{a}$, then for the vector $v^{\vec{a}}_{i',j'} \in \{0,1\}^{l*n}$ such that:

$$v^{\vec{a}}_{i',j'} = \begin{cases} 1 & j' = a_{i'} \\ 0 & otherwise \end{cases}$$

There is a congestion of exactly n(because it is held by all players). For this vector, we define: $c_{v^{\vec{a}}}(k) = \phi(\overrightarrow{a})$ for $k = n$ and 0 otherwise.

If these resources were the only ones with non-zero-cost, then each player would have a cost of $\phi(\vec{a})$. Therefore, we have to fix it for each player i, by $u_1^i(\overrightarrow{a}) - \phi(\overrightarrow{a})$.

In order to do this, we find a resource $r^i \in V$ that only player$i$ holds. For $\vec{a}$, we define:

$$r^i_{i',j'} = \begin{cases} 1 & i' = i \\ 0 & i' \neq i, j' \in a_{i'} \\ 1 & \text{Otherwise} \end{cases}$$

Meaning: r^i is 1 anywhere, except at the locations corresponding to the actions of the rest of the players
The cost we define for these resources are:

$$c_{r^i}(1) = u^i(\overrightarrow{a}) - \phi(\overrightarrow{a})$$

We have to show that $c_{r^i}(1)$ is well defined for player i. That is, that it doesn't change if the player changes it's action. We do it by using the fact that ϕ is an exact potential, and therefore, for actions a_i, b_i of player i:

$$u_1^i(a_i, a^{-i}) - \phi(a_i, a^{-i}) = u_1^i(b_i, a^{-i}) - \phi(b_i, a^{-i})$$

$$\Rightarrow u_1^i(a_i, a^{-i}) - u^i(b_i, a^{-i}) = \phi(a_i, a^{-i}) - \phi(b_i, a^{-i})$$

$$\Rightarrow \Delta u_1^i(a_i \to b_i) = \Delta\phi(a_i \to b_i)$$
$$\Rightarrow \Delta c_{r^i} = \Delta u_1^i(a_i \to b_i) - \Delta\phi(a_i \to b_i) = 0$$

All other resources have cost=0.

The cost of player i:
$u_2^i(\vec{a}) = \sum_{j \in a_i} c_j(n_j(\vec{a})) = c_{v^{\vec{a}}}(n) + c_{r^i}(1) = \phi(\vec{a}) + u^i(\vec{a}) - \phi(\vec{a}) = u^i(\vec{a})$

Therefore, by the definition above, the potential game is equivalent to the congestion game we've built.

Notice, that the number of resources is exponential in this representation. That means that a general potential game might be much harder then a congestion game.

□

6.6 Bibliography

[1] *The Complexity of Pure Nash Equilibria*, A. Fabrikant, C. Papadimitriou, K. Talawar
[2] *Potential Games*, D. Monderer, L. Shapley, 1994
[3] *How Easy Is Local Search?*, D. Johnson, C. Papadimitriou, M. Yannakakis, 2004, http://faculty.cs.tan