Non-cooperative scheduling of multiple bag-of-task applications

Arnaud Legrand, Corinne Touati

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Non-cooperative scheduling of multiple bag-of-task applications

Arnaud Legrand — Corinne Touati

N° 5819

January 2006

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--5819--FR+ENG

INRIA
RHÔNE-ALPES

Non-cooperative scheduling of multiple bag-of-task applications

Arnaud Legrand, Corinne Touati

Thème NUM — Systèmes numériques
Projet MESCAL

Rapport de recherche n° 5819 — January 2006 — 31 pages

Abstract: Multiple applications that execute concurrently on heterogeneous platforms compete for CPU and network resources. In this paper we analyze the behavior of K non-cooperative schedulers using the optimal strategy that maximize their efficiency. Meanwhile fairness is ensured at a system level ignoring applications characteristics. We limit our study to simple single-level master-worker platforms and the case where applications consist of a large number of independent tasks. The tasks of a given application all have the same computation and communication requirements, but these requirements can vary from one application to another. Therefore, each scheduler aims at maximizing its throughput. We give closed-form formula of the equilibrium reached by such a system and study its performances. We characterize the situations where this Nash equilibrium is Pareto-optimal and show that even though no catastrophic situation (Braess-like paradox) can occur, such an equilibrium can be arbitrarily bad for any classical performance measure.

Key-words: heterogeneous processors, master-slave tasking, steady-state, throughput, non-cooperative scheduling, Nash equilibrium, Braess-like paradox

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Ordonnancement non-coopératif d'applications régulières

Résumé : Lorsque plusieurs applications s'exécutent simultanément sur une plateforme de calcul hétérogène, elles entrent en compétition pour l'accès aux ressources de calcul et de communication. Dans ce rapport, nous analysons le comportement de K ordonnanceurs ne coopérant pas et utilisant la stratégie qui optimise leur performance. L'équité d'accès aux ressources est donc assurée au niveau du système sans tenir compte des spécificités des différentes applications. Nous limitons notre étude aux simples plates-formes maître-esclave arborescentes à un seul niveau et au cas où chaque application est constituée d'un très grand nombre de tâches indépendantes. Les tâches d'une même application ont toutes les mêmes besoins en terme de calcul et de communications mais ces besoins peuvent varier d'une application à l'autre. Dans un tel contexte, il est naturel que chaque application cherche à optimiser son propre débit. Nous donnons des formes closes de l'équilibre atteint par un tel système et nous étudions ses performances. Nous caractérisons les situations où cet équilibre de Nash est Pareto-optimal et montre que même si aucune situation catastrophique de type paradoxe de Braess ne peut se produire, cet équilibre est arbitrairement mauvais pour toutes les mesures de performances classiques.

Mots-clés : Ressources hétérogènes, maître/esclaves, débit, régime permanent, ordonnancement non-coopératif, équilibre de Nash, paradoxe de Braess

1 Introduction

The recent evolutions in computer networks technology, as well as their diversification, yield to a tremendous change in the use of these networks: applications and systems can now be designed at a much larger scale than before. Large-scale distributed platforms (Grid computing platforms, enterprise networks, peer-to-peer systems) result from the collaboration of many people. Thus, the scaling evolution we are facing is not only dealing with the amount of data and the number of computers but also with the number of users and the diversity of their behavior. Therefore computation and communication resources have to be *fairly* shared between users, otherwise users will leave the group and join another one. However, the variety of user profiles requires resource sharing to be ensured at a system level. We claim that even in a perfect system where each application competing on a given resource always receives the same share as the other ones and where no degradation of resource usage (e.g. packet loss or context switching overhead) occurs when a large number of applications use a given resource, non-cooperative usage of the system leads to important application performance degradation and resource wasting. In this context, we make the following contributions:

- We present a simple yet realistic situation where the *system-level* fairness fails to achieve a relevant *application-level* fairness. More precisely, we study the situation where multiple applications consisting of large numbers of independent identical tasks execute concurrently on heterogeneous platforms and compete for CPU and network resources. As the tasks of a given application all have the same computation and communication requirements (but these requirements can vary for different applications), each scheduler aims at maximizing its throughput. This framework had previously been studied in a cooperative centralized framework [BLCJF+06]. The resource sharing aspect was therefore not present and is extensively described in Section 2.

- We first characterize in Section 3.1 the optimal selfish strategy for each scheduler (i.e. the scheduling strategy that will maximize its own throughput in all circumstances and adapt to external usage of resources) and then propose in Section 3.2 equivalent representations of such non-cooperative schedulers competition.

- The particularities of these representations enable us to characterize the structure of the resulting Nash equilibrium as well as closed-form values of the throughput of each application (see Section 3.3 and 3.4).

- Using these closed-form formulas, we derive in Section 4 necessary and sufficient conditions on the system parameters (in term of bandwidth, CPU speed, ...) for the non-cooperative equilibrium to be Pareto-optimal. We also quickly study the degree of inefficiency of this equilibrium through a simple example.

- When studying properties of Nash equilibria, it is important to know whether paradoxical situations like the ones exhibited by Braess in his seminal work [Bra68] can

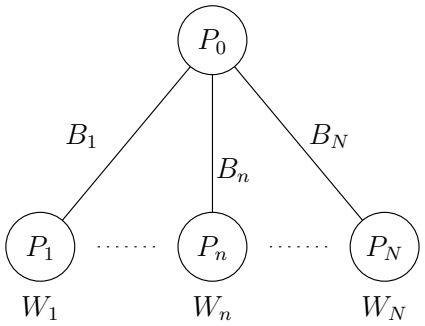
occur. In such situations, the addition of resource (a new machine, more bandwidth or more computing power in our framework) can result in a simultaneous degradation of the performance of *all* the users. Such situations only occur when the equilibrium is not Pareto-optimal, which may be the case in this framework. We investigate in Section 4.4 whether such situations can occur in our considered scenario and conclude with a negative answer to this question.

- Last, we show in Section 5, that even when the non-cooperative equilibrium is Pareto-optimal, the throughput of each application is far from being monotonous with a resource increase. This enables us to prove that this equilibrium can be arbitrarily bad for any of the classical performance measures (average throughput, maximal throughput, minimum throughput).

Section 6 concludes the paper with a discussion of extensions of this work and future directions of research.

2 Platform and Application Model

2.1 Platform Model



Our master-worker platform is made of $N+1$ processors $P_0, P_1, \ldots, P_N$. P_0 denotes the master processor, which doesn't perform any computation. Each processor P_n is characterized by its computing power W_n (in Mflop.s^{-1}) and the capacity of its connection with the master B_n (in Mb.s^{-1}). Last, we define the *communication-to-computation ratio* C_n of processor P_n as B_k/W_k. This model leads us to the following definition:

Definition 1. *We denote by* ***physical-system*** *a triplet* (N, B, W) *where* N *is the number of machines, and* B *and* W *the vectors of size* N *containing the link capacities and the computational powers of the machines.*

We assume that the platform performs an ideal fair sharing of resources among the various requests. More precisely, let us denote by $N_n^{(B)}(t)$ (resp. $N_n^{(W)}(t)$) the number of ongoing communication (resp. computation) from P_0 to P_n (resp. on P_n) at time t. Resources are shared "fairly" between the different users. Thus, the platform ensures that the amount of bandwidth received at time t by a communication from P_0 to P_n is

exactly $B_n/N_n^{(B)}(t)$. Likewise, the amount of processor power received at time t by a computation on P_n is exactly $W_n/N_n^{(W)}(t)$. Therefore, the time T needed to transfer a file of size b from P_0 to P_n starting at time t_0 is such that

$$\int_{t=t_0}^{T} \frac{B_n}{N_n^{(B)}(t)}.dt = b.$$

Likewise, the time T needed to perform a computation of size w on P_n starting at time t_0 is such that

$$\int_{t=t_0}^{T} \frac{W_n}{N_n^{(W)}(t)}.dt = w.$$

Last, we assume that communications to different processors do not interfere. This amounts to say that the network card of P_0 has a sufficiently large bandwidth not to be a bottleneck. This hypothesis makes sense when workers are spread over the Internet and are not too numerous.

2.2 Application Model

We consider K applications, A_k, $1 \leqslant k \leqslant K$. Each application is composed of a large set of independent, same-size tasks. We can think of each A_k as bag of tasks, and the tasks are files that require some processing. A task of application A_k is called a task of *type* k. SETI@home [SET], factoring large numbers [CDu+96], the Mersenne prime search [Pri], ClimatePrediction.NET [BOI], Einstein@Home [EIN], processing data of the Large Hadron Collider [LHC] are a few examples of such typical applications. We let w_k be the amount of computation (in floating point operations) required to process a task of type k. Similarly, b_k is the size (in Mb) of (the file associated to) a task of type k. We assume that the only communication required is outwards from the master, i.e. that the amount of data returned by the worker is negligible. This is a common hypothesis [BOBLC+04] as in steady-state, the output-file problem can be reduced to an equivalent problem with bigger input-files. Last, we define the *communication-to-computation ratio* c_k of tasks of type k as b_k/w_k. This model leads us to the following definition:

Definition 2. *We define a* ***user-system*** *a triplet* (K, b, w) *where* K *is the number of applications, and* b *and* w *the vectors of size* K *representing the size and the amount of computation associated to the different applications.*

2.3 Rules of the game

In the following our K applications run on our N processors and compete for the network and CPU access:

Definition 3. *A* ***system*** *is a sextuplet* (K, b, w, N, B, W)*, with* K*,*b*,*w*,*N*,*B*,*W *defined as for a user-system and a physical-system.*

We assume that each application is scheduled by its own scheduler. As each application comprises a very large number of independent tasks, trying to optimize the makespan is known to be vainly tedious [Dut04] especially when resource availability may vary over time. Maximizing the throughput of a single application is however known to be a much more relevant metric in our context [BLCJF+06, HP04]. Let us denote by $\alpha_{n,k}$ the average number of tasks of type k performed per unit of time on the processor P_n. In the rest of this document, we will denote by $\alpha_k = \sum_n \alpha_{n,k}$. The scheduler of each application thus aims at maximizing its own throughput, i.e. α_k. However, as they are sharing the same set of resources, we have the following general constraints[1]:

Computation $\forall n \in [\![0, N]\!] : \sum_{k=1}^{K} \alpha_{n,k}.w_k \leqslant W_n$

Communication $\forall n \in [\![1, N]\!] : \sum_{k=1}^{K} \alpha_{n,k}.b_k \leqslant B_n$

3 A non-cooperative game

Is this article, we assume that a process running on P_0 can communicate with as many processors as it wants. This model is called *multi-port* [BSP+99, BNGNS00] and is reasonable if the number of workers is relatively small and the process can make use of threads to handle communications. We first study the situation where only one application is scheduled on the platform. This will enable us to simply define the scheduling strategy that will be used by each players in the more general case where many applications are considered.

3.1 One application

When there is only one application, our problem reduces to the following linear program:

$$\begin{array}{l} \text{MAXIMIZE } \sum_n \alpha_{n,1}, \\ \text{UNDER THE CONSTRAINTS} \\ \left\{ \begin{array}{ll} \text{(1a)} & \forall n \in [\![0, N]\!] : \alpha_{n,1}.w_1 \leqslant W_n \\ \text{(1b)} & \forall n \in [\![1, N]\!] : \alpha_{n,1}.b_1 \leqslant B_n \\ \text{(1c)} & \forall n, \quad \alpha_{n,1} \geqslant 0 \end{array} \right. \end{array} \tag{1}$$

We can easily show that the optimal solution to this linear program is obtained by setting $\alpha_{n,1} = \min\left(\frac{W_n}{w}, \frac{B_n}{b}\right)$. In a practical setting, this amounts to say that the master process will saturate each of the workers by sending them as much tasks as possible. On a stable platform, W_n and B_n can easily be measured and the $\alpha_{n,1}$'s can thus easily be computed. On an unstable one this may be more tricky. However a simple acknowledgement mechanism enables the master process to ensure that it is not over-flooding the workers, while always converging to the optimal throughput.

In a multiple-applications context, each player (process) strives to optimize its own performance measure, considered here to be its throughput α_k. Hence, we will assume

[1] The notation $[\![a, b]\!]$ denotes the set of integers comprised between a and b, i.e. $[\![a, b]\!] = \mathbb{N} \cap [a, b]$

that each of them constantly floods the workers while ensuring that all the tasks they sends are performed (e.g. using an acknowledgement mechanism). This adaptive strategy automatically cope with other schedulers usage of resource and *selfishly* maximize the throughput of each application. We suppose a purely non-cooperative game where no scheduler decides to "ally" to any other. As the players constantly adapt to each others' actions, they may (or not) reach some equilibrium, known in game theory as *Nash equilibrium* [Nas50, Nas51]. In the remaining of this paper, we will denote by $\alpha_{n,k}^{(nc)}$ the rates achieved at such stable states.

3.2 Many applications: canonical representations

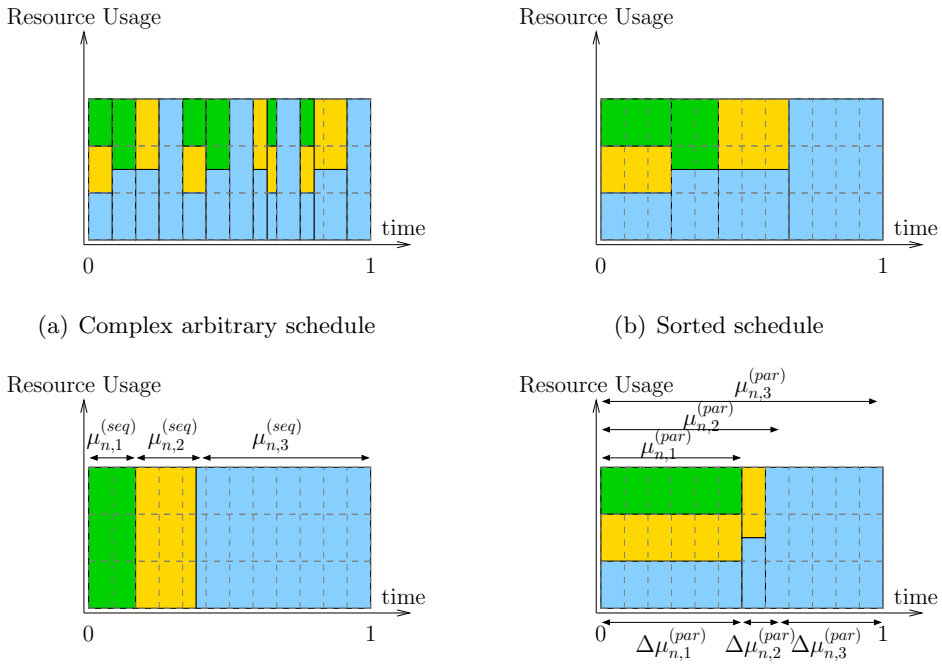
First, we can see that in the multi-port model, communication and computation resources are all independent. Let us consider in this section the use of resources on a arbitrary worker n. We only need to ensure that the number of tasks sent to P_n is equal to the number of tasks processed on P_n. In a steady state, actions of the players will interfere on each resource in (a priori) a non predictable order. The resource usage may be arbitrarily complex (see Figure 1(a)). However, for any given subset S of $[\![1, K]\!]$, we can define the fraction of time where all players of S (and only them) use a given resource. This enables us to reorganize the schedule into an equivalent representation (see Figure 1(b)) with at most 2^K time intervals (the number of possible choices for the subset S). In this representation, the fractions of time spent using a given resource (which can be a communication link or a processor) are perfectly equal to the ones in the original schedule. However such a representation is still too complex (2^K is a large value). Hence, we now explain how to build two more compact "equivalent" canonical representation (see Figure 1(c) and 1(d))

Sequential canonical representation The first compact form we define is called *sequential canonical representation* (see Figure 1(c)). If the schedulers were sending data one after the other on this link, the k^{th} scheduler would have to communicate during exactly $\mu_{n,k}^{(seq)} = \frac{\alpha_{n,k}^{(nc)} b_k}{B_n}$ of the time to send the same amount of data as in the original schedulers[2]. This value is called *sequential communication time ratio*. Similarly, we can define the *sequential computation time ratio* $\nu_{n,k}^{(seq)}$ as $\frac{\alpha_{n,k}^{(nc)} w_k}{W_n}$. We have the following relation between $\mu_{n,k}^{(seq)}$ and $\nu_{n,k}^{(seq)}$:

$$\mu_{n,k}^{(seq)} = \frac{c_k}{C_n} \nu_{n,k}^{(seq)} \tag{2}$$

We can therefore obtain a canonical schedule (see Figure 1(c)) with at most $K+1$ intervals whose respective sets of players are $\{1\}$, $\{2\}$, $\ldots$, $\{K\}$, $\emptyset$. This communication scheme is thus called sequential canonical representation and has the same $\alpha_{n,k}^{(nc)}$ values.

[2]Note that we have $\mu_{n,1}^{(seq)} \leqslant \mu_{n,2}^{(seq)} \leqslant \cdots \leqslant \mu_{n,K}^{(seq)}$.



(a) Complex arbitrary schedule

(b) Sorted schedule

(c) Sequential canonical representation: areas are preserved but using times are minimized

(d) Parallel canonical representation: areas are preserved but using times are maximized

Figure 1: Various schedule representations. Each application is associated to a color: Application 1 is green, application 2 is yellow and application 3 is blue. The area of each application is preserved throughout all transformations.

However, communication and computation times have all been decreased as each scheduler is now using the network link and the CPU exclusively. We will see later that this information loss does not matter for multi-port schedulers.

Parallel canonical representation The second compact form we define is called *parallel canonical representation* (see Figure 1(d)). In this scheme, resource usage is as conflicting as possible. Let us denote by $\mu_{n,k}^{(par)}$ (resp. $\nu_{n,k}^{(par)}$) the fraction of time spent by player k to communicate with P_n (resp. to compute on P_n) in such a configuration. $\mu_{n,k}^{(par)}$ is the *parallel communication time ratio* and $\nu_{n,k}^{(par)}$ is the *parallel computation time ratio*. Let us focus on the canonical representation of resources in the communication channel, the case of processor sharing being similar. To simplify the analysis, let us suppose that $\mu_{n,1}^{(par)} \leqslant \mu_{n,2}^{(par)} \leqslant \cdots \leqslant \mu_{n,K}^{(par)}$. Thus, communication times verify the

following set of equations (where $\Delta\mu_{n,k}^{(par)} = \mu_{n,k}^{(par)} - \mu_{n,k-1}^{(par)}$):

$$\begin{cases} \alpha_{n,1}^{(nc)} b_1 = \dfrac{B_n}{K}\Delta\mu_{n,1}^{(par)} \\ \alpha_{n,2}^{(nc)} b_2 = \dfrac{B_n}{K}\Delta\mu_{n,1}^{(par)} + \dfrac{B_n}{K-1}\Delta\mu_{n,2}^{(par)} \\ \quad\vdots \qquad\qquad \vdots \\ \alpha_{n,K}^{(nc)} b_K = \displaystyle\sum_{k=1}^{K} \frac{B_n}{K-k+1}\Delta\mu_{n,k}^{(par)} \end{cases} \tag{3}$$

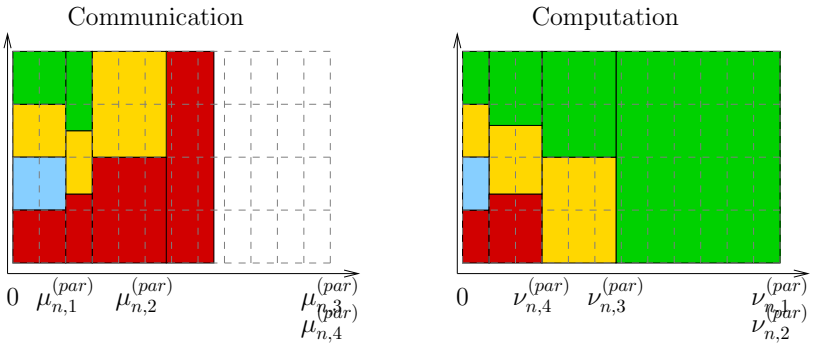
We can easily check that this system has a unique solution, that all the $\Delta\mu_{n,k}^{(par)}$ are non-negative and that their sum is no larger than 1. From these $\Delta\mu_{n,k}^{(par)}$'s we can therefore obtain a canonical schedule (see Figure 1(d)) with at most $K+1$ intervals whose respective player sets are respectively $\{1,\ldots,K\}$, $\{2,\ldots,K\}$, $\ldots$, $\{K\}$, $\emptyset$. This communication scheme is thus called parallel canonical representation and has the same $\alpha_{n,k}^{(nc)}$ values. However, communication times have all been increased as each scheduler is now interfering with as much other scheduler as possible.

Particularities of multi-port selfish schedulers The same reasonings can be applied to computation resources and therefore, for a given worker, both communication and computation resources can be put in canonical form (see Figure 2(a)). As we have seen in Section 3.1, the scheduling algorithm used by the players consist in constantly flooding workers. Hence it is impossible that both $\mu_{n,k}^{(par)}$ and $\nu_{n,k}^{(par)}$ are smaller than 1. A player is thus said to be either *communication-saturated* ($\mu_{n,k}^{(par)} = 1$) or *computation-saturated* ($\nu_{n,k}^{(par)} = 1$).

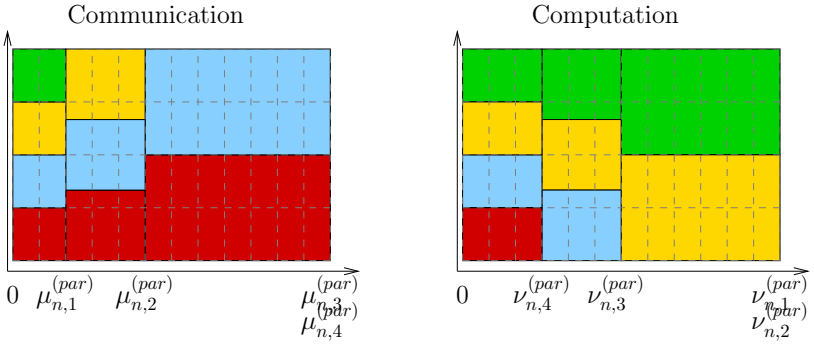
Proposition 1. *If there exists a communication-saturated application then* $\sum_k \mu_{n,k}^{(seq)} = 1$. *Similarly, if there exists a computation-saturated application then* $\sum_k \nu_{n,k}^{(seq)} = 1$.

As two computation-saturated players k_1 and k_2 receive the same amount of computation power and compute during the same amount of time, we have $\alpha_{n,k_1}^{(nc)} w_{k_1} = \alpha_{n,k_2}^{(nc)} w_{k_2}$. Therefore if $c_{k_1} \leqslant c_{k_2}$, we have $\alpha_{n,k_1}^{(nc)} b_{k_1} \leqslant \alpha_{n,k_2}^{(nc)} b_{k_2}$. The same reasoning holds for two communication-saturated players as well as for a mixture of both. As a consequence, in a multi-port setting, players should be first sorted according the c_k to build the canonical schedule. The very particular structure of this schedule (see Figure 2(b)) will enable us in the following section to give closed-form formula for the $\alpha_{n,k}^{(nc)}$. All these remarks can be summarized in the following proposition:

Proposition 2. *If there is an equilibrium, let us denote by* $\mathcal{B}_n$ *the set of communication-saturated applications and by* $\mathcal{W}_n$ *the set of computation-saturated applications. If* $c_1 \leqslant c_2 \leqslant \cdots \leqslant c_K$, *then there exists* $m \in [\![0, K]\!]$ *such that* $\mathcal{W}_n = [\![1, m]\!]$ *and* $\mathcal{B}_n = [\![m+1, K]\!]$. *We have:*



(a) Parallel canonical form of an arbitrary schedule



(b) Parallel canonical schedule for a given processor under the non-cooperative multi-port assumptions. Application 3 (blue) and 4 (red) are *communication saturated*: they receive the same amount of communication resource. Application 1 (green) and 2 (yellow) are *computation saturated*: they receive the same amount of computation resource.

Figure 2: Parallel canonical schedules

- *Parallel representation:*

$$\begin{cases} \textit{Communications:} & \mu_{n,1}^{(par)} \leqslant \cdots \leqslant \mu_{n,m}^{(par)} < \overbrace{\mu_{n,m+1}^{(par)} = \cdots = \mu_{n,K}^{(par)}}^{\mathcal{B}_n} = 1 \quad \textit{and} \\ \textit{Computations:} & 1 = \underbrace{\nu_{n,1}^{(par)} = \cdots = \nu_{n,m}^{(par)}}_{\mathcal{W}_n} > \nu_{n,m+1}^{(par)} \geqslant \cdots \geqslant \nu_{n,K}^{(par)} \end{cases}$$

- *Sequential representation:*

$$\begin{cases} \textit{Communications:} & \mu_{n,1}^{(seq)} \leqslant \cdots \leqslant \mu_{n,m}^{(seq)} < \overbrace{\mu_{n,m+1}^{(seq)} = \cdots = \mu_{n,K}^{(seq)}}^{\mathcal{B}_n} = G_n \quad \textit{and} \\ \textit{Communications:} & F_n = \underbrace{\nu_{n,1}^{(seq)} = \cdots = \nu_{n,m}^{(seq)}}_{\mathcal{W}_n} > \nu_{n,m+1}^{(seq)} \geqslant \cdots \geqslant \nu_{n,K}^{(seq)} \end{cases}$$

3.3 Closed-form solution of the equations

Theorem 1. *We assume $c_1 \leqslant c_2 \leqslant \cdots \leqslant c_K$. Let us denote by $\mathcal{W}_n$ the set of players that are computation-saturated and by $\mathcal{B}_n$ the set of players that are communication-saturated.*

1. *If $\mathcal{B}_n$ and $\mathcal{W}_n$ are non-empty, we have:*

$$\begin{cases} \alpha_{n,k}^{(nc)} = \frac{B_n}{b_k} \frac{|\mathcal{W}_n| - \sum_{p \in \mathcal{W}_n} \frac{c_p}{C_n}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p \in \mathcal{W}_n} c_p \sum_{p \in \mathcal{B}_n} \frac{1}{c_p}} & \text{if } k \in \mathcal{B}_n \\ \alpha_{n,k}^{(nc)} = \frac{W_n}{w_k} \frac{|\mathcal{B}_n| - \sum_{p \in \mathcal{B}_n} \frac{C_n}{c_p}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p \in \mathcal{W}_n} c_p \sum_{p \in \mathcal{B}_n} \frac{1}{c_p}} & \text{if } k \in \mathcal{W}_n \end{cases} \tag{4}$$

2. *If $\mathcal{W}_n = \emptyset$:*

$$\alpha_{n,k}^{(nc)} = \frac{B_n}{K.b_k}$$

3. *If $\mathcal{B}_n = \emptyset$:*

$$\alpha_{n,k}^{(nc)} = \frac{W_n}{K.w_k}$$

Proof. Let us start by the two easy cases where $\mathcal{W}_n = \emptyset$ or $\mathcal{B}_n = \emptyset$:

- If $\mathcal{B}_n = \emptyset$, then all applications use the CPU of P_n at any time. Therefore they all receive the exact same amount of CPU, i.e. B_n/K, hence $\alpha_{n,k}^{(nc)} = \frac{W_n}{K.w_k}$.
- If $\mathcal{W}_n = \emptyset$, the exact same reasoning holds and we get $\alpha_{n,k}^{(nc)} = \frac{B_n}{K.b_k}$.

Let us now focus on the more interesting case where both $\mathcal{B}_n \neq \emptyset$ and $\mathcal{W}_n \neq \emptyset$. Using the definition of sequential communication and computation times, we have:

$$\begin{cases} \sum_{p \in \mathcal{B}_n} \mu_{n,p}^{(seq)} + \sum_{p \in \mathcal{W}_n} \mu_{n,p}^{(seq)} = 1 \\ \sum_{p \in \mathcal{B}_n} \nu_{n,p}^{(seq)} + \sum_{p \in \mathcal{W}_n} \nu_{n,p}^{(seq)} = 1 \end{cases} \tag{5}$$

As we have $\alpha_{n,k}^{(nc)} = \frac{b_k}{B_n} \mu_{n,k}^{(seq)}$ and $\alpha_{n,k}^{(nc)} = \frac{w_k}{W_n} \nu_{n,k}^{(seq)}$, $\nu_{n,p}^{(seq)}$ and $\mu_{n,p}^{(seq)}$ are linked by the following relation:

$$\forall p, \nu_{n,p}^{(seq)} = \frac{C_n}{c_p} \mu_{n,p}^{(seq)} \tag{6}$$

Two applications from $\mathcal{B}_n$ communicate all the time. Therefore the send the exact same amount of data and thus:

$$\forall p \in \mathcal{B}_n, \mu_{n,p}^{(seq)} = G_n$$

Similarly, we get

$$\forall p \in \mathcal{W}_n, \nu_{n,p}^{(seq)} = F_n$$

Using these relations, equations (5) can be written:

$$\begin{cases} |\mathcal{B}_n| G_n + F_n \sum_{p \in \mathcal{W}_n} \frac{c_p}{C_n} = 1 \\ |\mathcal{W}_n| F_n + G_n \sum_{p \in \mathcal{B}_n} \frac{C_n}{c_p} = 1 \end{cases}$$

Solving this system, we get:

$$\begin{cases} G_n = \frac{|\mathcal{W}_n| - \sum_{p\in\mathcal{W}_n} \frac{c_p}{C_n}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p\in\mathcal{W}_n} c_p \sum_{p\in\mathcal{B}_n} \frac{1}{c_p}} \\ F_n = \frac{|\mathcal{B}_n| - \sum_{p\in\mathcal{B}_n} \frac{C_n}{c_p}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p\in\mathcal{W}_n} c_p \sum_{p\in\mathcal{B}_n} \frac{1}{c_p}}. \end{cases}$$

And finally:

$$\begin{cases} \alpha_{n,k}^{(nc)} = \frac{B_n}{b_k} \frac{|\mathcal{W}_n| - \sum_{p\in\mathcal{W}_n} \frac{c_p}{C_n}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p\in\mathcal{W}_n} c_p \sum_{p\in\mathcal{B}_n} \frac{1}{c_p}} & \text{if } k \in \mathcal{B}_n \\ \alpha_{n,k}^{(nc)} = \frac{W_n}{w_k} \frac{|\mathcal{B}_n| - \sum_{p\in\mathcal{B}_n} \frac{C_n}{c_p}}{|\mathcal{W}_n||\mathcal{B}_n| - \sum_{p\in\mathcal{W}_n} c_p \sum_{p\in\mathcal{B}_n} \frac{1}{c_p}} & \text{if } k \in \mathcal{W}_n \end{cases}$$

□

3.4 Conditions on the sets

Let us first start with a technical lemma that will prove to be very useful in the following.

Lemma 1. *Let $\gamma_1 < \cdots < \gamma_K$ be K positive numbers. We have:*

1. *If $\sum_k 1/\gamma_k \leqslant K$ then $\sum_k \gamma_k > K$;*
2. *If $\sum_k \gamma_k \leqslant K$ then $\sum_k 1/\gamma_k > K$;*
3. *If $\sum_k \gamma_k > K$ and $\sum_k 1/\gamma_k > K$, then there exists exactly one $m \in [\![1, K]\!]$ such that:*

$$\gamma_m < \frac{\sum_{k=1}^{m} 1 - \gamma_k}{\sum_{k=m+1}^{K} 1 - \frac{1}{\gamma_k}} < \gamma_{m+1}$$

Proof. Let us prove each statement one after the other:

1. Let us assume that $\sum_k 1/\gamma_k \leqslant K$. As $(1-\gamma_k)^2 \geqslant 0$, we have $(2-\gamma_k) \leqslant 1/\gamma_k$. Therefore, we have $\sum_k (2-\gamma_k) < K$, hence $\sum_k \gamma_k > K$.
2. The same proof as before can be used by setting $\gamma'_k = 1/\gamma_k$.
3. Let us now assume that $\sum_k \gamma_k < K$ and $\sum_k 1/\gamma_k > K$ and let us define ϱ by $\varrho(m) = \frac{\sum_{k=1}^{m} 1-\gamma_k}{\sum_{k=m+1}^{K} 1-\frac{1}{\gamma_k}}$. Let us also define f and g by

$$f(m) = \gamma_{m+1} \sum_{k=m+1}^{K} \left(1 - \frac{1}{\gamma_k}\right) - \sum_{k=1}^{m} (1-\gamma_k)$$

$$g(m) = \gamma_m \sum_{k=m+1}^{K} \left(1 - \frac{1}{\gamma_k}\right) - \sum_{k=1}^{m} (1-\gamma_k)$$

From these definitions, we can easily see that:

$$(\exists! m \,|\, \gamma_m < \varrho(m) < \gamma_{m+1}) \Leftrightarrow (\exists! m \,|\, f(m) > 0 \text{ and } g(m) < 0) \tag{7}$$

By noticing that we have

$$f(m) - g(m+1) = \gamma_{m+1}\left(1 - \frac{1}{\gamma_{m+1}}\right) + (1 - \gamma_{m+1}) = 0,$$

condition (7) can be rewritten as

$$(\exists! m \,|\, \gamma_m < \varrho(m) < \gamma_{m+1}) \Leftrightarrow (\exists! m \,|\, f(m) > 0 \text{ and } f(m-1) < 0)$$

Therefore, to prove our lemma, we should study the variations of f. We have[3]:

$$\begin{cases} f(0) & = \gamma_{K+1}\left(K - \sum_{k=1}^{K} \frac{1}{\gamma_k}\right) < 0 \text{ (by hypothesis)} \\ f(K) & = \left(\sum_{k=1}^{K} \gamma_k\right) - K > 0 \text{ (by hypothesis)} \\ f(m+1) - f(m) & = (\gamma_{m+2} - \gamma_{m+1}) \underbrace{\sum_{k=m+2}^{K} \left(1 - \frac{1}{\gamma_k}\right)}_{d(m)} \end{cases}$$

Then, we have

$$\begin{cases} d(0) & = K - \sum_{k=1}^{K} \frac{1}{\gamma_k} < 0 \text{ (by hypothesis)} \\ d(K-1) & = 0 \\ d(m+1) - d(m) & = \frac{1}{\gamma_{m+2}} - 1 \end{cases}$$

So let us denote by M the first value k such that $\gamma_k > 1$ and $\gamma_{k-1} < 1$ (we

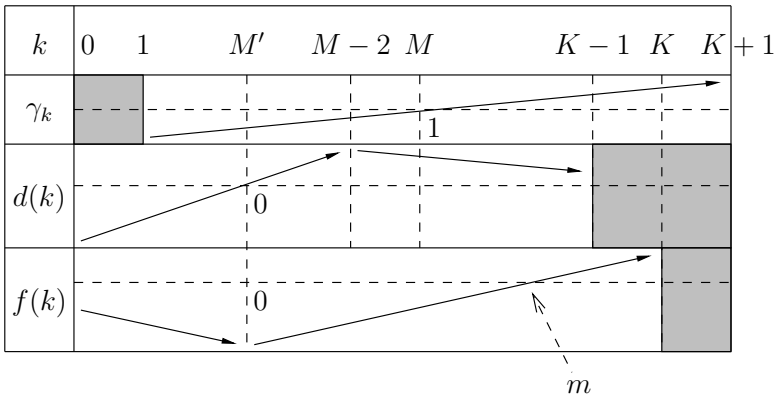


Figure 3: Variation table of f, h and γ.

know that such value exist otherwise we would have $\sum_k \gamma_k \leqslant K$ or $\sum_k 1/\gamma_k \leqslant K$). Therefore, we know that d is strictly increasing on $[\![0, M-2]\!]$ and strictly decreasing on $[\![M-2, 0]\!]$ (see Figure 3). Since $d(0) < 0$, and $d(K-1) = 0$, we know that

[3]We can freely set $\gamma_{K+1} = \gamma_K + 1$ so that f is defined on $[\![0, K]\!]$.

$d(M-2) > 0$. Therefore, there exists a $M' \in [\![1, K-1]\!]$ such that d is negative on $[\![0, M'-1]\!]$ and positive on $[\![M', K-1]\!]$. Last, since $f(0) < 0$ and $f(K) > 0$, we know there exists exactly one m such that $f(m) > 0$ and $f(m-1) < 0$, hence the result. □

Theorem 2. *Assuming $c_1 \leqslant ... \leqslant c_K$, $\mathcal{B}_n$ and $\mathcal{W}_n$ can be determined using the following formula:*

1. *If $\sum_k \frac{c_k}{C_n} \leqslant K$ then $\mathcal{B}_n = \emptyset$.*
2. *Else if $\sum_k \frac{C_n}{c_k} \leqslant K$ then $\mathcal{W}_n = \emptyset$.*
3. *Else we have $\mathcal{W}_n \neq \emptyset$ and $\mathcal{B}_n \neq \emptyset$. Let m be the integer such that*[4]*:*

$$\frac{c_m}{C_n} < \frac{m - \sum_{k=1}^{m} \frac{c_k}{C_n}}{K - m - \sum_{k=m+1}^{K} \frac{C_n}{c_k}} < \frac{c_{m+1}}{C_n}$$

Then we have $\mathcal{W}_n = \{1, \ldots, m\}$ and $\mathcal{B}_n = \{m+1, \ldots, K\}$.

Proof. First of all, we shall remark that the previous cases are exclusives using Lemma 1 with $\gamma_k = c_k/C_n$. Therefore, we only have to prove that, depending on the structure of $\mathcal{B}_n$ and $\mathcal{W}_n$, we fall in either one or the other of the previous cases:

- Suppose that $\mathcal{B}_n = \emptyset$, then $\forall k, \nu_{n,k}^{(seq)} = 1/K$ and $1 \geqslant \sum_k \mu_{n,k}^{(seq)} = \sum_k \frac{c_k}{C_n} \nu_{n,k}^{(seq)} = \sum_k \frac{c_k}{KC_n}$. Hence $\mathcal{B}_n = \emptyset \Rightarrow \sum_k \frac{C_n}{c_k} \leqslant K$.
- Suppose that $\mathcal{W}_n = \emptyset$, then $\forall k, \mu_{n,k}^{(seq)} = 1/K$ and $1 \geqslant \sum_k \nu_{n,k}^{(seq)} = \sum_k \frac{C_n}{c_k} \mu_{n,p}^{(seq)} = \sum_k \frac{C_n}{Kc_k}$. Hence $\mathcal{W}_n = \emptyset \Rightarrow \sum_k \frac{C_n}{c_k} \leqslant K$.
- Suppose that both $\mathcal{W}_n \neq \emptyset$ and $\mathcal{B}_n \neq \emptyset$. Remember that $\nu_{n,p}^{(seq)}$ and $\mu_{n,p}^{(seq)}$ are linked by the following relation:

$$\forall p, \nu_{n,p}^{(seq)} = \frac{C_n}{c_p} \mu_{n,p}^{(seq)}$$

Let m such that $m \in \mathcal{B}_n$ and $m+1 \in \mathcal{W}_n$. Recalling proposition 2, we have:

$$\mu_{n,1}^{(seq)} \leqslant \cdots \leqslant \mu_{n,m}^{(seq)} < \mu_{n,m+1}^{(seq)} = \cdots = \mu_{n,K}^{(seq)} \text{ and } \nu_{n,1}^{(seq)} = \cdots = \nu_{n,m}^{(seq)} > \nu_{n,m+1}^{(seq)} \geqslant \cdots \geqslant \nu_{n,K}^{(seq)}$$

From $\nu_{n,m+1}^{(seq)} = \frac{C_n}{c_{m+1}} \mu_{n,m+1}^{(seq)}$ and equation (4) of Theorem 1, we get:

$$\frac{c_{m+1}}{C_n} = \frac{\mu_{n,m+1}^{(seq)}}{\nu_{n,m+1}^{(seq)}} > \frac{\mu_{n,m+1}^{(seq)}}{\nu_{n,m}^{(seq)}} = \frac{m - \sum_{k=1}^{m} \frac{c_k}{C_n}}{K - m - \sum_{k=m+1}^{K} \frac{C_n}{c_k}}$$

[4] We know such an integer exists using Lemma 1 with $\gamma_k = c_k/C_n$

Similarly, we have:

$$\frac{c_m}{C_n} = \frac{\mu_{n,m}^{(seq)}}{\nu_{n,m}^{(seq)}} < \frac{\mu_{n,m+1}^{(seq)}}{\nu_{n,m}^{(seq)}} = \frac{m - \sum_{k=1}^{m} \frac{c_k}{C_n}}{K - m - \sum_{k=m+1}^{K} \frac{C_n}{c_k}}$$

Therefore, we have:

$$\frac{c_m}{C_n} < \frac{m - \sum_{k=1}^{m} \frac{c_k}{C_n}}{K - m - \sum_{k=m+1}^{K} \frac{C_n}{c_k}} < \frac{c_{m+1}}{C_n}$$

that is to say

$$\gamma_m < \frac{m - \sum_{k=1}^{m} \gamma_k}{K - m - \sum_{k=m+1}^{K} \frac{1}{\gamma_k}} < \gamma_{m+1}. \qquad \square$$

Remark 1. *From these equations, we see that there always exists exactly one non-cooperative equilibrium.*

4 Inefficiencies and paradoxes

In this section, we study the inefficiency of the Nash equilibria, in the Pareto sense, and their consequences. Let us start by recalling the definition of the Pareto optimality.

Definition 4. *Let G be a game with K players. Each of them is defined by a set of possible strategies S_k and utility functions u_k defined on $S_1 \times \cdots \times S_K$.*
A vector of strategy is said to be Pareto optimal if it is impossible to strictly increase the utility of a player without strictly decreasing the one of another. In other words,

$$\Big((s_1, \ldots, s_K) \in S_1 \times \cdots \times S_K \text{ is Pareto optimal }\Big) \text{ iff}$$
$$\Big(\forall (s_1^*, \ldots, s_K^*) \in S_1 \times \cdots \times S_K,$$
$$\exists i, u_i(s_1^*, \ldots, s_K^*) > u_i(s_1, \ldots, s_K) \Rightarrow \exists j, u_j(s_1^*, \ldots, s_K^*) < u_j(s_1, \ldots, s_K)\Big)$$

We recall that in the considered system, the utility functions are the α_k, that is to say, the average number of tasks of application k processed per time-unit, while the strategies are the scheduling algorithms (i.e. which resource to use and when to use them).

After expressing the utility set, i.e. the achievable utility vectors (subsection 4.1), we comment of the efficiency of the Nash equilibrium, in the case of a single worker (subsection 4.2), and then of multiple workers (4.3). Additionally, it is known that when Nash Equilibria are inefficient, some paradoxical phenomenon can occur (see, for instance [Bra68]). We hence study in subsection 4.4, the occurrence of Braess phenomena in this system.

4.1 Convex and compact utility set

Let consider a system with K applications running over N machines. We recall that the set of achievable utilities is:

$$\left\{ (\alpha_k)_{1\leqslant k\leqslant K} \;\middle|\; \begin{array}{l} \exists \alpha_{1,k}, \ldots, \alpha_{n,k}, \\ \quad \forall k \in [\![1,K]\!] : \sum_{n=1}^{N} \alpha_{n,k} = \alpha_k \\ \quad \forall n \in [\![1,N]\!] : \sum_{k=1}^{K} \alpha_{n,k}.w_k \leqslant W_n \\ \quad \forall n \in [\![1,N]\!] : \sum_{k=1}^{K} \alpha_{n,k}.b_k \leqslant B_n \\ \quad \forall n \in [\![1,N]\!], \forall k \in [\![1,K]\!] : \alpha_{n,k} \geqslant 0 \end{array} \right\}$$

The utility set is hence convex and compact.

4.2 Single processor

In this subsection we show that when the players (here the applications) compete in a non-cooperative way over a single processor, the resulting equilibrium (see Section 3.2) is Pareto optimal.

Theorem 3. *On a single-processor system, the allocation at the Nash equilibrium is Pareto optimal.*

Proof. Suppose that the non-cooperative allocation $\alpha^{(nc)}$ is not Pareto optimal on P_n. Then, there exists another allocation $\widetilde{\alpha}$. such that

$$\forall k, \alpha_{n,k}^{(nc)} \leqslant \widetilde{\alpha}_{n,k}, \quad \text{and} \quad \exists q \in [\![1,K]\!], \alpha_{n,q}^{(nc)} < \widetilde{\alpha}_{n,q}.$$

Then $\forall k, \alpha_{n,k}^{(nc)} = \frac{\mu_{n,k}^{(seq)} B_n}{b_k} \leqslant \frac{\widetilde{\mu}_{n,k}^{(seq)} B_n}{b_k} = \widetilde{\alpha}_{n,k}$ and $\frac{\mu_{n,q}^{(seq)} B_n}{b_k} < \frac{\widetilde{\mu}_{n,q}^{(seq)} B_n}{b_k}$, which implies

$$\sum_{k=1}^{K} \mu_{n,k}^{(seq)} < \sum_{k=1}^{K} \widetilde{\mu}_{n,k}^{(seq)} \tag{8}$$

Similarly, we get

$$\sum_{k=1}^{K} \nu_{n,k}^{(seq)} < \sum_{k=1}^{K} \widetilde{\nu}_{n,k}^{(seq)} \tag{9}$$

We have two possibilities:

- If in the non-cooperative solution, at least one application is communication-saturated (i.e. $\mathcal{B}_n \neq \emptyset$), then $1 = \sum_k \mu_{n,k}^{(seq)}$ (recall proposition 1). However, as $\sum_k \widetilde{\mu}_{n,k}^{(seq)} \leqslant 1$ by definition, we have $\sum_{k=1}^{K} \mu_{n,k}^{(seq)} \geqslant \sum_{k=1}^{K} \widetilde{\mu}_{n,k}^{(seq)}$, which is in contradiction with (8).
- We get a similar contradiction with (9) when at least one application is computation-saturated (i.e. $\mathcal{W}_n \neq \emptyset$) in the non-cooperative solution. □

4.3 Multi-processors and inefficiencies

Interestingly, although we have seen that the Nash Equilibria are Pareto optimal on each worker, we show in this section that this Equilibrium is not Pareto optimal for the whole system.

We show that the Nash Equilibrium can be inefficient and exhibit such an example on a system consisting of two machines and two applications (subsection 4.3.1). We then give a formal necessary and sufficient condition for the equilibrium to be Pareto optimal (subsection 4.3.2 and 4.3.3). We finally comment on different measures of inefficiency (subsection 4.3.4).

4.3.1 Example of Pareto inefficiency

The Pareto optimality is a global notion. Hence, although at each processor, the allocation is Pareto optimal, the result may not hold for a random number of machines. We will see later (Theorem 4) that the result holds if we have on all the machines $\mathcal{W}_n \neq \emptyset$ OR on all the machines $\mathcal{B}_n \neq \emptyset$. Yet, if there are two workers n_1 and n_2 such that $\mathcal{W}_{n_1} = \emptyset$ and $\mathcal{B}_{n_2} = \emptyset$, then the allocation may be Pareto sub-optimal. We illustrate this on a simple example with a two-applications and two-machines system.

Example 1. *Consider a system with two computers A and B, with parameters $B_1 = 50$, $W_1 = 100$, $B_2 = 100$, $B_2 = 50$ and two applications of parameters $b_1 = 1$, $w_1 = 2$, $b_2 = 2$ and $w_2 = 1$.*
If the applications were collaborating such that application 1 was processed exclusively to computer A and application 2 in computer B, their respective throughput would be

$$\alpha_1^{(coop)} = \alpha_2^{(coop)} = 50.$$

Yet, with the non-cooperative approach they only get a throughput of

$$\alpha_1^{(nc)} = \alpha_2^{(nc)} = 37.5.$$

4.3.2 Sufficient condition

Theorem 4. *If, in a system with a finite number N of machines, we have*

$$\forall n \in [\![1, N]\!], \mathcal{B}_n \neq \emptyset$$

then, the non-cooperative allocation is Pareto optimal.

Proof. Suppose that for the N machines, $\mathcal{B}_n \neq \emptyset$. Suppose that the non-cooperative allocation $\alpha^{(nc)}$ is not Pareto optimal. Then, there exists another allocation $\widetilde{\alpha}$ such that

$$\forall k, \alpha_k^{(nc)} \leqslant \widetilde{\alpha}_k, \quad \text{and} \quad \exists q \in [\![1, K]\!], \alpha_q^{(nc)} < \widetilde{\alpha}_q.$$

Then $\forall k, \sum_{n=1}^{N} \frac{\mu_{n,k}^{(seq)} B_n}{b_k} = \sum_{n=1}^{N} \alpha_{n,k}^{(nc)} = \alpha_k^{(nc)} \leqslant \widetilde{\alpha}_k = \sum_{n=1}^{N} \widetilde{\alpha}_{n,k} = \sum_{n=1}^{N} \frac{\widetilde{\mu}_{n,k}^{(seq)} B_n}{b_k}$ and $\sum_{n=1}^{N} \frac{\mu_{n,q}^{(seq)} B_n}{b_q} < \sum_{n=1}^{N} \frac{\widetilde{\mu}_{n,q}^{(seq)} B_n}{b_q}$, which implies

$$\sum_{k=1}^{K} \sum_{n=1}^{N} \mu_{n,k}^{(seq)} B_n < \sum_{k=1}^{K} \sum_{n=1}^{N} \widetilde{\mu}_{n,k}^{(seq)} B_n \tag{10}$$

However, as on each machine, at least one application is communication saturated (i.e. $\forall n, \mathcal{B}_n \neq \emptyset$) then $\sum_k \mu_{n,k}^{(seq)} = 1$, which implies that $\forall n, \sum_k \mu_{n,k}^{(seq)} \geqslant \sum_k \widetilde{\mu}_{n,k}^{(seq)}$. Hence finally

$$\sum_{n=1}^{N} \sum_{k=1}^{K} \mu_{n,k}^{(seq)} B_n \geqslant \sum_{n=1}^{N} \sum_{k=1}^{K} \widetilde{\mu}_{n,k}^{(seq)} B_n,$$

which is in contradiction. with (10). □

Corollary 1. *If, in a system with a finite number N of machines, we have*

$$\forall n \in N, \mathcal{W}_n \neq \emptyset$$

then, the non-cooperative allocation is Pareto optimal.

4.3.3 Necessary and sufficient condition

Suppose that the applications are not all identical, that is to say that there exists k_1 and k_2 such that $c_{k_1} < c_{k_2}$. We have seen in Section 4.3.2, that two sufficient conditions for the Nash equilibrium to be Pareto optimal is that:

- $\forall n, \mathcal{W}_n \neq \emptyset$ (i.e. $\forall n, \sum_k \frac{C_n}{c_k} > K$)
- or $\forall n, \mathcal{B}_n \neq \emptyset$ (i.e. $\forall n, \sum_k \frac{c_k}{C_n} > K$).

In this section, we show that this condition is actually necessary.

Theorem 5. *Suppose that the applications are not all identical, that is to say that there exists k_1 and k_2 such that $c_{k_1} < c_{k_2}$. Suppose that there exists two workers, namely n_1 and n_2 such that:*

- $\mathcal{W}_{n_1} = \emptyset$
- *and* $\mathcal{B}_{n_2} = \emptyset$,

then the allocation at the Nash Equilibrium is Pareto inefficient.

Proof. Let renumber the applications such that $c_1 \leqslant c_2 \leqslant \cdots \leqslant c_K$ and suppose that there exists $k \in [\![1, K-1]\!]$ such that $c_k < c_{k+1}$. To show the Pareto inefficiency of the Nash Equilibrium, we construct another allocation $\widetilde{\alpha}$ in which:

$$\forall k \in [\![2, K-1]\!], \alpha_k^{(nc)} = \widetilde{\alpha}_k \qquad \alpha_1^{(nc)} < \widetilde{\alpha}_1 \qquad \alpha_K^{(nc)} < \widetilde{\alpha}_K$$

The idea for this is to keep the allocations of the applications $2, \ldots, K-1$ unchanged. We note that n_1 is relatively poor in bandwidth while n_2 relatively lacks of computational capability. Also, as $c_1 < c_K$, the idea of the construction is to increase the fraction of jobs from application 1 (respectively K) sent to worker n_1 (resp. n_2). More precisely, we take:

- $\forall n \notin \{n_1, n_2\}, \forall k \in [\![1; K]\!], \mu_{n,k}^{(seq)} = \widetilde{\mu}_{n,k}^{(seq)}, \nu_{n,k}^{(seq)} = \widetilde{\nu}_{n,k}^{(seq)}$,
- $\forall n \in \{n_1, n_2\}, \forall k \in [\![2; K-1]\!], \mu_{n,k}^{(seq)} = \widetilde{\mu}_{n,k}^{(seq)}, \nu_{n,k}^{(seq)} = \widetilde{\nu}_{n,k}^{(seq)}$,
- $\nu_{n_2,1}^{(seq)} = \widetilde{\nu}_{n_2,1}^{(seq)} - \varepsilon_1$, $\nu_{n_2,K}^{(seq)} = \widetilde{\nu}_{n_2,K}^{(seq)} + \varepsilon_3$, $\mu_{n_1,1}^{(seq)} = \widetilde{\mu}_{n_1,1}^{(seq)} + \varepsilon_2$, and $\mu_{n_1,K}^{(seq)} = \widetilde{\mu}_{n_1,K}^{(seq)} - \varepsilon_4$.

Hence $\forall k \in [\![2, K-1]\!], \widetilde{\alpha}_k = \alpha_k^{(nc)}$ and $\forall n \notin \{n_1, n_2\}, \left(\widetilde{\alpha}_{n,1} = \alpha_{n,1}^{(nc)} \text{ and } \widetilde{\alpha}_{n,K} = \alpha_{n,K}^{(nc)}\right)$. These definitions leads us to the following equations for ε_1, ε_2, ε_3 and ε_4:

$$\begin{cases} \sum_k \mu_{n_2,k}^{(seq)} = \left(\sum_k \widetilde{\mu}_{n_2,k}^{(seq)}\right) + \left(\frac{c_K}{C_{n_2}}\varepsilon_3 - \frac{c_1}{C_{n_2}}\varepsilon_1\right) \\ \sum_k \nu_{n_2,k}^{(seq)} = \left(\sum_k \widetilde{\nu}_{n_2,k}^{(seq)}\right) + (\varepsilon_3 - \varepsilon_1) \end{cases} \quad \begin{cases} \sum_k \mu_{n_1,k}^{(seq)} = \left(\sum_k \widetilde{\mu}_{n_1,k}^{(seq)}\right) + (\varepsilon_2 - \varepsilon_4) \\ \sum_k \nu_{n_1,k}^{(seq)} = \left(\sum_k \widetilde{\nu}_{n_1,k}^{(seq)}\right) + \left(\frac{C_{n_1}}{c_1}\varepsilon_2 - \frac{C_{n_1}}{c_K}\varepsilon_4\right) \end{cases}$$

As $\mathcal{B}_{n_2} = \emptyset$, we have $\mathcal{W}_{n_2} \neq \emptyset$ hence $\sum_k \nu_{n_2,k}^{(seq)} = 1$. By setting $\varepsilon_1 = \varepsilon_3$, we get $\sum_k \widetilde{\nu}_{n_2,k}^{(seq)} = 1$. Also, as $\mathcal{B}_{n_2} = \emptyset$, we have $\sum_k \mu_{n_2,k}^{(seq)} < 1$, hence for ε_1 sufficiently small $\sum_k \widetilde{\mu}_{n_2,k}^{(seq)} < 1$.

As $\mathcal{W}_{n_1} = \emptyset$, we have $\mathcal{B}_{n_1} \neq \emptyset$ hence $\sum_k \mu_{n_1,k}^{(seq)} = 1$. By setting $\varepsilon_3 = \varepsilon_4$, we get $\sum_k \widetilde{\mu}_{n_1,k}^{(seq)} = 1$. Also, as $\mathcal{W}_{n_1} = \emptyset$, we have $\sum_k \nu_{n_1,k}^{(seq)} < 1$, hence for ε_2 sufficiently small $\sum_k \widetilde{\nu}_{n_1,k}^{(seq)} < 1$. Therefore, by setting ε_1 and ε_2 sufficiently small, $\widetilde{\alpha}$ is a valid allocation.

Finally, since $\mathcal{W}_{n_1} = \mathcal{B}_{n_2} = \emptyset$, then $\alpha_{n_1,k}^{(nc)} = \dfrac{B_{n_1}}{K.b_k}$ and $\alpha_{n_2,k}^{(nc)} = \dfrac{W_{n_1}}{K.w_k}$. Hence finally, $\widetilde{\alpha}_1 > \alpha_1^{(nc)}$ and $\widetilde{\alpha}_K > \alpha_K^{(nc)}$ are equivalent to $\widetilde{\alpha}_{n_1,1} + \widetilde{\alpha}_{n_2,1} > \alpha_{n_1,1}^{(nc)} + \alpha_{n_2,1}^{(nc)}$ and $\widetilde{\alpha}_{n_1,K} + \widetilde{\alpha}_{n_2,K} > \alpha_{n_1,K}^{(nc)} + \alpha_{n_2,K}^{(nc)}$. We have:

$$\widetilde{\alpha}_{n_1,1} + \widetilde{\alpha}_{n_2,1} - \alpha_{n_1,1}^{(nc)} + \alpha_{n_2,1}^{(nc)} = \frac{B_{n_1}}{b_1}\varepsilon_2 - \frac{W_{n_2}}{w_1}\varepsilon_1$$

and

$$\widetilde{\alpha}_{n_1,K} + \widetilde{\alpha}_{n_2,K} > \alpha_{n_1,K}^{(nc)} + \alpha_{n_2,K}^{(nc)} = \frac{W_{n_2}}{w_K}\varepsilon_1 - \frac{W_{n_1}}{w_K}\varepsilon_2$$

Therefore, the conditions $\widetilde{\alpha}_1 > \alpha_1^{(nc)}$ and $\widetilde{\alpha}_K > \alpha_K^{(nc)}$ are equivalent to:

$$\frac{W_{n_2}}{B_{n_1}}c_1 < \frac{\varepsilon_2}{\varepsilon_1} < \frac{W_{n_2}}{B_{n_1}}c_K \tag{11}$$

Hence, for ε_1 and ε_2 sufficiently small and satisfying (11), the allocation $\widetilde{\alpha}$ is strictly Pareto superior to the Nash equilibrium of the system. □

4.3.4 Degree of inefficiency

We have seen that the Nash equilibrium of the system can be Pareto inefficient. An natural question then raising is "how much inefficient" is it ? Unfortunately, defining Pareto inefficiency is still an open question. We recall in the following some possible definitions and study their properties.

Definition of degree of inefficiency Let us denote by f an efficiency measure on the α_k. Classical efficiency measure are:

Profit $f(\alpha_1, \ldots, \alpha_K) = \sum_{k=1}^{K} \alpha_k$

Max-min $f(\alpha_1, \ldots, \alpha_K) = \min_{k=1}^{K} \alpha_k$

Proportional $f(\alpha_1, \ldots, \alpha_K) = \prod_{k=1}^{K} \alpha_k$

Inverse $f(\alpha_1, \ldots, \alpha_K) = \left(\sum_{k=1}^{K} \frac{1}{\alpha_k}\right)^{-1}$

For a given system S (i.e. platform parameters along with the description of our K applications), we denote by $\alpha_k^{(nc)}(S)$, the rates achieved on system S by the non-cooperative algorithm of Section 3. Let us denote by $\alpha_k^{(f)}(S)$, the optimal rates on system S for the metric f.

One can define the inefficiency of the non-cooperative allocation for a given metric and a given system as:

$$\frac{f\left(\alpha_1^{(f)}(S), \ldots, \alpha_K^{(f)}(S)\right)}{f\left((\alpha_1^{(nc)}(S), \ldots, \alpha_K^{(nc)}(S)\right)} \geqslant 1$$

The degree of inefficiency ϕ_f can be defined as the largest inefficiency:

$$\phi_f = \max_I \frac{f\left(\alpha_1^{(f)}(S), \ldots, \alpha_K^{(f)}(S)\right)}{f\left(\alpha_1^{(nc)}(S), \ldots, \alpha_K^{(nc)}(S)\right)} \geqslant 1$$

Similarly, Papadimitriou [Pap98] introduced the now popular measure "price of anarchy", which corresponds to the inefficiency degree for the profit metrics.

Properties Consider again a system with two applications and two machines with parameters:

$$b_1 = 1 \quad w_1 = N \quad b_2 = N \quad w_2 = 1$$
$$B_A = 100 \quad W_A = 100 * N \quad B_B = 100 * N \quad W_B = 100$$

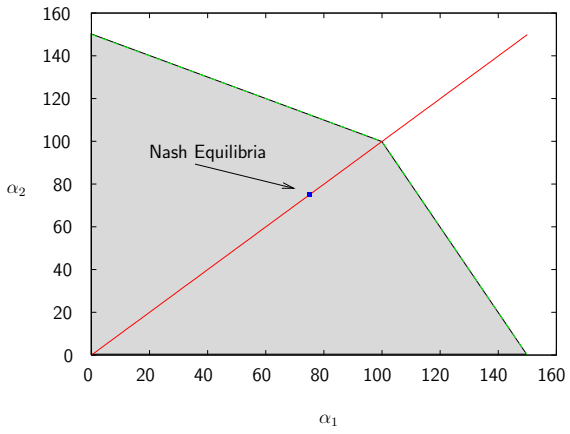


Figure 4: Pareto-inefficiency of the Nash Equilibrium on a two-machines/two-applications system

Consider then the non-cooperative approach. We have $\mathcal{W}_A = \emptyset$ and $\mathcal{B}_B = \emptyset$. We have:

$$\alpha_{n,1}^{(nc)} = \frac{B_A}{2b_1} + \frac{W_B}{2w_1} = 50\left(1 + \frac{1}{N}\right)$$
$$\alpha_{n,2}^{(nc)} = \frac{B_A}{2b_2} + \frac{W_B}{2w_2} = 50\left(1 + \frac{1}{N}\right)$$

However,

$$\alpha_{n,1}^{(\text{coop})} = 100 \qquad \alpha_{n,2}^{(\text{coop})} = 100.$$

As the utility set is clearly symmetrical for this system (see Figure 4), and the efficiency measure do not favor a particular application, the optimal point for all the previous measures is $\alpha^{(\text{coop})}$. As $\frac{\alpha_{n,1}^{(\text{coop})}}{\alpha_{n,1}^{(nc)}} = \frac{\alpha_{n,2}^{(\text{coop})}}{\alpha_{n,2}^{(nc)}} \xrightarrow[N\to+\infty]{} 2$, the efficiency degree of all previous measures is larger than two (except for the Proportional measure that is equal to 4).

4.4 Braess-like paradoxes

When studying properties of Nash equilibria in routing systems, Braess exhibited an example in which, by adding resource to the system (in his example, a route), the performance of all the users were degraded. We investigate in this section whether such situations can occur in our considered scenario.

Based on the definition of the Pareto optimality, we define the concept of strict Pareto-superiority.

Definition 5. *We say that a utility point a is* strictly Pareto-superior *to a point b is for all players $a_i < b_i$.*

Obviously, a Pareto-optimal point is such that there exists no achievable point strictly-Pareto superior to it.

Let us consider a system (called "initial") and a second one (referred to as the "augmented" system), derived from the first one by adding some quantity of resource. Intuitively, the Nash equilibrium *aug* in the augmented system should be Pareto-superior to the one in the initial system *ini*. We say that a Braess paradox happens when *ini* is strictly Pareto-superior to point *aug*.

Obviously, every achievable state in the initial system is also achievable in the augmented system. Hence if a is an achievable point in the initial system and if b is a *Pareto* optimal point is the augmented one, then a cannot be strictly Pareto superior to b. Hence Braess paradoxes are consequences of the Pareto inefficiencies of the Nash equilibria.

4.4.1 Property

We show that, even though the Nash equilibria may not be Pareto optimal, in the considered scenario, Braess paradoxes cannot occur.

Theorem 6. *In the non-cooperative multi-port scheduling problem, Braess like paradoxes cannot occur.*

To prove this proposition, we need to use two lemmas and the following definition.

Definition 6 (Equivalent subsystem)**.** *Consider a system $S = (K, b, w, N, B, W)$. We define the new subsystem $\widetilde{S} = (K, b, w, N, \widetilde{B}, \widetilde{W})$ by: For each worker n,*

$$\widetilde{W}_n = \begin{cases} \sum_k \frac{B_n}{K c_k} & \text{if } \mathcal{W}_n = \emptyset, \\ W_n & \text{otherwise.} \end{cases} \qquad \text{and} \qquad \widetilde{B}_n = \begin{cases} \sum_k \frac{W_n c_k}{K} & \text{if } \mathcal{B}_n = \emptyset, \\ B_n & \text{otherwise.} \end{cases}$$

We shall now precise why $\widetilde{S}$ is said to be an equivalent subsystem of S.

Lemma 2 (Equivalent subsystem)**.** *Consider a system $S = (K, b, w, N, B, W)$ and its Nash equilibrium $\alpha^{(nc)}$.*

i) The system $\widetilde{S}$ is a subsystem of S, i.e. for all worker n: $\widetilde{B}_n \leqslant B_n$ and $\widetilde{W}_n \leqslant W_n$.

ii) The Nash equilibrium $\widetilde{\alpha}^{(nc)}$ of the subsystem $\widetilde{S}$ verifies:

$$\forall n, \forall k, \alpha^{(nc)}_{n,k} = \widetilde{\alpha}^{(nc)}_{n,k}$$

iii) The Nash equilibrium $\widetilde{\alpha}^{(nc)}$ of the subsystem $\widetilde{S}$ is Pareto-optimal.

Proof. Let $n \in [\![1; N]\!]$.

i) If $\mathcal{W}_n = \emptyset$, then using theorem 2, we know that $\sum_k \frac{C_n}{c_k} \leqslant K$, hence $\widetilde{W}_n = \sum_k \frac{B_n}{K c_k} \leqslant W_n$. If $\mathcal{W}_n \neq \emptyset$, then $\widetilde{W}_n = W_n$, hence the result.

Similarly, if $\mathcal{B}_n = \emptyset$, then using theorem 2, we know that $\sum_k \frac{c_k}{C_n} \leqslant K$, hence $\widetilde{B}_n = \sum_k \frac{W_n c_k}{K} \leqslant B_n$. If $\mathcal{B}_n \neq \emptyset$, then $\widetilde{B}_n = B_n$, hence the result.

ii) If $\mathcal{W}_n = \emptyset$, then using theorem 1, we know that $\alpha_{n,k}^{(nc)} = \frac{B_n}{K.b_k}$. As $\mathcal{W}_n = \emptyset$, we have $\widetilde{B}_n = B_n$. Therefore $\sum_k \frac{\widetilde{C}_n}{Kc_k} = 1$, hence $\widetilde{\mathcal{W}}_n = \emptyset$ and $\widetilde{\alpha}_{n,k}^{(nc)} = \frac{\widetilde{B}_n}{K.b_k} = \frac{B_n}{K.b_k} = \alpha_{n,k}^{(nc)}$.

If $\mathcal{B}_n = \emptyset$, then using theorem 1, we know that $\alpha_{n,k}^{(nc)} = \frac{W_n}{K.w_k}$. As $\mathcal{B}_n = \emptyset$, we have $\widetilde{W}_n = W_n$. Therefore $\sum_k \frac{c_k}{K\widetilde{C}_n} = 1$, hence $\widetilde{\mathcal{B}}_n = \emptyset$ and $\widetilde{\alpha}_{n,k}^{(nc)} = \frac{\widetilde{W}_n}{K.w_k} = \frac{W_n}{K.w_k} = \alpha_{n,k}^{(nc)}$.

Last, if neither $\mathcal{B}_n = \emptyset$ nor $\mathcal{W}_n = \emptyset$, then $(B_n, W_n) = (\widetilde{B}_n, \widetilde{W}_n)$, and $\widetilde{\alpha}_{n,k}^{(nc)} = \alpha_{n,k}^{(nc)}$.

iii) Finally, note that $\forall n, \widetilde{\mathcal{W}}_n \neq \emptyset$ and $\widetilde{\mathcal{B}}_n \neq \emptyset$. Hence, from Prop. 4 the Nash equilibrium $\widetilde{\alpha}^{(nc)}$ is Pareto optimal. □

Lemma 3. *Consider two systems $S = (K, b, w, N, B, W)$ and $S' = (K, b, w, N, B', W')$ and their respective equivalent subsystems $\widetilde{S} = (K, b, w, N, \widetilde{B}, \widetilde{W})$ and $\widetilde{S}' = (K, b, w, N, \widetilde{B}', \widetilde{W}')$. Suppose that $\forall n, B'_n \geqslant B_n$ and $W'_n \geqslant W_n$ then $\forall n, \widetilde{B}'_n \geqslant \widetilde{B}_n$ and $\widetilde{W}'_n \geqslant \widetilde{W}_n$.*

Proof. Let $\alpha^{(nc)}$ and $\alpha^{(nc)'}$ be the Nash equilibrium of S and S'.
If $\alpha^{(nc)'}$ is Pareto optimal then $\widetilde{B}'_n = B'_n$ and $\widetilde{W}'_n = W'_n$. Hence, $\forall n, \widetilde{B}_n \leqslant B_n \leqslant B'_n = \widetilde{B}'_n$.
Similarly $\forall n, \widetilde{W}_n \leqslant \widetilde{W}'_n$ hence the result.
Suppose that $\alpha^{(nc)'}$ is not Pareto optimal. Let $n \in [\![1, N]\!]$.

- If $\mathcal{W}_s \neq \emptyset$ and $\mathcal{B}_s \neq \emptyset$ then $BB'_s = B'_s$ and $WW'_s = W'_s$. Hence $BB_s \leqslant B_s \leqslant B'_s = BB'_s$, similarly $WW_s \leqslant WW'_s$.
- If $\mathcal{B}_s = \emptyset$ then $WW'_s = \sum_k c_k B'_s/K$. But, as $B_s \leqslant B'_s$, then $\sum_k c_k B_s/K \leqslant WW'_s$. As $\frac{WW_s}{BB_s} \leqslant \frac{\sum c_k}{K}$ (from the construction of (N, BB, WW)), then $WW_s \leqslant WW'_s$ and $BB_s \leqslant B_s \leqslant B'_s = BB'_s$.
- Similarly, if $\mathcal{B}_n = \emptyset$, then $BB'_s = \frac{W'_s \sum 1/c_k}{K}$. As $W \leqslant W'$ then $BB_s \leqslant BB'_s$ while $WW_s \leqslant W_s \leqslant W'_s = WW'_s$. □

We can then finally prove Theorem 6:

Proof. Consider a user-system $S = (K, b, w)$, and two physical-systems: the initial system (N, B, W) and its Nash equilibrium $\alpha^{(nc)}$ and a second system obtained by adding some quantity of resource to the first one (N', B', W'), and its Nash equilibrium $\alpha^{(nc)'}$. We want to show that $\alpha^{(nc)}$ cannot be Pareto superior to $\alpha^{(nc)'}$.

Suppose that the second system is obtained by adding some machines to the system (i.e. $N' > N$). Then, as the non cooperative game at each machine is independent, the equilibrium on the N original machines will not be affected by the new machines. On the other hand, as in the new machine the allocation of throughput of each application will be strictly positive, then necessary $\alpha^{(nc)'}$ is strictly Pareto superior to $\alpha^{(nc)}$.

Suppose now that the second system consists of N machines with respective bandwidth and computational capacities B' and W'. By definition of the augmented system we have $\forall n \in [\![1, N]\!], B'_n \geqslant B_n$ and $W'_n \geqslant W_n$.

Consider the equivalent subsystems of the initial and the augmented systems $(N, \widetilde{B}, \widetilde{W})$ and $(N, \widetilde{B}', \widetilde{W}')$ (defined as in Lemma 2). Then, from Lemma 3, we have $\forall n, \widetilde{B}_n \leqslant \widetilde{B}'_n$ and $\widetilde{W}_n \leqslant \widetilde{W}'_n$. As the Nash equilibria $\alpha^{(nc)}$ and $\alpha^{(nc)'}$ are both Pareto optimal in these systems, then $\alpha^{(nc)}$ cannot be Pareto superior to $\alpha^{(nc)'}$. □

5 Performance measures

In this section we show that unexpected behavior of some typical performance can occur even for Pareto optimal situations. To ensure optimality of the Nash equilibrium, we consider applications running on a single processor (Prop. 3).

We recall that the Pareto optimality is a global performance measure. Yet, it is possible that, while the resources of the system increase (either by the adding of capacity to a link or of computational capabilities to a processor), the performance measure of a given application decreases. The aim of this section is to illustrate this phenomenon on some typical performance measures.

More precisely, we show through examples the non-monotonicity of the maximal throughput (Subsection 5.2), of the minimal throughput (Subsection 5.3) and of the average throughput (Subsection 5.4). We finally end this section with an example where all these performance measures decrease simultaneously with the increase of the resources.

We can note that all of these performance measures are consequences of the non-monotonicity of the throughput of a given application $(\alpha_k^{(nc)})$, which we hence analytically study (Subsection 5.1).

In the following, we suppose that only one of the resource of the system increases. By symmetry, we suppose that the computational capacity (W_n) is constant, while the link capacity B_n increases.

5.1 Variation of the throughput of a given application

Even on a single processor, the throughput of the application is not a increasing function. Even worse, the degradation can be arbitrarily large.

When considering the equations at the Nash equilibrium (1), we can distinguish 3 cases :

- 2 "saturated" situations that are :

  satW_n If $\mathcal{W}_n = \emptyset$ (i.e. $B_n \leqslant W_n / \sum_k \frac{1}{Kc_k}$), then $\alpha_{n,k}^{(nc)} = \frac{B_n}{K.b_k}$, i.e. the throughput of each application is proportional to B_n. We will note $sat\mathcal{W}_n =]0, W_n / \sum_k \frac{1}{Kc_k}]$.

  satB_n If $\mathcal{B}_n = \emptyset$ (i.e. $\sum_k \frac{c_k W_n}{K} \leqslant B_n$), then $\alpha_{n,k}^{(nc)} = \frac{W_n}{K.w_k}$, i.e. the throughput of each application is constant with respect with B_n. We will note $sat\mathcal{B}_n =]\sum_k \frac{c_k W_n}{K}, +\infty]$.

- 1 "continuous" situation when $W_n / \sum_k \frac{1}{Kc_k} < B_n < \sum_k \frac{c_k W_n}{K}$.

Obviously, in the "saturated" situations, the throughput $\alpha_{n,k}^{(nc)}$ are increasing or constant and the order between the applications is preserved (i.e. if for $B_n \in sat\mathcal{W}_n$ (resp. $sat\mathcal{B}_n$), $\alpha_{n,k_1}^{(nc)} \leqslant \alpha_{n,k_2}^{(nc)}$ then for all $B'_n \in sat\mathcal{W}_n$ (resp. $sat\mathcal{B}_n$) we have $\alpha_{n,k_1}^{(nc)} \leqslant \alpha_{n,k_2}^{(nc)}$.

To simplify the analysis, we consider the degradation obtained when $B_n = \sum_k \frac{c_k W_n}{K}$ compared to the situation where $B_n = W_n / \sum_k \frac{1}{Kc_k}$. It is hence a lower bound on the actual maximum achievable degradation.

Consider now a system of K applications. Suppose that W_n is given. For $B_n = W_n / \sum_k \frac{1}{Kc_k}$, then, for all k, $\alpha_{n,k}^{(nc)} = \frac{B_n}{Kb_k} = \frac{W_n}{\sum_p \frac{b_k}{c_p}}$. When B_n is larger than $\sum_k \frac{c_k W_n}{K}$, then $\alpha_{n,k}^{(nc)} = \frac{W_n}{Kw_k}$. Hence:

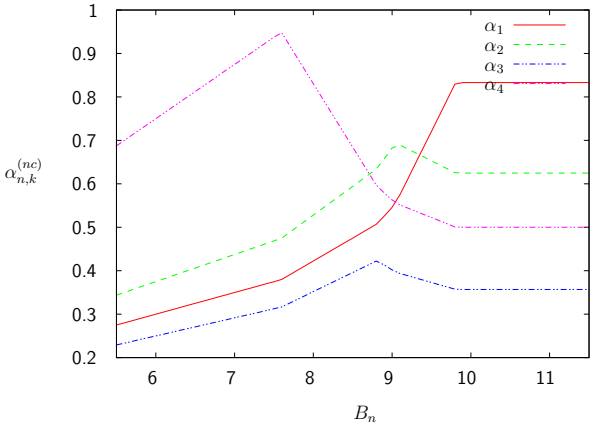
$$\frac{\alpha_{n,k}^{(nc)}{}_{\text{before}}}{\alpha_{n,k}^{(nc)}{}_{\text{after}}} = \frac{K}{\sum_p c_k/c_p}$$

Suppose now that $\forall p \neq k, c_p = K$ and $c_k = 1$. Then $\frac{\alpha_{\text{before}}}{\alpha_{\text{after}}} \sim K/2$. Hence, when the number of applications grows to infinity, the degradation of the application having the smaller c_k also grows to infinity.

Remark 2. *The applications with the smaller coefficient c_k is the most penalized by an increase of the communication resource.*

5.2 Maximal performance

We show in this section that even in a single processor system, the maximal throughput can strictly decrease with the adding of resource, and illustrate it with a numerical example (Fig. 5).



$b = \{5, 4, 6, 2\}$,
$w = \{3, 4, 7, 5\}$,
$K = 4$,
$W_n = 10$.

Figure 5: The maximal throughput can decrease while the resource (the bandwidth) increases.

Theorem 7. *In a system with K applications and a single processor, for a given W_n,*

- *if* $B_n \leqslant \frac{W_n \sum_k c_k}{K}$ *the application* k *having the higher throughput* $\alpha_{n,k}^{(nc)}$ *is the one having the smaller value of* b_k*.*
- *if* $B \geqslant \sum_k \frac{c_k W_n}{K}$ *the application* k *having the higher throughput* $\alpha_{n,k}^{(nc)}$ *is the one whose* w_k *is the smallest.*

Additionally, for given values of c_k*, the degradation of the higher throughput when* B_n *increases can be unbounded. More precisely, a lower bound of the maximal degradation is proportional to* $\frac{K}{\sum_k 1/c_k} \frac{\min_k w_k}{\min_k b_k}$*. Hence, for appropriate choices of* $\min_k b_k$ *and* $\min_k w_k$ *(for all* c_k *fixed), the degradation can be chosen arbitrarily large.*

Proof. Let us consider a system with K applications, with given values of c_k and a given value of W_n. Then,

- when $B_n = W_n / \sum_k \frac{1}{K c_k}$ then $\mathcal{W}_n = \emptyset$ and $\max_k \alpha_{n,k}^{(nc)} = \frac{B_n}{K. \min_k b_k} = \frac{W_n}{(\min_k b_k) \sum_k \frac{1}{c_k}}$
- and when $B_n = \frac{c_k W_n}{K}$ we have $\mathcal{B}_n = \emptyset$ and $\max_k \alpha_{n,k}^{(nc)} = \frac{W_n}{K \min_k w_k}$.

Hence, the application having the higher throughput can be different in $sat\mathcal{B}_n$ and $sat\mathcal{W}_n$ (as illustrated in Fig 5) and the maximum degradation when B grows from $sat\mathcal{B}_n$ to $sat\mathcal{W}_n$ is hence $\frac{K}{\sum 1/c_k} \frac{\min_k w_k}{\min_k b_k}$. □

Consider for example the example depicted in Fig. 5. As $\max_k w_k = w_1$ and $\min_k b_k = b_4$, then the application having the higher throughput is application 1 in $sat\mathcal{B}_n$ and application 4 in $sat\mathcal{W}_n$, and a lower bound of the degradation is 90/79.

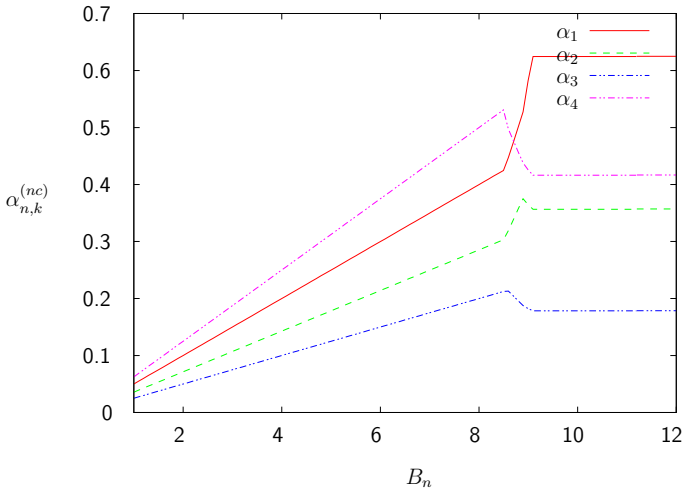
5.3 Minimal performance

We show here a similar result as what we obtained in Section 5.2 for the minimal throughput. That is, even in a single processor system, the minimal performance can decrease with an increase of the resource. We illustrate this property with a numerical example (Fig. 6).

Theorem 8. *In a system with* K *applications and a single processor, for a given* W_n*,*

- *if* $B_n \leqslant W_n / \sum_k \frac{1}{K c_k}$ *the application* k *having the smaller throughput* $\alpha_{n,k}^{(nc)}$ *is the one having the largest value of* w_k*.*
- *if* $B_n \geqslant \sum_k \frac{c_k W_n}{K}$ *the application* k *having the lower throughput* $\alpha_{n,k}^{(nc)}$ *is the one whose* b_k *is the largest.*

Additionally, for given values of c_k*, the degradation of the smaller throughput when* B *increases can be unbounded. More precisely, a lower bound of the maximal degradation is proportional to* $\frac{K}{\sum 1/c_k} \frac{\max_k w_k}{\max_k b_k}$*. Hence, for appropriate choices of* $\max_k b_k$ *and* $\max_k w_k$ *(for all* c_k *fixed), the degradation can be chosen arbitrarily large.*



$b = \{5, 7, 10, 4\}$,
$w = \{4, 7, 14, 6\}$,
$K = 4$,
$W = 10$.

Figure 6: The minimal throughput can decrease with the resource (here the bandwidth)

Proof. Let us consider a system with K applications, with given values of c_k and a given value of W_n. Then,

- when $B_n = W_n / \sum_k \frac{1}{Kc_k}$ then $\mathcal{W}_n = \emptyset$ and $\min_k \alpha_{n,k}^{(nc)} = \dfrac{B_n}{K \max_k b_k} = \dfrac{W_n}{\max_k b_k . \sum_k 1/c_k}$
- and when $B_n = \sum_k \frac{c_k W_n}{K}$ we have $\mathcal{B}_n = \emptyset$ and $\min_k \alpha_{n,k}^{(nc)} = \dfrac{W_n}{K \max_k w_k}$.

Hence, the application having the smaller throughput can be different in $sat\mathcal{B}_n$ and $sat\mathcal{W}_n$ (as illustrated in Fig 6) and the maximum degradation when B_n grows from $sat\mathcal{B}_n$ to $sat\mathcal{W}_n$ is hence $\dfrac{K}{\sum 1/c_k} \dfrac{\max_k w_k}{\max_k b_k}$. □

Consider for example the example depicted in Fig. 6. As $\text{argmax}\{w_k\} = \text{argmax}\{b_k\} = 3$, then the application having the lower throughput is application 3 in both $sat\mathcal{B}_n$ and $sat\mathcal{W}_n$, and a lower bound of the degradation is 56/47.

5.4 Average performance

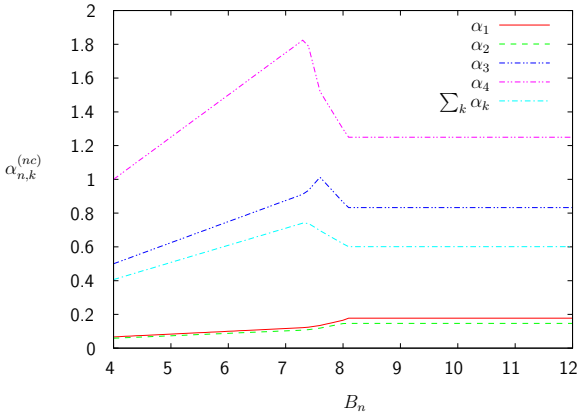
In this section we focus on the monotonicity of the average throughput, and illustrate our result with a numerical example (Fig. 7).

Theorem 9. *Consider, as a performance measure, the average of the applications throughputs. Then, the degradation of these performance measure when B increase can be arbitrarily large.*

Proof. Consider a system with K applications, with given values of c_k and a given value of W_n. Then, when $B_n = W_n / \sum_k \frac{1}{Kc_k}$ then $\mathcal{W}_n = \emptyset$ and $\sum_k \alpha_{n,k}^{(nc)} = \frac{W_n}{\sum_k 1/c_k} \sum_k \frac{1}{b_k}$

and when $B_n = \sum_k \frac{c_k W_n}{K}$ we have $\mathcal{B}_n = \emptyset$ and $\sum_k \alpha_{n,k}^{(nc)} = \frac{W_n}{K} \sum_k \frac{1}{w_k}$. Hence, the performance degradation becomes $\frac{p_{bef}}{p_{aft}} = \frac{K}{\sum_k \frac{w_k}{b_k}} \frac{\sum_k \frac{1}{b_k}}{\sum_k \frac{1}{w_k}}$. Suppose further that the applications are such that $b_1 = w_1 = 1$ and $\forall k \neq 1, b_k = P^2 \quad w_k = P$. Then, the lower bound becomes: $\frac{p_{bef}}{p_{aft}} = K \frac{1 + \frac{K-1}{P^2}}{\left(1 + \frac{K-1}{P}\right)^2} \xrightarrow[P \to \infty]{} K$. □

Consider the example depicted in Fig. 7. A lower bound of the degradation is $17409/14018 \simeq 1.24$.



$b = \{ 15,17,2,1\}$,
$w = \{ 14,17,3,2\}$,
K=4,
W_n =10.

Figure 7: The average throughput can decrease with the resource (here the bandwidth)

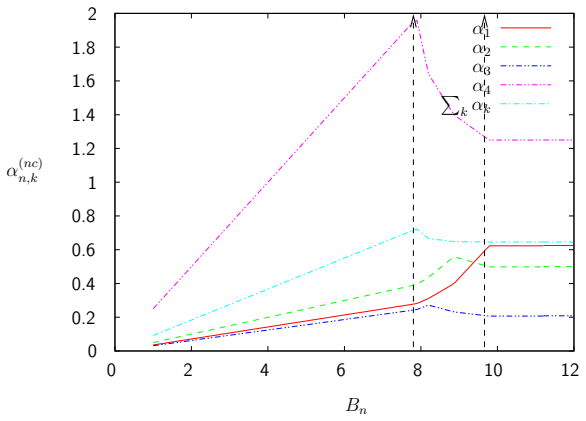
5.5 All measures considered simultaneously

We end this section with an example in which all the performance measures we considered are simultaneously degraded when the bandwidth B_n of the link connecting the master to the slave is increased.

Consider the example represented in Fig. 8. In this example, when the bandwidth B is 9.5 the three measures (namely the higher throughput, the lower throughput and the average throughput) have lower values than when the bandwidth B is only equal to 7.9.

6 Conclusion

We have presented a simple yet realistic situation where the *system-level* fairness fails to achieve a relevant *application-level* fairness. Even though the system achieves a perfect sharing of resources between applications, the non-cooperative usage of the system leads to important application performance degradation and resource wasting. We have proved the existence and uniqueness of the Nash equilibrium in our framework and extensively studied its property. Surprisingly, the equilibrium is Pareto-optimal on each worker independently. However, it may not be globally Pareto-optimal. We have proved that no



$b = \{7, 5, 8, 1\}$,
$w = \{4, 5, 12, 2\}$,
$K = 4$,
$W_n = 10$.

Figure 8: The three performance measures can simultaneously decrease with the resource

Braess-like paradoxical situations could occur, which is, to the best of our knowledge, the first situation where Pareto-inefficient non-cooperative equilibrium cannot lead to Braess-like paradox. However, even if such situations cannot occur, the performances of the equilibrium are relatively poor and can be arbitrarily bad for any classical performance measure.

The key hypothesis for deriving a closed-form description of the equilibria is the multi-port hypothesis. Under this hypothesis, some time information could be lost when using equivalent representations, which resulted in simpler equations than if a 1-port model had been used. Preliminary simulations with this model show that Braess-like paradoxes may occur. The understanding of such situation is crucial to large-scale system planing and development as there is no way to predict their apparition so far. Analytical characterizations of such a framework could provide significant insights on the key ingredients necessary to the occurrence of Braess-like paradoxes.

Last, we can see from this study that cooperation between applications is essential even for simple applications constituted of a huge number of independent identical tasks. As far as the framework of this article is concerned, some steps in this direction have been given in [BLCJF+06] where some distributed algorithms are proposed and compared to an optimal but centralized one. However in this work, there is a single scheduler whose duty is to achieve the best throughput for all applications while ensuring a max-min fairness criteria. In a fully-decentralized setting, some form of cooperation (e.g. similar to the one proposed by [YRRMCR00] for elastic traffic in broadband networks) between different schedulers should be designed.

References

[BLCJF+06] Olivier Beaumont, Larry Carter, Jeanne Ferrante, Arnaud Legrand, Loris Marchal, and Yves Robert. Centralized versus distributed schedulers mul-

tiple bag-of-task applications. In *International Parallel and Distributed Processing Symposium IPDPS'2006*. IEEE Computer Society Press, 2006.

[BNGNS00] Amotz Bar-Noy, Sudipto Guha, Joseph (Seffi) Naor, and Baruch Schieber. Message multicasting in heterogeneous networks. *SIAM Journal on Computing*, 30(2):347–358, 2000.

[BOBLC+04] Cyril Banino, Olivier Beaumont, Larry Carter, Jeanne Ferrante, Arnaud Legrand, and Yves Robert. Scheduling strategies for master-slave tasking on heterogeneous processor platforms. *IEEE Trans. Parallel Distributed Systems*, 15(4):319–330, 2004.

[BOI] Berkeley Open Infrastructure for Network Computing. http://boinc.berkeley.edu.

[Bra68] D. Braess. Über ein paradoxien aus der verkehrsplanung. *Unternehmensforschung*, 12:258–68, 1968.

[BSP+99] M. Banikazemi, J. Sampathkumar, S. Prabhu, D.K. Panda, and P. Sadayappan. Communication modeling of heterogeneous networks of workstations for performance characterization of collective operations. In *HCW'99, the 8th Heterogeneous Computing Workshop*, pages 125–133. IEEE Computer Society Press, 1999.

[CDu+96] James Cowie, Bruce Dodson, R.-Marije Elkenbracht-H uizing, Arjen K. Lenstra, Peter L. Montgomery, and Joerg Zayer. A world wide number field sieve factoring record: on to 512 bits. In Kwangjo Kim and Tsutomu Matsumoto, editors, *Advances in Cryptology - Asiacrypt '96*, volume 1163 of *LNCS*, pages 382–394. Springer Verlag, 1996.

[Dut04] Pierre-François Dutot. Complexity of master-slave tasking on heterogeneous trees. *European Journal of Operational Research*, 2004. Special issue on the Dagstuhl meeting on Scheduling for Computing and Manufacturing systems.

[EIN] Einstein@Home. http://einstein.phys.usm.edu.

[HP04] B. Hong and V.K. Prasanna. Distributed adaptive task allocation in heterogeneous computing environments to maximize throughput. In *International Parallel and Distributed Processing Symposium IPDPS'2004*. IEEE Computer Society Press, 2004.

[LHC] Large Hadron Collider. http://lhc.web.cern.ch/lhc/.

[Nas50] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences USA*, 36:48–49, 1950.

[Nas51] John F. Nash. Noncooperative games. *Annal of Mathematics*, 54:286–295, 1951.

[Pap98] Koutsoupias Papadimitriou. Worst-case equilibria. In *STACS*, 1998.

[Pri] Prime. URL: http://www.mersenne.org.

[SET] SETI. URL: http://setiathome.ssl.berkeley.edu.

[YRRMCR00] Haïkel Yaïche, Ravi R. Mazumdar, and Catherine Rosenberg. A Game Theoretic Framework for Bandwidth Allocation and Pricing in Broadband Networks. *IEEE/ACM Transactions on Networking*, 8(5):667–678, 2000.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399