Vrije Universiteit Brussel

Faculty of Science
Department of Mathematics

Game Theory: Distributed Selfish Load Balancing on Networks

Bachelor Thesis I

Filip Moons

Promotor: Prof. Dr. Ann Nowé

January 2013

Abstract for non-mathematicians

Explaining the subject of my Bachelor thesis to those who do not study either Mathematics or Computer Science isn't an easy task, but I can give an idea of the problem studied by giving an example. Imagine, for example, that there are only two roads from Paris to Brussels: road A & road B. If you take road A you'll always drive 5 hours, independent from the usage of road A by other drivers (*agents* is the more game theoretical word). If you take road B, the time you need depends of the number of other agents using road B: you'll drive $5\frac{x}{100}$ hours, with x the number of agents, but also on this road the maximum time spend is 5 hours, so mathematically that becomes $\min(5, 5\frac{x}{100})$ hours. Because every agent acts selfish and rational, the result under this circumstances and with only this information will be that every agent will take at any time road B: with road A they drive always 5 hours, with road B there is a chance to get to Brussels in less time. But consider now the following situation: exact 100 agents decide at the same time to go to Brussels, that means that all of them will take road B and so everyone will drive 5 hours. It would be much better that 50 agents would take road A (and thus drive 5 hours) and the other 50 take road B (and drive $5\frac{50}{100} = 2.5$ hours), that would reduce the average time to 3.75 hours! This kind of problems where the selfishness of agents reduce the power (load) of a general system take place in a wide range of real life problems: not only in traffic, but also in economics, politics and, that's more my interest, also in Computer Science: in a distributed system computers interact on a selfish base because there isn't a central computer that organizes the network. It's extremely interesting to study which information these kind of agents (computers) need to know to reduce their selfishness and reach the optimal power of the distributed network.

Contents

1 Introduction

Whenever a set of task should be executed on a set of resources, the problem of load balancing evidently arouses. We need to balance the load among the resources in order to exploit the available resources efficiently and fair.

1.1 Why we need Game Theory

One of the most fundamental load balancing problems is *makespan scheduling on uniformly related machines*. This problem is defined by m machines with speeds $s_1, ...s_m$ and n tasks with weights $w_1, ..., w_n$. Let $[n] = \{1, ..., n\}$ be to set of tasks and $[m]$ the set of machines. Now, the problem is to find an assignment function $A : [n] \to [m]$ of the tasks to the machines that is as balanced as possible. The load of machine $j \in [m]$ under A is defined as:

$$\ell_j = \sum_{\substack{i \in [n] \\ j = A(i)}} \frac{w_i}{s_j}$$

The *makespan* is defined as

$$\max_{j \in [m]} (\ell_j)$$

Now, of course, the objective is to minimize the makespan. When there is a central machine, it ins't that hard to design algorithms that compute a mapping A that minimizes the makespan. Suppose, however, that there is not a central machine that can enforce an efficient mapping of the tasks to the machines (e.g. in P2P Networks). This naturally leads to the following game theoretic setting in which we will be able to analyze what happens to the makespan if there is no global authority but selfish agents aiming at maximizing their individual benefit decide about the assignment of tasks. The understand the problem and it's solution, we first give the most important game theoretical results you'll need.

2 Introducing Load Balancing Games

Important note: This section doesn't aim to give the reader an introduction in Game Theory. Instead, it gives only the relevant results that the reader must know for understanding the rest of this paper.

2.1 Strategic games

2.1.1 Defenition

Definition 2.1. *[JOARU1994] A strategic game $\langle N, (A_i), \succeq_i \rangle$ consists of:*

- *a finite set N (the set of **players**),*
- *for each player $i \in N$ a nonempty set A_i (the **set of actions** available to player i),*
- *for each player $i \in N$ a preference relation $\succeq_i$ on $A = \times_{j \in N} A_j$ (the **preference relation** of player i).*

Remark 2.2. *Under a wide range of circumstances the preference relation $\succeq_i$ of player i in a strategic game can be represented by a* ***payoff function*** *$u_i : A \to \mathbb{R}$, in the sense that $u_i(a) \geq u_i(b)$ whenever $a \succeq_i b$. Sometimes this function is also called a utility function. A strategic game is then often denoted as $G = \langle N, (A_i), (u_i) \rangle$. In this paper, we assume that every strategic game has a payoff function, because the more general case is irrelevant for the subject studied.*

2.1.2 Mixed and pure strategies

A *pure strategy* provides a complete definition of how a player will play a game. In particular, it determines the action a player will make for any situation he or she could face. Mathematically, an element $a = (a_1, ..., a_n) \in A$ is called *pure strategy profile*. The components a_i of a contain an action for each player at any stage of the game.

A *mixed strategy* is an assignment of a probability to each action that is available to the player. This allows for a player to randomly select a pure strategy. Since probabilities are continuous, there are infinitely many mixed strategies available to a player, even if their strategy set is finite.

Of course, one can regard a pure strategy as a degenerate case of a mixed strategy, in which that particular pure strategy is selected with probability 1 and every other strategy with probability 0.

Let in a *mixed strategy*, $\alpha_i(a_j)$ (with $a_j \in A_i$) denote the probability that player i choose action j, thus: $\alpha_i(a_j) = \mathbb{P}[A_i = a_j]$. A *mixed strategy profile*

$$\alpha = (\alpha_i)_{i \in N}$$

specifies the probabilities for all players for all their possible actions. The probability of obtaining a specific pure strategy profile $a = (a_1, ..., a_n)$ is:

$$\mathbb{P}[\alpha = A] = \prod_{i \in N} \alpha_i(a_i)$$

We can now define the *expected payoff* for player i under a mixed strategy profile α:

$$U_i(\alpha) = \sum_{a \in A} \left(\prod_{j \in N} \alpha_j(a_j) \right) u_i(a)$$

Or, by using the properties of the (discrete) expected value (we iterate over the pure strategy profiles $a \in A$)[1]:

$$U_i(\alpha) = \mathbb{E}[u_i(a)]$$

The set of all mixed strategy profiles in a strategic game for a specific player i is denoted as $\Delta(A_i)$. Note that $(\alpha_i(a_1), ..., \alpha_i(a_k), ...)$ with the a_j's the pure actions of player i (thus $a_j \in A_i$) are the (vector)elements in $\Delta(A_i)$. The set of all mixed strategy profiles in a strategic game is denoted as $\Delta(A)$ and is defined as: $\Delta(A) = \Delta(A_1) \times ... \times \Delta(A_n)$.

[1] This notation is a little bit ambiguous, because in advanced game theory, also the payoff function may have a distribution. The reader must keep in mind that in this paper, payoff functions do not have a distribution.

2.1.3 Nash Equilibra

One of the most fundamental concepts in game theory is that of Nash equilibrum. This notion captures a steady state of the play of a strategic game in which no player can improve his cost by unilaterally changing his strategy. Of course, we distinguish pure and mixed Nash Equilibra.

Definition 2.3. *(Pure Nash Equilibrum) [THAR2011] A pure strategy profile $a^* \in A$ is a **pure Nash Equilibrum** if for each player $i \in N$:*

$$u_i(a^*_{-i}, a^*_i)) \geq u_i(a^*_{-i}, a_i) \quad \forall a_i \in A_i$$

Definition 2.4. *(Mixed Nash Equilibrum)[SUR2004] A mixed strategy profile α^* is a **mixed Nash Equilibrum** if for each player $i \in N$:*

$$U_i(\alpha^*_{-i}, \alpha^*_i)) \geq U_i(\alpha^*_{-i}, \alpha_i) \quad \forall \alpha_i$$

Remark 2.5. *The notation (a^*_{-i}, a_i) for a pure strategy profile a^* is a slight abuse of notation that is quite common in Game Theory, meaning that $a^*_i \in A_i$ and $Sa_{-i} \in A_1 \times ... \times A_{i-1} \times A_{i+1} ... \times A_n$. The same holds for the mixed strategy profiles. It's important to realize that players have no exact order in a strategy profile, so they can always be re-ordered.*

Although there is not enough space to give a deep explanation of the concept of a Nash equilibrum, it's important to know that a Nash equilibrum is not necessary an optimal solution of a game. It's only a profile in which no player will benefit from changing his strategy on it's own.

2.1.4 Theorem of Nash

Now we have defined the concept of Nash Equilibrum, we can look at one of the most fundamental theorems in Game Theory: *the Theorem of Nash.* This theorem states that every finite strategic game has at least one mixed Nash Equilibrum.

Nash proofed his theorem in 1950 by using the Brouwer fixed point theorem. Later on, a lot easier version of the proof is found by using the Kakutani's fixed point theorem. However, the (one dimensional version of) the Brouwer fixed point theorem is much more familiar because it's proved in almost every intermediate Analysis course. Therefore, we will give the original proof of Nash.

Definition 2.6. *(Fixed point)[MEU2011] Let X be a set and $f : X \to X$ a function. A point $x \in X$ is called a **fixed point** of f if $f(x) = x$.*

Property 2.7. *(One dimensional version Brouwer fixed point theorem) [MEU2011] Every continuous function*

$$f : [a, b] \to [a, b]$$

has a fixed point.

Proof. As the codomain of f is $[a, b]$ it follows that the image of f is a subset of $[a, b]$. Thus $f(a) \geq a$ and $f(b) \leq b$. Consider the function

$$g : [a, b] \to \mathbb{R} : x \mapsto f(x) - x.$$

Then is g also continuous and $g(a) \geq 0$ and $g(b) \leq 0$. By the theorem of Bolzano (see [MEU2011]): $\exists c \in [a, b] : g(c) = 0$, but this means that $f(c) = c$. □

The previous result is a very easy case of the Brouwer fixed point theorem. Proving the general theorem (see below) is very hard and falls behind the scope of this paper.

Lemma 2.8. *(Brouwer fixed point theorem) Let X be a non-empty, compact and convex set. If $f : X \to X$ is continuous, then there must exist $x \in X$ such that $f(x) = x$.*

Theorem 2.9. *(Theorem of Nash) Every finite strategic game has a mixed Nash equilibrum.*

Proof. For every player i, let the set of actions A_i is $\{a_1, ..., a_m\}$. For $1 \leq i \leq n$. Let α be a mixed strategy profile of the the game and define $g_{ij}(\alpha)$ to be the gain for player i form switching to his (pure) action a_j,:

$$g_{ij}(\alpha) = \max\{U_i(\alpha_{-i}, a_j) - U_i(\alpha), 0\}$$

We can now define a map between mixed strategies of player i, $y : \Delta(A_i) \to \Delta(A_i)$ by

$$y_{ij}(\alpha) = \frac{\alpha_i(a_j) + g_{ij}(\alpha)}{1 + \sum_{j=1}^{m} g_{ij}(\alpha)}$$

We now make two observations about this mapping:

- For every player i and his action a_j, the mapping $g_{ij}(\alpha)$ is continuous. This is due to the fact that $U_i(\alpha)$ is obviously continuous (it consist of the sum of products between a probability function and the continuous u_i), making $g_{ij}(\alpha)$ and consequently $y_{ij}(\alpha)$ continuous.
- For every player i, the vector $(y_{ij}(\alpha))_{j=1}^{m}$ is a distribution. This is due to the fact that the denominator of $y_{ij}(\alpha)$ is a normalization constant for any given i.
- Remember that $\Delta(A_i)$ has (vector)elements of the form $(\alpha_i(a_1), ..., \alpha_i(a_k), ...)$. Because the given strategic game is finite, we can identify $\Delta(A_i)$ with the set of vectors $(\alpha_i(a_1), ..., \alpha_i(a_k))$, for which $\alpha_i(a_j) \geq 0 \ \ \forall j$ and $\sum_{j=1}^{k} \alpha_i(a_j) = 1$. We now proof that the set $\Delta(A_i)$ satisfies the conditions for the Brouwer fixed point theorem:
  - The set $\Delta(A_i)$ is *non-empty* by definition of a strategic game.
  - To proof that the set $\Delta(A_i)$ is *convex*, take $\vec{x} = (\alpha_i^x(a_1), ..., \alpha_i^x(a_k))$ and $\vec{y} = (\alpha_i^y(a_1), ..., \alpha_i^y(a_k))$ then $\vec{z} = \theta\vec{x} + (1 - \theta)\vec{y}$ for some $\theta \in [0, 1]$ is in $\Delta(A_i)$ because $\vec{z}$ is also a mixed strategy for player i (the sum of the components of $\vec{z}$ is 1).
  - The *compactness* in $\mathbb{R}^k$ can be shown by proving that the set is closed and bounded. The set is bounded because $0 \leq \alpha_i(a_j) \leq 1$. To proof closeness in $\mathbb{R}^k$, we'll proof that the limit of every convergent sequence in $\Delta(A_i)$ is an element of $\Delta(A_i)$. Consider a convergent sequence in $\Delta(A_i) : ((\alpha_i^n(a_1), ..., \alpha_i^n(a_k))_n \to (\alpha_i^*(a_1), ..., \alpha_i^*(a_k))$. Remember that a convergent sequence of vectors converges componentwise and that the addition is a continuous function, so:

$$\begin{aligned} \sum_{j=1}^{k} \alpha_i^*(a_j) &= \sum_{j=1}^{k} \lim_{n\to\infty} \alpha_i^n(a_j) \\ &= \lim_{n\to\infty} \sum_{j=1}^{k} \alpha_i^n(a_j) \\ &= \lim_{n\to\infty} 1 \\ &= 1 \end{aligned}$$

this means that $(\alpha_i^*(a_1), ..., \alpha_i^*(a_k))$ is also a mixed strategy for player i, but by definition of $\Delta(A_i)$, this limit belongs to $\Delta(A_i)$.

Therefore y fulfills the conditions of Brouwer's Fixed Point Theorem. Using the theorem, we conclude that there is a fixed point $\alpha*$ for y. This point satisfies

$$\alpha_i^*(a_j) = \frac{\alpha_i^*(a_j) + g_{ij}(\alpha^*)}{1 + \sum_{j=1}^{m} g_{ij}(\alpha^*)}$$

Notice that this is possible only in one of two cases. Either $g_{ij}(\alpha) = 0$ for every i and j, in which case we have an equilibrium (since no one can profit from switching their strategy). If this is not the case, then there is a player i such that $g_{ij}(\alpha) > 0$. This would imply,

$$\alpha_i^*(a_j) \left(1 + \sum_{j=1}^{m} g_{ij}(\alpha^*)\right) = \alpha_i^*(a_j) + g_{ij}(\alpha^*)$$

or

$$\alpha_i^*(a_j) \left(\sum_{j=1}^{m} g_{ij}(\sigma)\right) = g_{ij}(\alpha^*)$$

.

This means that $g_{ij}(\alpha^*) = 0$ if and only if $\alpha_i^*(a_j) = 0$, and therefore, $\alpha_i^*(a_j) > 0 \implies g_{ij}(\alpha*) > 0$. However, this is impossible by the definition of $g_{ij}(\alpha)$. Recall that $U_i(\alpha)$ gives the expected payoff for a player under a mixed strategy α. Therefore, it cannot be that player i can profit from 'every' pure action in the support of α_i (with respect to $U_i(\alpha)$).

We are therefore left with the former possibility, i.e. $g_{ij}(\alpha) = 0$ for all i and j, implying a mixed Nash Equilibrium. □

2.2 Congestion games

2.2.1 Definition

Strategic games contains a wide range of games. We will now look at *congestion games*: a specific kind of strategic games for which we can prove the existence of a pure Nash equilibrum. We will use this result for defining in the next section *load balancing games*, a subclass of congestion games, this game is the one we'll need to handle the subject of this paper. Because we only need the theorem of the existence of a pure Nash equilibrum for congestion games, we will not take mixed strategies in consideration in this section.

Definition 2.10. *[YMAN2003, ICAR2008] A* ***congestion model*** $(N, M, (A_i)_{i\in N}, (c_j)_{j\in M})$ *is defined as follows:*

- *a finite set N of players. Each player i has a weight (or demand) $w_i \in \mathbb{N}$,*
- *a finite set M 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 $j \in M, c_j$ is a cost function $\mathbb{N} \to \mathbb{R}$, $c_j(k)$ denotes the cost related to the use of facility j under a certain load k;*

Definition 2.11. *[JOARU1994, YMAN2003] A* ***congestion game*** *associated with a congestion model is a strategic game with:*

- *a finite set N of players,*
- *for each player $i \in N$, there is a nonempty set of actions (strategies) A_i,*
- *the preference relation $\succeq_i$ for each player i is defined by a payoff function $u_i : A \to \mathbb{R}$. For any $a \in A$ and for any $j \in M$, let $\ell_j(a)$ be* the expected load on facility j*, assuming a is the current pure strategy profile, so $\ell_j(a) = \sum_{\substack{i\in[n]\\ j\in a_i}} w_i$. Then the payoff function for player i becomes: $u_i(a) = \sum_{j\in a_i} c_j(\ell_j(a))$.*

Remark 2.12. *Congestion models aren't always defined with the notion of weights of players (especially not in more economic game theory books). In those definitions, players have an equal weight. Those definitions match with ours if you give all players weight 1. Note that only the function $\ell_j(a)$ becomes much easier: it will just return the number of players using facility j under a pure strategy profile a. From a computer scientific point of view, such definitions are not sufficient because players (*tasks*) don't have the same weight. Watching the Eurovision Song Contest through a live stream is for example a much heavier task then sending an e-mail.*

Remark 2.13. *In order to preserve the generality in the definition of congestion games, note that we only stated that c_j are cost functions, without the need to explicitly define them. This is sufficient in this stage of the paper, however, the subject studied will require explicit definitions for c_j. These are given in section 3.1.*

2.2.2 The existence of a pure Nash equilbrum

Rosenthal proved in 1973 that every congestion game has a pure Nash Equilibrum. The proof of this statement use the notion of a potential function. We will first define a potential function, give a concrete potential function for congestion games and proof that it holds the right properties. With this result we can then finally proof the existence of a pure Nash Equilibrum in congestion games.

Definition 2.14. *Consider a function $\Phi : A \to \mathbb{R}$ defined on the space of pure stategy profiles of a (strategic) game G. If player i switches unilaterally from a_i to a_i^*, taking us from profile a to profile $\vec{a^*} = (a_i^*, \vec{a_{-i}})$ and the following property holds:*

$$\Phi(a) - \Phi(\vec{a^*}) = u_i(a) - u_i(\vec{a^*})$$

then the function Φ is called a ***potential function****.*

Lemma 2.15. *The function*

$$\begin{aligned} \Phi : A &\to \mathbb{R} \\ a &\mapsto \sum_{j=1}^{m} \sum_{k=1}^{\ell_j(a)} c_j(k) \end{aligned}$$

is a potential function for congestion games.

Proof. (Rosenthal 1973)[ROSENTHAL73, ETE2007] In remark 2.5 we already mentioned that players can be re-ordered. In particular, re-index player i as player n and vice versa.

Then, for $i' \in \{1, ..., n\}$, define:

$$\ell_j^{i'}(a) = \sum_{\substack{l \in [1,...,i'] \\ j \in a_l}} w_l$$

Now, by using the commutativity, you can exchange the order of summation and become:

$$\Phi(a) = \sum_{i=1}^{n} \sum_{j \in a_i} c_j(\ell_j^i(a))$$

Denote that $\ell_j^n(a) = \ell_j(a)$, thus:

$$\sum_{j \in a_i} c_j(\ell_j^n(a)) = \sum_{j \in a_i} c_j(\ell_j(a)) = u_n(a)$$

So, if player n switches from strategy a_n to a_n^*, taking us from profile a to profile $\vec{a^*} = (a_n^*, a_{-n}^{\rightarrow})$ then:

$$\Phi(a) - \Phi(\vec{a^*}) = \left(\sum_{j \in a_1} c_j(\ell_j^1(a)) + ... + u_n(a) \right) - \left(\sum_{j \in a_1} c_j(\ell_j^1(\vec{a^*})) + ... + u_n(\vec{a^*}) \right)$$

By definition of ℓ_j^i, this becomes (the n'th player doesn't count for ℓ_j^i with $i < n$):

$$\begin{aligned} &= \left(\sum_{j \in a_1} c_j(\ell_j^1(a)) + ... + u_n(a) \right) - \left(\sum_{j \in a_1} c_j(\ell_j^1(a)) + ... + u_n(\vec{a^*}) \right) \\ &= u_n(a) - u_n(\vec{a^*}) \end{aligned}$$

We switched player i with player n, so this holds for every player i.

□

Theorem 2.16. *(Rosenthal 1973)[ROSENTHAL73] Every congestion game has a pure Nash equilibrum.*

Proof. Start from a random strategy vector a, and let repeatedly one player reduce it's costs. That means, that at each step Φ is reduced identically. Since Φ can accept a finite amount of values (A is finite because it's the finite cartesian product of sets A_i ($i \in \{1, ..., n\}$), and every A_i is finite because it's by definition a subset of the finite set of facilities M), this procedure will stop and reach a local minimum. At this point, no player can achieve any improvements, and we reach a pure *Nash Equilibrum.* □

2.3 Load balancing games

2.3.1 Definition

Now we have introduced congestion games and proved that they always have a pure Nash Equilibrum, we take a closer look at a more specific kind of congestion games. Looking at the definition of a congestion game, it's immediately clear that this model is too general for giving a deeper understanding of the subject studied. Their is indeed a little problem with the actions (strategies) that a player can undertake: in a general congestion game, under a pure strategy profile a the strategy of player i, a_i, consist of multiple facilities j. That means that the weight of a player (*task*) counts for every function c_j where $j \in a_i$. Of course, this is not realistic: a single task (*player*) is not executed multiple times on different facilities. Therefore load balancing games are congestion games where the possible (pure) actions of players are singletons. So each $a_i = \{j\}$ $(i \in N, j \in M)$ in a pure strategy profile $a \in A$. So, for every player i, $A_i \subset M$. In load balancing terminology, we use the terms *machines* and *tasks* instead of the terms facility and player.

Definition 2.17. *[ICAR2008] A* ***load balancing game*** *is congestion game based on a congestion model with:*

- *a finite set N of tasks,*
- *for each player $i \in N$, there is a nonempty set of machines A_i with $A_i \subset M$. The elements of A_i are the possible machines on which task i can be executed.*
- *the preference relation $\succeq_i$ for each client i is defined by a payoff function $u_i : A \to \mathbb{R}$. For any $a \in A$ and for any $j \in M$, let $\ell_j(a)$ be* the expected load on machine j*, assuming a is the current pure strategy profile ($\ell_j(a) = \sum_{\substack{i \in [n] \\ j = a_i}} w_i$) . Then the payoff function for task i becomes: $u_i(a) = c_{a_i}(\ell_{a_i}(a))$.*

3 Linear cost functions in Load Balancing Games

Now we have defined *load balancing games*, we can take a closer look at the subject studied. As already mentioned in remark 2.13, the cost functions in the previous section aren't explicitly defined. In this section we look at the situation where the cost functions c_j for each machine j are linear (thus of the form: $c_j(x) = \lambda_j x + \mu_j$ with λ_j, μ_j non-negative constants) and every server has a certain *speed* $s_j \in N$.

3.1 Payoffs and makespan

3.1.1 Pure strategies

In the most easy case, not only the players have a certain *weight* $w_i \in N$, also the servers have certain *speed* $s_j \in N$. Intuitively, such s_j is the maximum amount of weight a server can handle. The cost functions c_j can then easily be defined as:

$$c_j(k) = \frac{k}{s_j}$$

The ***payoff functions*** become then:

$$u_i(a) = c_{a_i}(\ell_{a_i}(a)) = \frac{\ell_{a_i}(a)}{s_{a_i}}$$

The ***social costs*** of a pure strategy is denoted as $\text{cost}(a)$ and is defined to be the ***makespan***, i.e.

$$\text{cost}(a) = \max_{j\in[m]} c_j(\ell_j(a)) = \max_{j\in[m]} \frac{\ell_j(a)}{s_j}$$

3.1.2 Mixed strategies

Of course, players may use mixed strategies. Let x_i^j be a random variable for a mixed strategy profile α that takes the value 1 if $a_i = j$. This means for pure strategy profiles:

$$\begin{aligned}
\mathbb{E}[x_i^j(a)] &= \sum_{a\in A}\left(\prod_{j\in N}\alpha_j(a_j)\right)x_i^j(a)\\
&= \sum_{a\in A}\left(\prod_{j\in N}\alpha_j(a_j)\right)\delta_{a_i,j}\\
&= \sum_{a\in A}\left(\mathbb{P}[\alpha = A]\right)\delta_{a_i,j}\\
&= \mathbb{P}[A_i = j]\\
&= \alpha_i(j)
\end{aligned}$$

Now we can also define ℓ_j on mixed strategies: $\ell_j : \Delta(A) \to \mathbb{R}$, we will define $\ell_j(\alpha)$ as the **expected load**:

$$\begin{aligned}
\ell_j(\alpha) &= \mathbb{E}[\ell_j(a)]\\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\ell_j(a)\\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\sum_{\substack{i\in[n]\\ j=a_i}} w_i\\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\sum_{i\in[n]} w_i\delta_{a_i,j}\\
&= \sum_{a\in A}\left(\mathbb{P}[\alpha = A]\right)\sum_{i\in[n]} w_i\delta_{a_i,j}\\
&= \sum_{i\in[n]} w_i\mathbb{P}[A_i = j]\\
&= \sum_{i\in[n]} w_i\alpha_i(j)
\end{aligned}$$

$$\ell_j(\alpha)\mathbb{E}[\ell_j(\alpha)] = \mathbb{E}\left[\sum_{i\in[n]} w_i x_i^j(\alpha)\right] = \sum_{i\in[n]} w_i \mathbb{E}[x_i^j(\alpha)] = \sum_{i\in[n]} w_i \alpha_i(j)$$

The social cost of a mixed strategy profile is defined as the expected makespan:

$$\text{cost}(\alpha) = \mathbb{E}[\text{cost}(a)] = \mathbb{E}\left[\max_{j\in[m]} \frac{\ell_j(a)}{s_j}\right]$$

For mixed strategies, the expected payoff for player i, denoted by U_i is defined as:

$$\begin{aligned}
U_i(\alpha) &= \mathbb{E}[u_i(a)] \\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\frac{\ell_{a_i}(a)}{s_{a_i}} \\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\left(\sum_{j\in[m]}\frac{\ell_j(a)}{s_j}\delta_{a_i,j}\right) \\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\left(\sum_{j\in[m]}\frac{\sum_{l\in[n]}\underset{j=a_l}{w_i}}{s_j}\delta_{a_i,j}\right) \\
&= \sum_{a\in A}\left(\prod_{k\in N}\alpha_k(a_k)\right)\left(\sum_{j\in[m]}\frac{\sum_{l\in[n]} w_i\delta a_l, j}{s_j}\delta_{a_i,j}\right) \\
&= \sum_{a\in A}(\mathbb{P}[\alpha = A])\left(\frac{\sum_{l\in[n]} w_i\delta a_l, 1}{s_1}\delta_{a_i,1} + ... + \frac{\sum_{l\in[n]} w_i\delta a_l, m}{s_m}\delta_{a_i,m}\right) \\
&= \sum_{a\in A}(\mathbb{P}[\alpha = A])\left(\frac{\sum_{l\in[n]} w_i\delta a_l, 1}{s_1}\delta_{a_i,1}\right) + ... + \sum_{a\in A}(\mathbb{P}[\alpha = A])\left(\frac{\sum_{l\in[n]} w_i\delta a_l, m}{s_j}\delta_{a_i,m}\right)
\end{aligned}$$

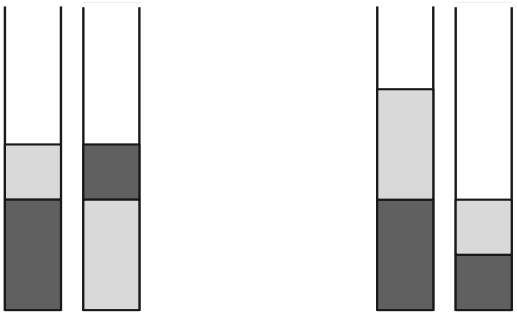
From point of view of client i, the ***expected cost on machine*** j under a mixed strategy profile α, denoted by U_i^j, is:

$$\begin{aligned}
U_i^j(\alpha) &= \mathbb{E}[U_i(\alpha)|A_i = j] \\
&= \frac{w_i + \sum_{k\neq i} w_k\alpha_k(j)}{s_j} \\
&= \frac{\ell_j(\alpha) + (1-\alpha_i(j))w_i}{s_j}
\end{aligned}$$

3.2 An example of a Load Balancing Game with linear costs

After more than 10 pages theory, we give an easy example in which all introduced concepts are contained.

Suppose that there are two identical machines both of which have speed 1 and four tasks, two *small tasks* of weight 1 and two *large tasks* of weight 2. An optimal assignment would map a small and a large task to each of the machines so that the load on both machines is 3. This assignment is illustrated in figure 3.1a.



(a) Illustration of the optimal assignment

(b) Illustration of a pure Nash equilibrum

Figure 3.1: (a) Illustration of the optimal assignment of an instance of the load balancing game with two large tasks of size 2 and two small tasks of size 1. The social cost of this assignment is 3. (b) Illustration of a pure Nash equilibrium for the same instance. The social cost of this assignment is 4, which is the maximum among all pure Nash equilibria for this instance.

Now consider a pure strategy profile a that maps the two large tasks to the first machines and the two small tasks to the second machine as illustrated in figure 3.1b. This way, the first machine has a load of 4 and the second machine has a load of 2. Obviously, a small task cannot improve its cost by moving from the second to the first machine. A large task cannot improve its cost by moving from the first tot the second machine either as its cost would remain 4 if it does. Thus this pure strategy profile a constitutes a *pure Nash equilibrium* with $\text{cost}(a) = 4$. This is beautiful example to show the already stated statement that Nash equilibra aren't mostly not the optimal situations. Observe that all assigments that yield a larger makespan than 4 cannot be a Nash equilibrium as, in this case, one of the machines has a load of at least 5 and the other has a load of at most 1 so that moving any task from the former to the latter would decrease the cost of this task. Thus, for this instance of the load balancing game, the social cost of the worst pure Nash Equilibrium is 4.

Clearly, the worst mixed equilibrium cannot be better than the worst pure equilibrium as the set of mixed equilibria is a superset of the set of pure equilibria, but can it really be worse? Suppose that each task is assigned to each of the machines with probability $\frac{1}{2}$. This corresponds to a mixed strategy profile α with the α_i's being constant functions,

$$\alpha = \Big(\underbrace{\frac{1}{2}}_{\alpha_1}, \underbrace{\frac{1}{2}}_{\alpha_2}, \underbrace{\frac{1}{2}}_{\alpha_3}, \underbrace{\frac{1}{2}}_{\alpha_4} \Big)$$

. The expected load on machine j is thus:

$$\ell_j(\alpha) = \sum_{1 \leq i \leq 4} w_i \alpha_i(j) = \frac{1}{2} + \frac{1}{2} + 1 + 1 = 3$$

It's important to notice that te expected cost of a task on a machine is larger than the expected load of the machine, unless the task is assigned with probability 1 to this machine. For example, if we assume that task 1 is a small task, then:

$$U_1^j = \frac{\ell_j(\alpha) + (1 - \alpha_1(j))w_1}{=} 3 + \frac{1}{2} = 3.5$$

and, if task 3 is a large task, then:

$$U_3^j = \frac{\ell_j(\alpha) + (1 - \alpha_3(j))w_3}{=} 3 + \frac{1}{2}.2 = 4$$

For symmetry reasons, the expected cost of each task under the considered mixed strategy profile α is the same on both machines so that α is a *mixed Nash equilibrium*. The social cost of this equilibrium, cost(α), is defined to be the expected makespan, $\mathbb{E}[cost(a)]$, of the random pure assignment A induced by α. The makespan, cost(a), is in fact a random variable. This variable can possibly take one of the four values $3, 4, 5$ or 6. There are $2^4 = 16$ different assignments of four tasks to two machines. The number of assigments that yield a makespan of 3 is 4, 4 is 6, 5 is 4 and 6 is 2. Consequently, the social cost of the mixed Nash equilibrium is:

$$cost(\alpha) = \mathbb{E}[\text{cost}(a)] = \frac{3.4 + 4.6 + 5.4 + 6.2}{16} = 4.25.$$

Thus mixed equilibria can, in fact, be worse than the worst pure equilibrium.

3.3 The Price of Anarchy

3.3.1 Definition

Since players ('tasks') act selfishly, load balancing games may reach assignments that do not minimize the makespan. We now introduce the notion of the ***price of the anarchy*** to quantify the degradation of the overall system performance.

Definition 3.1. *[SUR2004] Let Nash(G) be the set of all (mixed) strategy profiles being a Nash equilibrium of a (load balancing) game G and let α_{opt} be the pure strategy profile being the social optimum. Then the* **price of anarchy** *is defined as:*

$$PoA(G) = \max_{\alpha \in Nash(G)} \frac{cost(\alpha)}{cost(a_{opt})}$$

The motivation behind studying the price of anarchy is to quantify the increase of the social cost due to selfish behavior. With this motivation in mind, does it make more sense to consider pure or mixed Nash equilibria? Firstly, every load balancing game has a pure and mixed Nash equilibrium because it's a congestion and thus a strategy game. In reality, the answer depends: if one wants to study a distributed system in which tasks repeatedly perform improvement steps until they reach a Nash equilibrium, the situation of proof 2.16 arises. However, there might be other means by which task come to a Nash equilibrium. Moreover, the upper bounds about the price of anarchy for mixed equilibria are more robust than upper bounds for pure equilibira as mixed equilibria are more general. We'll study both equilibria in two situations: the case in which all machines have an equal speed and the case they don't. It will be clear that if the server speeds are relatively bounded an the number of task is large compared to the number of machines, then *every Nash assignment approaches the social optimum*!

3.3.2 Bachmann-Landau notations

For understanding the four cases for which we'll study the price of the anarchy, it's important to know the *Bachmann-Landau notations*. These notation are used to describe the limiting behavior of a function in terms of simpler functions. These notations are used a lot in computer science to classify algorithms by how they respond to change in input size.

Definition 3.2. ***(Big Oh)****[MEU2011]*
Big Omega is the set of all functions f that are bounded above by g asymptotically (up to constant factor).

$$O(g(n)) = \{f | \exists c, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

Definition 3.3. ***(Big Omega)****[MEU2011]*
Big Omega is the set of all functions f that are bounded below by g asymptotically (up to constant factor).

$$\Omega(g(n)) = \{f | \exists c, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

Definition 3.4. ***(Big Theta)****[MEU2011]*
Big Theta is the set of all functions f that are bounded above and below by g asymptotically (up to constant factors).

$$\Theta(g(n)) = \{f | \exists c_1, c_2 > 0, n_0 \geq 0 : \forall n \geq n_0 : 0 \leq c_1.g(n) \leq f(n) \leq c_2.g(n)\}$$

3.3.3 Pure Nash Equilibria

Identical Machines

Theorem 3.5. *Let G be a load balancing game with n tasks of weight $w_1, ..., w_n$ and m identical machines. Under a pure Nash equilibrium $a \in A$ it holds:*

$$\frac{cost(a)}{cost(a_{opt}} = 2 - \frac{2}{m+1}$$

Proof. Let $j*$ be a machine with the highest load under profile a, and let $i*$ be a task of smallest weight assigned to this machine. WLOG, there are at least two tasks assigned to machine $j*$ as, otherwise, $\text{cost}(a) = \text{cost}(a_{\text{opt}}$ so that the upper bound given in the theorem follows trivially. Thus $w_{i^*} \leq \frac{1}{2}\text{cost}(a)$.

Suppose there is a machine $j \in [m] \backslash j^*$ with load less than $l_{j^*} - w_{i^*}$. Then moving the task i^* from j^* to j would decrease the cost for this task. Hence, as a is a Nash equilibrium it holds:

$$\ell_j \geq \ell_{j^*} - w_{i^*} \geq \text{cost}(a) - \frac{1}{2}\text{cost}(a) = \frac{1}{2}\text{cost}(a).$$

Now observe that the cost of an optimal assignment cannot be smaller than the average load over all machines, so:

$$\begin{aligned} \text{cost}(a_{\text{opt}}) &\geq \frac{\sum_{i \in [n]} w_i}{m} \\ &= \frac{\sum_{j \in [m]} \ell_j}{m} \\ &\geq \frac{cost(a) + \frac{1}{2}cost(a)(m-1)}{m} \\ &= \frac{(m+1)\text{cost}(a)}{2m} \end{aligned}$$

□

Uniformly Related Machines

3.3.4 Mixed Nash Equilibria

Identical Machines

Uniformly Related Machines

3.3.5 Summary

References

[BVOCK2007] B. Vöcking, *Selfish Load Balancing*, Chapter 20 in Algorithmic Game Theory, Cambridge University Press, December 2007.

[JOARU1994] d J. Osborne and A. Rubinstein, *A course in Game Theory*, The MIT Press, 1994.

[MANN2008] S. Mannor, *Advancded Topics in Systems, Learning and Control, Lecture 3: Lecture 3: Mixed Actions, Nash and Correlated Equilibria*, Technicon, November 2008.

[COL2011] E. Colebunders, *Analyse II*, Vrije Universiteit Brussel, 2011.

[CW2007] C. Witteveen, Intreerede: De Prijs van de Onafhankelijkheid, TU Delft 2007.

[YMAN2003] Y. Mansour, *Lecture 6: Congestion and potential games*, Computational Learning Theory, University of Tel Aviv, 2003.

[JMAR] Jason R. Marden, *Lecture 12: Game Theory Course*, University of Colorado.

[THAR2011] T. Harks, M. Klimm, R. H. MÃűhring, *Characterizing the Existence of Potential Functions in Weighted Congestion Games*, February 2011

[ROSENTHAL73] R. W. Rosenthal, *A class of games possessing pure-strategy Nash equilibria*. International Journal of Game Theory, 2:65âĂŞ67, 1973

[ETE2007] K. Etessami, *Algorithmic Game Theory - Lecture 16 Best response dynamics and pure Nash Equilibria*, University of Edingburgh, 2007.

[ICAR2008] I. Caragiannis, C. Kaklamanis, P. Kanellopoulos, *Improving the Efficiency of Load Balancing Games through Taxes*, University of Patras, 2008.

[SUR2004] S. Suri, C. D. TÃşth, Y. Zhou. *Selfish Load Balancing and Atomic Congestion Games*, University of California, 2004.

[MEU2011] W. De Meuter. *Algoritmen en Datastructuren I*, Vrije Universiteit Brussel, 2011.